UNIVERSIDAD POLITÉCNICA DE MADRID

FACULTAD DE INFORMÁTICA

Clasificación Supervisada basada en Redes Bayesianas. Aplicación en Biología Computacional

Tesis Doctoral

Víctor Robles Forcada

Licenciado en Informática

Madrid, 2003

DEPARTAMENTO DE ARQUITECTURA Y
TECNOLOGÍA
DE SISTEMAS INFORMÁTICOS

FACULTAD DE INFORMÁTICA

Clasificación Supervisada basada en Redes Bayesianas. Aplicación en Biología Computacional

Tesis Doctoral

Autor: Víctor Robles Forcada
Licenciado en Informática
Directores: Pedro Larrañaga Múgica
Doctor en Informática
Pedro de Miguel Anasagasti
Doctor en Informática

Madrid, 2003

Título:
Clasificación Supervisada basada en Redes Bayesianas. Aplicación en Biología Computacional

Autor:
Víctor Robles Forcada

TRIBUNAL

Presidente : Antonio Pérez Ambite

Vocales : Juan Miguel Marín Diazaraque

Iñaki Inza Cano

Jose María Carazo García

Secretario : José María Peña Sánchez

Suplentes : Alfonso Rodríguez-Patón Aradas

Xavier Messeger Peypoch

Acuerdan otorgar la calificación de:

Madrid, de de 2003

A mi familia

Agradecimientos

Aquí finaliza un largo camino que realmente ha merecido la pena. Se acaba de cumplir uno de mis sueños en esta vida. Lo mejor de todo es que ha habido mucha gente que me ha ayudado, y a todos ellos van dirigidas estas líneas.

Esta tesis se ha podido realizar gracias al esfuerzo de mucha gente. En primer lugar, quiero agradecer a mis directores, Pedro de Miguel y Pedro Larrañaga, su esfuerzo en la dirección y su apoyo personal. Siempre han estado ahí cuando lo he necesitado.

Quería también agradecer a Pedro Larrañaga el haberme abierto las puertas de su grupo de investigación y el haber confiado en mi desde el primer momento. Siempre he sido muy bien recibido en el País Vasco, y trabajar con gente tan excepcional como vosotros es un lujo.

Detrás de esta memoria está el trabajo de mucha gente. Quiero agradecer a Chema el haber estado siempre ahí ayudándome, respondiendo a mis dudas y aguantando mis charlas. Muchas gracias a Vane, Fernando y Wifre por su esfuerzo y por haber realizado sus proyectos conmigo. Y por supuesto a María y Fran, esos geniales compañeros de despacho que siempre han tenido tiempo para mi.

Por último agradecer a los revisores su arduo trabajo. Gracias a Guille, Chema, Pedro, Pedro y Antonio por su esfuerzo en este sentido.

En el plano personal hay mucha más gente todavía. Para empezar mi familia, a quien dedico esta tesis –sobre todo a las nuevas adquisiciones, Blanca y el gamusino–.

También está toda la gente del trabajo. Gracias a todos los profesores del departamento de arquitectura y sobre todo a mis compis, los sosos, ya que gracias a su apoyo estoy pudiendo escribir estas líneas.

Por último, quiero agradecer el apoyo que me han dado todos mis amigos –a los que prometo dedicar más tiempo a partir de ahora–, y animar a Hugo y a Jose Carlos con sus tesis y a Guille con su residencia en el Hospital de León.

Víctor

Resumen

Los trabajos realizados en esta tesis se encuadran dentro de dos grandes campos: la clasificación supervisada con modelos gráficos probabilísticos y su aplicación a la biología computacional.

La idea fundamental de las propuestas que se han realizado dentro del campo de la clasificación supervisada con modelos gráficos probabilístico, es el uso de los algoritmos heurísticos de optimización EDA en la búsqueda de estructuras de redes Bayesianas para clasificación.

Gracias a la aplicación de los algoritmos EDA, se ha desarrollado un nuevo algoritmo de clasificación supervisada denominado *Interval Estimation naïve-Bayes* y se han mejorado varios de los algoritmos de clasificación propuestos en la literatura. Los resultados experimentales obtenidos han sido muy satisfactorios, ya que demuestran la superioridad de nuestra idea.

Además, con el objetivo de mejorar su rendimiento, se ha desarrollado una versión paralela de nuestro algoritmo, el *Parallel Interval Estimation naïve-Bayes*. Las pruebas experimentales han superado nuestras expectativas iniciales, ya que no sólo se ha logrado un *speedup* superlineal, si no que se han obtenido mejores resultados que en la versión secuencial.

En el campo de la biología computacional la predicción de la estructura secundaria de las proteínas es de vital importancia, ya que proporciona un punto de partida para la predicción de su estructura tridimensional, lo cual ayuda a la determinación de sus funciones.

Dentro de este campo, se ha estudiado la aplicación de los métodos de clasificación supervisada en dos niveles diferentes. Por un lado, se ha desarrollado un nuevo método basado en redes Bayesianas, para la predicción de la estructura secundaria de las proteínas. Aunque en primera instancia los resultados obtenidos no han sido brillantes, en esta tesis se sugieren refinamientos de la idea original que, confiamos, los mejorarán.

Por otra parte, se ha creado un multiclasificador con los métodos de predicción existentes, basado en el paradigma *stacked generalization*. Los resultados obtenidos por este multiclasificador han sido altamente satisfactorios, ya que se han mejorado los resultados de los métodos individuales.

Como resultado de las propuestas realizadas, han surgido multitud de futuras líneas de investigación, que se recogen a lo largo de esta tesis.

Abstract

This thesis is based on two different fields: supervised classification with probabilistic graphical models and its application to computational biology.

The main idea behind our proposals in the field of supervised classification is the use of heuristic optimization algorithms in the search of Bayesian network structures for classification.

Thanks to the application of EDAs, we have developed a new supervised classification algorithm named *Interval Estimation naïve-Bayes*, and we have improved some classification algorithms proposed in the literature. Experimental results are quite satisfactory, and show the superiority of our idea.

Besides, with the aim of improving its performance, we have developed a parallel version of the algorithm, the *Parallel Interval Estimation naïve-Bayes*. Experimental results show an improvement in both the performance, achieving a superlineal speedup, and the obtained results.

In the field of computational biology, the secondary structure prediction of proteins is really important, because it provides a starting point for tertiary structure prediction and function determination.

Inside this field, we have studied the application of supervised classification at two different levels. On one hand, we have developed a new method based on Bayesian networks for the secondary structure prediction. Although the results are not brilliant, in this thesis we suggest further works to improve them.

On the other hand, we have developed a multiclassifier with the existing methods, based on the paradigm named *stacked generalization*. The results obtained by the multiclassifier are really satisfactory. We have improved the results of the individual methods.

As a consequence of all the proposals, we have suggested a lot of further future work.

Índice general

Índice de figuras

Índice de tablas

Parte I

INTRODUCCIÓN Y OBJETIVOS

Capítulo 1

Introducción y objetivos

Durante los últimos años, la comunidad biomédica ha incrementando en gran medida el uso de las ciencias de la computación y de su tecnología. Esta fusión de la biomedicina y de la tecnología computacional está ofreciendo sustanciales beneficios que mejoran la calidad de la salud de los individuos e incrementan el conocimiento biológico.

En la línea marcada por esta fusión, los trabajos realizados en esta tesis se encuadran dentro de dos grandes campos: la clasificación supervisada y la biología computacional. Día a día, son cada vez más las aportaciones que la clasificación supervisada realiza en el campo de la biología computacional, siendo la presente tesis un paso más dentro de este amplio mundo.

Este capítulo explica las relaciones existentes entre la clasificación supervisada y la biología computacional, y establece los objetivos de la tesis. El índice del capítulo es el siguiente:

- En la sección 1.1 se realiza una breve introducción a la jerarquía del aprendizaje, dentro de la cual se puede localizar el aprendizaje con clasificación supervisada.
- En la sección 1.2 se definen dos términos estrechamente relacionados, la bioinformática y la biología computacional.
- En la sección 1.3 se establece la relación entre la clasificación supervisada y la biología computacional.
- En la sección 1.4 se presentan los objetivos de esta tesis.
- La sección 1.5 contiene la organización del resto de la tesis.

1.1. Aprendizaje con clasificación supervisada

Preguntas siempre difíciles de responder son: ¿qué es el aprendizaje? y ¿qué es la clasificación supervisada? En la figura 1.1 se encuentra representada la jerarquía del aprendizaje automático realizada por Kohavi [Koh95], dentro de la cual se localiza el aprendizaje con clasificación supervisada.

El **aprendizaje** se puede definir [Sim83] como "cambios adaptativos en un sistema, en el sentido de que permiten al sistema realizar las mismas tareas pero de forma más

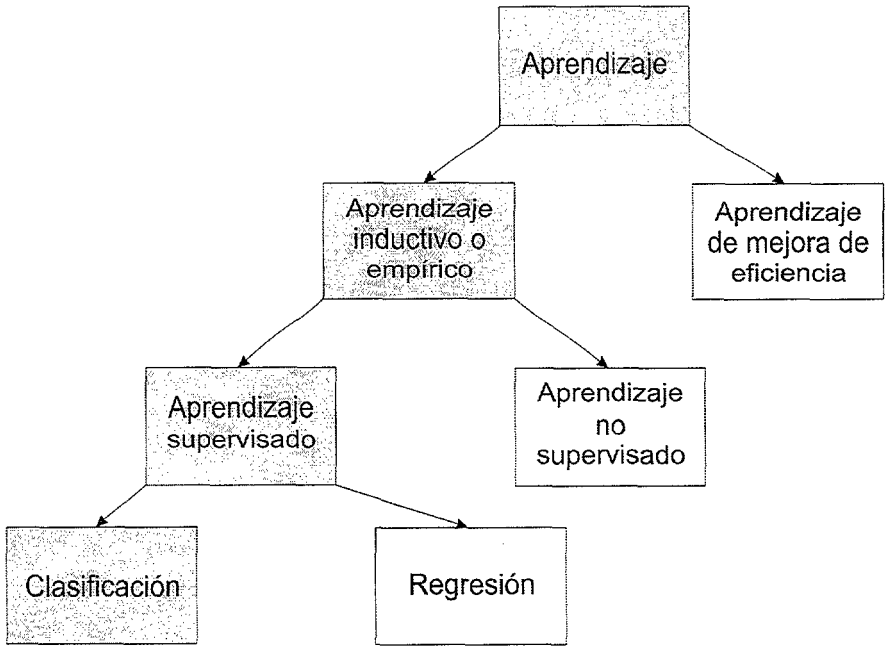


Figura 1.1: La jerarquía del aprendizaje automático. Los cuadrados sombreados llevan al aprendizaje con clasificación supervisada, objeto de esta tesis

eficiente y más efectiva la siguiente vez". El aprendizaje se puede dividir en dos grandes ramas, el aprendizaje de mejora de eficiencia y el aprendizaje inductivo o empírico, que se define a continuación.

El **aprendizaje empírico** "se realiza a través de razonamientos de ejemplos suministrados externamente, para producir reglas de carácter general" [Die86]. Dentro del dominio del aprendizaje inductivo esta tesis se centra en el aprendizaje con clasificación supervisada.

En el **aprendizaje con clasificación supervisada** el algoritmo de inducción recibe un conjunto de ejemplos, denominado conjunto de entrenamiento, en el que cada instancia consiste en una lista de valores de características y una etiqueta discreta. La tarea del algoritmo es aprender modelos, en amplio sentido, que clasifiquen correctamente instancias sin etiquetar. El término "supervisado" sugiere que algún proceso ha etiquetado las instancias del conjunto de entrenamiento. El término "clasificación" denota el hecho de que la etiqueta es *discreta*, es decir, que consiste en un número finito de valores.

En el mundo de la clasificación supervisada existen muchos clasificadores que se pueden construir a partir de los datos. Esta tesis se centra en el uso de algoritmos de aprendizaje que permiten construir clasificadores basados en redes Bayesianas. Estos algoritmos contribuyen a una mejor comprensión de los datos, ya que se aprenden estructuras comprensibles, que ayudan a entender el dominio [New82].

Tal y como se muestra en la figura 1.1, el aprendizaje con clasificación supervisada es un subconjunto del campo del aprendizaje automático. El **aprendizaje para mejorar la eficiencia** (*speedup learning*), que tiene como su máximo representante al aprendizaje basado en explicaciones [MKKC86], intenta mejorar la eficiencia de procesos existentes

mediante el aprendizaje automático. El **aprendizaje no supervisado**, como los métodos de *clustering* [And73], trata de descubrir estructuras en instancias no etiquetadas. Por último, la **regresión** intenta asociar una función de proyección de instancias no etiquetadas a una etiqueta con valor real [BFOS84, DS81].

1.1.1. Motivaciones del aprendizaje con clasificación supervisada

Existen tres motivaciones principales para la existencia del aprendizaje con clasificación supervisada: la minería de datos o *data mining*, la mejora en la adquisición de conocimientos y el conseguir mejorar el conocimiento de los expertos. A continuación se detalla cada uno de ellos.

Data mining

Se puede definir *data mining* como "el proceso de identificar patrones válidos, novedosos, potencialmente útiles y comprensibles en los datos" [FPSS96]. *Data mining*, como punto de encuentro de varias disciplinas, trata de extraer conocimiento de grandes cantidades de datos usando técnicas estadísticas, de aprendizaje automático, de análisis de patrones o de bases de datos.

La cantidad de datos recopilados en algunas bases de datos está creciendo rápidamente. Este hecho se debe a que las tecnologías de almacenamiento y de recolección de datos están mejorando. El campo de la biología computacional es uno de los ejemplos más representativos. Bases de datos como la PDB (*Protein Data Bank*) [BW03], que contiene la secuencia de las proteínas actualmente conocidas, tiene gigabytes de información.

La habilidad de extraer información interesante y de comprender los datos almacenados en estas bases de datos es de vital importancia. El campo de *data mining* está creciendo rápidamente y al mismo ritmo que las necesidades de almacenamiento de datos.

Por último, es importante destacar que el campo de *data mining* paralelo tiene mucha importancia debido a la gran cantidad de datos con los que normalmente se trabaja y a lo costoso que son (computacionalmente hablando) los procesos de análisis de los mismos. Esto hace que *data mining* se encuadre dentro del marco de las aplicaciones de supercomputación.

Mejorar la adquisición de conocimientos

Los sistemas expertos [BS88] solucionan problemas que normalmente son resueltos por expertos humanos. Para resolver problemas al nivel de un experto, estos sistemas necesitan construir una base de conocimientos de tamaño considerable. Una tarea que normalmente es realizada por un ingeniero de conocimiento. La construcción de la base de conocimientos es la fase que está considerada como el cuello de botella de un sistema experto.

Una motivación de la investigación en el aprendizaje automático, especialmente de la clasificación supervisada, es conseguir recortar el tiempo de adquisición de conocimientos de forma drástica. En la construcción de un sistema experto estándar "uno de los objetivos del ingeniero es convertir el saber-cómo (*know-how*) de un humano en el decir-cómo (*say-how*)... y programar en máquina el saber-cómo" [Mic87]. La forma de acortar el tiempo

de adquisición de conocimientos es evitar el estrecho canal, que supone la ingeniería del conocimiento, utilizando *data mining* sobre ejemplos reales.

Mejorar el conocimiento de los expertos

Existen diversos trabajos en los que se asegura que los sistemas de ayuda a la decisión han superado a los diagnósticos de los expertos.

Por ejemplo, Kononenko [Kon93] referencia 24 artículos en los que los sistemas de inducción han sido utilizados para su aplicación a dominios médicos tales como la oncología, los pronósticos de hepatitis, etc. Además, se remarca que "típicamente, las reglas de diagnóstico generadas automáticamente mejoran significativamente la exactitud de los diagnósticos de los especialistas".

Otro ejemplo se encuentra en Fayyad e Irani [FI93], que aseguran que su árbol de decisión mejora a los astrónomos en la observación del cielo, y que, para la mayoría de estos objetos, los astrónomos "no eran capaces de determinar las clases examinando las imágenes de observación".

1.2. Biología computacional y bioinformática

La biología computacional y la bioinformática son enfoques interdisciplinares que parten de ciencias específicas como las matemáticas, la física, las ciencias de la computación, la biología y las ciencias del comportamiento.

La bioinformática aplica el conocimiento de las tecnologías y de las ciencias de la información al vasto, diverso y complejo mundo de los datos de las ciencias de la vida, para hacerlos más comprensibles y usables. La biología computacional utiliza enfoques matemáticos y computacionales para resolver cuestiones teóricas y experimentales de la biología. Aunque la bioinformática y la biología computacional son diferentes, existe un solapamiento significativo entre ellas, como se puede observar a continuación.

1.2.1. Definiciones

Las siguientes definiciones provienen del NIH–BISTIC (*National Institute of Health–Biomedical Information Science and Technology Initiative Consortium*) de Maryland [NB03]. Ellos mismos reconocen que las definiciones de bioinformática y biología computacional nunca podrán eliminar el solapamiento con otras actividades, o impedir diferentes interpretaciones.

- Bioinformática: Investigación, desarrollo o aplicación de herramientas computacionales para expandir el uso de datos biológicos, médicos, de comportamiento o de salud, incluyendo aquellas que permiten adquirir, almacenar, organizar, archivar, analizar o visualizar tales datos.

- Biología computacional: Desarrollo y aplicación de métodos teóricos y de análisis de datos, modelado matemático y técnicas de simulación al estudio de la biología, el comportamiento y los sistemas sociales.

1.3. *Data mining* en biología computacional

Como se vió en el apartado anterior, la biología computacional se puede definir como el desarrollo y la aplicación de métodos teóricos y de análisis de datos a la biología molecular. La biología computacional ha evolucionado desde su aparición. De ser una simple herramienta de soporte a la investigación, ha pasado a ser, hoy en día, una disciplina científica. Uno de los principales objetivos de la biología computacional es la investigación y el desarrollo de herramientas útiles que permitan comprender la información biológica lo mejor posible. Las estructuras moleculares, las funciones bioquímicas y el comportamiento biológico de los genes y proteínas se tratan en esta disciplina con el objetivo de descubrir su influencia en las enfermedades y en la salud. La **genómica** (ciencia que estudia el genoma –conjunto completo de genes de un organismo–) y la **proteómica** (ciencia que estudia la estructura, función, localización e interacción de las proteínas, dentro y fuera de la célula) son actualmente dos de los campos más importantes de la biología computacional, en los que las técnicas de *data mining* están siendo más utilizadas.

La genómica y la proteómica están permitiendo a los investigadores realizar diferentes mejoras médicas. El estudio de la genómica está proporcionando información útil que puede ser utilizada en la investigación biomédica. A día de hoy, esta información está siendo utilizada en la práctica clínica, como terapia y como diagnosis. Por otra parte, la proteómica está generando importante información sobre las proteínas, proporcionando datos sobre su función y su estructura. Esta información está permitiendo grandes avances en el campo del diseño de fármacos y de la industria protéica.

Gracias a las nuevas tecnologías para la obtención de información de genes y proteínas, se hace posible obtener mayores avances en la ciencia de la biomedicina. Sin embargo, debido a la gran cantidad de información obtenida por estas tecnologías, se hace necesaria la utilización de técnicas de aprendizaje automático para el procesado de la información y la posterior obtención de conocimiento. Las técnicas de *data mining* tales como las redes Bayesianas, las redes neuronales, los árboles de decisión, etc., están obteniendo importantes resultados en el análisis de datos dentro de estos campos.

Los campos de la biología computacional son múltiples. Sirvan como ejemplos los siguientes:

- Localización de enfermedades asociadas a determinados genes o proteínas.
- Localización de información de genes o proteínas.
- *Microarrays* de ADN.
- *Microarrays* de proteínas.
- Investigación biofarmacéutica y de desarrollo de fármacos.
- Predicción de la estructura de proteínas.
- Evolución molecular.
- Ensamblado, agrupamiento y proyección de ADN.
- Algoritmos de reconstrucción biogenéticos.

1.4. Objetivos

Los trabajos que se llevan a cabo en esta tesis se enmarcan dentro de dos grandes áreas: la clasificación supervisada con redes Bayesianas y la biología computacional. Además, dentro del área de la biología computacional, se centra en un determinado campo de la proteómica como es el de la predicción de la estructura secundaria de las proteínas (PSSP).

En los siguientes apartados se describen los principales objetivos.

1.4.1. Clasificación supervisada con modelos gráficos probabilísticos

En los últimos años ha habido un interés creciente en la utilización de modelos gráficos probabilísticos para clasificación. Éstos han demostrado acomodarse a la naturaleza flexible de numerosos conceptos y ,además, gozan de una sólida base en la teoría de la probabilidad.

El método gráfico probabilístico para clasificación más ampliamente utilizado es conocido como naïve-Bayes [DH73, HY01] y se basa en la aplicación del teorema de Bayes. Este método parte de la suposición de la independencia condicional de los atributos predictores dada la clase.

Además, las redes Bayesianas también se han incorporado a tareas de clasificación supervisada [FG96, SL98]. De esta forma, es posible utilizar factorizaciones de probabilidad representadas por las redes Bayesianas para realizar clasificaciones.

Dentro de este campo se plantean los objetivos que a continuación se detallan.

Desarrollo de un nuevo algoritmo de clasificación basado en naïve-Bayes

Como ya se ha comentado, el método probabilístico para clasificación más utilizado es conocido como naïve-Bayes. Dicho método se basa en un producto de probabilidades condicionales, que son estimadas a partir de los datos del conjunto de entrenamiento gracias a una estimación puntual.

Nos proponemos analizar la implementación de un nuevo algoritmo en el cual, en lugar de estimar probabilidades puntuales de los datos como se hace en naïve-Bayes, se realicen estimaciones por intervalo. A continuación, a través de algoritmos de optimización heurísticos, se tratará de buscar la combinación de valores dentro de los intervalos que maximiza el tanto por ciento de bien clasificados. Estos valores serán las nuevas y mejoradas probabilidades que se usarán en naïve-Bayes.

Aprendizaje de clasificadores semi naïve-Bayes

No es de extrañar que, debido al gran éxito cosechado por el clasificador naïve-Bayes, sean muchas las variantes de este método que intentan mejorar sus resultados. Las variantes de este método son genéricamente denominadas enfoques semi naïve-Bayes [HY01]. En el capítulo 3 se describen los principales algoritmos semi naïve-Bayes existentes en la literatura.

Los clasificadores semi naïve-Bayes están normalmente basados en la realización de búsquedas de determinadas estructuras. Estas búsquedas se suelen realizar a través de algoritmos de búsqueda voraz (*greedy*).

En esta tesis estudiaremos la realización de búsquedas heurísticas de estructuras propuestas en diversos enfoques semi naïve-Bayes, frente a las búsquedas voraces realizadas en la actualidad.

Aprendizaje de clasificadores basados en redes Bayesianas

Para poder utilizar las redes Bayesianas como clasificadores supervisados es fundamental que la estructura de red utilizada sea aprendida con objetivos clasificatorios, esto es, que se haya tenido en cuenta en el proceso de aprendizaje estructural, el hecho de que existe una variable cuya probabilidad a priori es primordial en el manejo de la red Bayesiana.

En la literatura existen diversos algoritmos que realizan búsquedas de este tipo de estructuras, basándose en la cantidad de información mutua condicionada a la variable clase. En esta tesis nos proponemos el desarrollo de algoritmos voraces y heurísticos, guiados por el tanto por ciento de bien clasificados, para la obtención de dichas estructuras.

1.4.2. Problemática computacional

Sabemos que algunas de las propuestas anteriormente citadas pueden ser muy costosas desde el punto de vista computacional, en gran medida por las búsquedas heurísticas que se deben realizar.

Por tanto, haremos un estudio y, en caso necesario, una paralelización de los algoritmos más costosos.

1.4.3. Predicción de la estructura secundaria de las proteínas con redes Bayesianas

Las importantísimas aplicaciones de las proteínas, tanto a nivel médico como a nivel industrial, han creado un enorme interés en el diseño e implementación de técnicas que permitan predecir la estructura tridimensional de las mismas. La estructura tridimensional de una proteína es la que nos indica su función y, por tanto, las aplicaciones que tendrá.

Esta parte de la tesis se encuadra dentro de la predicción de la estructura secundaria de las proteínas (PSSP). El estudio de la estructura secundaria de las proteínas es de vital importancia ya que [SLB00] proporciona un punto de partida para la predicción de la estructura tridimensional de las proteínas y puede mejorar significativamente el análisis de secuencias o las técnicas de *threading* [RCB96], que ayudan en la determinación de la estructura y función de las proteínas.

Dentro del campo de la predicción de la estructura secundaria de las proteínas nos marcamos diversos objetivos que a continuación se detallan.

Modelización matemática de las proteínas y del problema PSSP

Nos proponemos realizar una modelización de las proteínas y del problema de la predicción de la estructura secundaria.

Esta modelización matemática resultaría de gran utilidad, ya que permitiría acercar el campo de la proteómica y de la predicción de la estructura secundaria a los científicos de

la comunidad del aprendizaje automático.

Desarrollo de un clasificador para la PSSP con redes Bayesianas

Desde que se comenzó a estudiar el problema PSSP [CF74] han sido varios los algoritmos de predicción utilizados. Es posible destacar los algoritmos basados en información estadística [GOR78], propiedades físico-químicas [CK89], redes de neuronas multicapa [QS88, Ros98, Jon99, PPRB01], teoría de grafos [MS97], reglas expertas [FA95, ZB96] y algoritmos del vecino más cercano [SS95]. De entre ellos, los algoritmos que más éxito han tenido en la predicción son los basados en redes neuronales.

Como se puede observar, por el momento, no hay ningún estudio realizado sobre la predicción con redes Bayesianas.

En consecuencia, proponemos el uso de las redes Bayesianas para realizar la predicción de la estructura secundaria de las proteínas. Este uso puede suponer dos ventajas principales:

- Mejoras en el proceso de aprendizaje, ya que uno de los principales problemas que presentan las redes neuronales es el tiempo que se debe emplear en el mismo, que puede llegar a ser de semanas.
- Mejoras en la transparencia del aprendizaje, ya que los clasificadores basados en redes Bayesianas generan estructuras comprensibles en contraposición con los modelos obtenidos con las redes neuronales.

Multiclasificador basado en redes Bayesianas para la PSSP

Como se ha podido ver en el apartado anterior, son varios los algoritmos de predicción existentes para el problema PSSP. Lo que proponemos en este punto es la realización de un multiclasificador basado en el conocido paradigma *stacked generalization* [Wol92].

1.5. Organización de la tesis

La tesis se divide en cinco bloques temáticos, cada uno de los cuales se divide a su vez en capítulos:

- El primer bloque presenta una introducción general a la tesis.
  - En el capítulo 1 se realiza una introducción general a la tesis y se presentan los principales objetivos.
- El segundo bloque recoge el estado actual de las áreas de investigación relacionadas con la presente tesis.
  - El capítulo 2 presenta una introducción general al aprendizaje con clasificación supervisada.
  - El capítulo 3 realiza una introducción a las redes Bayesianas y analiza el estado del arte de la clasificación supervisada con modelos gráficos probabilísticos.

- El tercer bloque, que es una de las partes centrales de esta tesis, presenta las diferentes propuestas que se realizan dentro del campo de la clasificación supervisada con modelos gráficos probabilísticos.
  - El capítulo 4 realiza un análisis general de las propuestas de la tesis e introduce los algoritmos de estimación de distribuciones.
  - En el capítulo 5 se presentan y evalúan diversas propuestas en el campo de la clasificación supervisada con modelos gráficos probabilísticos.
- El cuarto bloque está dedicado a la aplicación de los resultados obtenidos en clasificación supervisada a la biología computacional.
  - El capítulo 6 presenta el estado del arte de la predicción de la estructura secundaria de las proteínas.
  - En el capítulo 7 se presentan diversas propuestas en el campo de la predicción de la estructura secundaria de las proteínas.
- El quinto y último bloque enumera las conclusiones y define las futuras líneas de trabajo.
  - El capítulo 8 analiza las aportaciones realizadas por la presente tesis y describe las posibles líneas de investigación y desarrollo que se derivan de la misma.

Parte II

ESTADO DEL ARTE

Capítulo 2

Aprendizaje con clasificación supervisada

Este capítulo presenta una introducción general al aprendizaje con clasificación supervisada. La organización del capítulo es la siguiente:

- En la sección 2.1 se define formalmente el aprendizaje con clasificación supervisada.
- En la sección 2.2 se describen los principales algoritmos de clasificación supervisada, dentro de los cuales se encuentran los basados en redes Bayesianas.
- Por último, en la sección 2.3 se analizan las diferentes técnicas de validación (estimación de la exactitud) de estos algoritmos.

2.1. Definiciones

Informalmente, la tarea de un algoritmo de clasificación supervisada es generar un buen clasificador a partir de un conjunto de ejemplos etiquetados. A continuación, este clasificador puede ser utilizado para identificar casos no etiquetados, con el objetivo de predecir la clase correcta. Un clasificador puede ser evaluado por su exactitud, comprensibilidad u otras propiedades deseables que determinen cómo de apropiado es para la tarea a realizar.

Una **instancia**, también llamada **caso** o **ejemplo**, es una lista fija de valores de **atributos**. Una instancia describe las entidades básicas con las que se trabajará, tales como una persona, una seta o una secuencia de ADN.

Un **atributo**, a veces llamado **variable**, describe alguna propiedad de una instancia. Se utilizan dos tipos de atributos: discretos, que a su vez pueden ser nominales u ordinales (por ejemplo, un atributo discreto nominal puede ser $color \in \{rojo, verde, azul\}$) y continuos (por ejemplo, $peso \in \mathbb{R}^+$)).

Cada instancia tiene un atributo especial, la **clase**, que describe el fenómeno que se quiere aprender o sobre el que se desea hacer predicciones. Una **instancia no clasificada** es la parte de la instancia sin la clase, es decir, la lista de los valores de las características o atributos.

Un **conjunto de datos** (*dataset*) es un conjunto de instancias clasificadas. La tabla 2.1 muestra un conjunto de datos con nueve instancias en el dominio de enfermedades del corazón. La última columna, *Enfermo*, es la que se intenta predecir a partir del resto de los atributos.

Edad (cont.)	Sexo {*M,F*}	Colesterol (cont.)	ECG restante {*norm,abn,hyp*}	Max. num. latidos (cont.)	Enfermo {*si,no*}
53	M	203	hyp	155	si
60	M	185	hyp	155	si
40	M	199	norm	178	no
46	F	243	norm	144	no
62	F	294	norm	162	no
43	M	177	hyp	120	si
76	F	197	abn	116	no
62	M	267	norm	99	si
57	M	274	norm	88	si

Tabla 2.1: Conjunto de datos en el dominio de enfermedades del corazón

Un **clasificador** es una función que obtiene la clase de una instancia sin clasificar. Todos los clasificadores tienen una estructura de datos almacenada que deben interpretar a la hora de generar la clase para la instancia sin clasificar. Por ejemplo, los árboles de decisión tienen almacenado un árbol que proyecta una instancia no clasificada a una determinada categoría siguiendo el camino desde la raíz hasta las hojas del árbol y devolviendo la categoría de la correspondiente hoja.

Un **algoritmo de inducción** construye un clasificador a partir de un conjunto de datos dado. Por ejemplo, **CART** [BFOS84] y **C4.5** [Qui93] son algoritmos de aprendizaje que construyen clasificadores basados en árboles de decisión a partir de conjuntos de datos.

La **exactitud** (*accuracy*) de un clasificador es la probabilidad de clasificar correctamente una instancia seleccionada al azar.

La tarea de un algoritmo de inducción es generar un clasificador con las siguientes características deseables:

- Que sea exacto. Este requisito es normalmente la característica más importante, y será la principal consideración a lo largo de esta tesis.
- Que sea comprensible. Dados dos clasificadores con aproximadamente la misma exactitud, se preferirá el más comprensible. Para algunos dominios, como los dominios médicos, la comprensibilidad es crucial. Para otros dominios, como el reconocimiento de caracteres ópticos, este aspecto no es muy importante. Una de las ventajas de los clasificadores supervisados basados en redes Bayesianas es que son muy comprensibles.
- Que sea compacto. Aunque está relacionada con la comprensibilidad, una característica no implica la otra. Un perceptrón puede ser un clasificador compacto, pero dada una instancia, entender el proceso de clasificación es muy complicado. En el otro extremo, un árbol de decisión puede ser muy grande, pero el proceso de la clasificación de las instancias es trivial.

A continuación se describe formalmente la notación que será utilizada a lo largo de toda la tesis. Al conjunto de posibles valores (dominio) de un atributo X_i se le denota por $Dom(X_i)$. Se supone que la cardinalidad de $Dom(X_i)$ es r_i. Cada instancia no etiquetada es un elemento del espacio de instancias no clasificadas $\mathcal{X} = Dom(X_1) \times Dom(X_2) \times \ldots \times Dom(X_n)$, donde n es el número de atributos. A una instancia no clasificada la denotamos por $\mathbf{x}$. Al valor de un atributo específico X_i se le denotará como x_i.

Sea $\mathcal{C}$ el conjunto de los posibles valores de la clase C, es decir, $\mathcal{C} = Dom(C)$. Cada posible valor de la clase se denota por c. Sea $\mathcal{X} \times \mathcal{C}$ el espacio de las instancias clasificadas y $\mathcal{D}$ un conjunto de datos con N instancias clasificadas donde $\mathcal{D} = \{(\mathbf{x}^{(1)}, c^{(1)}), \ldots, (\mathbf{x}^{(N)}, c^{(N)})\}$

Un clasificador genera una clase $c \in \mathcal{C}$ para cada instancia no clasificada $\mathbf{x} \in \mathcal{X}$ y un algoritmo de aprendizaje $\mathcal{I}$ genera un clasificador dado un conjunto de datos $\mathcal{D}$. La notación $\mathcal{I}(\mathcal{D}, \mathbf{x})$ denotará la clase asignada a una instancia no etiquetada $\mathbf{x}$ por el clasificador construido por el algoritmo de aprendizaje $\mathcal{I}$ sobre el conjunto de datos $\mathcal{D}$.

2.2. Algoritmos de inducción

En esta sección se realiza una breve introducción a los principales algoritmos de clasificación supervisada existentes.

Conjuntos de reglas

Uno de los paradigmas más fácilmente interpretable del aprendizaje automático es el constituido por **conjuntos de reglas** del tipo si-entonces (*if-then*). El objetivo de este paradigma es el aprendizaje de un conjunto de reglas cortas, simples y comprensibles en dominios con ruido, que sirvan para discriminar entre las categorías del problema. Dentro de los diferentes enfoques de este paradigma están los algoritmos **CN2** [CN89], **RIPPER** [Coh95] y **OneR** [Hol94], los cuales guardan numerosas similitudes entre sí.

Árboles de clasificación

Los árboles de clasificación son uno de los paradigmas más clásicos y ampliamente usados del mundo del aprendizaje automático. En la literatura acerca de este paradigma, se encuentran numerosas variantes como **CART** [BFOS84], **ID3** [Qui86], **OC1** [MS94] o **C4.5** [Qui93]. Un árbol de clasificación está formado por nodos, ramas y hojas. Cada nodo representa un test univariado o decisión sobre los valores de un atributo concreto. El primer nodo del árbol es conocido como el nodo raíz. Finalmente están los nodos terminales u hojas en los que se toma una decisión acerca de la clase a asignar. Así, a la hora de clasificar un nuevo caso, tendrán que compararse los valores de los atributos con las decisiones o tests que se toman en los nodos, siguiendo la rama que coincida con dichos valores en cada test. Finalmente se llega a un nodo terminal u hoja que predice la clase para el caso tratado. Un árbol de decisión también se puede ver como un conjunto de reglas si-entonces, si bien la diferencia más obvia entre los dos paradigmas es que las reglas de decisión son independientes entre sí, mientras que las reglas extraídas del árbol de decisión no lo son.

K vecinos más próximos

El paradigma de los **K vecinos más próximos** (*K nearest neighbour*) [Das91] es uno de los paradigmas de más sencilla comprensión del aprendizaje automático. Para clasificar un nuevo individuo se guardan en memoria todas las instancias del conjunto de entrenamiento. Se calculan (mediante el uso de una distancia específica) las distancias entre la nueva instancia y todas las instancias del conjunto de entrenamiento. Y, por último, teniendo en cuenta las K distancias más cercanas del conjunto de entrenamiento, se obtiene la etiqueta de clase.

Así expuesto, el algoritmo admite varias variantes, sea por el tipo de distancia que se utilice (euclidea, de Mahalanobis, ...), sea por la ponderación de cada atributo en el cálculo de las distancias o por la forma en que las K instancias más cercanas asignan la clase a la nueva instancia (ponderando cada instancia mediante su distancia, voto por mayoría, ...).

Aunque de sencilla comprensión, este algoritmo puede ser inabordable para bases de datos grandes, dada la cantidad de memoria necesaria para almacenar las instancias del conjunto de entrenamiento y la cantidad de cálculos en el cómputo de las distancias.

Redes neuronales

Una red neuronal puede ser definida como [Hay99]: "modelo computacional con un conjunto de propiedades específicas, como son la habilidad de adaptarse o aprender, generalizar u organizar la información, todo ello basado en un procesamiento eminentemente paralelo".

Se define como arquitectura de una red neuronal a la manera en que se interconectan los distintos elementos de proceso (neuronas artificiales) que forman la red. Normalmente los elementos de proceso se organizan como una secuencia de capas con un determinado patrón de interconexión entre los diferentes elementos de proceso que las forman, y con un patrón de conexión entre los elementos de proceso de las distintas capas.

Una de las principales ideas sobre las que se basan las redes neuronales artificiales, es la de responder a los estímulos del entorno mediante un proceso de aprendizaje por el cual va adaptando los pesos de las conexiones de sus elementos de proceso, de tal forma que se "memorizan" los ejemplos de entrenamiento que se le presentan. El paradigma de aprendizaje indica la forma en que el entorno influye en ese proceso de aprendizaje.

Dentro de la clasificación supervisada, la arquitectura de red neuronal más utilizada es la de perceptrón o perceptrón multicapa. Los algoritmos de aprendizaje utilizados en este tipo de red neuronal son la retropropagación del error, ADALINE [WH60] y MADALINE [Wid87] entre otros.

Naïve-Bayes

En los últimos años ha habido un interés creciente en la utilización de métodos probabilísticos para clasificación. Éstos han demostrado acomodarse a la naturaleza flexible de numerosos conceptos, y, además, gozan de una sólida base en la teoría de la probabilidad. El método probabilístico para clasificación más ampliamente utilizado es conocido como el método **naïve-Bayes** o simple naïve-Bayes [DH73, HY01]. Éste método se basa en una aplicación del teorema de Bayes, pero con unas restricciones y suposiciones de partida.

Una parte importante de esta tesis está basada en este método, por lo que se explicará en profundidad en el capítulo 3.

Redes Bayesianas

Las redes Bayesianas se han incorporado recientemente a las tareas de clasificación supervisada [CGK+02, CG99, FGG97], ya que es posible utilizar las factorizaciones de probabilidad representadas por las mismas para realizar clasificadores. Para ello se debe considerar la existencia de una variable especial, la variable a clasificar, que tiene que ser predicha por el resto de las variables. De esta forma, la estructura de red obtenida se puede utilizar en la predicción del valor de la clase de la variable especial, mediante la asignación de valores a las variables predictoras y el cálculo de la probabilidad a posteriori del nodo asociado a dicha variable.

Para realizar esta labor es fundamental que la estructura de red utilizada sea aprendida con objetivos clasificatorios, esto es, que se haya tenido en cuenta en el proceso de aprendizaje estructural el hecho de que existe una variable cuya probabilidad a posteriori es primordial en el manejo de la red Bayesiana obtenida.

Las redes Bayesianas como clasificadores supervisados son parte central de esta tesis. El estado del arte relativo a este campo será revisado en el capítulo 3.

Por último, es importante destacar que el paradigma naïve-Bayes es un caso particular de red Bayesiana.

2.3. Validación

Estimar la exactitud (*accuracy*) de un clasificador inducido por un algoritmo de aprendizaje automático, es decir, validar un clasificador, es importante no sólo para predecir su futuro comportamiento, sino también para poder escoger un clasificador (**selección de modelo**) [Sch83] dentro de un conjunto de posibilidades, o para combinar clasificadores [Wol92] [Bre94]. Para estimar la exactitud final de un clasificador, lo deseable es tener un método con poca varianza.

Como ya se comentó, la **exactitud** de un clasificador es la probabilidad con la que dicho clasificador clasifica correctamente una instancia seleccionada al azar. Algunos investigadores, sobre todo en la comunidad estadística, usan **ratios de error** (uno menos la exactitud) en lugar de la exactitud. En esta tesis se ha escogido la exactitud porque es la medida más usual en la comunidad del aprendizaje automático. Sin embargo, se utilizarán ratios de error cuando las ventajas sean claras.

Además se utilizará la **reducción relativa del error** entre un algoritmo A y un algoritmo B con mayor exactitud definida como $(error(A) - error(B)/error(A)) = (exactitud(B) - exactitud(A))/(1 - exactitud(A))$. Por ejemplo, si el algoritmo A tiene una exactitud del 98 % y el algoritmo B presenta una exactitud del 99 %, la reducción relativa del error será del 50 %. Aunque esta medida es a veces más apropiada que la diferencia absoluta de las exactitudes, mejorar del 80 % al 82 % puede ser más complicado que mejorar del 96 % al 98 % porque en el primer caso, la mejor predicción posible sea 82 % mientras que en el segundo puede ser del 100 %. Todos los conjuntos de datos artificiales usados en esta tesis son conceptos determinísticos, por lo que es posible llegar al 100 % de predicción.

Para los conjuntos de datos reales, el valor de la exactitud más alta no es conocido, pero probablemente no sea 100 % en la mayor parte de los dominios.

El poder computacional ha crecido hasta un punto en que los métodos de computación intensivos para estimar la exactitud son utilizados más a menudo y en conjuntos de datos más grandes. En los restantes apartados de esta sección se tratarán los siguientes aspectos:

- Las matrices de confusión, que permiten ver mediante una tabla la distribución de los errores cometidos.
- Los métodos estándares de validación, que permiten obtener la exactitud de los clasificadores.
- Las curvas ROC (*Receiver Operation Characteristic*), un procedimiento que permite evaluar la calidad de los clasificadores.

2.3.1. Matriz de confusión

Una matriz de confusión permite ver mediante una tabla de contingencia, la distribución de los errores cometidos por un clasificador a lo largo de las distintas categorías del problema. En dicha matriz se cruza la clase predicha por el clasificador con la clase real.

	Clasificado como		
Real	*Clase 0*	*Clase 1*	
Clase 0	a	b	π_0
Clase 1	c	d	π_1
	p_0	p_1	N

Tabla 2.2: Matriz de confusión genérica para dos clases

Una matriz de confusión, para el caso de dos clases, tiene la forma que se puede apreciar en la tabla 2.2. En la misma se tiene:

- π_0 denota la probabilidad a priori de la clase 0.
- π_1 denota la probabilidad a priori de la clase 1; $\pi_1 = 1 - \pi_0$.
- p_0 denota la proporción de casos que el clasificador predice con la clase 0.
- p_1 denota la proporción de casos que el clasificador predice con la clase 1; $p_1 = 1 - p_0$.
- Número de casos total $N = a + b + c + d$.

De una matriz de confusión también se pueden extraer los siguientes conceptos, enriquecedores a la hora de comprender la distribución y naturaleza de los errores cometidos por el clasificador:

- *Sensibilidad* $Se = a/(a + c)$ proporción de verdaderos positivos.
- *Especificidad* $Es = d/(b + d)$ proporción de verdaderos negativos.

- Proporción de *falsos positivos* $c/(a+c)$.
- Proporción de *falsos negativos* $b/(b+d)$.

Otros dos conceptos importantes en la validación son:

- *Ratio de error verdadero*: es la probabilidad de que un modelo construido clasifique incorrectamente nuevos casos no utilizados en su construcción. El objetivo de la validación es realizar una estimación, lo más realista posible, de dicho ratio de error verdadero.
- *Ratio de error aparente*: es el ratio de error obtenido por el clasificador a la hora de clasificar las instancias utilizadas en su construcción. Este ratio es demasiado optimista respecto a la realidad (o tasa de error verdadera), ya que las instancias utilizadas para inducir el modelo suelen adaptarse mejor a él que las instancias nuevas no utilizadas en su construcción, conociéndose este fenómeno como sobreentrenamiento u *overfitting*.

2.3.2. Métodos de validación

En este apartado se tratan los métodos de validación que, comparando el etiquetado real de una instancia con la clase predicha por el clasificador, permiten estimar la futura exactitud de un clasificador.

Excepto la estimación de la restitución, todos los estimadores no paramétricos que se explican están basados en la idea del remuestreo (*resampling*) (ver figura 2.1).

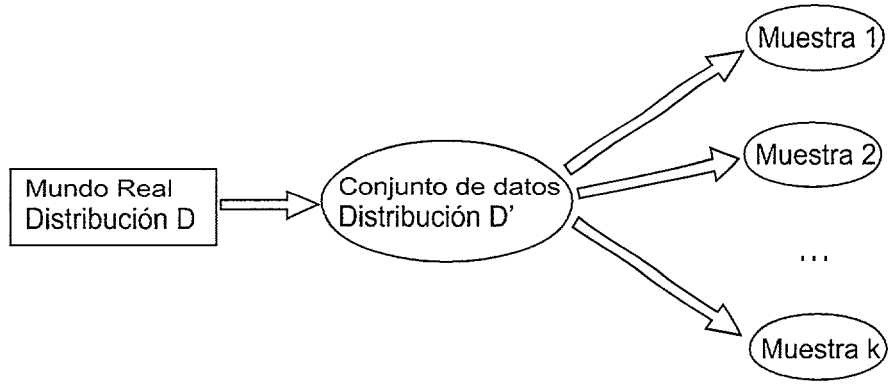


Figura 2.1: Las técnicas de estimación de la exactitud, tales como *hold-out*, *cross-validation* o *bootstrap*, están basadas en la idea del remuestreo

Estimación de la resustitución

Este método calcula la precisión aparente, ya que prueba el clasificador con los mismos datos usados por el algoritmo de inducción. Formalmente se puede expresar como,

$$acc_r = \frac{1}{N} \sum_{(\mathbf{x}^{(i)}, c^{(i)}) \in \mathcal{D}} \delta(\mathcal{I}(\mathcal{D}, \mathbf{x}^{(i)}), c^{(i)}) \tag{2.1}$$

donde $\delta(l, k) = 1$ si $l = k$ y 0 en otro caso.

La estimación de la restitución es una estimación de la precisión muy optimista porque los procedimientos de clasificación tienden a minimizarlo. Para muchos algoritmos de inducción que se ajustan perfectamente a los datos, tales como los árboles de decisión o los K vecinos más cercanos, este método es muy optimista, y, si no hay instancias conflictivas, la precisión estimada será del 100 %.

Holdout y remuestreo

El método *holdout* particiona los datos aleatoriamente en dos conjuntos mutuamente exclusivos, denominados conjunto de entrenamiento y conjunto de validación (o conjunto de *holdout*). El conjunto de entrenamiento es usado para inducir un modelo clasificatorio, utilizando el conjunto de validación para estimar la predicción verdadera. El conjunto de entrenamiento viene a ser habitualmente las dos terceras partes de todos los datos, utilizando el resto para el conjunto de validación. Formalmente se tiene $\mathcal{D}_h$ como el conjunto de validación, un subconjunto de $\mathcal{D}$ de tamaño N_h, y $\mathcal{D}_t$ definido por $\mathcal{D} \backslash \mathcal{D}_h$ como el conjunto de aprendizaje siendo $N_t = N - N_h$. La estimación de la distribución del método *holdout* viene definida como,

$$acc_h = \frac{1}{N_h} \sum_{(\mathbf{x}^{(i)}, c^{(i)}) \in \mathcal{D}_h} \delta(\mathcal{I}(\mathcal{D}_t, \mathbf{x}^{(i)}), c^{(i)}) \tag{2.2}$$

donde $\delta(l, k) = 1$ si $l = k$ y 0 en otro caso.

Asumiendo que la predicción del inductor se incrementa con el número de instancias utilizadas en el aprendizaje, el método *holdout* es una estimación pesimista, porque el algoritmo de inducción sólo utiliza una porción del conjunto de datos. Cuantas más instancias se dejen para el conjunto de validación, más pesimista será la predicción.

El método de remuestreo (*random subsampling*) viene a ser una generalización del método *holdout*, realizándose éste múltiples veces sobre diferentes particiones independientes del conjunto de entrenamiento y conjunto de validación. De esta forma, la exactitud se calcula a partir de la media de las exactitudes obtenidas en los diferentes experimentos.

Cross-validation, leave-one-out y stratification

El método de la validación cruzada (*cross-validation*), viene también a ser una generalización del *holdout*. El conjunto de datos $\mathcal{D}$ se divide aleatoriamente en k subconjuntos mutuamente excluyentes $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$ de aproximadamente el mismo tamaño. El clasificador es entrenado y validado k veces. Cada instante de tiempo $t \in \{1, 2, \ldots, k\}$ es entrenado en $\mathcal{D} \backslash \mathcal{D}_t$ y validado en $\mathcal{D}_t$. La estimación de la exactitud por medio del método *cross-validation* es el número total de bien clasificados, dividido entre el número total de instancias del conjunto de datos. Formalmente, la estimación de la predicción del *cross-validation* es,

$$acc_{cv} = \frac{1}{N}\sum_{t=1}^{k}\sum_{(\mathbf{x}^{(i)},c^{(i)})\in\mathcal{D}_t}\delta(\mathcal{I}(\mathcal{D}\backslash\mathcal{D}_t,\mathbf{x}^{(i)}),c^{(i)}). \tag{2.3}$$

La estimación *cross-validation* es un número aleatorio que depende de la división previamente realizada.

Un caso particular del *cross-validation* es el dejar-uno-fuera (**leave-one-out**), en el cual el parámetro k viene a ser igual al número de instancias N que existen para inducir el modelo final. De esta forma, los N subconjuntos de validación están formados por una única instancia y los de entrenamiento por los de la cardinalidad del conjunto total menos esa única instancia que ha sido llevada a la validación.

En **stratified cross-validation**, los subconjuntos son estratificados, de tal forma que contienen aproximadamente la misma proporción de casos respecto a las clases que el conjunto de datos original. Este método ha sido utilizado en múltiples ocasiones. De esta forma, la estratificación, especialmente para regresión, es el método idóneo para seleccionar el tamaño de árbol ideal [BFOS84] y es también la utilizada en los algoritmos CART y C4.5.

Bootstrapping

El método de *bootstraping* es quizás el de más difícil comprensión. En un conjunto de casos de cardinalidad N se escoge una muestra aleatoria con reemplazamiento del mismo tamaño que el conjunto de entrenamiento, dejando los casos no seleccionados como conjunto de validación. Ya que las instancias son escogidas aleatoriamente con reemplazamiento, la probabilidad de que una instancia cualquiera no sea escogida tras N muestras es $(1-\frac{1}{N})^N \approx e^{-1} \approx 0{,}368$, y el número esperado de instancias diferentes en la muestra se aproxima por medio de $0{,}632N$. De esta forma, denotando por α la tasa de error en el conjunto de validación y, si se realizaran b procesos de selección de N muestras aleatorias con reemplazamiento en cada una de ellas, la estimación de la tasa de error real del *bootstrapping* es,

$$acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0{,}632\cdot\alpha_i + 0{,}368\cdot acc_s), \tag{2.4}$$

donde acc_s es la precisión estimada del conjunto de datos calculada con la estimación de la resustitución.

2.3.3. Área bajo la curva ROC

Evaluación de clasificadores

En muchas ocasiones se tiene la necesidad de evaluar si un clasificador supervisado es mejor que otro. Una posible comparación que se puede realizar es que el clasificador que mejor tanto por ciento de bien clasificados tenga es el mejor clasificador.

Sin embargo, hay un enfoque más formal, basado en el cálculo del área bajo la curva

ROC del clasificador. Cuanto mayor sea el área bajo la curva ROC, mejor será el clasificador.

Las curvas ROC

En esta sección se definen las curvas ROC. La definición dada se centra en los casos en los que sólo hay dos posibles valores de clase a clasificar. Sin embargo, las curvas ROC son extensibles para cualquier número de clases.

El análisis ROC (*Receiver Operating Characteristic*) es una metodología desarrollada en el seno de la *Teoría de la Decisión* en los años 50, y cuya primera aplicación fue motivada por problemas prácticos en la detección de señales por radar. Con el tiempo se comenzó a utilizar en el área de la biomedicina [SP82], inicialmente en radiología.

Para centrar ideas se escoge, por ejemplo, un conjunto de datos en el que cada individuo de una población se clasifica como demócrata o republicano. Se supone que la variable de decisión, que representa el resultado de la clasificación realizada, se distribuye normalmente, con media y desviación típica conocidas. En la figura 2.2 se pueden ver las funciones de densidad para ambas clases, las cuales muestran un determinado nivel de solapamiento. Si consideramos un valor arbitrario del resultado de la clasificación, x –en adelante valor de corte–, la *sensibilidad* y $1 - especificidad$ se corresponderán respectivamente con el área a la derecha de ese punto bajo la función de densidad de la población demócrata (áreas clara y oscura) y de la población republicana (área oscura). La curva ROC se obtiene representando, para cada posible elección del valor de corte, la $1 - especificidad$ en el eje x y la *sensibilidad* en el eje y.

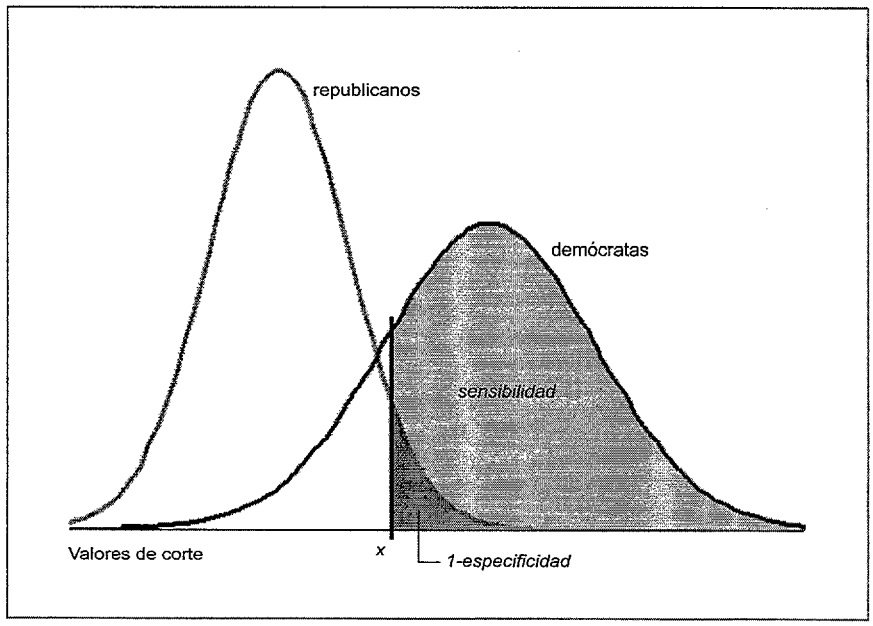


Figura 2.2: Funciones de densidad de probabilidad para las poblaciones de demócratas y de republicanos

Mediante esta representación de los pares (1-especificidad, sensibilidad) obtenidos al considerar todos los posibles valores de corte de la prueba, la curva ROC proporciona una representación global de la exactitud (véase figura 2.3). La curva ROC es necesariamente creciente, propiedad que refleja el compromiso existente entre la sensibilidad y la especificidad: el modificar el valor de corte para obtener mayor sensibilidad, sólo se puede hacer a expensas de disminuir al mismo tiempo la especificidad. Si la prueba no permitiera discriminar entre clases, la curva ROC sería la diagonal que une los vértices inferior izquierdo y superior derecho. La exactitud de la prueba aumenta a medida que la curva se desplaza desde la diagonal hacia el vértice superior izquierdo. Si la discriminación fuera perfecta (100 % de sensibilidad y 100 % de especificidad) la curva pasaría por dicho punto.

Obviamente, el escenario en que se ha presentado la curva ROC es completamente teórico, por dos razones relacionadas entre sí:

- En la práctica no se dispone de la población de gente demócrata y republicana, sino simplemente de una muestra de ella.
- En general, no se conocen las distribuciones de los valores de la clasificación en dicha población.

Estas limitaciones obligan a considerar el problema práctico de la construcción de curvas ROC, que se trata a continuación, desde un punto de vista meramente estadístico.

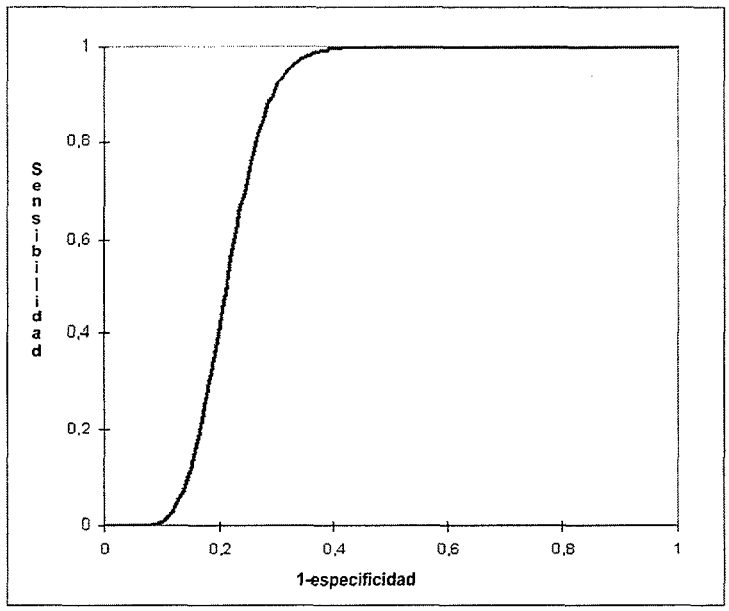


Figura 2.3: Ejemplo de curva ROC

Métodos de cálculo de las curvas ROC

Un primer grupo de métodos para construir la curva ROC lo constituyen los llamados métodos no paramétricos. Se caracterizan por no hacer ninguna suposición sobre la

distribución de los resultados del clasificador. El más simple de estos métodos es el que suele conocerse como empírico, que consiste simplemente en representar todos los pares $(1 - especificidad, sensibilidad)$ para todos los posibles valores de corte que se puedan considerar con la muestra particular de que se disponga. Desde un punto de vista técnico, este método sustituye las funciones de densidad teóricas por una estimación no paramétrica de ellas, es decir, la función de densidad empírica construida a partir de los datos. Informalmente, es como si en la figura 2.3 se sustituyeran las funciones de densidad por histogramas obtenidos a partir de las clasificaciones realizadas y se construyera la curva ROC a partir de ellos.

La representación gráfica obtenida por este método tiene una forma escalonada. En efecto, para cada variación mínima del valor de corte que produzca cambios en la sensibilidad o en la especificidad, al menos un caso pasa a ser considerado bien como verdadero positivo, lo que se corresponde con un trazo vertical, bien como falso positivo, lo que da lugar a un trazo horizontal. Existe aún otra posibilidad, derivada de que se produzcan empates, es decir, dos o más casos con el mismo valor de la prueba: si el empate ocurre entre un caso del grupo demócrata y otro del grupo republicano aparecerá un trazo diagonal en la representación.

Existen otros métodos no paramétricos [ZHS97] que permiten obtener curvas ROC suavizadas, en contraposición con la forma dentada de la curva obtenida por el método empírico. La idea es básicamente obtener estimaciones no paramétricas suavizadas de las funciones de densidad de las dos distribuciones de resultados de la clasificación, empleando, generalmente, estimadores de tipo núcleo. A partir de dichas densidades –en lugar de a partir de los histogramas, como en el método anterior– se obtiene directamente una curva ROC que habrá sido suavizada.

Análisis estadístico de las curvas ROC: área bajo la curva

Como se comentó en la anterior sección, la mayor exactitud diagnóstica de una prueba se traduce en un desplazamiento hacia arriba y a la izquierda de la curva ROC. Esto sugiere que el área bajo la curva ROC se puede emplear como un índice conveniente de la exactitud global de la prueba: la exactitud máxima correspondería a un valor del área bajo la curva de 1 y la mínima a un valor de 0.5.

Cuando la curva ROC se genera por el método empírico, independientemente de que haya empates o no, el área puede aproximarse mediante la regla trapezoidal, es decir, como la suma de las áreas de todos los rectángulos y trapecios (correspondientes a los empates) que se forman bajo la curva.

Capítulo 3

Clasificación supervisada con modelos gráficos probabilísticos

En este capítulo se analiza el estado del arte de la clasificación supervisada con modelos gráficos probabilísticos. El índice del capítulo es el siguiente:

- En la sección 3.1 se realiza una introducción a las redes Bayesianas, el modelo gráfico probabilístico que será utilizado a lo largo de toda la tesis.
- En la sección 3.2 se describe en profundidad el clasificador naïve-Bayes, tanto con variables discretas como con variables continuas y se detalla cómo estimar sus probabilidades a priori utilizando la corrección de Laplace.
- En la sección 3.3 se analizan diferentes métodos de discretización de variables continuas.
- En la sección 3.4 se describen diferentes formas de tratar los datos perdidos que puedan existir dentro del conjunto de datos.
- En la sección 3.5 se enumeran, clasifican y describen los principales enfoques semi naïve-Bayes.
- En la sección 3.6 se analizan los clasificadores del tipo naïve-Bayes aumentado.
- Por último, en la sección 3.7 se encuentra descrito el enfoque basado en el manto de Markov de la variable a clasificar.

3.1. Redes Bayesianas

Utilizando un lenguaje coloquial, las redes Bayesianas (RRBB) son una representación gráfica para manejar incertidumbre en sistemas expertos. Dentro de este campo se tienen dos divisiones en cuanto a la forma de construir el sistema: el enfoque tradicional y el enfoque basado en el aprendizaje. En el enfoque tradicional, la determinación de la topología o estructura de la red y de los parámetros asociados con dicha topología se propone por el experto humano. En el enfoque basado en el aprendizaje, tanto la topología como los

parámetros se determinan a partir de una muestra de datos estadísticos, sin la intervención directa del experto humano (en la mayoría de los casos). A fecha de hoy, la búsqueda de esquemas de aprendizaje en redes probabilistas es un campo de investigación abierto y sumamente activo, y en esta línea de trabajo es donde se mueve esta tesis.

3.1.1. Fuentes de incertidumbre

Una de las propiedades esenciales de los sistemas expertos (SSEE), y a la vez una de las más complejas, es el tratamiento de la incertidumbre. En esta sección se enumeran y clasifican las fuentes de incertidumbre habituales, tomando como ejemplo el campo de la medicina, con el fin de mostrar la importancia del tema.

Observando la historia de los SSEE, y en particular de los métodos de razonamiento incierto, se comprueba que casi todos los primeros sistemas desarrollados (cronológicamente) y muchos de los más importantes, lo han ido en el campo de la medicina. Si se trata de averiguar la causa, se descubre que éste es un campo donde se dan todos los tipos de incertidumbre. Veamos algunos ejemplos:

- **Información incompleta**. En muchos casos la historia clínica completa no está disponible, y el paciente es incapaz de recordar todos los síntomas que ha experimentado y cómo se ha desarrollado la enfermedad.
- **Información errónea**. En cuanto a la información suministrada por el paciente, puede que éste describa incorrectamente sus síntomas e incluso que trate de mentir deliberadamente al médico. También es posible que el diagnóstico anterior, contenido en la historia clínica, haya sido erróneo. Por último, tampoco es extraño que las pruebas de laboratorio den falsos positivos y falsos negativos.
- **Información imprecisa**. Hay muchos datos en medicina que son difícilmente cuantificables. Tal es el caso, por ejemplo, de los síntomas como el dolor o la fatiga.
- **Mundo real no determinista**. Muchas veces las mismas causas producen efectos diferentes en distintas personas, sin que haya ninguna explicación aparente. Por ello, el diagnóstico médico debe estar siempre abierto a admitir la aleatoriedad y las excepciones.
- **Modelo incompleto**. Por un lado, hay muchos fenómenos médicos cuya causa aún se desconoce. Por otro, es frecuente la falta de acuerdo entre los expertos de un mismo campo.
- **Modelo inexacto**. Por último, todo modelo que trate de cuantificar la incertidumbre, por cualquiera de los métodos que existen, necesita incluir un elevado número de parámetros. Por ejemplo, en el caso de las RRBB, necesitamos especificar todas las probabilidades a priori y condicionales. Sin embargo, una gran parte de esta información no suele estar disponible, por lo que debe ser estimada de forma subjetiva.

Se ha escogido el campo de la medicina como ejemplo paradigmático de dominio incierto, aunque todas estas fuentes de incertidumbre pueden darse, y de hecho se dan, en cualquier otro campo de las ciencias naturales, la ingeniería, el derecho, las humanidades, etc.

En resumen, el tratamiento de la incertidumbre es, junto con la representación del conocimiento y el aprendizaje, uno de los problemas fundamentales de la inteligencia artificial. Por ello no es extraño que casi desde los orígenes de este campo se le haya prestado tanta atención y hayan surgido tantos métodos.

3.1.2. Tratamiento de la incertidumbre

Los métodos de razonamiento incierto se clasifican en dos grandes grupos: métodos numéricos y métodos cualitativos. Cuando el razonamiento incierto se realiza mediante métodos numéricos suele hablarse de razonamiento aproximado (aunque tampoco es una cuestión en la que haya acuerdo unánime, pues algunos autores, al hablar de "razonamiento aproximado", piensan sobre todo en la lógica difusa y en modelos afines).

Entre los **métodos cualitativos** para el tratamiento de la incertidumbre, destacan los basados en lógicas no monótonas, tales como los modelos de razonamiento por defecto (el más conocido es el de Reiter [Rei80]), los sistemas de suposiciones razonadas (originalmente llamados *"truth maintenance systems"*, aunque sería más correcto denominarlos *"reason maintenance systems")* de Doyle [Doy79] y la teoría de justificaciones (*"theory of endorsements")* de Cohen y Grinberg [CG83]. Estos métodos consisten en que, cuando no hay información suficiente, se hacen suposiciones, que posteriormente podrán ser corregidas al recibir nueva información. El problema principal que presentan se debe a su naturaleza cualitativa, por lo que no pueden considerar los distintos grados de certeza o incertidumbre de las hipótesis. Además, suelen presentar problemas de explosión combinatoria. En consecuencia, se estudian más por su importancia teórica (fundamentación de la IA) que por las aplicaciones prácticas a que puedan dar lugar.

En cuanto a los **métodos numéricos**, que son los que se estudian en este trabajo, destacan los siguientes:

- **Método probabilista clásico**. Ya en el siglo XVIII, Bayes y Laplace propusieron la probabilidad como una medida de la creencia personal. A principios del siglo XX surgen las interpretaciones de la probabilidad como la frecuencia (a largo plazo) asociada a situaciones o experimentos repetibles; en esta línea, destacan especialmente los trabajos estadísticos de Fisher. A principios de los años 30, en cambio, debido sobre todo a los trabajos de L.J. Savage y B. de Finetti, entre otros muchos, se redescubre la probabilidad como medida de la creencia personal. Unos años más tarde, se inventan los computadores y poco después surge la inteligencia artificial –IA– (suele tomarse como punto de referencia el año 1956, en el que se celebró la *Conferencia de Darmouth*).

- **Modelo de factores de certeza**. El éxito obtenido por el DENDRAL, considerado por muchos como el primer sistema experto, mostró las grandes ventajas de la programación mediante reglas. Por ello, los creadores de MYCIN buscaron un método de computación eficiente que pudiera adaptarse al razonamiento mediante encadenamiento de reglas. La incapacidad de los métodos probabilistas para encajar en este esquema llevaron a los responsables del proyecto a desarrollar un método propio, consistente en asignar a cada regla un factor de certeza.

- **Redes Bayesianas**. A principios de los años 80 ocurrió un acontecimiento que cambió completamente el escenario: la aparición de las RRBB, un modelo probabilista

inspirado en la causalidad, cuya virtud principal consiste en que lleva asociado un modelo gráfico en que cada nodo representa una variable y cada enlace representa, generalmente, un mecanismo causal. El extraordinario desarrollo experimentado por las RRBB en las dos últimas décadas ha permitido construir modelos de diagnóstico y algoritmos eficientes para problemas de tamaño considerable, a veces con cientos de variables, o incluso con miles de variables en algunos problemas de genética. Por ello, algunos de los antiguos detractores del uso de la probabilidad en IA son hoy en día defensores entusiastas de los modelos gráficos probabilistas.

- **Lógica difusa.** En paralelo con esta evolución histórica de crisis y resurgimiento de la probabilidad, se desarrolló la teoría de los conjuntos difusos, frecuentemente llamada, con cierta impropiedad, **lógica difusa**. La motivación inicial no fue el estudio de la incertidumbre, sino el estudio de la vaguedad, que es algo diferente. Por ejemplo, si sabemos que Juan mide 1,78 m., no se puede decir con rotundidad que es alto, pero tampoco se puede decir que no lo es. Se trata de una cuestión de grado; en este caso hay vaguedad intrínseca, pero no hay incertidumbre, con lo que se demuestra que son dos conceptos en principio independientes, aunque existe una cierta relación en el sentido de que si recibimos una información imprecisa (por ejemplo, si nos dicen que Juan es alto, pero sin decirnos su estatura exacta) tenemos una cierta incertidumbre.

Los cuatro métodos que se acaban de comentar, **método probabilista clásico**, **modelo de factores de certeza**, **RRBB** y **lógica difusa**, corresponden cuatro métodos utilizados para representar la incertidumbre. Es importante señalar que, mientras las RRBB y la lógica difusa son temas de gran actualidad, como lo prueba la intensa labor investigadora que se está realizando en cada uno de ellos, el método probabilista clásico y el modelo de factores de certeza se consideran temas "muertos" desde el punto de vista de la investigación, por razones diversas.

Como conclusión, señalar que el debate sobre cuál es el método más adecuado para representar la incertidumbre sigue abierto hoy en día. Por un lado, está el grupo de los bayesianos, que defienden que la teoría de la probabilidad es el único método correcto para el tratamiento de la incertidumbre. Por otro, están quienes señalan que los modelos probabilistas, a pesar de sus cualidades, resultan insuficientes o inaplicables en muchos problemas del mundo real, por lo que conviene disponer de métodos alternativos.

3.1.3. Thomas Bayes

Las redes Bayesianas (RRBB) deben su nombre al inglés Thomas Bayes (Londres, 1702–1761). Su padre fue uno de los primeros seis ministros presbiterianos que fueron ordenados en Inglaterra. La educación de Thomas fue privada, un hecho que se antoja necesario para el hijo de un ministro presbiteriano de aquellos tiempos. Parece ser que de Moivre fue su maestro particular, pues se sabe que por aquel entonces ejercía como profesor en Londres. Bayes fue ordenado ministro presbiteriano y asistió a su padre en Holborn. Al final de la década iniciada en 1720 fue nombrado pastor en Turnbridge Wells (Kent, Inglaterra). Aunque trató de retirarse de su puesto eclesiástico en 1749, permaneció en él hasta 1752. Una vez retirado siguió viviendo en Turnbridge Wells hasta el día de su muerte, el 17 de abril de 1761. Sus restos descansan en el cementerio londinense de Bunhill

Fields. La traducción de la inscripción en su tumba puede leerse como sigue: *"Reverendo Thomas Bayes. Hijo de los conocidos Joshua y Ann Bayes. 17 de abril de 1761"*. En reconocimiento al importante trabajo que realizó Thomas Bayes en el desarrollo de la teoría de la probabilidad, su tumba fue restaurada en 1969 con donativos de estadísticos de todo el mundo.

Teólogo, matemático y miembro de la *Royal Society* desde 1742, Bayes fue el primero en utilizar la probabilidad inductivamente y establecer una base matemática para la inferencia probabilística. Los únicos trabajos que se sabe que Thomas Bayes publicó en vida son: *Divine Providence and Government is the Happiness of His Creatures* (1731) y *An Introduction to the Doctrine of Fluxions, and a Defence of The Analyst* (1736), que fueron blanco de críticas por parte del obispo Berkeley, quien sustentaba sus ideas en los fundamentos lógicos del cálculo de Newton. En 1763 se publicó póstumamente *Essay Towards Solving a Problem in the Doctrine of Chances*, donde el reverendo Bayes abordó el problema de las causas a través de los efectos observados, y donde se enuncia el teorema que lleva su nombre. Este trabajo fue entregado a la Royal Society por Richard Price y resulta ser la base para la técnica estadística conocida como estadística bayesiana, que se utiliza para calcular la probabilidad de la validez de un enunciado tomando como bases la estimación de la probabilidad previa y las evidencias relevantes más recientes.

3.1.4. El teorema de Bayes

De entre los descubrimientos que Thomas Bayes realizó, uno de los más importantes es el llamado teorema de Bayes, que es en el que se basan las RRBB. Antes de profundizar en el teorema conviene introducir algunos conceptos importantes:

Probabilidad condicional

La probabilidad de que el enunciado B sea verdadero, condicionado a que el enunciado A sea verdadero, se denota por $p(B \mid A)$, y se define, en caso de que $p(A) > 0$, como:

$$p(B \mid A) = \frac{p(A \wedge B)}{p(A)} \tag{3.1}$$

La probabilidad condicional se interpreta como una implicación especial, que es lo característico del razonamiento probabilístico, entendiéndose que "si A es verdadero, entonces B tiene una probabilidad $p(B \mid A)$ de ser verdadero".

A partir de la denominada *regla de la multiplicación* se puede expresar la probabilidad correspondiente a la intersección de un número finito de enunciados, por medio del producto de probabilidades condicionadas, de la manera siguiente:

$$p(A_1 \wedge A_2 \wedge A_3 \wedge \cdots \wedge A_n) = p(A_n) \cdot \prod_{j=1}^{(n-1)} p(A_j \mid A_{j+1} \wedge \cdots \wedge A_n) \tag{3.2}$$

Fórmula de Bayes

La definición de probabilidad condicionada anteriormente expuesta tiene dos versiones en función de cual sea el enunciado que condiciona. De esta forma, se tiene que, si A y B son dos enunciados tales que $p(A) > 0$ y $p(B) > 0$, se verifica que:

$$p(B \mid A) = \frac{p(A \wedge B)}{p(A)} \tag{3.3}$$

$$p(A \mid B) = \frac{p(A \wedge B)}{p(B)} \tag{3.4}$$

De las anteriores igualdades se obtiene que:

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B)} \tag{3.5}$$

Además, si aplicamos el denominado teorema de la probabilidad total al enunciado B, se tendría:

$$p(B) = p(B \mid A) \cdot p(A) + p(B \mid \neg A) \cdot p(\neg A) \tag{3.6}$$

obteniendo de las dos últimas fórmulas la denominada fórmula de Bayes:

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B \mid A) \cdot p(A) + p(B \mid \neg A) \cdot p(\neg A)} \tag{3.7}$$

Razonamiento probabilístico

Mediante el teorema enunciado es posible realizar lo que se ha dado en llamar razonamiento probabilístico, algunas de cuyas características son las siguientes:

- El sistema deberá ser capaz de manejar desigualdades a la hora de realizar razonamientos encadenados.
  Si suponemos que tratamos de determinar el valor del enunciado B, una vez conocidos los valores asignados a $p(A)$ y $p(B \mid A)$, a partir del teorema de la probabilidad total, se tiene que:

$$p(B) = p(A) \cdot p(B \mid A) + p(\neg A) \cdot p(B \mid \neg A) \tag{3.8}$$

  Al no conocerse el valor de $p(B \mid \neg A)$, de la fórmula anterior se obtiene que:

$$p(B) \geq p(A) \cdot p(B \mid A) \tag{3.9}$$

  Análogamente se deduce que:

$$p(\neg B) \geq p(A) \cdot p(\neg B \mid A) \tag{3.10}$$

  De las dos desigualdades anteriores se obtiene:

$$p(A) \cdot p(B \mid A) \leq p(B) \geq 1 - (p(A) \cdot (1 - p(B \mid A))) \tag{3.11}$$

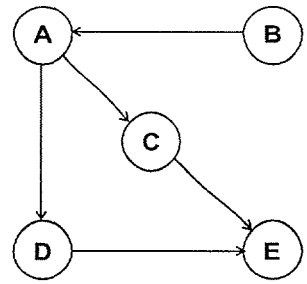


Figura 3.1: Grafo Dirigido

- Posibilidad de utilizar otros esquemas de razonamiento que denominaremos *plausibles*, no aceptables en la lógica bivalente.
  Ejemplos de lo anterior son los siguientes esquemas:
  - Si se sabe que $p(B \mid A) = 1$, y A es falso, entonces parece que B tiene que ser menos creíble.
  - Si se sabe que $p(B \mid A) = 1$, y B es verdadero, entonces parece que A tiene que ser más creíble.
- Carácter *no monótono*. Es decir, a medida que obtenemos nueva información (hechos) sobre el problema, los valores de incertidumbre de los enunciados varían.
- A partir de la fórmula de Bayes, se deduce que la implicación probabilística es *bidireccional*.

3.1.5. Estructura de la red. Teoría de grafos

La estructura de las RRBB viene representada por un grafo. Se presentan algunas nociones básicas sobre teoría de grafos que pueden ser de utilidad para comprenderlas mejor:

- Un grafo G viene definido por un par (N, A), en donde N representa los nodos que lo forman y A es el conjunto de aristas que hay entre los mismos.
- Si las aristas son dirigidas, se denominan arcos dirigidos, o simplemente arcos.
- Un grafo dirigido es un par (N, A) en donde todas las aristas son arcos dirigidos.
- Un nodo X es padre de un nodo Y si existe un arco que va de X a Y, $X \rightarrow Y$. Se dice también que Y es hijo de X.
- Se denomina familia del nodo X al conjunto formado por X y los padres $pa(X)$ de éste.
- Un camino es una sucesión de nodos $\{X_1, \ldots, X_L\}$, pertenecientes a un grafo (N, A), de modo que $X_i \neq X_j$ para $1 < i < j < L$ y
  $(X_i, X_{i+1}) \in A$ ó $(X_{i+1}, X_i) \in A, \forall i = 1, \ldots, L-1$

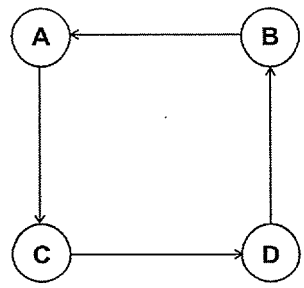


Figura 3.2: Grafo con un ciclo de longitud 4

- Un ciclo es una sucesión de nodos $\{X_1, \ldots, X_L\}$, pertenecientes al grafo (N, A), tal que $X_i \neq X_j$ para $1 < i < j < L$, y $\forall i < L-1, \exists (X_i, X_i + 1) \in A$ y existe el arco $(X_L, X_1) \in A$ que cierra el ciclo.

La figura 3.1 muestra un ejemplo de un grafo dirigido. En la figura 3.2 se muestra un ejemplo de un ciclo de longitud 4.

A continuación, se muestran los conceptos básicos del tipo de grafos más importante para la teoría de las RRBB: los grafos acíclicos dirigidos.

- Sea $G = (N, A)$ un grafo (dirigido o no dirigido). Se dice que G es acíclico si G no contiene ciclos.
- Un grafo $G = (N, A)$ es un grafo acíclico dirigido (DAG) si G es acíclico y dirigido.
- Sea $G = (N, A)$ un grafo dirigido. Sean U y V vértices en N. Se dice que U es un padre de V, o bien que V es un hijo de U, si y sólo si $(U, V) \in A$. Se dice que U es un ancestro de V, o bien que V es un descendiente de U, si existe un camino de U a V. Un vértice sin padres se denomina vértice raíz.

  La figura 3.3 ilustra los conceptos anteriores.

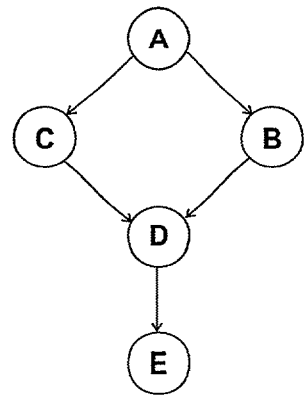


Figura 3.3: DAG. B y C son padres de D. A es un vértice raíz

- Un DAG, $G = (N, A)$, está simplemente conectado si dados dos vértices cualesquiera, $U, V \in N$, existe como máximo un camino entre U y V. En caso contrario se dice que el DAG está múltiplemente conectado. En el contexto de las RRBB, a los grafos acíclicos simplemente conectados se les denomina habitualmente poliárboles.
- Un DAG, $G = (N, A)$, se denomina un bosque, si cada vértice $V \in N$ tiene como máximo un padre.
- Un bosque se denomina árbol si exactamente un vértice no tiene padres. Dicho vértice es la raíz del árbol.

  En la figura 3.4 se muestran diferentes tipos de estructuras DAG.

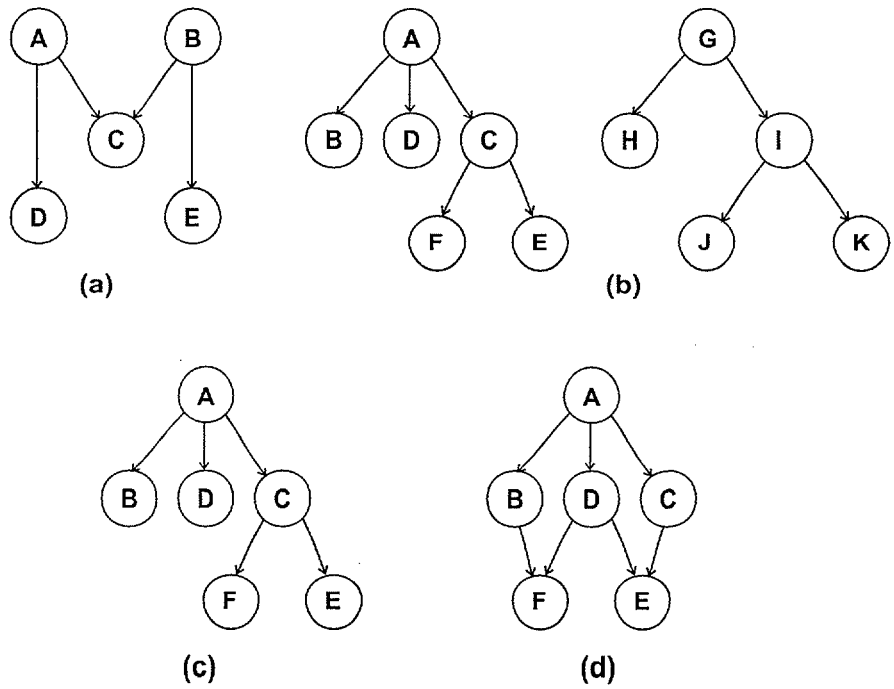


Figura 3.4: El DAG en (a) está simplemente conectado, el de (b) es un bosque, el de (c) es un árbol y el de (d) está múltiplemente conectado

Se define en este punto un aspecto fundamental de las RRBB: el concepto de separación entre los nodos de la red. Mediante el criterio de separación conocido como *D-separation* (Pearl, 1988) se puede indicar si un grafo verifica o no una relación de independencia dada. Presentamos formalmente el concepto, junto con las definiciones previas necesarias para ello.

- Dados un grafo dirigido y un camino no dirigido $(\cdots - U - A - V - \ldots)$, el nodo A se denomina *nodo de aristas convergentes en un camino* si las dos aristas del camino convergen a este nodo en el grafo dirigido, es decir, si el grafo dirigido contiene las aristas $U \rightarrow A$ y $V \rightarrow A$.

- Sean X, Y y Z tres subconjuntos disjuntos de nodos en un grafo acíclico dirigido D. Se dice que Z D-separa X e Y si y sólo si a lo largo del camino no dirigido entre cualquier nodo de X y cualquier nodo de Y existe un nodo intermedio A tal que,

  1. A es un nodo de aristas convergentes en el camino y ni A ni sus descendientes están en Z, o bien
  2. A no es un nodo de aristas convergentes en el camino y A está en Z.

Cuando Z D-separa X e Y en G, se escribe $I(X, Y \mid Z)_G$, para indicar que la relación de independencia viene dada por el grafo G; en caso contrario, se escribe $D(X, Y \mid Z)_G$, para indicar que X e Y son condicionalmente dependientes dado Z en el grafo G.

3.1.6. Estimación de los parámetros de una red Bayesiana

Las RRBB tienen dos partes fundamentales: la estructura y los valores de probabilidad asociados a los nodos. A menudo se estiman los valores de probabilidad de una base de datos utilizando la frecuencia relativa, es decir, el estimador máximo de verosimilitud de probabilidades. A pesar de que esta estimación aparenta ser una buena aproximación hay que tener en cuenta que, para problemas reales, suele suceder que los casos de la base de datos no abarcan todas las posibilidades de combinaciones entre valores de variables, con lo que este tipo de estimación puede llevar a una estimación de parámetros con abundancia de ceros, o, lo que es peor, a situaciones en las que la estimación de la frecuencia indica que hay cero casos de cero posibles. Otro problema de este tipo de estimación es el sobreajuste, ya que la tendencia a estimar probabilidades nulas acarrea una tendencia a estimar unos, lo cual puede hacer que la red Bayesiana no se comporte de forma adecuada para datos no pertenecientes a la BBDD. Existen varios métodos que intentan solucionar estos problemas. Una de ellas, denominada *ley de la sucesión de Laplace* (Good, 1965), en lugar de estimar la probabilidad directamente como $\frac{CasosFavorables}{CasosTotales}$, utiliza como estimación inicial el número que se obtiene al dividir $\frac{N_{ijk}+1}{N_{ij}+r_i}$, donde N_{ijk} es el número de casos favorables[1], N_{ij} es el número de casos posibles[2] y r_i es el número de valores posibles de la variable i-ésima.

Para especificar la distribución de probabilidad de una red Bayesiana se debe proporcionar la distribución de probabilidad a priori de todos los vértices raíz (vértices sin precedentes), así como las probabilidades condicionadas de todos los vértices no raíz, para cada posible combinación de sus padres o predecesores directos. Estos números, en conjunción con el DAG especifican totalmente la red Bayesiana. La probabilidad conjunta de cualquier punto n dimensional $(X_1, \ldots, X_n)$ puede calcularse como:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid pa(x_i)) \tag{3.12}$$

donde x_i representa el valor de la variable X_i y $pa(x_i)$ representa el valor de los padres de X_i.

[1] N_{ijk} representa el número de veces que la variable i toma el valor k de entre aquellos casos en los que sus variables padres toman su j-ésima combinación.

[2] N_{ij} representa el número de veces que las variables de la i-ésima variable toman su j-ésima combinación.

3.1.7. Definición formal de red Bayesiana

Hay varias maneras de dar una definición formal de red Bayesiana. Una definición que encaja con los apartados anteriores es la que se da en (Neapolitan, 1990):

Sea n un conjunto finito de variables discretas definidas en el mismo espacio probabilístico, sea P su distribución de probabilidad conjunta y sea $G = (V, A)$ un DAG. Para cada $X \in V$ sea $pa(X) \subseteq V$ el conjunto de todos los padres de X y sea $d(X) \subseteq V$ el conjunto de todos los descendientes de X. Además, para cada $X \in V$ se define $a(X) \subseteq V$ como $a(X) = V \setminus (d(X) \cup X)$, es decir, el conjunto de variables en V, excluyendo X y sus descendientes. Si para cada subconjunto $W \subseteq a(X)$, W y X son condicionalmente independientes dado $pa(X)$, es decir, si se cumple alguna de las dos condiciones siguientes:

1. $P(X \mid pa(X)) = 0$ *ó* $P(W \mid pa(X)) = 0$
2. $P(X \mid W \wedge pa(X)) = P(X \mid pa(X))$

entonces diremos que $C = (V, A, P)$ es una red Bayesiana.

Según la definición anterior, dado un grafo acíclico dirigido y una distribución de probabilidad sobre sus variables, se dice que hay separación direccional si, dado un vértice X cualquiera, el conjunto de sus padres, $pa(X)$, separa condicionalmente este vértice de cualquier otra variable Y en que no haya descendientes de X. Es decir, si se verifica que:

$$P(X \mid pa(X), Y) = P(X \mid pa(X)).$$

Se puede, por tanto, definir una red Bayesiana como un grafo acíclico dirigido más una distribución de probabilidad sobre sus variables, que cumple la propiedad de separación direccional. En la figura 3.5 se presenta un ejemplo de una estructura de red Bayesiana múltiplemente conectada.

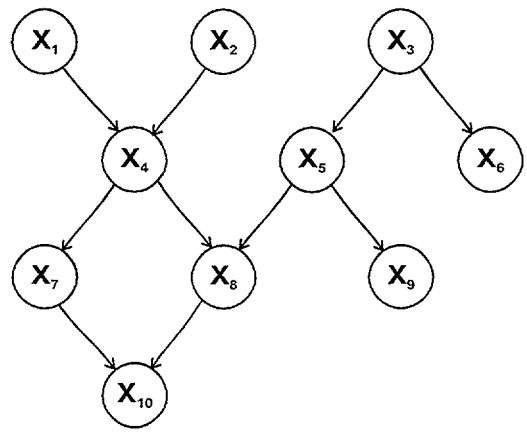


Figura 3.5: Red Bayesiana múltiplemente conectada

3.2. Naïve-Bayes

El modelo gráfico probabilístico para clasificación supervisada más ampliamente utilizado es conocido como naïve-Bayes [DH73, HY01], y está basado en la aplicación del teorema de Bayes. Naïve-Bayes se basa en un modelo de independencia condicional de los atributos predictores dada la clase, a pesar de lo cual garantiza una clasificación óptima si se cumplen un conjunto de suposiciones explícitas [DP96]. En la figura 3.6 se puede ver una representación gráfica de este modelo.

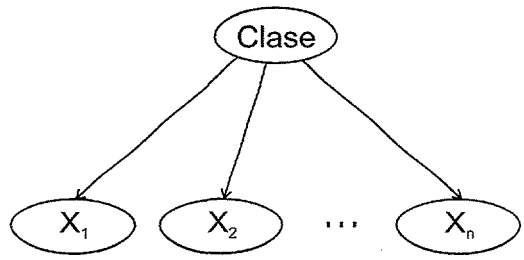


Figura 3.6: El clasificador naïve-Bayes

En el momento de clasificar una nueva instancia I, se basa en el teorema de Bayes para calcular la probabilidad a posteriori con la que la instancia puede pertenecer a cada una de las clases del problema,

$$P(c_i|X_1 = x_1, \ldots, X_n = x_n) = \frac{P(c_i)P(X_1 = x_1, \ldots, X_n = x_n|c_i)}{P(X_1 = x_1, \ldots, X_n = x_n)} \tag{3.13}$$

donde $P(c_i)$ es la probabilidad a priori de la clase c_i en el conjunto de entrenamiento. De todas formas, sabiendo que la instancia I es una conjunción de los n valores de sus atributos descriptivos $X_1 = x_1, \ldots, X_n = x_n$ y que $P(I)$ es igual para todas las clases, se puede restringir la regla anteriormente expuesta, quedando de la siguiente forma,

$$P(C = c_i|X_1 = x_1, \ldots, X_n = x_n) \propto P(C = c_i)P(X_1 = x_1, \ldots, X_n = x_n|C = c_i) \tag{3.14}$$

donde $P(X_1 = x_1, \ldots, X_n = x_n|C = c_i)$ supone la probabilidad de que ocurra la instancia I cuando la clase es c_i. Estas probabilidades se pueden estimar mediante las frecuencias del conjunto de entrenamiento. Sin embargo, esta expresión no es operacional, ya que el término $P(X_1 = x_1, \ldots, X_n = x_n|C = c_i)$ será habitualmente cero, debido al número de parámetros tan elevado –del orden de 2^n para variables predictoras dicotómicas– que se debe estimar para cada valor de la variable C. Para hacer operativa la expresión anterior se debe tener en cuenta la suposición de que los atributos que definen cada instancia son independientes entre sí dada la clase del problema. De esta forma se puede utilizar la siguiente expresión,

$$P(X_1 = x_1, \ldots, X_n = x_n|C = c_i) = \prod_{k=1}^{n} P(X_k = x_k|C = c_i) \tag{3.15}$$

donde las probabilidades a priori y condicionales del valor de cada atributo X_k dada la clase c_i se estiman a través de las frecuencias del conjunto de entrenamiento. Utilizando la última igualdad se calculará la probabilidad a posteriori de cada clase dada la instancia, clasificándose la misma con la clase que computa la mayor de estas probabilidades.

A pesar de su simplicidad y la asunción de la independencia entre las variables en la que se basa (la cual no es cierta para la mayoría de los problemas reales), ha demostrado su eficacia en numerosos problemas de muy distinta naturaleza [HT01]. Dentro del campo de la medicina se pueden citar varios ejemplos como los estudios de datos de lesiones cerebrales [TMS+81], los clasificadores para la predicción del cáncer de mama recurrente [MPW97] o los estudios de las enfermedades del corazón [RKF83]. Fuera del ámbito de la medicina también se encuentran estudios que demuestran que es muy eficiente, a menudo mucho mejor que otras alternativas. Dentro de estos estudios se pueden resaltar los siguientes [CKB87] [CN89] [Ces90] [LIT92] [PMB96] [FGG97] y [DP97].

El clasificador naïve-Bayes puede ser utilizado tanto con atributos discretos como con atributos continuos. Sin embargo, se ha demostrado [DKS95] que la correcta discretización de los valores de los atributos continuos puede llevar a una mejora significativa de su exactitud. En la sección 3.2.2 se realiza una introducción a naïve-Bayes con atributos continuos, mientras que en la sección 3.3 se examinan los distintos tipos de discretización que se pueden llevar a cabo antes de aplicar naïve-Bayes.

Otro aspecto importante que se debe tener en cuenta es cómo tratar los valores ausentes (*missing values*) [KBS97], es decir, qué hacer cuando el valor de algún atributo de alguna de las instancias del conjunto de entrenamiento es desconocido. En la sección 3.4 se detalla un estudio realizado sobre este tema.

Un último aspecto a tener en cuenta cuando se aplica naïve-Bayes es la forma de estimar las probabilidades condicionales del conjunto de entrenamiento [KBS97]. Este aspecto es de especial relevancia cuando alguna de las estimaciones de las probabilidades condicionales vale cero, ya que esto supone que el producto de la ecuación 3.15 se hace cero, desvirtuando los resultados conseguidos. A continuación se analiza esta cuestión.

3.2.1. Estimación de probabilidades a priori en el clasificador naïve-Bayes

En el clasificador naïve-Bayes, la estimación de las probabilidades de la clase y de las probabilidades condicionadas se basa en contar su frecuencia. En este caso, si un determinado valor de un atributo $X = a$ no aparece junto a una etiqueta de clase c_i, la probabilidad condicionada estimada será cero, $\hat{P}(X = a|C = c_i) = 0$, eliminando la clase c_i de toda consideración. Para solucionar el problema de que sólo un valor controle los resultados obtenidos hay dos posibles enfoques [KBS97]:

- El enfoque de no coincidencia (*no-match*). Consiste en reemplazar una probabilidad condicionada cero por $P(X = a, C = c_i)$ con un factor que es inversamente proporcional al número de casos N. Dentro de esta variante existen diferentes soluciones, dependiendo del numerador utilizado. De esta forma, algunos autores utilizan $P(C = c_i)/N$ [CN89, DP96] o en las bibliotecas *MLC++* [KJL+94] se usa por defecto $0{,}5/N$.

- El enfoque de Laplace. Dado un valor predefinido f, si hay s coincidencias de N posibles casos para un problema con k valores, la probabilidad estimada será $(s + f)/(N + kf)$. Así por ejemplo, si en un problema con dos clases se usa $f = 1$, se obtiene la famosa regla de sucesión de Laplace [Goo65] $(s + 1)/(N + 2)$

En la tabla 3.2.1 se muestran los resultados experimentales obtenidos [KBS97] con estos enfoques. Como se puede apreciar, contar la frecuencia es la peor de las soluciones y la ley de sucesión de Laplace la segunda peor. El método *no-matches* con $P(C = c_i)/N$ está por el medio de la tabla. Valores muy pequeños para *no-matches*, tales como $0{,}01/N$ y correcciones similares para Laplace parecen funcionar mejor.

Método	Error medio
Laplace $k = 1/N$	18.58
No-matches $P = 0{,}01/N$	18.51
Laplace $k = 0{,}01$	18.70
No-matches $P = P(C = c_i)/N$	18.62
No-matches $P = 0{,}1/N$	18.64
No-matches $P = 0{,}5/N$	18.76
Laplace $k = 0{,}1$	18.83
Laplace $k = 1$	19.59
Sin método $(P = 0)$	20.16

Tabla 3.1: Comparación de métodos para cálculo de probabilidades en naïve-Bayes

3.2.2. Naïve-Bayes con atributos continuos

El clasificador naïve-Bayes también puede ser utilizado sobre atributos continuos [JL95]. En este sentido, una suposición muy común, aunque no sea intrínseca al método naïve-Bayes, es que en cada atributo continuo los valores siguen una distribución normal. Por tanto, es posible representar esta distribución a través de su media y su desviación típica y calcular la probabilidad de cualquier valor a través de estos parámetros. De esta forma, para cada clase c_i y cada atributo continuo k queda,

$$\hat{p}_{k,i} = \hat{P}(X_k = x_k | C = c_i) = g(x; \mu_{k,i}, \sigma_{k,i}) \tag{3.16}$$

donde,

$$g(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.17}$$

siendo g la función de densidad de probabilidad de una distribución normal o gaussiana.

El anterior modelo deja un pequeño conjunto de parámetros a estimar del conjunto de entrenamiento. Para cada clase y atributo discreto se debe estimar la probabilidad que tomará el atributo en cada valor del dominio dada la clase. Para cada clase y atributo continuo se deben calcular la media y desviación típica del atributo dada la clase. La estimación máximo verosímil de estos parámetros es directa. La estimación de probabilidad

de que una variable aleatoria discreta tome un valor determinado es igual a la frecuencia de la muestra –número de veces que el valor es observado– dividido por el número total de observaciones. La estimación máximo verosímil de la media y desviación típica de una distribución normal son la media de la muestra y la desviación típica de la muestra [Sch95].

Aunque es típico basarse en la presunción de que las variables continuas siguen una distribución normal o gaussiana, existen más enfoques. Entre ellas destaca el algoritmo de aprendizaje *Flexible naïve-Bayes* [JL95]. Este algoritmo muestra como única diferencia, el método utilizado para la estimación de la densidad de los atributos continuos el cual se basa en la estimación de densidad del núcleo (*Kernel density estimation*).

Los resultados experimentales muestran que el método *Flexible naïve-Bayes* mejora al naïve-Bayes con estimaciones normales o gaussianas en la mayoría de los conjuntos de datos. Sin embargo, la correcta discretización de las variables continuas consigue mejorar a ambos métodos [JL95].

El desarrollo de esta tesis se centra en el uso de variables discretas, por lo que siempre que se tengan variables continuas éstas serán discretizadas antes de su uso. En la siguiente sección se analizan las distintas técnicas de discretización.

3.3. Discretización de los atributos continuos

Durante los últimos años se ha comenzado a prestar gran atención a la discretización de los atributos continuos. Hay tres razones principales que han llevado a esta situación [DKS95]:

- Muchos algoritmos desarrollados en aprendizaje automático sólo son capaces de aprender desde variables discretas. Debido a que muchas tareas de clasificación de la vida real presentan valores continuos, se hace necesaria una discretización para poder utilizar estos algoritmos.
- Algunos clasificadores, como ocurre con naïve-Bayes o C4.5, pueden ser utilizados tanto sobre datos discretos como sobre datos continuos, aunque la correcta discretización de los datos puede llevar a una mejora en su rendimiento.
- Mejora de velocidad en los algoritmos de inducción que utilizan atributos discretos.

Existen tres criterios diferentes a la hora de clasificar los métodos de discretización [DKS95]:

- Métodos locales vs. métodos globales: En los métodos locales los atributos se discretizan de forma independiente, mientras que en las discretizaciones efectuadas por los métodos globales se proponen políticas de discretización que tienen en cuenta a todas las variables al mismo tiempo.
- Supervisados vs. no supervisados: Los métodos no supervisados no hacen uso de las etiquetas de clase de los casos del conjunto de entrenamiento durante la discretización. Por contra, los métodos supervisados utilizan esta etiqueta de clase.

- Estáticos vs. dinámicos: Muchos métodos de discretización requieren un parámetro k que indica el máximo número de intervalos que puede producir durante la discretización. Por otra parte, los métodos dinámicos realizan una búsqueda en el espacio de los posibles valores de k sin necesidad de fijar el valor de dicho parámetro.

Se ha constatado, a través de resultados experimentales [DKS95] sobre los algoritmos C4.5 y naïve-Bayes, que el método de discretización que mejores resultados presenta es el descrito en [Tin94], una variante global del método de discretización local presentado en [FI93].

Este método supervisado utiliza una heurística recursiva de minimización de la entropía asociada a un criterio que controla el número de intervalos producidos.

La siguiente notación es la utilizada por Fayyad e Irani para su método de discretización supervisado. Si se tiene un conjunto de instancias S, un atributo X y una partición T, que divide a S en S_1 y S_2, la entropía de la clase de la información inducida por T y llamada $E(X,T;S)$ viene dada por,

$$E(X,T;S) = \frac{|S_1|}{|S|}Ent(S_1) + \frac{|S_2|}{|S|}Ent(S_2) \tag{3.18}$$

donde $Ent(S_1)$ y $Ent(S_2)$ denotan respectivamente la entropía de X en los subconjuntos de instancias S_1 y S_2.

Para un atributo dado X se selecciona la partición T_{min} que minimize la función de entropía sobre todas las posibles particiones como una discretización binaria. Este método es aplicado recursivamente a las particiones inducidas por T_{min} hasta que se llega a una condición de parada, creando de esta forma los múltiples intervalos del atributo X.

Para determinar la condición de parada se hizo uso del principio de mínima distancia de descripción (*Minimum Description Lenght Principle*) [Ris86]. La partición recursiva de un conjunto de valores S se acaba cuando:

$$Gain(X,T;S) < \frac{log_2(N-1)}{N} + \frac{\Delta(X,T;S)}{N} \tag{3.19}$$

donde N es el número de instancias en el conjunto S, siendo

$$Gain(X,T;S) = Ent(S) - E(X,T;S) \tag{3.20}$$

y

$$\Delta(X,T;S) = log_2(3^k - 2) - N \cdot Ent(S) - k_1 \cdot Ent(S_1) - k_2 \cdot Ent(S_2) \tag{3.21}$$

con k_i el número de etiquetas de clase representadas en el conjunto de datos S_i. Como las particiones de cada rama de la discretización recursiva se evalúan de forma independiente usando este criterio, algunas áreas del espacio continuo se particionarán de manera muy fina, mientras que otras, que tienen la entropía relativamente baja, se particionarán menos. En esta descripción no se ha mostrado cómo se derivan estas fórmulas, refiriendo al lector al artículo original para más detalles.

3.4. Tratamiento de los valores ausentes

Ya sea por la naturaleza de los datos o por la discretización realizada sobre los mismos, es posible que en el conjunto de datos aparezcan valores ausentes.

Existen diferentes formas de tratar los valores ausentes:

- Considerar los valores ausentes como un nuevo valor del atributo en el que aparecen [DP96].
- Ignorarlos, no incluyendo estos términos en el algoritmo de aprendizaje.
- Eliminar las instancias que contienen valores ausentes.
- Eliminar un atributo completo en caso de que éste presente valores ausentes.

Existen otros enfoques más novedosos como el que se presenta en el algoritmo *Robust Bayesian Classifier (RBC)* [RS01]. Este algoritmo es una extensión de naïve-Bayes en la que se tienen en cuenta todas las posibles complexiones de los valores ausentes, obteniendo, a partir de dichas complexiones, intervalos para cada una de las probabilidades a priori y condicionadas.

Según un estudio realizado sobre 37 conjuntos de datos del repositorio de UCI [KBS97], el error medio considerando los valores ausentes como un nuevo valor fue de 20,30 % e ignorando estos valores fue de 20,20 %. Además, en la mayor parte de los conjuntos de datos el tratamiento que se dio a los valores ausentes no fue significativo.

De esta forma Kohavi, Becker y Sommerfield [KBS97] concluyeron en su artículo que prácticamente en todos los casos es mejor ignorar los valores ausentes. Esta ha sido, por tanto, la filosofía que se ha utilizado a lo largo de toda la tesis.

3.5. Enfoques semi naïve-Bayes

Existen múltiples enfoques del clasificador naïve-Bayes que consiguen mejorar su exactitud. Estos enfoques son conocidos con el término genérico de variantes **semi naïve-Bayes**.

Tomando como base una clasificación previa [WP98] y extendiéndola, se puede considerar que los clasificadores semi naïve-Bayes son divisibles en varios grupos dependiendo de las actividades pre/post-proceso que realicen:

- Enfoques que procesan las variables a ser empleadas antes de aplicar naïve-Bayes.
- Enfoques que corrigen las probabilidades producidas por naïve-Bayes.
- Enfoques que seleccionan subconjuntos de instancias antes de la aplicación de naïve-Bayes.

A continuación se describen más en profundidad cada una de estas variantes.

3.5.1. Enfoques que tratan las variables a ser empleadas antes de aplicar naïve-Bayes

En este apartado se analizan las variantes semi naïve-Bayes que manipulan las variables predictoras antes de aplicar naïve-Bayes.

El principal grupo de trabajos que se pueden considerar dentro de este apartado son todos los relacionados con la selección de variables.

El enfoque *Selective Bayesian Classifier* [LS94] tiene como objetivo la modificación del clasificador naïve-Bayes para mejorar su exactitud en dominios con atributos redundantes. De esta forma, presenta un algoritmo voraz hacia delante que en cada iteración selecciona, de entre los atributos que todavía no han sido considerados, aquel cuya inclusión produzca una mejor ganancia. El algoritmo finaliza cuando la inclusión de cualquier atributo reduce la exactitud obtenida hasta ese momento. En esta búsqueda cada posible candidato se evalúa a través del tanto por ciento de bien clasificados con la técnica de validación *leave-one-out*.

Cuando se está realizando una selección de variables se pueden utilizar dos enfoques diferentes: *filter* y *wrapper* [KJ97]. En el enfoque *filter* los atributos se seleccionan en base a propiedades intrínsecas de los datos y de forma totalmente independiente al algoritmo de inducción. En el enfoque *wrapper* la selección de variables se realiza utilizando el algoritmo de inducción como si fuera una caja negra. Por tanto, durante la selección se realiza una búsqueda en la que el clasificador se utiliza como parte de la función de evaluación. En el citado trabajo se demuestra experimentalmente que el enfoque *wrapper* realizado a través de una búsqueda voraz es mejor que el enfoque *filter* que realiza el algoritmo *Relief* [Kon94] para la selección de variables.

Por último, en relación a la selección de variables, también es posible realizar búsquedas heurísticas de tipo *wrapper* del mejor conjunto de variables. En este sentido se han utilizado tanto algoritmos genéticos [LLD01] [ILES00] como algoritmos de estimación de las distribuciones (EDAs) [ILES00].

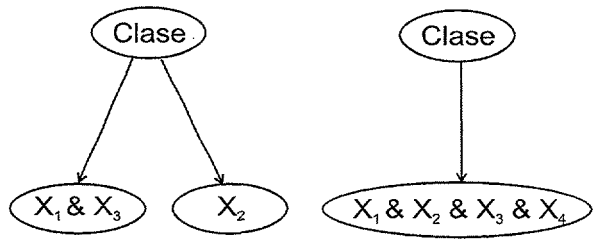


Figura 3.7: Ejemplo de particiones del conjunto de atributos con Pazzani

Otro enfoque semi naïve-Bayes que manipula las variables predictoras y que ha tenido bastante buena acogida es el presentado por Pazzani [PB97]. Véase figura 3.7 para una ilustración de posibles modelos obtenidos al aplicar dicho algoritmo. En este enfoque se construye un modelo naïve-Bayes a través de un algoritmo voraz capaz de detectar variables irrelevantes y variables dependientes entre sí. La idea central del algoritmo es que se pueden obtener clasificadores más exactos como resultado de la unión de grupos de atributos correctos. De esta manera, cuando se detectan atributos dependientes se crea

un nuevo atributo a partir del producto cartesiano de los mismos. El algoritmo está guiado por el tanto por ciento de bien clasificados obtenido por validaciones *leave-one-out*. En el trabajo de Pazzani se proponen dos algoritmos voraces para la unión de atributos: *FSSJ (Forward Sequential Selection and Joining)*, que realiza una búsqueda hacia delante y *BSEJ (Backward Sequential Elimination and Joining)*, que realiza una búsqueda hacia atrás.

El algoritmo voraz FSSJ inicializa el conjunto de atributos a ser utilizado en el clasificador como el conjunto vacío. Por tanto, todas las instancias se clasificarán como la clase más frecuente. A continuación, se utilizan dos operadores para generar el siguiente clasificador:

- Añadir alguno de los atributos que no está en el clasificador actual como un nuevo atributo condicionalmente independiente del resto de los atributos.
- Unir alguno de los atributos que no está en el clasificador actual con los atributos del clasificador.

En cada paso del algoritmo se consideran todas las inclusiones y todas las uniones de los atributos no utilizados hasta ese momento, evaluando cada posible candidato con *leave-one-out* y tomando el que mejor resultado presente. El algoritmo finaliza cuando no se produce ninguna mejoría.

El algoritmo voraz BSEJ crea un clasificador inicial con todos los atributos como condicionalmente independientes. A la hora de generar el nuevo clasificador tiene en cuenta dos operadores:

- Reemplazar cada par de atributos utilizados en el clasificador con un nuevo atributo que los una.
- Eliminar cada atributo utilizado en el clasificador.

Al igual que el algoritmo FSSJ, el algoritmo BSEJ considera todas las posibles modificaciones del clasificador, las evalúa utilizando *leave-one-out* y toma el mejor resultado posible. El algoritmo finaliza cuando no se produce ninguna mejoría.

3.5.2. Enfoques que corrigen las probabilidades producidas por naïve-Bayes

En este apartado se analizan las principales variantes semi naïve-Bayes que de alguna forma varían las probabilidades obtenidas por el clasificador naïve-Bayes.

El enfoque *APNBC (Adjusted Probability naïve-Bayesian Induction)* [WP98] realiza un ajuste lineal del peso de la probabilidad de cada clase. De esta forma, se asigna un factor de ajuste a cada clase, y la probabilidad estimada para cada clase se multiplica por este factor. El ajuste lineal propuesto por los autores se puede expresar de la siguiente manera:

$$P(C = c_i|X_1 = x_1, \ldots, X_n = x_n) \propto w_i P(C = c_i) \prod_{k=1}^{n} P(X_k = x_k|C = c_i) \qquad (3.22)$$

Los pesos relativos a cada clase se buscan a través de un algoritmo voraz guiado por el tanto por ciento de bien clasificados, el cual se obtiene por el método de resustitución.

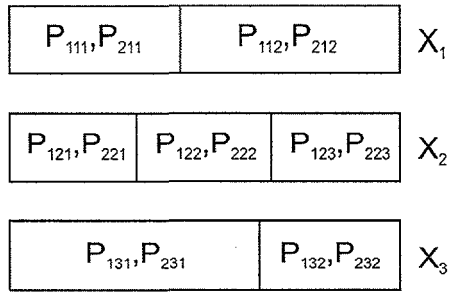


Figura 3.8: Ejemplo de estructura de particiones en un problema con dos clases y tres variables predictoras

El enfoque *WenBay (Weighted naïve-Bayes)* [FDH01], permite asociar pesos a particiones, en lugar de a cada clase. Las instancias de cada variable predictora son particionadas de manera recursiva con el objetivo de maximizar la entropía. En la figura 3.8 se muestra un ejemplo de una posible estructura de partición, en donde los valores P_{ikj} representan la partición número j realizada en la clase i y la variable k. El ajuste realizado por los autores se expresa de la siguiente manera:

$$P(C = c_i|X_1 = x_1, \ldots, X_n = x_n) \propto P(C = c_i) \prod_{k=1}^{n} [P(X_k = x_k|C = c_i)]^{w(X_k,k,i)} \quad (3.23)$$

siendo $w(X_k, k, i)$ la función de peso para la variable X_k y la clase c_i. En los resultados experimentales sobre seis bases de datos del repositorio de UCI [MA95] se consigue mejorar a naïve-Bayes una media de 0,6 % en porcentaje de bien clasificados con la técnica de validación *10-fold cross-validation*.

El algoritmo *Iterative Bayes* [Gam00] tiene como idea fundamental mejorar de forma iterativa las probabilidades condicionadas usadas en el modelo naïve-Bayes. El algoritmo itera sobre el conjunto de casos por medio de un algoritmo de "ascenso por la colina". Todas las instancias del conjunto de entrenamiento se clasifican en cada iteración utilizando las probabilidades actuales, que son evaluadas a través de la siguiente función

$$\frac{1}{N} \sum_{i=1}^{N} (1 - \max_j p(C = c_j|\mathbf{x}^{(i)})) \quad (3.24)$$

donde N representa el número de instancias y j recorre el conjunto de posibles valores de la clase. El proceso iterativo se repite mientras se consiga decrementar la anterior función de evaluación, hasta un máximo de diez iteraciones. La función de actualización tiene la siguiente heurística:

1. Si la instancia está correctamente clasificada entonces el incremento es positivo. En caso contrario, el incremento es negativo. El valor del incremento se calcula con la

siguiente fórmula heurística $(1{,}0 - p(clase\ predicha|\mathbf{x}))/num.\ clases$. Es decir, el valor del incremento depende de la clase predicha y del número de clases.

2. Las probabilidades condicionadas de naïve-Bayes deben incrementarse por cada instancia del conjunto de datos si la clase predicha es verdadera o disminuirse en caso contrario.

Las probabilidades condicionadas son actualizadas cada vez que se recorre el conjunto de datos. Esto significa que el orden de las instancias en el conjunto de datos influye en el resultado final. Por último, es importante destacar que no hay garantía de mejora con este procedimiento. Las pruebas experimentales fueron realizadas sobre 27 conjuntos de datos del repositorio de UCI, consiguiendo reducir el error medio en un 1,24 %.

Basándose en el anterior algoritmo de *Iterative Bayes*, se han creado dos versiones adaptativas de naïve-Bayes [GC02]. La primera denominada *Incremental Adaptive Bayes* construye el modelo naïve-Bayes desde el conjunto de entrenamiento y, posteriormente, lo actualiza cada vez que se le presenta un nuevo caso. La segunda *On-line Adaptive Bayes* tan sólo actualiza el modelo cuando es capaz de predecir la clase a la que pertenece el nuevo caso.

El algoritmo *RBC (Robust Bayesian Classifier)* [RS01] es un clasificador naïve-Bayes diseñado para el caso en que el conjunto de datos presente valores ausentes. Este algoritmo tiene en cuenta todas las posibles combinaciones de los valores ausentes, obteniendo de esta forma intervalos para cada una de las probabilidades a priori y condicionales estimadas por naïve-Bayes. A partir de estos intervalos es posible calcular intervalos para la distribución de probabilidad a posteriori de la clase. Los resultados experimentales sobre 21 conjuntos de casos del repositorio UCI demuestran la superioridad de este método.

El enfoque *NCC (Naïve Credal Classifier)* [Zaf02] es una extensión del clasificador naïve-Bayes basada en conjuntos credales, también denominados conjuntos convexos de distribuciones de probabilidad. Para escapar del requerimiento de la precisión de los valores puntuales que se deben estimar en naïve-Bayes, NCC representa las distribuciones de probabilidad como puntos pertenecientes a regiones geométricas cerradas y acotadas que se encuentran descritas por restricciones lineales. Como consecuencia, el resultado de una clasificación es un conjunto de clases, candidatas a ser la categoría correcta.

Por último cabe destacar el trabajo [GZ02]. En él se propone encontrar los parámetros que maximizan la verosimilitud condicional en lugar de la verosimilitud de la muestra en una red Bayesiana. Su propuesta se encuadra dentro del aprendizaje discriminativo en el que se tiene en cuenta la existencia de una variable especial, la variable a clasificar. Propone un algoritmo de descenso por el gradiente con el que proporciona una aproximación a los valores óptimos de los parámetros.

3.5.3. Enfoques que seleccionan subconjuntos de instancias antes de la aplicación de naïve-Bayes

Existen una serie de variantes que dividen el conjunto de aprendizaje en diferentes subconjuntos y que realizan una clasificación diferente sobre cada uno de ellos.

La variante *NBTREE (Naïve-Bayes Tree)* [Koh96] presenta un algoritmo híbrido entre los árboles de clasificación y el clasificador naïve-Bayes. Se puede definir NBTREE como

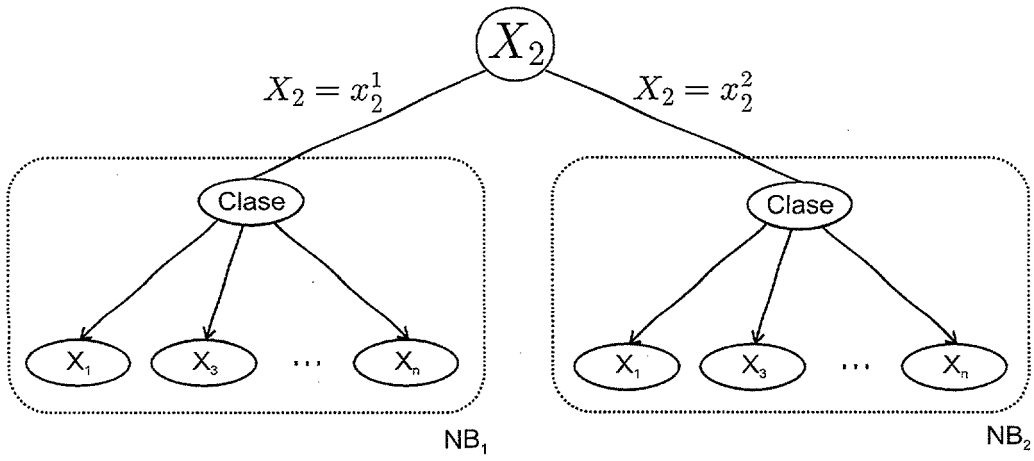


Figura 3.9: Ejemplo de estructura NBTREE con un nodo de decisión con el atributo X_2 y dos clasificadores naïve-Bayes como hojas

un árbol de clasificación cuyas hojas son clasificadores naïve-Bayes (véase figura 3.9). Cada hoja del NBTREE contiene un clasificador naïve-Bayes local que no considera las variables que se encuentran involucradas en los test univariados que están en el camino que lleva hasta la hoja. Las propiedades de este árbol de clasificación son: (i) cada nodo interno representa una variable, (ii) cada nodo interno tiene tantos hijos o ramas salientes como estados tiene la variable representada en dicho nodo, (iii) todas las hojas están al mismo nivel y (iv) en cualquier camino que se recorra desde la raíz hasta las hojas no existen variables repetidas. La condición (iii) se impone para simplificar el modelo, pero puede ser no tenida en cuenta en la práctica. En el trabajo [PLL02] se presenta un algoritmo heurístico para el aprendizaje de este tipo de estructuras. Este algoritmo está basado en la verosimilitud marginal de los datos para realizar la búsqueda.

El algoritmo *LBR (Lazy Bayesian Rule)* [ZW00] trata de solventar los problemas derivados de los subconjuntos disjuntos de tamaño reducido, que pueden aparecer en NBTREE, a través de un aprendizaje perezoso que retrasa el cálculo hasta el último momento. El algoritmo propuesto construye reglas cuyos consecuentes son clasificadores locales naïve-Bayes. El antecedente de cada regla es una conjunción de pares atributo-valor, mientras que el consecuente es un clasificador local naïve-Bayes inducido a través de aquellos ejemplos que satisfacen el antecedente de la regla. Las pruebas experimentales se han realizado sobre 29 conjuntos de datos del repositorio UCI, consiguiendo demostrar su superioridad respecto al naïve-Bayes, NBTREE, C4.5 y BSEJ.

Por último, el enfoque *Recursive Bayesian Classifier* [Lan93] se puede interpretar como un ejemplo de aproximación jerárquica. Se comienza utilizando el clasificador naïve-Bayes. Si se consigue clasificar todas las instancias de forma correcta se concluye el algoritmo, en caso contrario se llama recursivamente al clasificador naïve-Bayes para cada clase para la que se hayan asignado casos pertenecientes a otras clases, usando todos los casos asignados a dicha clase como conjunto de entrenamiento.

3.6. Naïve-Bayes aumentado

La exactitud de las redes Bayesianas como clasificadores supervisados aumenta si se tiene en cuenta la existencia de una variable especial, la variable a clasificar. Una forma sencilla de asegurar que esto suceda es a través de la estructura de la red, como sucedía con el clasificador naïve-Bayes, en el que había un arco entre la clase y cada uno de los atributos. Otro tipo de estructura que asegura una buena exactitud es la denominada *Naïve-Bayes Aumentado (Augmented Naïve-Bayes)* o *BAN*, que es la estructura del naïve-Bayes aumentada con arcos entre las variables.

En [ZL01] se plantea un paradigma perteneciente a BAN, en el que la única limitación entre los arcos de las variables predictoras es que no se formen ciclos (véase figura 3.10). Además demuestra que este tipo de modelos pueden representar cualquier clasificador Bayesiano.

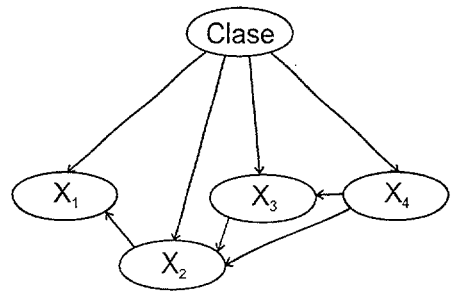


Figura 3.10: Ejemplo de estructura *BAN* cuya única restricción es que no se formen ciclos entre las variables predictoras

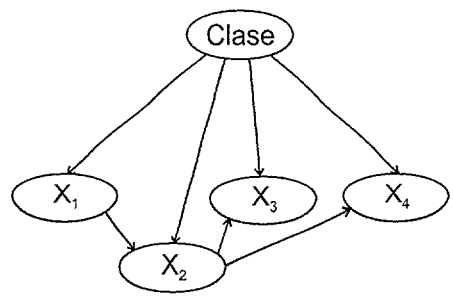


Figura 3.11: Ejemplo de estructura TAN

Otro enfoque diferente también perteneciente al paradigma *Augmented Naïve-Bayes* es la denominada *TAN (Tree Augmented Naïve-Bayesian)* [FGG97], en la que cada atributo tiene como padre la variable clase y como mucho otro atributo. Véase la figura 3.11. Los autores proponen un algoritmo, adaptación de uno anterior [CL68], que utiliza el concepto de información mutua entre las variables predictoras condicionada a la variable a clasificar. La función se define como:

$$I_P(X,Y|C) = \sum_{x,y,c} P(x,y,z) \log \frac{P(x,y|c)}{P(x|c)P(y|c)} \tag{3.25}$$

De manera simple se puede decir que la función anterior mide la información que la variable Y proporciona sobre la variable X cuando el valor de C es conocido.

El algoritmo propuesto [FGG97], que garantiza que la estructura TAN obtenida tiene asociada la máxima verosimilitud entre todas las posibles estructuras TAN, es como sigue:

1. Calcular $I_P(X_i, X_j|C)$ para cada par de variables predictoras con $i \neq j$.
2. Construir un grafo no dirigido completo en el cual los vértices son las variables predictoras $X_1, \ldots, X_n$. Asignar a cada arista conectando las variables X_i y X_j el peso dado por $I_P(X_i, X_j|C)$.
3. Construir un árbol expandido de máximo peso.
4. Transformar el árbol resultante no dirigido en uno dirigido, escogiendo una variable raíz y direccionando todas las aristas partiendo del nodo raíz.
5. Construir un modelo TAN añadiendo un nodo etiquetado como C y, posteriormente, un arco desde C a cada variable predictora X_i.

Por último, se encuentra el algoritmo *ACO (Archs in Correct Order)* [RSL02], un algoritmo heurístico para inducir estructuras de árbol aumentado. El algoritmo propuesto parte de una estructura de árbol aumentado, la cual se revisa en caso de que los nuevos datos invaliden la estructura vigente. Dicha revisión se efectúa en el caso en que las ramas derivadas de los nuevos datos no estén en el mismo orden decreciente que el que ha propiciado la estructura actual. La reconstrucción del árbol se lleva a cabo desde la primera rama incorrecta encontrada.

3.7. Enfoque manto de Markov

El método naïve-Bayes tiene en consideración el hecho de que existe una variable especial en el problema a tratar, la variable clase. Sin embargo, debido a su simplicidad estructural, no alcanza a considerar de forma adecuada la semántica intrínseca de las redes Bayesianas [Pea88, Jen01, CGH97, Cow01].

Teniendo en cuenta que en una red Bayesiana cualquier variable tan sólo se encuentra influenciada por el denominado *MB (Markov Blanket)* relativo a la misma –es decir, por el conjunto de sus variables padre, sus variables hijas y las variables que son padre de las hijas–, parece intuitivo tener en cuenta modelos clasificatorios que sean MB de la variable a clasificar. En definitiva, se deben buscar estructuras en las que todas las variables formen parte del MB de la variable a clasificar. Véase figura 3.12.

Existen varios procedimientos para buscar dentro del espacio de posibles MB de la variable a clasificar. Por ejemplo, es posible utilizar algoritmos genéticos [SL98] para llevar a efecto tal búsqueda.

Debido a los problemas de sobreajuste –constatados en algunos experimentos–, que sufre el método anterior, se pueden reducir las condiciones impuestas por el modelo, con el

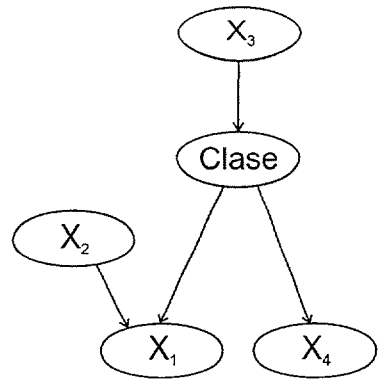


Figura 3.12: Ejemplo de clasificador MB

objeto de obtener redes Bayesianas más simples pero a la vez con mayor poder generalizador. Para ello se pueden, por ejemplo, efectuar las siguientes dos relajaciones del modelo anterior:

1. No es necesario que todas las variables del modelo formen parte del MB de la variable a clasificar. De este modo se reduce el espacio de búsqueda, a la vez que se realiza de forma implícita una selección de variables.

2. Una variable que sea padre de la variable a clasificar no puede ser padre de un hijo de la variable a clasificar. Además, una variable tan sólo puede ser padre de un hijo de la variable a clasificar.

Parte III

ALGORITMOS DE BÚSQUEDA

Capítulo 4

Enfoque general

En este capítulo se realiza un análisis general de nuestras propuestas dentro del campo de la clasificación supervisada con modelos gráficos probabilísticos.

La idea fundamental dentro de estas propuestas, tal y como se verá en la primera sección, es hacer una apuesta por el uso de algoritmos heurísticos de optimización para realizar las búsquedas que muchas de las técnicas actuales realizan a través de algoritmos voraces (*greedy*).

Como algoritmo heurístico de optimación se ha optado por el uso de los algoritmos de estimación de distribuciones (EDAs). Estos algoritmos ya han demostrado su potencia dentro de muchos otros campos.

El índice del capítulo es el siguiente:

- En la sección 4.1 se realiza una descripción general de las propuestas que se realizan dentro del campo de clasificación supervisada con modelos gráficos probabilísticos.
- En la sección 4.2 se introducen los algoritmos de estimación de distribuciones (EDAs), los algoritmos heurísticos que serán utilizados para la búsqueda de los clasificadores.

4.1. Descripción general de las propuestas

En esta tesis se realiza una fuerte apuesta por el uso de los algoritmos heurísticos de optimización EDA, como herramienta válida para la realización de búsquedas de clasificadores supervisados basados en redes Bayesianas.

Dentro de este enfoque se realizan varias propuestas concretas que a continuación se describen.

4.1.1. Diseño y paralelización de un nuevo algoritmo de clasificación semi naïve-Bayes

Proponemos un nuevo algoritmo semi naïve-Bayes denominado *Interval Estimation naïve-Bayes* (IENB) [RLP+03b, RLP+03a, RLM+03]. En este enfoque, en lugar de calcular la estimación puntual de las probabilidades necesarias para el clasificador naïve-Bayes, se calcula una estimación por intervalo. Después, a través de la búsqueda heurística de la

mejor combinación de valores dentro de estos intervalos, se trata de superar la asunción de independencia condicional entre variables predictoras dada la clase, premisa sobre la que se construye el naïve-Bayes. La búsqueda se realiza con algoritmos heurísticos de optimización (EDAs) y está guiada por la exactitud de los clasificadores.

Este algoritmo se puede situar dentro de las variantes semi naïve-Bayes que corrigen las probabilidades producidas por dicho clasificador.

IENB puede llegar a ser muy costoso desde el punto de vista computacional debido a la búsqueda heurística que debe realizar. Como solución a este coste computacional de los EDAs, proponemos una paralelización de los mismos basada en islas [RPP+03]. Cada isla contendrá una población diferente y, cada cierto tiempo y con un esquema de migración predeterminado, las islas se intercambiarán los mejores individuos entre sí. Esta forma de paralelización ya ha sido utilizada con éxito en los algoritmos genéticos [CP01, Ste93].

4.1.2. Aprendizaje de clasificadores en el espacio de estructuras

Las variantes semi naïve-Bayes están normalmente basadas en la realización de búsquedas de determinadas estructuras o determinados valores. Estas búsquedas se suelen realizar a través de algoritmos de búsqueda voraz (*greedy*).

En esta tesis proponemos el uso de los algoritmos de optimización EDA, como herramienta para la realización de las búsquedas de estructuras semi naïve-Bayes [RLP+03d].

De esta forma, proponemos extensiones a los algoritmos *BSEJ (Backward Sequential Elimination and Joining)* y *FSSJ (Forward Sequential Selection and Joining)* [Paz97] y al enfoque *APNBC (Adjusted Probability Naïve-Bayes Classification)* [WP98]. Estas extensiones están basadas en la realización de la búsqueda del mejor clasificador con algoritmos heurísticos de optimización, frente a las búsquedas voraces realizadas actualmente.

Además, como se vio en el capítulo 3, existen varias estructuras de red Bayesiana que pueden ser utilizadas para realizar clasificación supervisada: TAN (*Tree Augmented Network*), BAN (*Bayesian Augmented Network*) y MB (*Markov Blanquet*). En todas estas estructuras se ha tenido en cuenta el hecho de que existe una variable especial, la variable a clasificar.

Frente a los métodos de búsqueda que se encuentran actualmente en la literatura [FGG97, Sah96], que se basan en la cantidad de información mutua entre las variables predictoras, en esta tesis proponemos el desarrollo de algoritmos voraces guiados por el tanto por ciento de bien clasificados para la búsqueda de estructuras TAN y BAN. También proponemos el uso de algoritmos heurísticos de optimización EDA para la búsqueda de estructuras TAN, BAN y MB.

4.2. Algoritmos de Estimación de Distribuciones (EDAs)

En las propuestas anteriormente realizadas, se usan los *algoritmos de estimación de distribuciones* (EDAs) como los algoritmos heurísticos de optimización que permiten realizar las búsquedas. Los EDAs son algoritmos de optimización que pertenecen a la familia de *algoritmos evolutivos*.

Los algoritmos evolutivos son técnicas implementadas para la resolución de problemas de optimización combinatoria, y están basados en la evolución de poblaciones de indivi-

duos, que representan posibles soluciones candidatas. A partir de una población inicial, habitualmente generada de forma aleatoria, se selecciona un conjunto de soluciones prometedoras. La información contenida en el conjunto de individuos seleccionados se utiliza para crear una nueva población hija. Los nuevos individuos reemplazan total o parcialmente a los de la antigua población. El procesamiento de la información contenida en el conjunto seleccionado y la incorporación de nuevos individuos en la población previa, se repite hasta que la población resultante satisface algún criterio de parada.

Los algoritmos genéticos, inspirados en la genética Mendeliana, procesan la información contenida en el conjunto, generando los nuevos individuos mediante operadores de cruce de pares de individuos y operadores de mutación. Recientemente han aparecido otro tipo de algoritmos evolutivos, que procesan la información contenida en el conjunto, calculando la estimación de la distribución de probabilidad conjunta de esos individuos y generando nuevos individuos mediante el muestreo de esa distribución. Son los llamados algoritmos EDA [MP96, LL02]. Su objetivo es generar nuevas soluciones usando información extraída del conjunto completo de soluciones prometedoras, construyendo un modelo explícito de las buenas soluciones encontradas hasta el momento para guiar la subsiguiente búsqueda. De este modo, se intenta tener en cuenta las interacciones o dependencias entre soluciones parciales que contribuyen a la calidad de la solución global, frente al proceso más ciego de recombinación de individuos utilizado por los algoritmos genéticos. La cuestión clave en los algoritmos EDA es cómo estimar fiable y eficientemente la distribución de probabilidad del conjunto de individuos prometedores seleccionado.

En este sentido, los métodos de factorización de la distribución basados en redes Bayesianas aparecen como una herramienta muy apropiada. Pero debido a su reciente desarrollo, su aplicación a los problemas de optimización clásicos y reales está aún comenzando.

4.2.1. Introducción

Los algoritmos EDA [MP96, LL02] son estrategias estocásticas de búsqueda heurística que forman parte del enfoque computacional evolutivo. En cada generación crean un número de soluciones o individuos, evolucionando una y otra vez hasta que se alcanza una solución satisfactoria. En resumen, la característica que más diferencia a los EDA de otras estrategias de búsqueda evolutiva, tales como los algoritmos genéticos (GAs), es que la evolución de una generación a la siguiente se hace estimando la distribución de probabilidad de los individuos mejor dotados y simulando el modelo inducido. Esto evita el uso de operadores de cruce o mutación, reduciendo considerablemente el número de parámetros que requieren los EDA.

Los EDAs están empezando a ser utilizados en múltiples dominios [LL02]: búsqueda de la mejor permutación [LBL$^+$01], el problema del viajante [RdML02], selección de variables, planificación de trabajos, etc.

En los algoritmos EDA no se dice que los individuos contienen genes, sino variables cuyas dependencias tienen que ser analizadas. Además, mientras que en otras heurísticas de computación evolutiva las interrelaciones entre las diferentes variables que representan a los individuos se mantienen implícitas, en los EDA las interrelaciones se expresan explícitamente a través de la distribución de probabilidad conjunta asociada con los individuos seleccionados en cada iteración. La tarea de estimar la distribución de probabilidad conjunta de los individuos seleccionados de las generaciones previas constituye el trabajo

más duro a realizar, debido a que requiere la adaptación de métodos de aprendizaje de modelos de datos, desarrollados en el dominio de modelos gráficos probabilísticos.

EDA

$D_0 \leftarrow$ Generar M individuos (la población inicial) aleatoriamente

Repetir para $l = 1, 2, \ldots$ hasta que se cumpla un criterio de parada

$D_{l-1}^N \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} de acuerdo a un criterio de selección

$\rho_l(\boldsymbol{x}) = \rho(\boldsymbol{x}|D_{l-1}^N) \leftarrow$ Estimar la distribución de probabilidad de que un individuo se encuentre entre los seleccionados

$D_l \leftarrow$ Simular M individuos (la nueva población) desde $\rho_l(\boldsymbol{x})$

Figura 4.1: Pseudocódigo de EDA

La figura 4.1 muestra el pseudocódigo de EDA, en el cual se distinguen 4 pasos principales:

1. Al comenzar, se genera la primera población D_0 de M individuos, asumiendo una distribución uniforme en cada variable (tanto en el caso discreto como en el caso continuo) y evaluando cada uno de los individuos.
2. En segundo lugar, se selecciona un número N ($N \leq M$) de individuos, normalmente los mejores candidatos.
3. En tercer lugar, se induce el modelo probabilístico n-dimensional expresando las interdependencias entre las n variables.
4. Por último, se obtiene la nueva población de M nuevos individuos, mediante la simulación de la distribución de probabilidad obtenida en el paso anterior.

Los pasos 2, 3 y 4 se repiten hasta que se verifique una condición de parada. El paso más importante de este nuevo paradigma es encontrar las interdependencias entre las variables (paso 3). Esta tarea se realizará usando técnicas del campo de los modelos gráficos probabilísticos.

A continuación se introduce la notación necesaria para profundizar en dichos modelos gráficos. Sea $\boldsymbol{X} = (X_1, \ldots, X_n)$ un conjunto de variables aleatorias, y sea x_i un valor de X_i, el i-ésimo componente de X. Sea $\boldsymbol{y} = (x_i)_{X_i \in \boldsymbol{Y}}$ un valor de $\boldsymbol{Y} \subseteq \boldsymbol{X}$. De esta forma, un modelo gráfico probabilístico de $\boldsymbol{X}$ es una factorización gráfica de la función de densidad de probabilidad conjunta, $\rho(\boldsymbol{X} = \boldsymbol{x})$ (o simplemente $\rho(\boldsymbol{x})$). La representación de este modelo viene dada por dos componentes: una estructura y un conjunto de densidades de probabilidad locales.

Respecto a la estructura del modelo, la estructura S para $\boldsymbol{X}$ es un grafo acíclico dirigido (DAG) que describe un conjunto de interdependencias condicionales entre las variables sobre $\boldsymbol{X}$. $\boldsymbol{Pa}_i^S$ representa el conjunto de padres –variables desde las cuales sale una flecha

en S– de la variable X_i en el modelo gráfico probabilístico, cuya estructura viene dada por S. La estructura S para $\mathbf{X}$ asume que X_i y sus no descendientes son condicionalmente independientes dado $\boldsymbol{Pa}_i^S$, $i = 2, \ldots, n$. Por lo tanto, la factorización puede ser escrita como sigue:

$$\rho(\boldsymbol{x}) = \rho(x_1, \ldots, x_n) = \prod_{i=1}^{n} \rho(x_i \mid \boldsymbol{pa}_i^S). \tag{4.1}$$

Además, las densidades locales de probabilidad asociadas con el modelo gráfico probabilístico son precisamente aquellas que aparecen en la ecuación 4.1.

Una representación de los modelos con las características descritas anteriormente asume que las densidades locales de probabilidad dependen de un conjunto finito de parámetros $\boldsymbol{\theta}_S \in \boldsymbol{\Theta}_S$, y como resultado de tal asunción la ecuación anterior se puede escribir como sigue:

$$\rho(\boldsymbol{x} \mid \boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i \mid \boldsymbol{pa}_i^S, \boldsymbol{\theta}_i) \tag{4.2}$$

donde $\boldsymbol{\theta}_S = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n)$.

Después de haber definido ambos componentes del modelo gráfico probabilístico, el modelo en sí se representará por $MO = (S, \boldsymbol{\theta}_S)$.

4.2.2. EDAs en dominios discretos

En el caso particular donde todas las variables $X_i \in \boldsymbol{X}$ son discretas, el modelo gráfico probabilístico utilizado es una red Bayesiana. Si la variable X_i tiene r_i posibles valores, $x_i^1, \ldots, x_i^{r_i}$, la distribución local, $p(x_i \mid \boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i)$ es:

$$p(x_i{}^k \mid \boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i) = \theta_{x_i^k \mid \boldsymbol{pa}_i^j} \equiv \theta_{ijk} \tag{4.3}$$

donde $\boldsymbol{pa}_i^{1,S}, \ldots, \boldsymbol{pa}_i^{q_i,S}$ denota los valores de $\boldsymbol{Pa}_i^S$, que es el conjunto de padres de la variable X_i en la estructura S; q_i es el número de las diferentes posibles instanciaciones de las variables padre de X_i. De este modo, $q_i = \prod_{X_g \in \boldsymbol{Pa}_i} r_g$. Los parámetros locales vienen dados por $\boldsymbol{\theta}_i = ((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i})$. En otras palabras, el parámetro θ_{ijk} representa la probabilidad condicional de que la variable X_i tome su valor k-ésimo, conociendo que el conjunto de sus variables padre toman su valor j-ésimo. Se asume que cada θ_{ijk} es mayor que cero.

Todos los EDAs se clasifican de acuerdo al número máximo de dependencias que se aceptan entre las variables (máximo número de padres que cada variable puede tener en el modelo gráfico probabilístico).

Sin interdependencias

El algoritmo UMDA (*Univariate Marginal Distribution Algorithm (UMDA)*) [Müh98], es un ejemplo representativo de esta categoría, que puede ser escrito como:

$$p_l(\boldsymbol{x};\boldsymbol{\theta}^l) = \prod_{i=1}^{n} p_l(x_i;\theta_i^l) \tag{4.4}$$

donde $\boldsymbol{\theta}_i^l = \left\{\theta_{ijk}^l\right\}$ es recalculado en cada generación usando la estimación de máxima verosimilitud, es decir $\widehat{\theta}_{ijk}^l = \frac{N_{ijk}^{l-1}}{N_{ij}^{l-1}}$. N_{ijk}^{l-1} es el número de casos en los cuales la variable X_i toma el valor x_i^k cuando sus padres están en su j-ésima combinación de valores para la $(l-1)$-ésima generación, y $N_{ij}^{l-1} = \sum_k N_{ijk}^{l-1}$.

Dependencias a pares

Un ejemplo de esta segunda categoría es el algoritmo denominado *MIMIC (Mutual Information Maximization for Input Clustering)* [DIV97]. La idea principal de MIMIC es describir la distribución de probabilidad conjunta verdadera tan bien como sea posible, usando únicamente una probabilidad marginal univariada y $n-1$ pares de funciones de probabilidad condicional.

Interdependencias múltiples

Se usará EBNA (Algoritmo de Estimación de red Bayesiana - *Estimation of Bayesian Network Algorithm*) como ejemplo de esta categoría. El enfoque EBNA se introdujo por primera vez en [EL99], donde los autores usan el criterio de información bayesiana *(Bayesian Information Criterion - BIC)*, como la métrica para evaluar la bondad de cada estructura encontrada durante la búsqueda.

El modelo inicial MO_0 en EBNA se forma mediante su estructura S_0, que es un grafo acíclico dirigido sin arcos, y las distribuciones de probabilidad locales vienen dadas por las n probabilidades marginales unidimensionales $p(X_i = x_i) = \frac{1}{r_i}$, $i = 1,\ldots,n$. Es decir, MO_0 asigna la misma probabilidad a todos los individuos. El modelo de la primera generación MO_1 se aprende usando el algoritmo B [Bun91], mientras que el resto de modelos son aprendidos siguiendo una estrategia de búsqueda local que recibe el modelo de la generación anterior como estructura inicial.

Simulación en redes Bayesianas

La simulación de redes Bayesianas se usa solamente en los EDAs como una herramienta para generar nuevos individuos para la siguiente población, basándose en la estructura aprendida previamente. El método usado para la simulación es el muestreo lógico probabilístico (*Probabilistic Logic Sampling - PLS*) [Hen88]. Siguiendo este método, las instanciaciones se hacen variable a variable siguiendo un orden ancestral. Es decir, una variable no es muestreada hasta que todos sus padres no lo hayan sido.

4.2.3. EDAs en dominios continuos

En esta sección se introduce un ejemplo de modelo gráfico probabilístico que asume que la función de densidad conjunta es una función de densidad Gaussiana multivariada.

La función de densidad local para la i-ésima variable es computada como el modelo de regresión lineal siguiente:

$$f(x_i \mid \boldsymbol{pa}_i^S, \boldsymbol{\theta}_i) \equiv \mathcal{N}(x_i; m_i + \sum_{x_j \in \boldsymbol{pa}_i} b_{ji}(x_j - m_j), v_i) \tag{4.5}$$

donde $\mathcal{N}(x; \mu, \sigma^2)$ es una distribución normal univariada con media μ y varianza σ^2.

Los parámetros locales vienen dados por $\boldsymbol{\theta}_i = (m_i, \boldsymbol{b}_i, v_i)$, donde $\boldsymbol{b}_i = (b_{1i}, \ldots, b_{i-1i})^t$ es un vector columna. Los parámetros locales son los siguientes: m_i es la media incondicional de X_i, v_i es la varianza condicional de X_i dado $\boldsymbol{pa}_i$, y b_{ji} es un coeficiente lineal que mide la fortaleza de la relación entre X_j y X_i. Todo modelo gráfico probabilístico que se construye con estas funciones de densidad locales se conoce como red Gaussiana. Las redes Gaussianas son de interés en los EDAs continuos debido a que el número de parámetros necesarios para especificar una densidad Gaussiana multivariada es menor que si dicha especificación se hiciese a partir de una normal multidimensional.

A continuación se realiza una clasificación análoga a la hecha en el dominio discreto, en la cual los EDAs continuos también se organizan de acuerdo al número de dependencias a tener en cuenta.

Sin dependencias

En este caso se asume que la función de densidad conjunta sigue una distribución normal n-dimensional y que, por tanto, se factoriza como un producto de n densidades normales unidimensionales e independientes. Usando la notación matemática $\boldsymbol{X} \equiv \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \sum)$, puede expresarse como:

$$\begin{aligned} f_{\mathcal{N}}(\boldsymbol{x}; \boldsymbol{\mu}, \sum) &= \prod_{i=1}^{n} f_{\mathcal{N}}(x_i; \mu_i, \sigma_i^2) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2}(\frac{x_i - \mu_i}{\sigma_i})^2}. \end{aligned} \tag{4.6}$$

Un ejemplo de EDAs continuos en esta categoría es UMDA$_c$ [LELP00].

Dependencias a pares

Un ejemplo de esta categoría es MIMIC$_c^G$ [LELP00], que es básicamente una adaptación del algoritmo MIMIC [DIV97] al dominio continuo.

Dependencias múltiples

Los algoritmos de esta sección son variantes de EDAs para dominios continuos, en los que no hay restricción en el aprendizaje de la función de densidad en cada generación. Un ejemplo de esta categoría es EGNA$_{BIC}$ (Algoritmo de Estimación de una Red Gaussiana – *Estimation of Gaussian Network Algorithm*) [LELP00]. El método usado para encontrar la estructura de red Gaussiana en cada generación es un algoritmo de métrica+búsqueda. En EGNA$_{BIC}$ se usa una búsqueda local para buscar buenas estructuras, en combinación con el criterio de información Bayesiana, BIC.

Simulación de redes Gaussianas

El "método condicionado" es una variante general para muestrear distribuciones normales multivariadas que genera instancias de $\boldsymbol{X}$ mediante el muestreo de X_1, X_2 condicionado a X_1, y así sucesivamente. La simulación de una distribución normal univariada se puede realizar –en base al teorema central del límite– con un método simple basado en la suma de doce variables uniformes.

Capítulo 5

Propuestas

En este capítulo se describen y evalúan las propuestas que hemos realizado en el campo de la clasificación supervisada con modelos gráficos probabilísticos.

El índice del capítulo es el siguiente:

- En la sección 5.1 se presentan los conjuntos de datos que serán utilizados para validar los algoritmos que proponemos a lo largo de todo el capítulo.
- En la sección 5.2 se define la metodología de trabajo que se sigue a la hora de validar los diferentes algoritmos propuestos.
- La sección 5.3.2 presenta un nuevo algoritmo semi naïve-Bayes denominado *Interval Estimation naïve-Bayes* (IENB).
- Debido al gran coste computacional que puede presentar el algoritmo IENB, se realiza una paralelización de dicho algoritmo, denominada Parallel *Interval Estimation naïve-Bayes* (PIENB) en la sección 5.4.
- La sección 5.5 presenta el algoritmo APNBC-EDA, una versión heurística del algoritmo voraz APNBC.
- En la sección 5.6 se propone el algoritmo Pazzani-EDA, una versión heurística de los algoritmos voraces FSSJ y BSEJ propuestos en el artículo original.
- En la sección 5.7 se proponen dos algoritmos diferentes para la búsqueda de estructuras de tipo BAN. El primero de ellos está basado en una búsqueda voraz y el segundo de ellos en una búsqueda heurística. Ambos algoritmos están guiados por el tanto por ciento de bien clasificados.
- De forma similar, en la sección 5.8 se presentan dos algoritmos para la búsquedas de estructuras de tipo TAN. El primero de ellos está basado en una búsqueda voraz y el segundo en una búsqueda heurística con EDAs. Ambos algoritmos están guiados por el tanto por ciento de bien clasificados.
- Por último, en la sección 5.9 se presenta un algoritmo heurístico para la búsqueda de clasificadores MB. Este algoritmo está guiado por el tanto por ciento de bien clasificados.

5.1. Conjuntos de datos

A lo largo de todo el capítulo la exactitud de los diferentes algoritmos se evaluará en dominios ampliamente utilizados por la comunidad de aprendizaje automático. Se dispone de un total de 21 conjuntos de datos, todos ellos provenientes del repositorio de UCI [MA95], excepto los conjuntos *m-of-n* y *waveform-21* que fueron diseñados para evaluar técnicas de selección de variables [KBS97]. La elección de los conjuntos de datos ha sido sencilla, ya que son los más utilizados en el campo de las redes Bayesianas como clasificadores supervisados. De esta forma, resultará mucho más simple la realización de comparaciones entre los algoritmos desarrollados y los ya existentes. En la tabla 5.1 se encuentran descritos los conjuntos de datos que serán utilizados en los experimentos.

	Atributos				Instancias	
Nombre	Total	Continuos	Nominales	Clases	Apren.	Valid.
breast	10	10	-	2	699	-
chess	36	-	36	2	3196	-
cleve	13	6	7	2	303	-
corral	6	-	6	2	128	-
crx	15	6	9	2	690	-
flare	10	2	8	2	1066	-
german	20	7	13	2	1000	-
glass	9	9	-	7	214	-
glass2	9	9	-	2	163	-
hepatitis	19	6	13	2	155	-
iris	4	4	-	3	150	-
lymphography	18	3	15	4	148	-
mofn-3-7-10	10	-	10	2	300	1024
pima	8	8	-	2	768	-
satimage	36	36	-	6	6435	-
segment	19	19	-	7	2310	-
shuttle-small	9	9	-	7	5800	-
soybean-large	35	-	35	19	683	-
vehicle	18	18	-	4	846	-
vote	16	-	16	2	435	-
waveform-21	21	21	-	3	300	4700

Tabla 5.1: Descripción de los conjuntos de datos utilizados en los experimentos

Los conjuntos de datos escogidos son los siguientes:

- **Breast cancer Wisconsin (breast):** Son 699 instancias de casos clínicos recopilados por el Dr. Wolberg en la Universidad de Winconsin. Fueron recopilados en un periodo de dos años y medio y el problema es averiguar si los tumores son benignos o malignos, basándose en los datos del cáncer de cada paciente. Tiene diez atributos, uno es un número de serie y los otros nueve diferentes características: uniformidad del tamaño de la célula, uniformidad de la forma de la célula, mitosis, núcleo normal, núcleo descubierto, etc. Estos atributos presentan valores enteros entre el 1 y el 10.

- **Chess endgame database (chess):** Este conjunto de datos tiene 3196 instancias. Representa una partida de ajedrez entre la reina y la torre blancas y la reina negra. Mueve la pieza negra y hay que averiguar si la posición es de empate o pérdida en N movimientos. Fue desarrollado por Michael Bain y Arthur van Hoff en el Instituto Turing de Glasgow. Tiene un total de 36 atributos nominales con dos posibles valores (si la pieza correspondiente está en una determinada fila o columna) y dos posibles clases.

- **Cleveland heart disease (cleve):** 303 instancias del Dr. Detrato. La tarea es diferenciar la presencia o ausencia de enfermedades cardíacas en los pacientes. Hay siete atributos nominales y seis continuos. Los atributos incluyen: edad, sexo, tipo de dolor de pecho, colesterol, etc.

- **Corral (corral):** Es un conjunto de datos artificial con un total de 128 instancias. Contiene seis atributos: $A0$, $A1$, $B0$, $B1$, "irrelevante" y "correlada". El concepto principal es $(A0 \wedge A1) \vee (B0 \wedge B1)$. El atributo denominado "irrelevante" es uniformemente aleatorio, y el atributo denominado "correlado" coincide con la etiqueta de la clase en el 75 % de las ocasiones.

- **Australian credit screening (crx):** Presenta un total de 690 instancias de una compañía de crédito australiana. La tarea es determinar si conceder o no una tarjeta de crédito a los solicitantes. Los atributos están codificados para preservar la confidencialidad. Este conjunto de datos fue usado por primera vez en al año 1986 [Qui86]. Tiene seis variables continuas y nueve nominales.

- **Solar flare database (flare):** Este conjunto de datos contiene un total de 1066 instancias. Cada instancia representa características particulares de una región activa del sol y es clasificada según haya llamas en esa determinada región o no. Tiene un total de 10 atributos, 2 continuos y 8 nominales.

- **German credit data (german):** Conjunto de datos con un total de 1000 instancias. Es muy similar al conjunto *crx*. A través de un total de 20 atributos (7 continuos y 13 nominales), hay que decidir si es posible la concesión de préstamos a clientes.

- **Glass identification database (glass):** El estudio de la clasificación de los cristales ha sido motivada por la investigación criminológica. En la escena del crimen, los cristales abandonados pueden servir de prueba siempre y cuando sea posible identificarlos correctamente. Este conjunto de datos tiene un total de 214 instancias, con 9 atributos continuos: contenido en sodio, en magnesio, en aluminio, etc. A través de estos atributos hay que deducir la clase a la que pertenece el cristal: de edificio, de coche, de contenedor, etc. Existe un total de 7 posibles clases.

- **Glass identification database 2 (glass2):** Es muy similar al conjunto de datos anterior *glass*, pero en este caso la clase indica si el cristal ha sido procesado por planchas o no. Tiene un total de 163 instancias descritas a través de 9 atributos continuos.

- **Hepatitis database (hepatitis):** Son 155 instancias sobre una base de datos de hepatitis. Tiene un total de 19 atributos y dos posibles clases. De los 19 atributos 6 son continuos y 13 nominales.

- **Iris plants database (iris):** Tiene un total de 150 instancias. El objetivo es realizar una clasificación de lirios a través de cuatro atributos de tipo continuo: el ancho y el largo tanto del pétalo como del sépalo. Hay un total de tres tipos diferentes de lirios.

- **Lymphography domain (lymphography):** El objetivo de este conjunto de datos es averiguar el estado de los ganglios linfáticos de los pacientes: normal, con metástasis, maligno o fibroso. Tiene un total de 148 instancias, descritas por 18 atributos, tres de los cuales son continuos. Este conjunto de datos proviene de la University Medical Centre, Institute of Oncology, Ljubljana, Eslovenia y, junto con el conjunto *breast*, es uno de los más utilizados en el aprendizaje automático.

- **m-of-n-3-7-10 (mofn-3-7-10):** Este conjunto de datos tiene un total de 300 instancias de entrenamiento y 1024 de validación. La clase indica si al menos tres bits, de los numerados tres a nueve, están a uno (bits uno, dos y diez son irrelevantes). El objetivo que se persigue en este conjunto de datos es muy frecuente en dominios médicos, donde un paciente necesita tener por lo menos m de un conjunto de n síntomas para que se le diagnostique la enfermedad [Spa88].

- **Pima indian diabetes (pima):** 768 instancias del Instituto Nacional de la diabetes y de enfermedades digestivas y de riñón de la Johns Hopkins University. La tarea es determinar si los pacientes muestran signos de diabetes de acuerdo con los criterios de la Organización Mundial de la Salud. Todos los pacientes son femeninos y viven cerca de Phoenix, Arizona, tienen al menos 21 años y son de herencia india. Este conjunto de datos está formado por ocho atributos continuos tales como: edad, número de veces embarazada, concentración de glucosa, tensión, etc.

- **Satellite image data (satimage):** Está formado por 6435 instancias. Tiene un total de 6 posibles clases y 36 atributos (4 bandas espectrales y 9 píxeles por imagen) con valores continuos en el rango 0–255. Este conjunto de datos viene dado en orden aleatorio y algunas líneas han sido borradas, de tal forma que no es posible reconstruir la imagen original. El objetivo es clasificar la imagen en 6 posibles tipos diferentes.

- **Image segmentation data (segment):** Este conjunto de datos está formado por 2310 instancias. Fue desarrollado en 1990 por el Vision Group, University of Massachusetts. Las instancias fueron seleccionadas aleatoriamente de una base de datos de siete imágenes de exterior. Las imágenes se segmentaron a mano para crear una clasificación para cada pixel. Cada instancia es una región de tres por tres descrita a través de 19 atributos continuos.

- **Shuttle dataset (shuttle-small):** Está formado por 5800 instancias. El objetivo es la clasificación de imágenes de satélite en siete clases diferentes. Este conjunto de datos está formado por nueve atributos continuos y proviene de Jason Catlett del Basser Department of Computer Science, University of Sydney.

- **Large soybean database (soybean-large):** Tiene un total de 683 instancias. La tarea es diagnosticar enfermedades del *soybean*, una legumbre asiática. Existe un total de 19 enfermedades diferentes (clases) y 35 atributos nominales con propiedades y varias anomalías de esta legumbre.

- **Vehicle silhouette dataset (vehicle):** Este conjunto de datos tiene un total de 846 instancias. El propósito es averiguar, a través de sus 18 atributos continuos, si un determinado vehículo es un Opel, un Saab, un autobús, o una furgoneta. El propósito original de estos datos, provenientes del Instituto Turing de Glasgow, Scotland, era encontrar un método para diferenciar objetos 3D a través de una imagen 2D, aplicando un conjunto de extractores de atributos de formas a las siluetas 2D de los objetos.

- **United States congressional voting records database (vote):** Con un total de 435 instancias este conjunto de datos incluye votos para los representantes al congreso de los Estados Unidos. A través de 16 atributos nominales se debe decidir si los votantes van a votar a los demócratas o a los republicanos.

- **Waveform database generator (waveform-21):** Contiene un total de 300 instancias de aprendizaje y 4700 de validación. El conjunto de datos generados consiste en 21 atributos con valores continuos y la clase a predecir con tres posibles valores. Cada clase es generada por una combinación de dos o tres ondas básicas.

5.2. Metodología experimental

Debido a que esta tesis está centrada en el análisis de atributos discretos, todos los conjuntos de datos utilizados han sido discretizados de la forma descrita en [Tin94]. Esta etapa de preprocesamiento de los datos ha sido llevada a cabo con la biblioteca *MLC++* [KJL+94].

Todos los algoritmos desarrollados en este capítulo son validados con los 21 conjuntos de datos antes descritos en la tabla 5.1.

Dos de estos conjuntos de datos están perfectamente definidos con un conjunto de aprendizaje y un conjunto de validación. En estos casos se actuará tal y como se puede observar en la figura 5.1. Los algoritmos de búsqueda buscarán el mejor individuo teniendo en cuenta sólo el conjunto de entrenamiento. Por tanto, cada uno de los individuos será validado con la técnica *leave-one-out* ó *10-fold cross-validation* sobre el conjunto de entrenamiento. Una vez obtenido el mejor individuo, éste será validado con el conjunto de validación.

En los 19 conjuntos de datos restantes se tiene un sólo conjunto. En este caso será utilizada la técnica de validación *leave-one-out* ó *10-fold cross-validation* según las capacidades de computación lo permitan. Este procedimiento se puede observar en la figura 5.2.

Por último, y a lo largo de toda la tesis, se mostrarán tablas o figuras de comparación de los algoritmos desarrollados cuando se considere necesario. De forma complementaria a estas tablas, en muchas ocasiones, se mostrarán los resultados del test no paramétrico de Mann-Whitney para comprobar la hipótesis nula de la misma distribución de densidad para los resultados de los diferentes métodos. Por último, también serán utilizadas las curvas ROC para la comparación de dos algoritmos determinados.

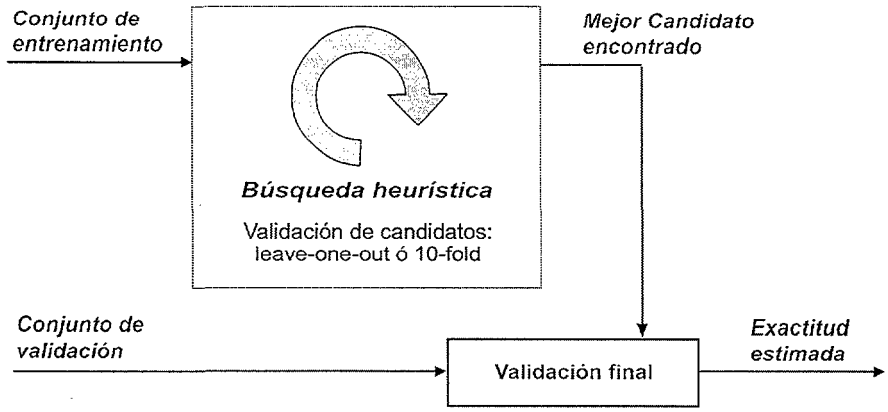


Figura 5.1: Validación de los clasificadores cuando el conjunto de datos está dividido en conjunto de aprendizaje y conjunto de validación

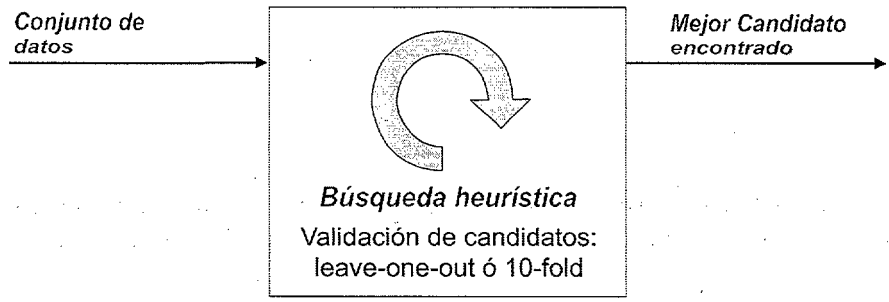


Figura 5.2: Validación de los clasificadores con *leave-one-out* o *10-fold cross-validation*

5.3. *Interval Estimation naïve-Bayes* – IENB

5.3.1. Propuesta

Se propone desarrollar un nuevo algoritmo de clasificación basado en el naïve-Bayes. Este algoritmo se puede encuadrar dentro de las variantes semi naïve-Bayes que corrigen las probabilidades producidas por dicho clasificador.

La idea de desarrollar este algoritmo esta motivada básicamente por tres razones:

- Las variantes semi naïve-Bayes que corrigen las probabilidades producidas por naïve-Bayes.
- La inferencia estadística, que propone el uso de estimaciones por intervalo frente a las estimaciones puntuales.
- El uso de algoritmos heurísticos de optimización para realizar la búsqueda del mejor clasificador dentro de un espacio de búsqueda predeterminado.

Como se vio en el capítulo 3, existen diversas variantes semi naïve-Bayes que corrigen las probabilidades producidas por el clasificador naïve-Bayes. Por ejemplo, el algoritmo

Iterative Bayes [Gam00] tiene como objetivo principal mejorar de forma iterativa las probabilidades condicionadas usadas en el modelo naïve-Bayes. La idea fundamental que se puede extraer de este algoritmo es la de buscar un método que, de forma efectiva, corrija las probabilidades de naïve-Bayes con el fin de mejorar su exactitud.

La primera pregunta que cabría responder es ¿dónde buscar estas probabilidades? La inferencia estadística proporciona una posible respuesta, ya que siempre se ha propuesto como una mejor solución realizar estimaciones por intervalo en lugar de estimaciones puntuales. Al realizar una estimación por intervalo se consigue un intervalo de confianza para un determinado parámetro, de forma que es posible asegurar que el parámetro está dentro del intervalo con un nivel de confianza específico previamente determinado.

La segunda y última pregunta a responder es ¿cómo encontrar las mejores probabilidades? En este caso la respuesta está en los algoritmos heurísticos de optimización, ya que es posible realizar una búsqueda de la mejor combinación de valores dentro de los intervalos antes obtenidos, con el objetivo de maximizar la exactitud del clasificador.

Por tanto, y a modo de resumen, en el nuevo algoritmo propuesto, en lugar de estimar probabilidades puntuales de los datos, como se hace en naïve-Bayes, se realizan estimaciones por intervalo. A continuación, a través de algoritmos de optimización heurísticos se consigue la combinación de valores dentro de los intervalos que maximiza el tanto por ciento de bien clasificados. Estos valores serán las nuevas y mejoradas probabilidades que se usarán en naïve-Bayes.

Para llevar a cabo esta búsqueda se está utilizando un enfoque de tipo *wrapper* [Koh95], ya que el algoritmo de inducción se considera como una caja negra que se utiliza para evaluar cada posible candidato.

Este algoritmo, que se detalla a continuación, ha sido denominado *Interval Estimation naïve-Bayes* (IENB) [RLP+03b, RLP+03a, RLM+03].

El resto de la sección tiene el siguiente contenido:

- A continuación, se realiza una introducción a la inferencia estadística, explicando los métodos de estimación puntual y de estimación por intervalo.
- En segundo lugar, se justificará la estimación puntual de los parámetros en el clasificador naïve-Bayes y cómo se deben realizar las estimaciones por intervalo de estos mismos parámetros.
- En el siguiente apartado se describe en profundidad el algoritmo desarrollado, IENB.
- Por último, se presentan los resultados experimentales sobre los 21 conjuntos de datos de UCI seleccionados y se presentan las conclusiones y las líneas de trabajo futuro.

5.3.2. Estimación por intervalos

La inferencia estadística es la parte de la estadística que estudia grandes colectivos a partir de una muestra de estos, siendo la muestra la parte de la población en la que se apoya para realizar el análisis.

Los dos problemas fundamentales que estudia la inferencia estadística son el "problema de la estimación" y el "problema del constraste de hipótesis". Esta sección se centra única

y exclusivamente en los problemas de estimación.

La estimación de un parámetro involucra el uso de los datos muestrales en conjunción con alguna estadística. Existen dos formas de llevar a cabo lo anterior: estimación puntual y estimación por intervalo. En la primera se busca un estimador, que, con base en los datos muestrales, dé origen a una estimación del valor del parámetro, y que recibe el nombre de estimador puntual. Para la segunda, se determina un intervalo en el que, en forma probable, se encuentra el valor del parámetro. Este intervalo recibe el nombre de intervalo de confianza estimado.

Se denomina **estimador de un parámetro** θ, a cualquier variable aleatoria $\theta^*(X_1, \ldots, X_n)$ que se expresa en función de una muestra aleatoria simple de tamaño n, $X_1, \ldots, X_n$, y que tiene por objetivo aproximar el valor de θ.

Es importante observar que el estimador **no es un valor concreto** sino una variable aleatoria, ya que aunque depende unívocamente de los valores de la muestra observados, la elección de la muestra es un proceso aleatorio. Una vez que la muestra ha sido elegida, se denomina **estimación** al valor numérico que toma el estimador en esa muestra.

Las características deseables para esta nueva variable aleatoria (que se usará para estimar el parámetro desconocido) deben ser:

- **Consistencia:** Cuando el tamaño de la muestra crece arbitrariamente, el valor estimado se aproxima al parámetro desconocido.
- **Carencia de sesgo:** El valor medio que se obtiene de la estimación para diferentes muestras debe ser el valor del parámetro.
- **Eficiencia:** Se busca que la dispersión del estimador con respecto al valor central, es decir, la varianza, sea tan pequeña como sea posible.
- **Suficiencia:** El estimador debería aprovechar toda la información existente en la muestra.

Métodos de estimación puntual

El objetivo de la estimación puntual es emplear una muestra para calcular un número que represente, en algún sentido, una buena presunción para el verdadero valor del parámetro. El número resultante se llama estimación puntual.

Los métodos de estimación puntual estudian cómo obtener estimadores que, de forma general, tengan buenas propiedades. Específicamente se consideran los métodos de máxima verosimilitud y el de los momentos. A continuación se introduce únicamente el método de máxima verosimilitud, ya que es el utilizado por los clasificadores supervisados basados en redes Bayesianas.

Estimación por máxima verosimilitud

Sea X una variable aleatoria con función de probabilidad

$$f(x;\theta) \tag{5.1}$$

Una muestra aleatoria simple de tamaño n, $X_1, X_2, \ldots, X_n$, tiene por distribución de probabilidad conjunta

$$V(\theta) = f(x_1, x_2, \ldots, x_n; \theta) \tag{5.2}$$

Esta función que depende de $n + 1$ parámetros puede ser considerada, fijando los $x_1, \ldots, x_n$ como consecuencia de los resultados de elegir una muestra mediante un experimento aleatorio, únicamente función de θ. A esta función de θ se le denomina función de verosimilitud asociada a la muestra.

Se puede plantear el que, dada una muestra sobre la que se han observado los valores $x_1, \ldots, x_n$, una estimación del parámetro es aquella que maximiza la función de verosimilitud.

Como es lo mismo maximizar una función que su logaritmo (al ser este una función estrictamente creciente), este máximo puede calcularse derivando con respecto a θ la función de verosimilitud (o bien su logaritmo) y tomando como estimador de máxima verosimilitud el que haga la derivada igual a cero,

$$\left.\frac{\partial \log V(\theta)}{\partial \theta}\right|_{\theta=\hat{\theta}} = 0 \tag{5.3}$$

De modo más preciso se define el **estimador de máxima verosimilitud** como la variable aleatoria

$$\hat{\theta} = \max_{\theta} f(X_1, X_2, \ldots, X_n; \theta) \tag{5.4}$$

Estimación por intervalo

La estimación por intervalo consiste en determinar un posible rango de valores o intervalo, en los que se pueda precisar –con una determinada probabilidad– que el valor de un parámetro se encuentra dentro de esos límites. Este parámetro será habitualmente una probabilidad de éxito en el caso de variables dicotómicas o la media (esperanza) y la varianza para variables Gaussianas.

La técnica de estimación por intervalos consiste en asociar a cada muestra un intervalo que se sospecha que debe contener al parámetro. A éste se le denomina intervalo de confianza.

Evidentemente, esta técnica no tiene por qué dar siempre un resultado correcto. A la probabilidad de que se haya acertado al decir que un parámetro estaba contenido en un intervalo se le denomina nivel de confianza.

5.3.3. Estimación de los parámetros en naïve-Bayes e IENB

Esta sección contiene un análisis formal de cómo se estiman puntualmente los parámetros en naïve-Bayes y los intervalos en IENB.

Estimación puntual de los parámetros en naïve-Bayes

En este apartado se detalla formalmente cómo aprender los parámetros en un modelo de datos naïve-Bayes. Los resultados obtenidos son intuitivos y, de hecho, ya se hizo uso de ellos en otros capítulos previos.

Se considera un conjunto de datos $\mathcal{D} = \{(\mathbf{x}^{(1)}, c^{(1)}), \ldots, (\mathbf{x}^{(N)}, c^{(N)})\}$ de atributos binarios. Es decir $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$ con $x_k^i \in \{0, 1\}$. Cada instancia $\mathbf{x}^{(i)}$ tiene asociada una etiqueta de clase $c^{(i)}$. Basándose en la etiqueta de clase es posible dividir las entradas en aquellas que pertenecen a una determinada clase: $\mathcal{D}_i = \{(\mathbf{x}^{(j)}, c^{(j)}) \in \mathcal{D} | c^{(j)} = i\}$. Se considera sólo el uso de dos clases (este caso es el denominado proceso de Bernoulli –el caso de más clases es también directo y se denomina proceso multinomial–).

Para cada una de las clases se deben estimar los valores $P(X_k = 1|C = c_i) \equiv \theta_k^i$. La otra probabilidad, $P(X_k = 0|C = c_i)$ viene dada por el requisito de normalización $P(X_k = 0|C = c_i) = 1 - P(X_k = 1|C = c_i) = 1 - \theta_k^i$.

Utilizando la suposición de independencia de las variables predictoras dada la clase, la probabilidad de que una instancia pertenezca a una determinada clase es

$$P(X_1 = x_1, \ldots, X_n = x_n|C = c_i) = \prod_{k=1}^{n} P(X_k = x_k|C = c_i) = \prod_{k=1}^{n} (\theta_k^i)^{x_k} (1-\theta_k^i)^{1-x_k} \quad (5.5)$$

Hay que recordar que en cada término del producto anterior, x_k es cero o uno, por tanto, sólo uno de los dos factores contribuirá. Si $x_k = 1$ contribuirá el factor θ_k^i y si $x_k = 0$ contribuirá el factor $1 - \theta_k^i$.

Utilizando la asunción estándar de que los datos son generados de forma idéntica e independiente, la función de verosimilitud para el conjunto $\mathcal{D}_i$ queda

$$V_{\mathcal{D}_i}((x^{(1)}, c^{(1)}), \ldots, (x^{(N)}, c^{(N)})) = \prod_{j=1}^{N_i} \left(\prod_{k=1}^{n} (\theta_k^i)^{x_k^j} (1 - \theta_k^i)^{1-x_k^j} \right) \quad (5.6)$$

donde N_i es el número de casos en que $C = c_i$

Tomando logaritmos se obtiene,

$$\log V_{\mathcal{D}_i} = \sum_{j=1}^{N_i} \sum_{k=1}^{n} x_k^j \log(\theta_k^i) + \sum_{j=1}^{N_i} \sum_{k=1}^{n} (1 - x_k^j) \log(1 - \theta_k^i) \quad (5.7)$$

Para $k = 1, \ldots, n$, derivando respecto a θ_k^i el logaritmo de la función de verosimilitud e igualando a cero,

$$\left. \frac{\partial \log V_{\mathcal{D}_i}}{\partial \theta_k^i} \right|_{\theta_k^i = \hat{\theta}_k^i} = \sum_{j=1}^{N_i} \frac{x_k^j}{\hat{\theta}_k^i} - \sum_{j=1}^{N_i} \frac{1 - x_k^j}{1 - \hat{\theta}_k^i} = 0 \quad (5.8)$$

Por lo que se llega al siguiente resultado

$$\hat{P}(X_k = 1|C = c_i) \equiv \hat{\theta}_k^i = \frac{\sum_{j=1}^{N_i} x_k^j}{N_i} \quad (5.9)$$

Con un desarrollo muy similar de estimación por máxima verosimilitud se obtiene

$$\hat{P}(C = c_i) = \frac{N_i}{N} \tag{5.10}$$

Estimación por intervalos de los parámetros en IENB

Cuando se tiene una variable dicotómica (o de Bernoulli) a menudo interesa saber en qué proporción de casos, p, ocurre el éxito en la realización de un experimento. En este caso, es posible realizar el cálculo del intervalo de confianza de p.

Sean $X_1, X_2, \ldots, X_M \rightsquigarrow Ber(p)$. La manera más natural de estimar el parámetro p consiste en definir la suma de éstas –lo que proporciona una distribución Binomial–.

$$X = X_1 + \ldots + X_M \rightsquigarrow \mathcal{B}(M, p) \tag{5.11}$$

y tomar como su estimador la variable aleatoria

$$\hat{p} = \frac{X}{M}. \tag{5.12}$$

Es decir, se toma como estimación de p la proporción de éxitos obtenidos en las M pruebas.

La distribución del número de éxitos es binomial, y puede ser aproximada a la normal cuando el tamaño de muestra M es grande, y p no es una cantidad muy cercana a cero o uno. Es decir,

$$X \rightsquigarrow B(M, p) \Rightarrow X \overset{\approx}{\rightsquigarrow} \mathcal{N}(Mp, Mp(1-p)) \tag{5.13}$$

El estimador $\hat{p}$ no es más que un cambio de escala de X, por tanto,

$$\frac{\hat{p} - p}{\sqrt{\frac{p(1-p)}{M}}} \approx \mathcal{Z} \rightsquigarrow \mathcal{N}(0, 1) \tag{5.14}$$

Al ser p desconocido, esta expresión presenta dificultades para el cálculo, siendo más cómodo sustituirla por la siguiente expresión, la cual proporciona una aproximación de la anterior.

$$\frac{\hat{p} - p}{\sqrt{\frac{\hat{p}(1-\hat{p})}{M}}} \approx \mathcal{Z} \rightsquigarrow \mathcal{N}(0, 1) \tag{5.15}$$

Para encontrar un intervalo de confianza para p al nivel de significación $1 - \alpha$ se considera el intervalo que hace que la distribución de $\mathcal{Z} \rightsquigarrow \mathcal{N}(0, 1)$ deje la probabilidad α fuera del mismo. Esto puede hacerse de infinitas maneras, siendo el intervalo con nivel de significación $1 - \alpha$ de menor longitud, aquel cuyos extremos son los cuantiles $\alpha/2$ y $1 - \alpha/2$. Así, se puede afirmar con una confianza $1 - \alpha$ que,

$$|Z| \leq z_\alpha \tag{5.16}$$

es decir,

$$\frac{|\hat{p}-p|}{\sqrt{\frac{\hat{p}(1-\hat{p})}{M}}} \leq z_\alpha \tag{5.17}$$

y por tanto,

$$|\hat{p}-p| \leq z_\alpha \sqrt{\frac{\hat{p}(1-\hat{p})}{M}} \tag{5.18}$$

que se resume en,

$$p \in \left(\hat{p} - z_\alpha \sqrt{\frac{\hat{p}(1-\hat{p})}{M}}; \hat{p} + z_\alpha \sqrt{\frac{\hat{p}(1-\hat{p})}{M}}\right) \tag{5.19}$$

donde z_α es el $(1-\frac{\alpha}{2})$ percentil de la $\mathcal{N}(0,1)$

5.3.4. Algoritmo

En la figura 5.3 se encuentra una comparación entre naïve-Bayes e *Interval Estimation naïve-Bayes.*

El algoritmo de inducción naïve-Bayes se basa en la estimación puntual de determinados parámetros del conjunto de datos. En la sección 5.3.3 se encuentra la demostración formal de la obtención de estos parámetros. En contraposición a esta aproximación, el algoritmo IENB se basa en la estimación de intervalos de confianza (véase la sección 5.3.3) de los parámetros necesarios.

Tal y como se puede observar en la figura 5.3, en el algoritmo IENB, una vez obtenidos los intervalos, y seleccionando al azar un valor dentro de cada uno de ellos, es posible generar tantas combinaciones de probabilidades como se desee. Estas combinaciones de probabilidades, bajo unas determinadas condiciones, representan clasificadores naïve-Bayes válidos.

El último paso que queda por realizar es la búsqueda de la mejor de las combinaciones de probabilidades, es decir, la búsqueda del mejor clasificador naïve-Bayes. Para realizar esta búsqueda se utilizarán algoritmos heurísticos de optimización. En el caso de esta tesis los algoritmos utilizados son los EDAs.

El objetivo que se busca con este algoritmo es doble:

- Se está realizando una búsqueda de la mejor combinación de probabilidades posible. En esta búsqueda se están teniendo en cuenta todas las probabilidades de forma global, por lo que se está relajando la presunción de independencia de las variables predictoras que impone naïve-Bayes.
- Al relajar la independencia de las variables predictoras y ajustar mejor las probabilidades estimadas, se mejora la exactitud del naïve-Bayes.

Estas ventajas conllevan también un inconveniente que se debe mencionar, ya que el tiempo de aprendizaje aumenta considerablemente debido a la realización de la búsqueda

heurística. En las situaciones en las que lo más importante sea el tiempo de aprendizaje, este algoritmo será de difícil utilización, y viceversa, en las situaciones en las que lo importante sea la exactitud, este algoritmo es ideal debido a los buenos resultados que presenta.

Sin embargo, la paralelización del algoritmo, tal y como se verá en la sección 5.4, lleva a una disminución considerable del tiempo requerido para su ejecución.

Tal y como se puede observar en la figura 5.3 hay tres aspectos principales a tener en cuenta en el algoritmo IENB: cálculo de los intervalos de confianza, definición del espacio de búsqueda y búsqueda heurística del mejor individuo. A continuación se describe cada uno de estos aspectos.

Cálculo de los intervalos de confianza

Dado el conjunto de datos, el primer paso a realizar consiste en calcular los intervalos de confianza de cada probabilidad condicional y de las probabilidades de los valores de la clase. Para realizar el cálculo de estos intervalos se deben calcular previamente las estimaciones puntuales de estos mismos parámetros (véase la sección 5.3.3).

De esta forma, cada probabilidad condicional, obtenida del conjunto de datos $\hat{p}^i_{k,r} = \hat{P}(X_k = x^r_k | C = c_i)$, debe ser estimada con el correspondiente intervalo de confianza, tal y como se demostró en la sección 5.3.3. La contribución de este algoritmo se fundamenta en este cálculo.

Para $k = 1, \ldots, n; i = 1, \ldots, r_0; r = 1, \ldots, r_k$

$$\left(\hat{p}^i_{k,r} - z_\alpha \sqrt{\frac{\hat{p}^i_{k,r}(1 - \hat{p}^i_{k,r})}{N_i}} ; \hat{p}^i_{k,r} + z_\alpha \sqrt{\frac{\hat{p}^i_{k,r}(1 - \hat{p}^i_{k,r})}{N_i}} \right) \tag{5.20}$$

denota las estimaciones por intervalo de las probabilidades condicionadas $p^i_{k,r}$, donde,

r son los posibles valores de la variable X_k,

i son los posibles valores de la clase,

$\hat{p}^i_{k,r}$ es la estimación puntual de las probabilidades condicionales $P(X_k = x^r_k | C = c_i)$,

z_α es el $(1 - \frac{\alpha}{2})$ percentil en la $\mathcal{N}(0,1)$.

Además, de forma similar, las probabilidades para los valores de la variable clase $\hat{p}_i = \hat{P}(C = c_i)$ deben ser estimadas con el siguiente intervalo de confianza,

$$\left(\hat{p}_i - z_\alpha \sqrt{\frac{\hat{p}_i(1 - \hat{p}_i)}{N}} ; \hat{p}_i + z_\alpha \sqrt{\frac{\hat{p}_i(1 - \hat{p}_i)}{N}} \right) \tag{5.21}$$

donde,

$\hat{p}^i_i$ es la estimación puntual de la probabilidad $P(C = c_i)$,

z_α es el $(1 - \frac{\alpha}{2})$ percentil en la $\mathcal{N}(0,1)$,

N es el número de instancias en el conjunto de datos.

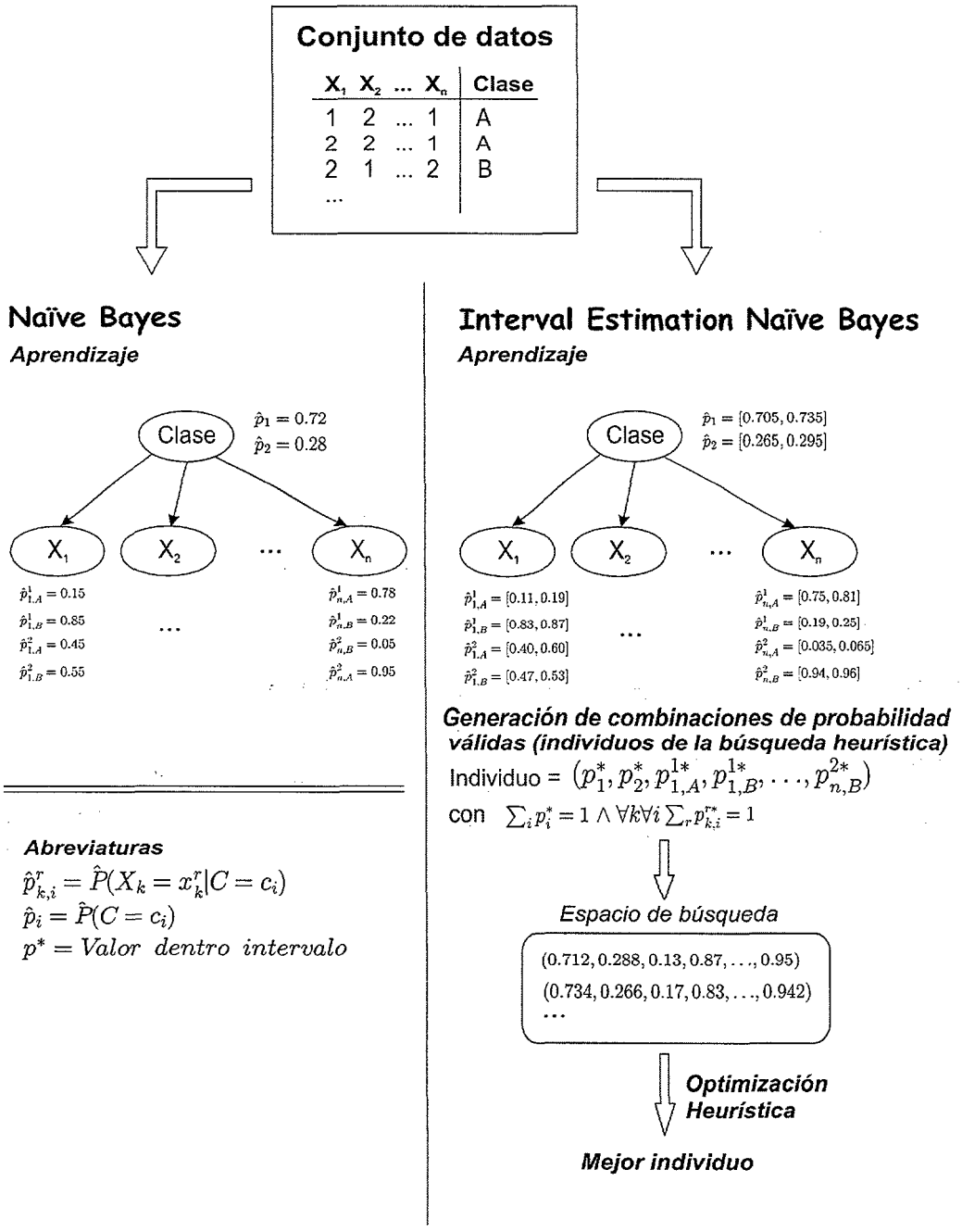


Figura 5.3: Comparación de naïve-Bayes e *Interval Estimation naïve-Bayes*

Definición del espacio de búsqueda

Una vez estimados los intervalos de confianza desde el conjunto de entrenamiento, es posible generar tantos clasificadores naïve-Bayes como se desee. Simplemente se debe escoger un determinado valor dentro de cada uno de los intervalos de confianza antes

estimados.

De esta forma, cada posible clasificador naïve-Bayes se va a representar con la siguiente tupla,

$$(p_1^*, \ldots, p_{r_0}^*, p_{1,1}^{*1}, \ldots, p_{1,1}^{*r_0}, \ldots, p_{1,r_1}^{*r_0}, \ldots, p_{n,r_n}^{*r_0}) \tag{5.22}$$

donde cada componente de la tupla p^* es el valor seleccionado dentro de su correspondiente intervalo de confianza, al que se le ha efectuado la correspondiente normalización.

El espacio de búsqueda para el algoritmo de optimización heurístico, que a continuación debe ser ejecutado, estará formado por todas las tuplas válidas (que serán los individuos para la búsqueda). Para que una tupla sea válida debe representar a un clasificador naïve-Bayes correcto, para lo que se debe cumplir el siguiente requisito:

$$\sum_{i=1}^{r_0} p_i^* = 1; \forall k \forall i \sum_{r=1}^{r_k} p_{k,r}^{*i} = 1 \tag{5.23}$$

Por último, hay que destacar que cada individuo debe ser evaluado con una función *fitness* (evaluación). La función de evaluación que se usará a lo largo de toda la tesis es el tanto por ciento de bien clasificados que se obtiene con dicho clasificador (individuo). Esto implica que se está llevando a cabo un enfoque de tipo *wrapper*. Para obtener este valor se realiza una validación *leave-one-out* de cada individuo.

Búsqueda heurística del mejor individuo

Una vez definidos los individuos y el espacio de búsqueda sólo queda ejecutar el algoritmo heurístico de optimización para realizar la búsqueda del mejor individuo posible. Como se ha comentado previamente se utilizarán los EDAs como algoritmos heurísticos de optimización.

5.3.5. Ejemplo

En la figura 5.4 se puede observar el resultado de la ejecución del algoritmo IENB sobre el conjunto de datos *pima*, utilizando un valor de z_α de $1,96$. En dicha figura se pueden observar todos los intervalos de confianza necesarios para este conjunto de datos. Los dos primeros intervalos son los pertenecientes a las dos posibles clases, mientras que el resto de los intervalos pertenecen a las diferentes probabilidades condicionadas necesarias. Lo que siempre se cumple en estos intervalos es que el valor central de los mismos es la probabilidad que utilizaría el clasificador naïve-Bayes estándar.

En relación a los intervalos, y debido a su forma de cálculo, cuanto más cerca se está de la incertidumbre $(0,5)$, más grande se hará el intervalo, y viceversa. Además, el tamaño de los intervalos es inversamente proporcional al número de instancias involucradas en su cálculo. En este ejemplo hay un total de 766 instancias, 266 de una clase y 500 de la otra.

En la gráfica de la figura, además de los intervalos, se encuentra señalado el valor seleccionado después de la ejecución del algoritmo heurístico de optimización. En esta ejecución el algoritmo IENB ha obtenido un resultado de 79,79 % de bien clasificados,

mientras que el resultado de naïve-Bayes sobre este mismo conjunto de datos es de 77,73 % de bien clasificados.

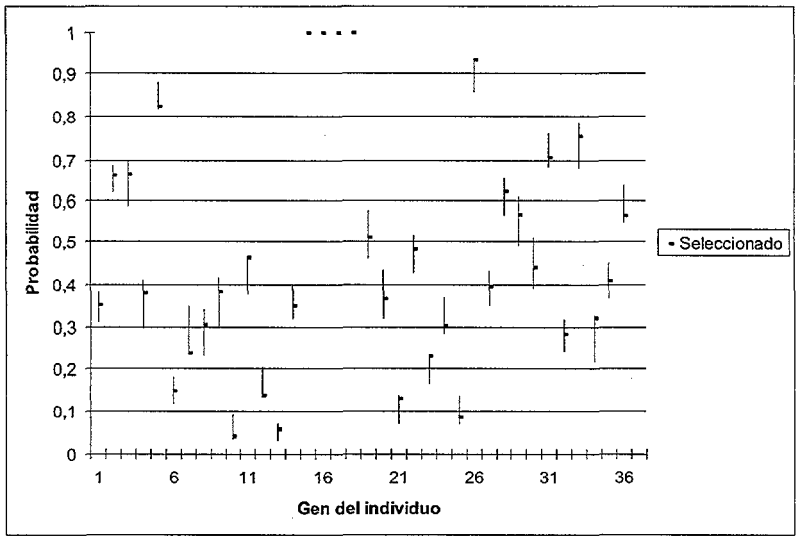


Figura 5.4: Intervalos de confianza y valores finales seleccionados por IENB para el conjunto de datos *pima*

5.3.6. Resultados

Resultados experimentales

Las pruebas realizadas con IENB se ejecutaron sobre un Athlon 2000+ con 1GByte de memoria RAM. Los parámetros utilizados para la ejecución de los EDAs fueron: tamaño de población 500 individuos, individuos seleccionados para aprendizaje 500, número de nuevos individuos cada generación (*offspring*) 1000, tipo de aprendizaje UMDA (*Univariate Marginal Distribution Algorithm*) [Müh98] y elitismo. Cada prueba ha sido ejecutada 10 veces con dos valores de percentil diferentes: 95 ($z_\alpha = 1{,}96$) y 98 ($z_\alpha = 2{,}33$).

Los resultados obtenidos se muestran en la tabla 5.3.6 para el percentil 95 y en la tabla 5.3 para el percentil 98. En ambas tablas las columnas tienen, por orden, el siguiente contenido: el tanto por ciento de bien clasificados obtenido con el clasificador naïve-Bayes, la media y la desviación típica del tanto por ciento de bien clasificados obtenido con IENB (el símbolo † indica que la diferencia entre naïve-Bayes e IENB es estadísticamente significativa), número medio de evaluaciones necesarias para obtener el resultado y mejora obtenida respecto al clasificador naïve-Bayes.

Los resultados obtenidos son muy satisfactorios, ya que, como se puede observar, el algoritmo IENB es claramente superior a naïve-Bayes. Para el percentil 95 se ha obtenido una mejora media de 4,65 % respecto al porcentaje de bien clasificados, mientras que con el percentil 98 esta mejora aumenta a 4,93 %.

Además, se ha utilizado el test no paramétrico de Mann-Whitney para comprobar la hipótesis nula de la misma distribución de densidad para cada uno de los conjuntos

Interval Estimation naïve-Bayes (Percentil 95)				
	Naïve-Bayes	*IENB 95*	*Num. eval.*	*Mejora*
breast	97,14	97,71 ± 0,00 †	8600	0,57
chess	87,92	93,31 ± 0,10 †	109000	5,39
cleve	83,82	86,29 ± 0,17 †	28200	2,47
corral	84,37	93,70 ± 0,00 †	43300	9,33
crx	86,23	89,64 ± 0,10 †	63100	3,41
flare	80,86	82,69 ± 0,13 †	16000	1,83
german	75,40	81,57 ± 0,10 †	188900	6,17
glass	74,77	83,00 ± 0,20 †	33000	8,23
glass2	82,21	88,27 ± 0,00 †	5500	6,06
hepatitis	85,16	92,60 ± 0,34 †	23400	7,44
iris	94,67	95,97 ± 0,00 †	5500	1,30
lymphography	85,14	94,56 ± 0,00 †	49700	9,42
mofn-3-7-10	86,33	95,31 ± 0,00 †	129100	8,98
pima	77,73	79,84 ± 0,09 †	19300	2,11
satimage	82,46	83,88 ± 0,32 †	93500	1,42
segment	91,95	96,38 ± 0,08 †	145100	4,44
shuttle-small	99,36	99,90 ± 0,00 †	16600	0,54
soybean-large	92,83	95,67 ± 0,10 †	61000	2,84
vehicle	61,47	71,16 ± 0,25 †	129100	9,69
vote	90,11	95,07 ± 0,19 †	28100	4,96
waveform-21	78,85	79,81 ± 0,10 †	28700	0,96

Tabla 5.2: Resultados experimentales de *Interval Estimation naïve-Bayes* (Percentil 95)

de prueba. Esta tarea se realizó con el paquete estadístico S.P.S.S. versión 11.50. Los resultados de estas pruebas se muestran a continuación.

- Pruebas del clasificador IENB:
  - Naïve-Bayes vs. IENB 95. Valor de ajuste en todos los conjuntos de datos: $p <$ 0,001.
  - Naïve-Bayes vs. IENB 98. Valor de ajuste en todos los conjuntos de datos: $p <$ 0,001.
  - IENB 95 vs. IENB 98. Valor de ajuste $p <$ 0,001 sólo en siete conjuntos de datos: *chess*, *cleve*, *crx*, *flare*, *satimage*, *segment* y *vehicle*.

Estos resultados demuestran que la diferencia entre naïve-Bayes e IENB es significativa en todos los conjuntos de datos, independientemente del valor del percentil utilizado en el clasificador IENB. Sin embargo, sólo en siete de los 21 conjuntos de datos hay diferencia significativa entre IENB con percentil 95 e IENB con percentil 98.

Respecto a *Iterative Bayes* [Gam00], el enfoque semi naïve-Bayes más similar a IENB, la comparación también muestra una superioridad de IENB. En la tabla 5.4 se puede ver esta comparación. El símbolo † indica que la diferencia respecto a naïve-Bayes es estadísticamente significativa. Excepto en los conjuntos de datos *iris* y *satimage*, IENB se muestra claramente superior.

Interval Estimation naïve-Bayes (Percentil 98)				
	Naïve-Bayes	*IENB 98*	*Num. eval.*	*Mejora*
breast	97,14	97,71 ± 0,00 †	7600	0,57
chess	87,92	93,87 ± 0,04 †	119300	5,95
cleve	83,82	86,92 ± 0,23 †	45200	3,10
corral	84,37	93,70 ± 0,00 †	56000	9,33
crx	86,23	90,11 ± 0,11 †	101100	3,88
flare	80,86	83,47 ± 0,00 †	51400	2,61
german	75,40	81,82 ± 0,12 †	188100	6,42
glass	74,77	83,00 ± 0,23 †	36400	8,19
glass2	82,21	88,27 ± 0,00 †	6700	6,06
hepatitis	85,16	92,99 ± 0,27 †	23100	7,83
iris	94,67	95,97 ± 0,00 †	4500	1,30
lymphography	85,14	94,83 ± 0,22 †	46100	9,69
mofn-3-7-10	86,33	95,31 ± 0,00 †	20500	8,98
pima	77,73	80,09 ± 0,06 †	28300	2,36
satimage	82,46	84,38 ± 0,35 †	123500	1,92
segment	91,95	96,75 ± 0,12 †	181500	4,80
shuttle-small	99,36	99,90 ± 0,00 †	19300	0,54
soybean-large	92,83	95,67 ± 0,10 †	73500	2,84
vehicle	61,47	72,41 ± 0,35 †	146400	10,94
vote	90,11	95,16 ± 0,17 †	20500	5,05
waveform-21	78,85	80,09 ± 0,11 †	35800	1,24

Tabla 5.3: Resultados experimentales de *Interval Estimation naïve-Bayes* (Percentil 98)

La calidad de los clasificadores también ha sido comparada a través de las curvas ROC. Así, por ejemplo, para el conjunto de datos *corral*, se obtienen las curvas de la figura 5.5. Estas curvas ROC se han obtenido y analizado con el programa *Analized-it*. Las áreas obtenidas bajo las curvas ROC son: 0,921 para el clasificador naïve-Bayes, 0,944 para el clasificador obtenido con IENB con percentil 95 y 0,960 para el clasificador obtenido con IENB con percentil 98. Estos datos indican que el mejor de los clasificadores es el obtenido con IENB con percentil 98.

Conclusiones

En esta sección se ha propuesto un nuevo algoritmo semi naïve-Bayes denominado *Interval Estimation naïve-Bayes* (IENB). Este algoritmo se puede encuadrar dentro de las variantes semi naïve-Bayes que corrigen las probabilidades producidas por dicho clasificador.

En IENB, en lugar de estimar probabilidades puntuales de los datos, como se hace en naïve-Bayes, se realizan estimaciones por intervalo. A continuación, a través de algoritmos de optimización heurísticos, se consigue la combinación de valores dentro de los intervalos que maximiza el tanto por ciento de bien clasificados.

Los resultados obtenidos son excelentes, tanto si se comparan con naïve-Bayes, como si se comparan con enfoques semi naïve-Bayes similares como *Iterative Bayes*.

Comparación: IENB vs Iterative Bayes			
Conjunto de datos	*Mejora IENB 95*	*Mejora IENB 98*	*Mejora Iterative Bayes*
breast	0,57 †	0,57 †	-0,16
chess	5,39 †	5,95 †	-
cleve	2,47 †	3,10 †	0,10
corral	9,33 †	9,33 †	-
crx	3,41 †	3,88 †	-
flare	1,83 †	2,61 †	-
german	6,17 †	6,42 †	-0,05
glass	8,23 †	8,19 †	-1,19
glass2	6,06 †	6,06 †	-
hepatitis	7,44 †	7,83 †	1,41
iris	1,30 †	1,30 †	1,40 †
lymphography	9,42 †	9,69 †	-
mofn-3-7-10	8,98 †	8,98 †	-
pima	2,11 †	2,36 †	-
satimage	1,42 †	1,92 †	3,59 †
segment	4,43 †	4,80 †	1,50 †
shuttle-small	0,54 †	0,54 †	-
soybean-large	2,84 †	2,84 †	-
vehicle	9,69 †	10,94 †	4,39 †
vote	4,96 †	5,05 †	1,45 †
waveform-21	0,96 †	1,24 †	-

Tabla 5.4: Comparación de los resultados obtenidos con IENB e *Iterative Bayes*

Líneas de trabajo futuro

IENB es un algoritmo de nueva creación, por lo que quedan muchos campos abiertos para futuros desarrollos:

- Adaptación del clasificador IENB para su uso con variables continuas.
- Realización de implementaciones de IENB que cambien la función objetivo del tanto por ciento de bien clasificados. Por ejemplo, se podría maximizar el área bajo la curva ROC del clasificador.
- Otra posible idea, sería realizar una selección de variables previa a la ejecución de IENB.

5.4. Parallel *Interval Estimation naïve-Bayes* – PIENB

5.4.1. Propuesta

El nuevo algoritmo de clasificación propuesto, *Interval Estimation naïve-Bayes* (IENB), puede llegar a ser muy costoso desde el punto de vista computacional, cuando existen gran cantidad de variables predictoras o cuando el conjunto de instancias utilizado para el aprendizaje es muy alto. Por ejemplo, con el conjunto de datos *segment* la ejecución

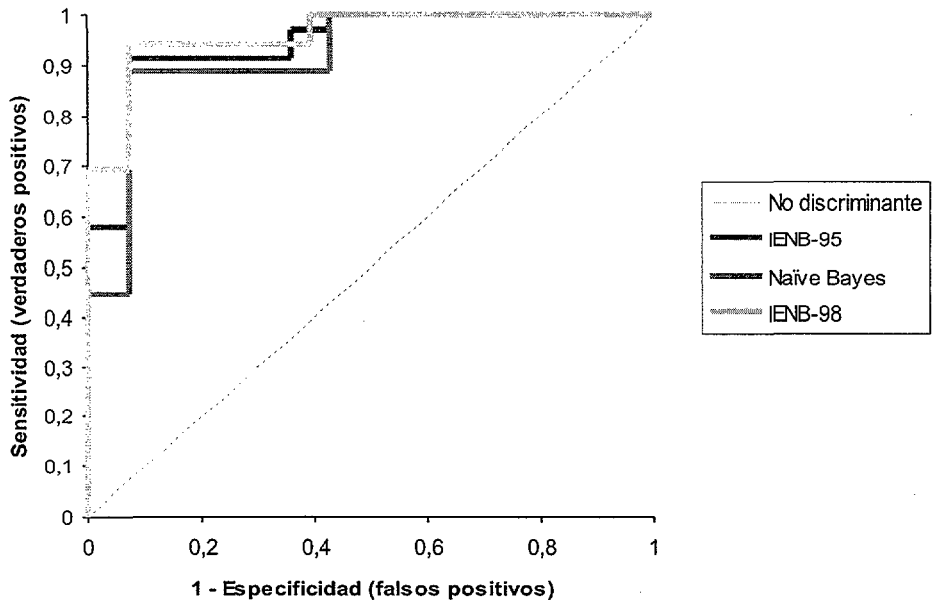


Figura 5.5: Curvas ROC de naïve-Bayes e IENB con percentiles 95 y 98 para el conjunto de datos *corral*

dura unos 692 minutos. En gran medida, este coste computacional es debido a que la parte central del algoritmo es una búsqueda heurística realiza con EDAs.

Como solución al coste computacional de los EDAs, se propone una posible paralelización de los mismos, basada en islas [RPP+03]. Ya existen algunas propuestas de paralelización de los EDAs como la presentada en [LSL02], que trata de mejorar el proceso de aprendizaje de las estructuras de red Bayesiana con técnicas de paralelismo, o la presentada en [Ben02], que paraleliza la creación de los individuos y el aprendizaje de la estructura. Sin embargo, no hay ningún enfoque basado en la paralelización por islas.

Cada isla contendrá una población diferente y, cada cierto tiempo y con un esquema de migración predeterminado, las islas se intercambiarán los mejores individuos entre sí. Este modelo de paralelización ya ha sido utilizado con éxito en los algoritmos genéticos paralelos [CP01, Ste93].

La versión paralela de IENB se denominará *Parallel Interval Estimation naïve-Bayes* (PIENB).

Además de mejorar el rendimiento, con PIENB también se persigue mejorar los resultados obtenidos por IENB. La paralelización por islas suele implicar que se recorran diferentes áreas del espacio de búsqueda, siendo muchas veces posible mejorar las soluciones encontradas por el algoritmo secuencial.

A continuación se analiza el estado del arte de la paralelización de algoritmos genéticos (ya que es el paradigma más cercano a los EDAs y es posible utilizar las mismas ideas de paralelización), se describe el algoritmo PIENB propuesto, y se presentan los resultados experimentales obtenidos con varios conjuntos de datos del repositorio UCI.

5.4.2. Estado del arte de la paralelización de algoritmos genéticos

La idea básica, subyacente en la mayor parte de los programas paralelos, es dividir una tarea grande en tareas más pequeñas, y resolver estas tareas simultáneamente utilizando múltiples procesadores. Este enfoque divide y vencerás, puede ser aplicado a los algoritmos genéticos (AGs) de formas muy diferentes. En la literatura se encuentran numerosos ejemplos de implementaciones paralelas exitosas. Algunos métodos de paralelización utilizan una población única, mientras que otros dividen la población en diversas subpoblaciones relativamente aisladas. Algunos métodos utilizan arquitecturas masivamente paralelas, mientras que otros son más apropiados para multiprocesadores con menos y más potentes elementos de proceso conectados por una red más lenta.

Es posible definir cuatro tipologías diferentes de AGs paralelos [Bel95, CP01, Lev94]:

- AGs maestro-esclavo con una población única.
- AGs multipoblación (modelo de islas).
- AGs de grano fino.
- Híbridos jerárquicos.

Los AGs maestro-esclavo tienen una población única. Un nodo maestro ejecuta el AG (selección, cruce y mutación), y la evaluación de los individuos se distribuye entre diversos procesadores esclavo. Los esclavos evalúan los individuos que reciben del maestro y le devuelven el resultado. Ya que en este tipo de AGs paralelos, el cruce y la mutación consideran a la población completa, también se les conoce como AGs paralelos "globales".

Los AGs con múltiples poblaciones [WRH97, MS00] son más sofisticados y provienen de la observación de la naturaleza. Es posible decir que el mundo de los humanos es una colección de subpoblaciones, que evolucionan de forma independiente en continentes o regiones restringidas. Sin embargo, cada cierto tiempo, algunos individuos de una región migran a otra región, permitiendo a las subpoblaciones compartir material genético. La idea es que entornos aislados, o islas competitivas, son más eficientes en la búsqueda que una única población en la que todos los miembros están juntos.

Aplicando esta idea a los algoritmos genéticos, es posible que cada procesador ejecute su propio algoritmo genético secuencial con su propia subpoblación, pero por supuesto, intentando maximizar la misma función. Si se define sobre las subpoblaciones una estructura de vecinos, y, con una determinada frecuencia, cada subpoblación manda sus mejores individuos a sus vecinos, se dice que se está ejecutando un *algoritmo genético distribuido.*

Como cada procesador comienza con una población inicial diferente, la búsqueda evolutiva dirigirá estas poblaciones en diferentes direcciones. A través de la introducción de la migración, el modelo de islas es capaz de aprovecharse de las diferencias de las subpoblaciones, creándose una fuente de diversidad genética. Migrar un gran número de individuos muy frecuentemente puede anular las diferencias entre las islas, destruyendo de esta forma la diversidad global. Por otra parte, si la migración no es muy frecuente, se puede producir una convergencia prematura de las subpoblaciones. A la hora de trabajar con el modelo de islas se debe determinar:

- El número y tamaño de las subpoblaciones.

- La frecuencia de migración.
- El número y el destino de los individuos migrados.
- Un método para seleccionar los individuos a migrar.

En la figura 5.6 se pueden observar diversas topologías que han sido utilizadas en los AGs paralelos por islas. Las topologías indican el número total de islas utilizadas y cómo estas islas se interconectan a la hora de migrar individuos.

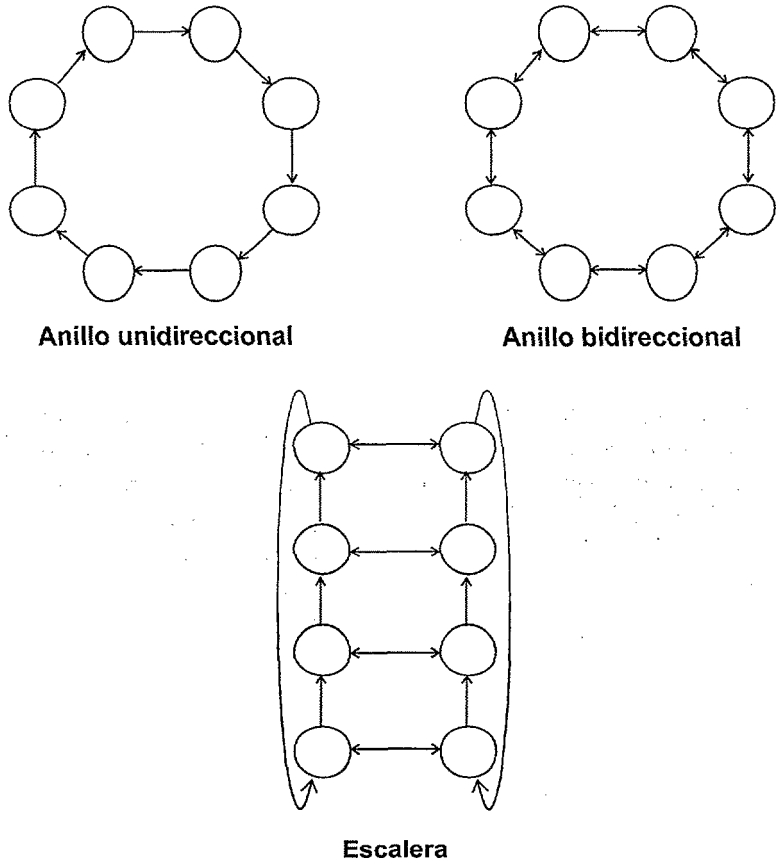


Figura 5.6: Diversas topologías de AGs paralelos con múltiples poblaciones

En cuanto al modelo de islas, hay diversos investigadores que aseguran que es posible conseguir *speedups* superlineales con este tipo de algoritmos, consiguiendo mejores resultados con un menor número de individuos evaluados. Aunque existe mucha controversia sobre esta discusión [Pun98], los estudios sobre el incremento de la presión selectiva [CP01] proporcionan una respuesta afirmativa apropiada.

El tercer tipo de AGs paralelos son los algoritmos de grano fino. Consisten en una única población estructurada espacialmente (véase figura 5.7). Normalmente, la estructura de la población es una rejilla rectangular de dos dimensiones, con un individuo en cada punto de la rejilla. Idealmente hay un procesador por cada individuo, por lo que las evaluaciones del *fitness* se pueden realizar simultáneamente a todos los individuos. La selección y cruce están

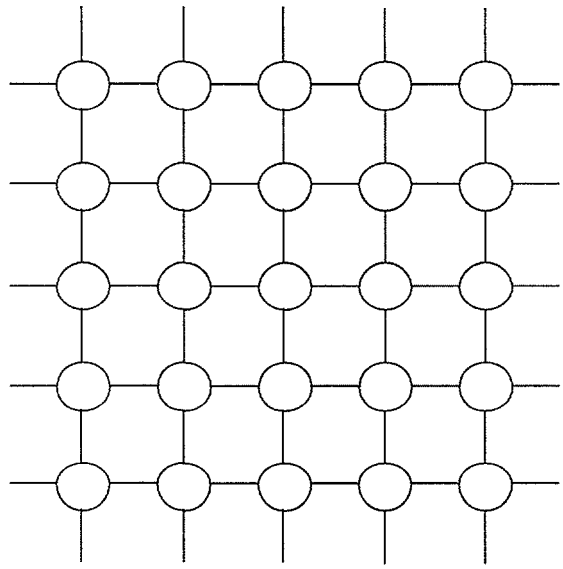


Figura 5.7: Esquema de AGs paralelos de grano fino

restringidos a un pequeño vecindario alrededor de cada individuo. Como los vecindarios están solapados, los buenos rasgos de los mejores individuos se pueden extender a toda la población, por lo que a este tipo de AGs paralelos también se les denomina "modelo de difusión" (análogo a la difusión aleatoria de partículas en un fluido). Este tipo de AGs paralelos son idóneos para computadores matriciales (SIMD), que ejecutan al mismo tiempo la misma instrucción en diferentes procesadores.

El último método de AGs paralelos, híbridos jerárquicos, combina el modelo de islas con los AGs maestro-esclavo o grano fino. Estos AGs combinan las ventajas de sus componentes, dando mejor rendimiento que cada uno de ellos por separado.

5.4.3. PIENB

Se ha desarrollado una versión paralela del algoritmo IENB, denominada PIENB, con el objetivo de mejorar su rendimiento y su exactitud. De todos los modelos de paralelización de AGs expuestos en la anterior sección se ha elegido el modelo de islas, con una topología de anillo unidireccional, como el más apropiado para las pruebas iniciales de paralelización debido a su simplicidad.

La figura 5.8 muestra el flujo de control de PIENB, en donde se puede observar que el pseudocódigo está ejecutando en todos los nodos. Las flechas muestran la topología de migración de cada nodo. Por supuesto, sería posible utilizar diferentes topologías de migración.

El algoritmo desarrollado tiene en cuenta los siguientes aspectos:

- Cada nodo genera y mejora su propia subpoblación, pero cada G generaciones, los mejores Me individuos se migran en forma circular (topología circular). De esta forma, cada nodo i envía sus mejores individuos al nodo $i+1$ y recibe los mejores

individuos del nodo $i-1$. Antes de enviar un individuo, el algoritmo chequea si el individuo ya ha sido enviado al nodo destino, enviando sólo aquellos individuos que no hayan sido enviados. Los parámetros G y Me son configurables, y dependen del tamaño de la población y del número de nodos. Los individuos migrados reemplazarán a los peores individuos de la población de destino.

- Cuando un nodo converge, no finaliza, ya que tiene enlaces con otros nodos. En este caso, el nodo toma el papel de "puente", recibiendo y enviando mensajes a los nodos correspondientes en la topología. Se considera que la aplicación finaliza cuando todos los nodos han convergido, siendo la solución la mejor de las soluciones parciales de cada nodo.
- PIENB aprovecha la mayor potencia de cálculo de un *cluster* de varios nodos, consiguiendo mejores resultados en un tiempo más corto.

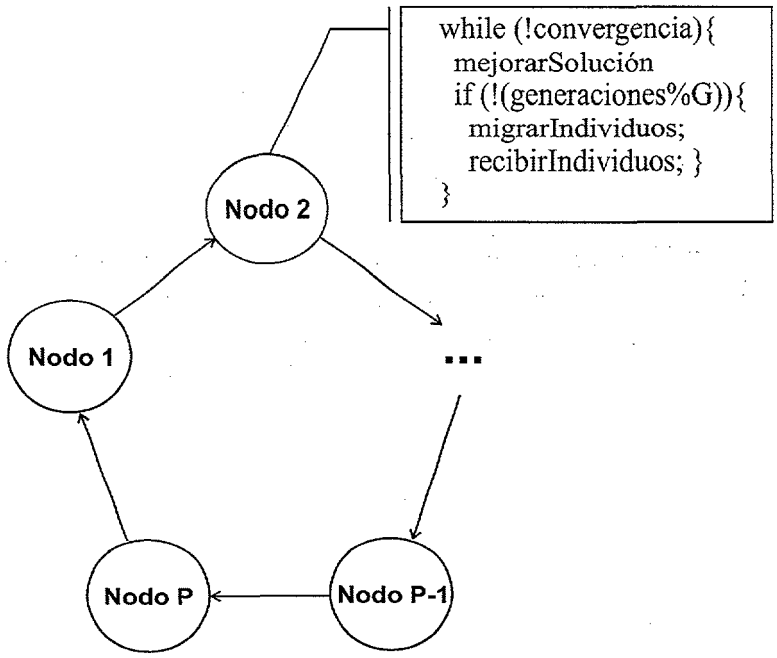


Figura 5.8: Esquema de *Parallel Interval Estimation naïve-Bayes*

Para la implementación de PIENB se ha utilizado MPI (*Message Passing Interface*) [MPI94]. Las principales razones de esta decisión son las siguientes:

- Es una interfaz estándar de paso de mensajes, que permite a los diferentes procesos comunicarse a través del uso de mensajes.
- Es ampliamente utilizado en *clusters* de estaciones de trabajo.

MPI ha sido utilizado como interfaz de paso de mensajes para los procesos de migración y de "puente".

5.4.4. Resultados

Resultados experimentales

Para realizar la experimentación con PIENB se han seleccionado los conjuntos de datos *chess*, *crx*, *german* y *vehicle*. Las pruebas se han realizado en un *cluster* con 8 nodos biprocesadores Intel Xeon a 2GHz con 1 GB de memoria ram y conectados a través de una red GigaEthernet.

En las primeras pruebas realizadas se ha fijado el tamaño de la población global a 500 individuos (el mismo número de individuos que el usado en las pruebas secuenciales) y se ha ido variando el número de islas. Se han realizado pruebas para 2, 4, 8 y 16 islas. El tamaño de la población de cada isla se puede obtener dividiendo el tamaño de la población global entre el número de islas.

Los parámetros de migración de los individuos se han fijado de la siguiente manera: cada cinco generaciones se migran el 10 % de los mejores individuos de cada población.

El resto de parámetros específicos de EDA fueron: tipo de aprendizaje UMDA, número de individuos creados en cada generación (*offspring*) el doble del tamaño de la población de la isla y elitismo.

Por último, cabe destacar, que cada una de las pruebas ha sido ejecutada 10 veces, siendo los valores que aparecen en las siguientes gráficas la media de estas ejecuciones.

En la figura 5.9 se encuentran representadas las medias de los valores del tanto por ciento de bien clasificados obtenidos en las ejecuciones con 1 –ejecución secuencial–, 2, 4, 8 y 16 nodos.

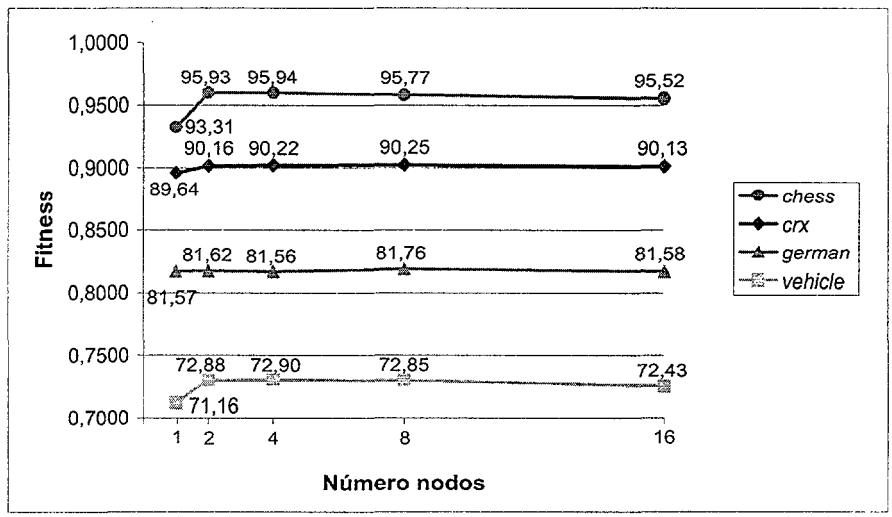


Figura 5.9: Resultados obtenidos por el algoritmo IENB con una población global de 500 individuos

Como se puede apreciar, los resultados de las pruebas paralelas mejoran bastante a los resultados secuenciales. En las ejecuciones con 2 nodos, la mínima mejora obtenida es de 0,05 % para el conjunto *german*, mientras que la máxima mejora es de 2,62 % para el

conjunto *chess*. Las ejecuciones con 4 nodos también dan excelentes resultados. Sin embargo, se nota un cierto empeoramiento en las ejecuciones de 8 y 16 nodos. Esto puede ser debido a que el tamaño de las poblaciones locales, 62 y 31 individuos para las ejecuciones con 8 y 16 nodos respectivamente, es demasiado pequeño y convergen prematuramente.

Por otra parte, en la figura 5.10 están representadas las medias del número total de evaluaciones necesarias para llegar al resultado. Los valores obtenidos son sorprendentes. Por ejemplo, para el conjunto de datos *vehicle*, mientras que en la versión secuencial son necesarias 129100 evaluaciones en total, en la ejecución paralela con 2 nodos fueron necesarias sólo 58600 evaluaciones, es decir, 29300 evaluaciones por nodo. Esto significa un *speedup* de 4,40 (ya que 129100/29300=4,40). Hay que tener en cuenta, que en el cálculo de este valor del *speedup* no se está teniendo en cuenta el tiempo necesario para la migración de los individuos entre poblaciones. En conclusión, además de conseguir mejorar los resultados obtenidos por IENB, se ha conseguido un *speedup* superlineal.

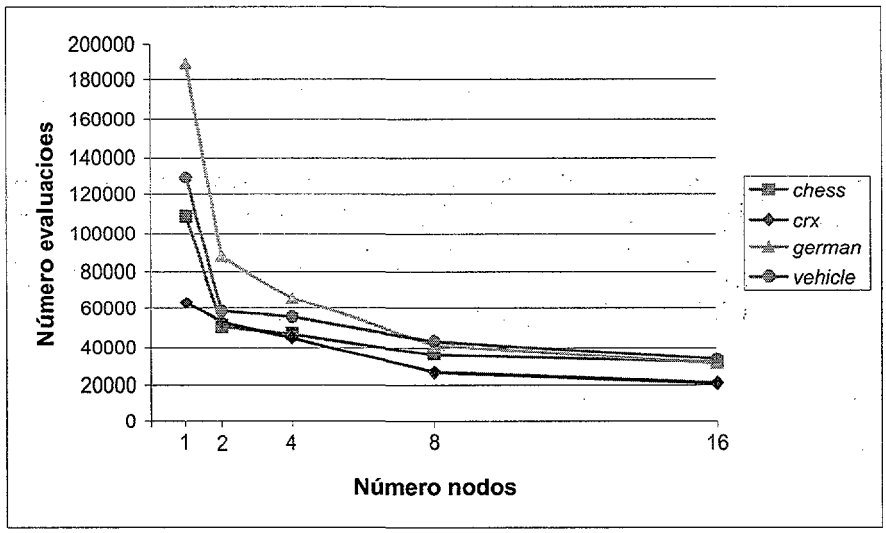


Figura 5.10: Número total de individuos evaluados por el algoritmo PIENB con una población global de 500 individuos

Por último, en la figura 5.11 están representadas las curvas de aprendizaje para el conjunto de datos *chess*. Se puede observar, incluso desde las primeras ejecuciones, la superioridad del enfoque paralelo.

Conclusiones

El algoritmo IENB puede llegar a ser muy costoso desde el punto de vista computacional debido a la búsqueda heurística que realiza.

En esta sección se ha presentado una paralelización basada en islas, en donde cada isla contiene una población diferente y, cada cierto tiempo y con un esquema de migración predeterminado, las islas se intercambian los mejores individuos entre sí.

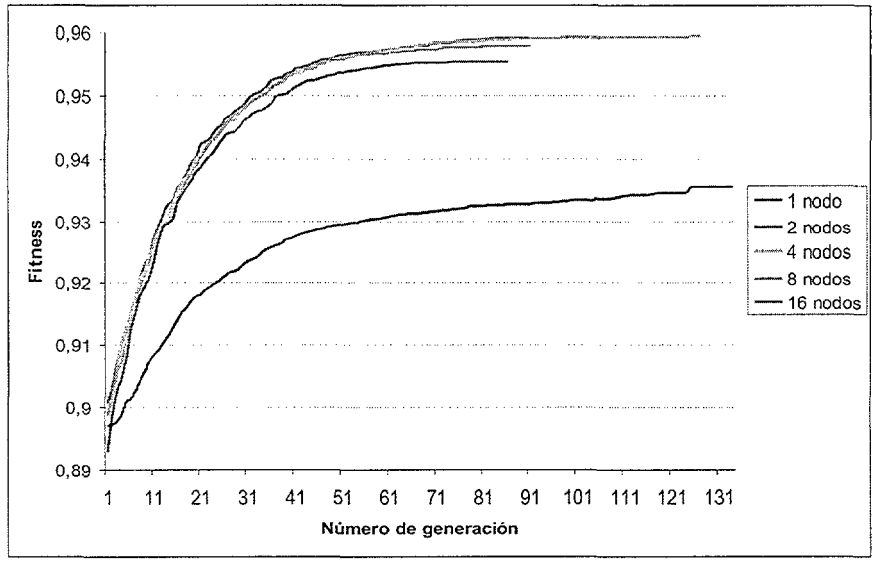


Figura 5.11: Curvas de aprendizaje del algoritmo PIENB para el conjunto de datos *chess*

Los resultados obtenidos son excelentes, ya que comparativamente se obtienen mejores resultados que con la versión secuencial y además se obtiene un *speedup* superlineal.

Líneas de trabajo futuro

La paralelización de IENB realmente ha supuesto paralelizar los algoritmos heurísticos EDAs. Dentro de este campo queda mucho trabajo por desarrollar. Algunas ideas son:

- Realizar un estudio detallado sobre los ratios de migración entre islas.
- Estudio del *speedup* y del número de procesadores óptimo dado un tamaño de población global.
- Probar diferentes topologías de migración entre islas.
- Estudio de algoritmos heurísticos paralelos híbridos AGs-EDAs. En este sentido sería posible tener unas islas que ejecuten una optimización con AGs y otras que lo hagan con EDAs.
- Probar otro tipo de paralelizaciones de EDAs, similares a las de grano fino que se realizan con los AGs.

5.5. APNBC-EDA

5.5.1. Propuesta

Como se comentó en el anterior capítulo, las variantes semi naïve-Bayes están normalmente basadas en la realización de búsquedas de determinadas estructuras o de deter-

minados valores. Estas búsquedas se suelen realizar a través de algoritmos de búsqueda voraz (*greedy*). Un ejemplo de esta situación es el enfoque *APNBC (Adjusted Probability naïve-Bayes Classification)* [WP98].

En esta sección se describe cómo se puede realizar el enfoque APNBC a través de algoritmos heurísticos de optimización EDAs [RLP+03d]. Para ello se define el formato de los individuos utilizados para la optimización heurística y se presentan los resultados experimentales conseguidos con los conjuntos de datos del repositorio UCI descritos en la sección 5.1.

5.5.2. Formato de los individuos

El enfoque APNBC realiza un ajuste lineal del peso de la probabilidad de cada clase. De esta forma, se asigna un factor de ajuste a cada clase y su probabilidad estimada se multiplica por este factor. El ajuste lineal propuesto por los autores se puede expresar de la siguiente manera:

$$P(C = c_i|X_1 = x_1, \ldots, X_n = x_n) \propto w_i P(C = c_i) \prod_{k=1}^{n} P(X_k = x_k|C = c_i) \qquad (5.24)$$

En los conjuntos de datos con dos clases sólo es necesario calcular uno de los factores de ajuste. Esto es debido a que el efecto conseguido por cualquier combinación de ajustes a_1 y a_2 para las clases c_1 y c_2 se puede obtener ajustando c_1 con a_1/a_2 y c_2 con 1. En el caso de múltiples clases la búsqueda se debe realizar de una forma similar, fijando uno de los factores de ajuste a 1 y realizando la búsqueda del resto de los factores. En el enfoque APNBC la búsqueda de estos valores de ajuste se realiza a través de un algoritmo de "ascenso por la colina", en donde cada posible conjunto de factores de ajuste es validado a través del método de resustitución. Hay que destacar el hecho de que, en diversos conjuntos de datos, el enfoque APNBC empeora los resultados obtenidos por el clasificador naïve-Bayes.

En el caso de utilizar EDAs continuos, para la búsqueda de los factores de ajuste, es necesario definir el formato de los individuos. Para cada gen del individuo, se deben definir un valor máximo y un valor mínimo dentro de los cuales se realizará la búsqueda. De esta forma, para un problema con n clases se utilizará un individuo con $n-1$ genes –ya que la última clase se ajustará siempre con 1– y cada gen del individuo tendrá un valor mínimo de 0 y un valor máximo de 5. Se ha demostrado experimentalmente que nunca se llegan a seleccionar estos valores extremos, por lo que se considera que el intervalo definido es idóneo para la búsqueda. Además, cada individuo será validado a través del método *leave-one-out*.

5.5.3. Resultados

Resultados experimentales

Las pruebas realizadas con APNBC-EDA se ejecutaron sobre un Athlon 2000+ con 1GByte de memoria RAM. Los parámetros utilizados para la ejecución de los EDAs fueron: tamaño de población 500 individuos, individuos seleccionados para aprendizaje 500,

número de nuevos individuos cada generación (*offspring*) 1000, tipo de aprendizaje UMDA (*Univariate Marginal Distribution Algorithm*) [Müh98] y elitismo. Cada prueba ha sido ejecutada 10 veces.

		APNBC-EDA			
	NB	*APNBC-EDA*	*Num. Eval.*	*Mejora*	*Mejora APNBC*
breast	97,14	97,57 ± 0,00 †	2600	0,43	-0,20
chess	87,92	88,14 ± 0,00 †	11300	0,22	
cleve	83,82	84,16 ± 0,00 †	7500	0,34	-1,00
corral	84,37	90,63 ± 0,00 †	1500	6,26	
crx	86,23	86,96 ± 0,00 †	10400	0,73	-0,20
flare	80,86	83,77 ± 0,00 †	4000	2,91	
german	75,40	75,90 ± 0,00 †	30600	0,50	
glass	74,77	79,44 ± 0,00 †	7500	4,67	
glass2	82,21	88,34 ± 0,00 †	2200	6,13	
hepatitis	85,16	88,39 ± 0,00 †	3300	3,23	
iris	94,67	96,00 ± 0,00 †	3200	1,33	0,60
lymphography	85,14	86,49 ± 0,00 †	3500	1,35	1,40
mofn-3-7-10	86,33	98,24 ± 0,04 †	5500	11,91	
pima	77,73	79,43 ± 0,00 †	5500	1,70	-0,50
satimage	82,46	85,19 ± 0,00 †	25500	2,73	
segment	91,95	92,68 ± 0,00 †	11400	0,73	
shuttle-small	99,36	99,47 ± 0,00 †	6500	0,11	
soybean-large	92,83	93,70 ± 0,00 †	13700	0,87	-1,50
vehicle	61,47	65,25 ± 0,17 †	6600	3,78	
vote	90,11	90,57 ± 0,00 †	5400	0,46	
waveform-21	78,85	80,21 ± 0,05 †	3100	1,36	5,00

Tabla 5.5: Resultados experimentales de APNBC-EDA

Los resultados experimentales de APNBC –extraídos del artículo original [WP98]– y de APNBC-EDA se pueden ver en la tabla 5.5. Las columnas de dicha tabla tienen, por orden, el siguiente contenido: el tanto por ciento de bien clasificados obtenido con el clasificador naïve-Bayes, la media y la desviación típica del tanto por ciento de bien clasificados obtenidos con APNBC-EDA, el número medio de evaluaciones necesarias para obtener el resultado, la mejora obtenida respecto al clasificador naïve-Bayes, y la mejora obtenida por APNBC según el artículo original.

Sólo se tiene el resultado de APNBC en ocho de los 21 conjuntos de datos utilizados. De estos ocho conjuntos, en seis APNBC-EDA se muestra muy superior a APNBC, en uno *lymphography* se consigue prácticamente el mismo resultado y extrañamente, en el conjunto *waveform-21*, mientras que APNBC mejora un 5,00 %, APNBC-EDA sólo consigue mejorar 1,36 %.

Sin embargo, es interesante reseñar otros aspectos. En los ocho conjuntos de datos citados, APNBC-EDA consigue una mejora media de 1,01 %, mientras que APNBC sólo consigue un 0,45 % de mejora. Si se tienen en cuenta los 21 conjuntos de datos, APNBC-EDA consigue una mejora media de 2,46 %. Por último, cabe destacar que APNBC sólo consigue mejorar los resultados de naïve-Bayes en tres de los ocho conjuntos, mientras que

APNBC-EDA mejora en absolutamente todos los conjuntos de datos. Estos datos indican la gran superioridad de APNBC-EDA respecto al enfoque APNBC.

También se ha utilizado el test no paramétrico de Mann-Whitney para validar la hipótesis nula de la misma distribución de densidad para cada uno de los conjuntos de prueba. Esta tarea se realizó con el paquete estadístico S.P.S.S. versión 11.50. Los resultados de estas pruebas se muestran a continuación.

- Pruebas del clasificador APNBC-EDA:
  - Naïve-Bayes vs. APNBC-EDA. Valor de ajuste en todos los conjuntos de datos: $p < 0{,}001$.

Estos resultados demuestran que la diferencia entre naïve-Bayes y APNBC-EDA es significativa en todos los conjuntos de datos.

Conclusiones

El enfoque APNBC realiza, sobre los valores obtenidos por el clasificador naïve-Bayes, un ajuste lineal del peso de la probabilidad de cada clase. De esta forma, se asigna un factor de ajuste a cada clase y su probabilidad estimada se multiplica por este factor. En APNBC la búsqueda de estos valores de ajuste se realiza a través de un algoritmo de "ascenso por la colina".

En esta sección se ha descrito una extensión del algoritmo APNBC, denominada APNBC-EDA, que realiza la búsqueda de los valores de ajuste por medio de los algoritmos heurísticos EDAs.

Los resultados obtenidos son muy satisfactorios, ya que se ha conseguido mejorar en gran medida los resultados de APNBC.

5.6. Pazzani-EDA

5.6.1. Propuesta

Siguiendo con la filosofía de la sección anterior (aplicar los algoritmos heurísticos EDA para la búsqueda de estructuras semi naïve-Bayes), en esta sección se propone la búsqueda heurística de estructuras basadas en el producto cartesiano de variables [RLP+03d].

Por tanto, esta sección describe cómo se puede realizar la búsqueda de la estructura de Pazzani que maximice el tanto por ciento de bien clasificados a través de algoritmos heurísticos de optimización EDAs. Para poder realizar la mejor comparación posible se han implementado los algoritmos voraces BSEJ y FSSJ descritos por Pazzani.

A continuación se describe el formato de los individuos utilizados en la optimización heurística y se presentan los resultados experimentales obtenidos con los conjuntos de datos del repositorio UCI descritos en la sección 5.1.

5.6.2. Formato de los individuos

En la figura 5.12 se pueden ver dos ejemplos de estructura de Pazzani y los individuos que representan dichas estructuras.

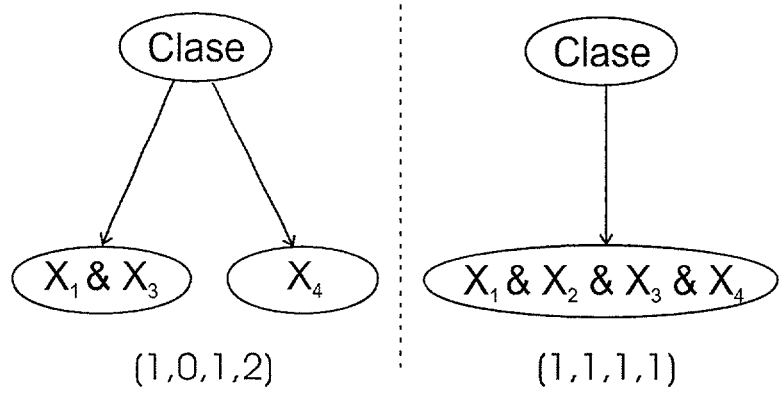


Figura 5.12: Estructuras de Pazzani e individuos que las representan

De esta forma, para un conjunto de datos con n atributos, los individuos tendrán n genes, cada uno de ellos con un valor entero de 0 a n. El valor 0 representa que el atributo correspondiente no se tiene en cuenta en la estructura de Pazzani, mientras que un valor entre 1 y n significa que el correspondiente atributo pertenece a dicho grupo dentro de la estructura de Pazzani.

5.6.3. Resultados

Resultados experimentales

Las pruebas realizadas con los algoritmos voraces BSEJ y FSSJ y el algoritmo Pazzani-EDA se ejecutaron sobre un Athlon 2000+ con 1GByte de memoria RAM. Los parámetros utilizados para la ejecución de los EDAs fueron: tamaño de población 500 individuos, individuos seleccionados para aprendizaje 500, número de nuevos individuos cada generación (*offspring*) 1000, tipo de aprendizaje UMDA (*Univariate Marginal Distribution Algorithm*) [Müh98] y elitismo. Cada prueba ha sido ejecutada 10 veces.

Los resultados experimentales obtenidos para los algoritmos voraces FSSJ y BSEJ se pueden ver en la tabla 5.6. Las cuatro primeras columnas presentan los resultados del algoritmo FSSJ y las cuatro últimas los resultados del algoritmo BSEJ. De la misma forma, en la tabla 5.6.3 se encuentran los resultados del algoritmo Pazzani-EDA. Para cada algoritmo se recogen los siguientes valores: tanto por ciento de bien clasificados obtenidos, número de evaluaciones necesarias para obtener el resultado, número de atributos usados en la estructura final y número de uniones (*joins*) en la estructura final.

En la tabla 5.6.3 se encuentra la comparación de los resultados obtenidos con los tres algoritmos que utilizan el concepto de producto cartesiano de variables: FSSJ, BSEJ y EDA. En la primera columna de la tabla se muestran los resultados del algoritmo FSSJ, en la segunda columna los del BSEJ y en la tercera y última columna los de Pazzani-EDA.

Algoritmos voraces FSSJ y BSEJ de Pazzani								
	FSSJ				*BSEJ*			
	Result.	*Eval.*	*Atrib.*	*Grupos*	*Result.*	*Eval.*	*Atrib.*	*Grupos*
breast	97,42	80	3	2	97,56	200	10	9
chess	94,33	396	5	1	97,06	14256	35	26
cleve	85,10	169	4	3	85,43	676	12	10
corral	74,80	18	1	1	93,70	108	6	4
crx	86,79	195	5	2	88,10	1125	14	11
flare	84,04	90	3	2	84,04	500	6	6
german	77,18	500	7	3	77,68	2400	19	15
glass	76,53	153	5	3	76,53	162	9	8
glass2	87,04	99	4	2	87,04	162	9	8
hepatitis	91,56	456	7	3	88,96	1083	18	17
iris	95,97	12	1	1	95,97	32	3	3
lymphography	84,35	252	4	3	90,48	972	18	16
mofn-3-7-10	77,32	30	1	1	88,27	200	9	9
pima	79,79	72	4	1	79,79	192	6	6
satimage	86,17	828	7	3	86,84	12914	36	22
segment	96,32	589	7	4	95,89	3971	18	10
shuttle-small	99,74	72	3	2	99,90	324	9	6
soybean-large	93,40	3010	13	8	94,57	7350	34	30
vehicle	70,77	162	4	1	72,66	3240	17	9
vote	96,77	128	3	2	94,47	1792	15	10
waveform-21	69,65	189	4	1	79,70	89	21	19

Tabla 5.6: Resultados experimentales de los algoritmos voraces FSSJ y BSEJ de Pazzani

Teniendo en cuenta todos los conjuntos de datos, el algoritmo FSSJ muestra una mejora media respecto a naïve-Bayes de 1,25 %, el algoritmo BSEJ de 3,61 % y el algoritmo Pazzani-EDA de 5,51 %.

También se ha utilizado el test no paramétrico de Mann-Whitney para validar la hipótesis nula de la misma distribución de densidad para cada uno de los conjuntos de prueba. Esta tarea se realizó con el paquete estadístico S.P.S.S. versión 11.50. Los resultados de estas pruebas se muestran a continuación.

- Pruebas del clasificador Pazzani-EDA:
  - Naïve-Bayes vs. Pazzani-EDA. Valor de ajuste en todos los conjuntos de datos: $p < 0{,}001$.
  - Pazzani-BSEJ vs. Pazzani-EDA. Valor de ajuste en 15 conjuntos de datos: $p < 0{,}001$. En todos los conjuntos de datos menos: *flare*, *glass2*, *iris*, *pima*, *satimage* y *shuttle-small*.

Estos resultados demuestran que la diferencia entre Pazzani-BSEJ y Pazzani-EDA son significativos en 15 conjuntos de datos.

Además, otra ventaja de utilizar la búsqueda heurística con EDAs es que las estructuras de Pazzani encontradas suelen ser más sencillas, tanto en número de atributos utilizados

	Pazzani-EDA			
	Pazzani-EDA	*Num. Eval.*	*Atributos*	*Grupos*
breast	97,82 ± 0,06 †	20600	10	7
chess	97,93 ± 0,24 †	44900	26	15
cleve	86,98 ± 1,18 †	49600	9	5
corral	100,00 ± 0,00 †	7400	5	3
crx	89,23 ± 0,07 †	80200	12	7
flare	84,04 ± 0,00 †	9700	5	4
german	78,41 ± 0,06 †	90900	17	11
glass	77,93 ± 0,00 †	40200	7	5
glass2	87,04 ± 0,00 †	7700	6	5
hepatitis	93,64 ± 0,29 †	55800	11	7
iris	95,97 ± 0,00 †	3600	1	1
lymphography	93,88 ± 0,00 †	48900	15	7
mofn-3-7-10	100,00 ± 0,00 †	39600	8	2
pima	79,82 ± 0,28 †	7300	7	4
satimage	86,94 ± 0,12 †	105300	12	7
segment	96,15 ± 0,06 †	24300	14	7
shuttle-small	99,90 ± 0,00 †	30200	7	5
soybean-large	95,67 ± 0,17 †	50500	32	20
vehicle	76,50 ± 0,32 †	61300	12	7
vote	96,77 ± 0,13 †	30200	7	4
waveform-21	79,72 ± 0,07 †	89200	17	9

Tabla 5.7: Resultados experimentales del algoritmo Pazzani-EDA

(hay que recordar en este punto que los algoritmos de Pazzani realizan, de forma implícita, una selección de variables), como en el número total de uniones empleadas. A la vista de los resultados se puede apreciar que el algoritmo BSEJ utiliza una media de 15 atributos y 12 uniones, mientras que el algoritmo Pazzani-EDA utiliza una media de 12 atributos y 7 uniones.

Conclusiones

En esta sección se ha presentado una extensión a los algoritmos heurísticos FSSJ y BSEJ de Pazzani, que permite realizar la búsqueda de la mejor estructura de Pazzani con los algoritmos heurísticos de optimización EDA.

Los resultados obtenidos por Pazzani-EDA son muy buenos. Por un lado se ha conseguido mejorar los resultados de los algoritmos FSSJ y BSEJ de Pazzani. Por otra parte, las estructuras encontradas por Pazzani-EDA son más sencillas que las encontradas por los algoritmos voraces FSSJ y BSEJ.

Líneas de trabajo futuro

Dentro de los 6 algoritmos semi naïve-Bayes presentados hasta el momento, el que mejores resultados ha obtenido es Pazzani-EDA con una mejora media de 5,51 %, seguido

	Comparación resultados experimentales		
	Pazzani-FSSJ	*Pazzani-BSEJ*	*Pazzani-EDA*
breast	0,28	0,42	0,68
chess	6,41	9,14	10,01
cleve	1,28	1,61	3,16
corral	-9,57	9,33	15,63
crx	0,56	1,87	3,00
flare	3,18	3,18	3,18
german	1,78	2,28	3,01
glass	1,76	1,76	3,16
glass2	4,83	4,83	4,83
hepatitis	6,40	3,80	8,48
iris	1,30	1,30	1,30
lymphography	-0,79	5,34	8,74
mofn-3-7-10	-9,01	1,94	13,67
pima	2,06	2,06	2,09
satimage	3,71	4,38	4,48
segment	4,37	3,94	4,20
shuttle-small	0,38	0,54	0,54
soybean-large	0,57	1,74	2,84
vehicle	9,30	11,19	15,03
vote	6,66	4,36	6,66
waveform-21	-9,20	0,85	0,87
Mejora media	1,25	3,61	5,51

Tabla 5.8: Comparación respecto a naïve Bayes de los resultados experimentales de los algoritmos que utilizan el concepto de producto cartesiano entre variables

por IENB con una mejora media de 4,93 %.

Sin embargo, estos algoritmos son totalmente compatibles, siendo posible realizar en una primera instancia la búsqueda de la mejor estructura de Pazzani con el algoritmo Pazzani-EDA, para a continuación aplicar la estimación por intervalo propuesta en el IENB.

5.7. Búsqueda de clasificadores BAN

5.7.1. Propuesta

Frente a los métodos de búsqueda que se encuentran actualmente en la literatura [FGG97, Sah96], que están basadas en la cantidad de información mutua entre las variables predictoras, en esta sección se presentan dos algoritmos para la búsqueda de la mejor estructura de tipo *Augmented naïve-Bayes* (BAN). El primer algoritmo está basado en una búsqueda voraz, mientras que el segundo está basado en una búsqueda heurística con EDAs. Ambos algoritmos están guiados por el tanto por ciento de bien clasificados.

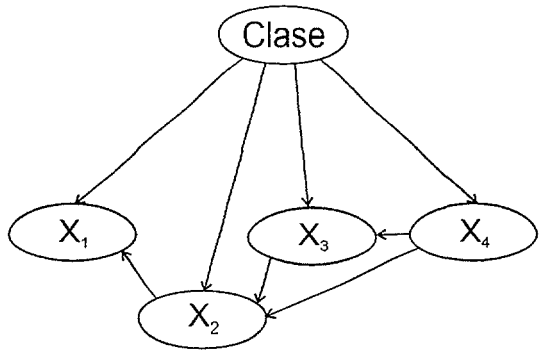
En las siguientes secciones se describe el formato de los individuos para este tipo de búsquedas y se analizan los dos algoritmos, el voraz y el heurístico. Se concluye con la

presentación de los resultados conseguidos.

5.7.2. Formato de los individuos

Como se vio en el capítulo 3, el único requisito que debe cumplir una red Bayesiana para ser de tipo BAN, es que no se formen ciclos entre las variables predictoras.

En la figura 5.13 está representada una red de tipo BAN y el individuo que la representa. Ya que siempre existen arcos entre la variable a clasificar y las variables predictoras, no hace falta incluir esta información en la representación del individuo.



	X_1	X_2	X_3	X_4
X_1	0	0	0	0
X_2	1	0	0	0
X_3	0	1	0	0
X_4	0	1	1	0

Figura 5.13: Representación de los individuos BAN

De esta forma, para un conjunto de datos con n variables predictoras, el individuo será una matriz cuadrada de rango n, donde cada posición puede tomar el valor 0 ó 1. Un valor de 1 en la posición (i, j) de la matriz, representa que existe un arco de la variable predictora X_i a la variable predictora X_j.

En la fase de búsqueda de clasificadores tipo BAN, dependiendo del algoritmo empleado, voraz o heurístico, la acción a realizar sobre los individuos será diferente. En el caso del algoritmo voraz, se debe verificar que cada individuo generado en la búsqueda cumple la restricción de no tener ciclos. En caso de que el individuo tenga ciclos, éste no se tiene el cuenta, y se continúa la búsqueda sin evaluarlo.

Por otra parte, en el algoritmo heurístico, se tratará de que todos los individuos de la población sean correctos. Cada individuo generado por el algoritmo será corregido en caso necesario, para su posterior evaluación e inserción en la población.

5.7.3. Algoritmo BAN-voraz

El pseudocódigo del algoritmo BAN-voraz es el siguiente:

- *Paso 1*: Inicializar con el modelo naïve-Bayes con todas las variables predictoras.
- *Paso 2*: Partiendo del mejor individuo del paso anterior:
  - Tentativamente añadir un arco entre dos variables predictoras.
  - Evaluar cada posible opción válida por medio de la estimación del porcentaje de bien clasificados.

- Hasta que ninguna opción produzca mejoras.

5.7.4. Algoritmo BAN-EDA

Dentro del algoritmo heurístico, como ya se comentó, es importante que todos los individuos de la población sean correctos. En la figura 5.14 se muestra el pseudocódigo utilizado para corregir los individuos. Con este algoritmo se asegura que, en un número finito de pasos, el grafo obtenido es acíclico.

$rango = numeroVariables$
$listaDescartadas = \emptyset$
$listaCandidatas = \emptyset$

while $rango \neq 0$ y $\exists fila$ t.q. $numUnos = 0$
  para cada $fila$
    Si $fila \notin listaDescartadas \Rightarrow$
      calcular $numUnos$
      Si $numUnos = 0 \Rightarrow$
        $fila \rightarrow listaDescartadas$
        dec($rango$)
        $fila = 1$
        $listaCandidatas = \emptyset$
      else $\Rightarrow$
        $fila \rightarrow listaCandidatas$

  Si no $\exists fila$ t.q. $numUnos = 0$ y $rango \neq 0$
    elegir aleatoriamente $fila_i \in listaCandidatas$
    poner 0 en columnas cuya $fila \notin listaDescartadas$
    $fila \rightarrow listaDescartadas$
    dec($rango$)

Figura 5.14: Pseudocódigo de la corrección de un individuo BAN

5.7.5. Resultados

Resultados experimentales

Las pruebas realizadas con el algoritmo voraz y el algoritmo BAN-EDA se ejecutaron sobre un Athlon 2000+ con 1GByte de memoria RAM. Los parámetros utilizados para la ejecución de los EDAs fueron: tamaño de población 500 individuos, individuos seleccionados para aprendizaje 500, número de nuevos individuos cada generación (*offspring*) 1000, tipo de aprendizaje UMDA (*Univariate Marginal Distribution Algorithm*) [Müh98] y elitismo. Cada prueba ha sido ejecutada 10 veces.

Los resultados experimentales obtenidos para el algoritmo BAN-voraz, el algoritmo heurístico BAN-EDA y el algoritmo de búsqueda de Friedman [FGG97] se pueden ver en la tabla 5.7.5. Las cuatro primeras columnas presentan los resultados del algoritmo BAN-voraz, las cuatro siguientes los resultados del algoritmo BAN-EDA y la última los resultados de Friedman. Para los algoritmos BAN-voraz y BAN-EDA se recogen los siguientes valores: tanto por ciento de bien clasificados obtenidos, número de evaluaciones necesarias para llegar al resultado, número de arcos usados en la estructura final y mejora conseguida respecto a naïve-Bayes.

	BAN-voraz				*BAN-EDA*				Fried.
	voraz	*Eval*	*Ar*	*Mej*	*BAN-EDA*	*Eval*	*Ar*	*Mej*	*Mej Fr*
breast	97,99	400	4	0,85	99,42 ± 0,52 †	8500	29	2,28	-0,44
cleve	86,14	468	2	2,32	97,32 ± 0,47 †	11000	54	13,50	-1,37
corral	100,00	90	2	15,63	100,00 ± 0,00 †	2000	10	15,63	11,72
crx	88,98	975	4	2,75	92,80 ± 1,03 †	8600	22	6,57	-0,62
flare	84,52	490	6	3,66	86,02 ± 0,13 †	3200	22	5,16	3,28
glass	84,58	270	3	9,81	85,05 ± 0,00 †	5500	19	10,28	-14,09
glass2	88,34	216	2	6,13	88,34 ± 0,00 †	4300	16	6,13	-3,68
iris	96,67	40	3	2,00	96,67 ± 0,00 †	400	3	2,00	0,67
mofn-3-7-10	98,43	340	3	12,10	98,57 ± 2,45 †	15200	16	12,24	-0,49
pima	80,08	208	3	2,35	81,90 ± 0,00 †	4500	10	4,17	-0,51
vote	96,32	1296	5	6,21	99,54 ± 1,05 †	15300	32	9,43	4,6

Tabla 5.9: Resultados experimentales de los algoritmos BAN-voraz, BAN-EDA y algoritmo de Friedman

En esta ocasión los resultados necesitan una lectura diferente. Si bien es cierto que el algoritmo que mejores resultados presenta es BAN-EDA, no menos cierto es que las estructuras encontradas por dicho algoritmo son demasiado complejas. Esto hace suponer que BAN-EDA se está sobreajustando a los datos de aprendizaje.

BAN-voraz consigue una mejora media de 5,80 % con una media de 3 arcos en las estructuras obtenidas. Por su parte, BAN-EDA consigue una mejora media de 7,94 %, pero el número medio de arcos en las estructuras obtenidas es de 21. Estos datos, unidos al número medio de evaluaciones necesario para obtener los resultados, hacen que la recomendación en este caso se decante por el algoritmo BAN-voraz.

Por último, cabe destacar que ambos algoritmos, BAN-voraz y BAN-EDA, son muy superiores al enfoque de Friedman basado en la cantidad de información mutua entre las variables predictoras.

Conclusiones

En esta sección se han presentado dos algoritmos para la búsqueda de estructuras tipo BAN. El primer algoritmo, denominado BAN-voraz, está basado en una búsqueda voraz, mientras que el segundo, denominado BAN-EDA, está basado en una búsqueda heurística con EDAs. Ambos algoritmos han sido comparados con las búsquedas realizadas por Friedman, que están basadas en la cantidad de información mutua entre las variables predictoras.

Los resultados obtenidos muestran que ambos algoritmos desarrollados son muy superiores al enfoque de Friedman. Sin embargo, esta vez, la complejidad de las estructuras encontradas y los valores del tanto por ciento de bien clasificados obtenidos, hacen que la recomendación se decante por BAN-voraz.

Líneas de trabajo futuro

En esta ocasión el algoritmo recomendado ha sido el basado en la búsqueda voraz, sobre todo por la enorme complejidad de las estructuras encontradas por los algoritmos EDAs.

Sería posible realizar búsquedas heurísticas en las que se penalizara la complejidad de la estructura encontrada. De esta forma, a lo mejor sería posible encontrar estructuras que mejoren los resultados de BAN-voraz, a la vez que mantienen su simplicidad estructural.

5.8. Búsqueda de clasificadores TAN

5.8.1. Propuesta

La estructura *Augmented naïve-Bayes* (BAN) es un buen intento de buscar las correlaciones que se pueden dar entre los parámetros de un conjunto de datos. Sin embargo, presenta un problema, y es que el espacio de búsqueda de estructuras válidas es tan grande, que resulta muy costoso encontrar buenos clasificadores. Ante este problema, Friedman [FGG97] propone acotar las posibles correlaciones que se pueden dar entre las variables, y de este modo disminuir el espacio de búsqueda. El método resultante, llamado estructura *Tree Augmented Naïve-Bayes* (TAN), aproxima las interacciones entre los atributos usando una estructura de árbol impuesta sobre la estructura naïve-Bayes. Para encontrar la estructura TAN, Friedman propone un algoritmo basado en la cantidad de información mutua condicionada a la variable clase.

Frente al enfoque de Friedman, en esta sección se proponen dos nuevos algoritmos: TAN-voraz, basado en una búsqueda voraz y TAN-EDA, basado en una búsqueda con algoritmos EDA. Ambos algoritmos están guiados por el tanto por ciento de bien clasificados.

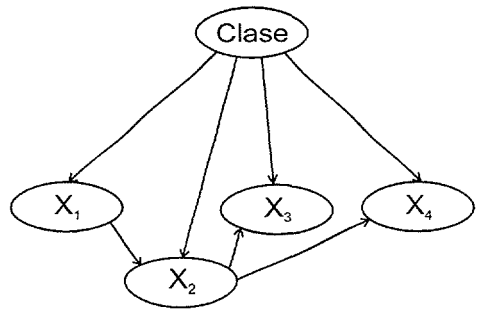
A continuación se define el formato necesario para los individuos en la búsqueda, se detallan los algoritmos voraz y heurístico, y se finaliza con los resultados obtenidos.

5.8.2. Formato de los individuos

La estructura TAN se define mediante dos condiciones [KP99]:

- Cada variable predictora tiene como padre la variable a clasificar.
- Las variables predictoras pueden tener como máximo a una y sólo una variable predictora como padre.

Al igual que en el caso de *Augmented Naïve-Bayes*, la representación de un individuo *Tree Augmented Naïve-Bayes* se realiza mediante una matriz cuadrada de rango el número de variables predictoras, con unos indicando las correlaciones que se dan entre ellas. Además, no se muestra la relación de parentesco entre la variable clase y el resto, dado que siempre existe (se parte de la estructura naïve-Bayes). En la figura 5.15 se muestra un ejemplo de esta representación.



	X_1	X_2	X_3	X_4
X_1	0	1	0	0
X_2	0	0	1	1
X_3	0	0	0	0
X_4	0	0	0	0

Figura 5.15: Representación de los individuos TAN

Tal y como se puede observar, la segunda condición que se impone a una red TAN –que una variable predictora sólo pueda tener a otra variable predictora como padre–, significa que en cada columna puede haber como máximo un uno.

5.8.3. Algoritmo TAN-voraz

El algoritmo es totalmente análogo a BAN-voraz, salvo que, a la hora de comprobar que el individuo es correcto, se debe comprobar también la condición de un único padre por cada variable predictora.

De este modo, es posible aprovechar el trabajo ya realizado, y concluir con la idea de que para comprobar si un individuo verifica las condiciones TAN, basta con comprobar que es una estructura BAN válida, y luego, comprobar que no hay más de un uno por columna en su matriz.

5.8.4. Algoritmo TAN-EDA

Para el algoritmo heurístico, aquellos individuos que no sean válidos se deben corregir. En el caso de estructuras TAN, la corrección de los individuos es muy sencilla, y aprovecha el trabajo realizado con las estructuras BAN. Una vez que un individuo ha sido generado en la fase de búsqueda, se corrige hasta que sea una estructura BAN válida, y luego, de forma aleatoria, se deja un único uno en cada columna. En la figura 5.16 se muestra el pseudocódigo del algoritmo utilizado.

```
Corregir individuo hasta que sea BAN
Para cada columna
    Si numUnos > 1 ⇒
        Elegir aleatoriamente uno y eliminar el resto
```

Figura 5.16: Pseudocódigo de la corrección de un individuo TAN

5.8.5. Resultados

Resultados experimentales

Las pruebas realizadas con el algoritmo TAN-voraz y el algoritmo TAN-EDA se ejecutaron sobre un Athlon 2000+ con 1GByte de memoria RAM. Los parámetros utilizados para la ejecución de los EDAs fueron: tamaño de población 500 individuos, individuos seleccionados para aprendizaje 500, número de nuevos individuos cada generación (*offspring*) 1000, tipo de aprendizaje UMDA (*Univariate Marginal Distribution Algorithm*) [Müh98] y elitismo. Cada prueba ha sido ejecutada 10 veces.

Los resultados experimentales obtenidos para el algoritmos voraz TAN-voraz, el algoritmo heurístico TAN-EDA y el algoritmo de búsqueda de Friedman [FGG97] se pueden ver en la tabla 5.8.5. Las cuatro primeras columnas presentan los resultados del algoritmo TAN-voraz, las cuatro siguientes los resultados del algoritmo TAN-EDA y la última los resultados de Friedman. Para los algoritmos TAN-voraz y TAN-EDA se recogen los siguientes valores: tanto por ciento de bien clasificados obtenidos, número de evaluaciones necesarias para llegar al resultado, número de arcos usados en la estructura final y mejora conseguida respecto a naïve-Bayes.

	TAN-voraz				*TAN-EDA*				Fried.
	voraz	*Eval*	*Ar*	*Mej*	*TAN-EDA*	*Eval*	*Ar*	*Mej*	*Mej Fr*
breast	97,99	300	3	0,85	98,26 ± 0,04 †	8100	7	1,12	-0,42
cleve	86,47	429	2	2,65	87,46 ± 0,00 †	10900	6	3,64	-1,00
corral	100,00	72	2	15,63	100,00 ± 0,00 †	1400	3	15,63	10,18
crx	89,71	1155	5	3,48	90,22 ± 0,09 †	6249	9	3,99	-0,46
flare	84,71	390	5	3,85	84,71 ± 0,00 †	2300	6	3,85	2,81
glass	84,58	234	10	9,81	85,05 ± 0,00 †	4600	7	10,28	-1,88
glass2	88,34	189	2	6,13	88,34 ± 0,00 †	2900	2	6,13	-1,25
iris	96,67	24	3	2,00	96,67 ± 0,00 †	200	3	2,00	0,67
mofn-3-7-10	96,09	440	7	9,76	96,09 ± 0,00 †	7400	8	9,76	4,68
pima	80,47	176	3	2,74	80,47 ± 0,00 †	3800	3	2,74	0,01
vote	96,32	1040	4	6,21	97,93 ± 0,00 †	9600	11	7,82	3,22

Tabla 5.10: Resultados experimentales de los algoritmos TAN-voraz, TAN-EDA y algoritmo de Friedman

Ambos algoritmos, TAN-voraz y TAN-EDA, son muy superiores a la búsqueda de

Friedman basada en la cantidad de información mutua entre las variables predictoras. TAN-voraz consigue una mejora media, respecto a naïve-Bayes, de 5,74 % con una media de 4 arcos en las estructuras obtenidas. Por su parte, TAN-EDA consigue una mejora media de 6,09 %, con un número medio de 6 arcos en las estructuras obtenidas.

También se ha utilizado el test no paramétrico de Mann-Whitney para validar la hipótesis nula de la misma distribución de densidad para cada uno de los conjuntos de prueba. Esta tarea se realizó con el paquete estadístico S.P.S.S. versión 11.50. Los resultados de estas pruebas se muestran a continuación.

- Pruebas del clasificador TAN-EDA:
  - Naïve-Bayes vs. TAN-EDA. Valor de ajuste en todos los conjuntos de datos: $p < 0{,}001$.
  - TAN-voraz vs. TAN-EDA. Valor de ajuste en 5 conjuntos de datos: $p < 0{,}001$. Los conjuntos de datos son: *breast*, *cleve*, *crx*, *glass* y *vote*.

Por tanto, TAN-EDA muestra una mejora significativa respecto a TAN-voraz en sólo 5 conjuntos de los 11 evaluados.

Conclusiones

En esta sección se han presentado dos algoritmos para la búsqueda de estructuras tipo TAN. El primer algoritmo, denominado TAN-voraz, está basado en una búsqueda voraz, mientras que el segundo, denominado TAN-EDA, está basado en una búsqueda heurística con EDAs. Ambos algoritmos han sido comparados con las búsquedas realizadas por Friedman, que están basadas en la cantidad de información mutua entre las variables predictoras.

Los resultados obtenidos muestran que ambos algoritmos desarrollados son muy superiores al enfoque de Friedman.

Líneas de trabajo futuro

Las estructuras TAN han demostrado ser muy superiores al clasificador naïve-Bayes. Sin embargo, es muy posible que los resultados de TAN puedan ser mejorados introduciendo la idea expuesta en el algoritmo IENB, es decir, realizando una estimación por intervalo de los parámetros necesarios.

5.9. Búsqueda de clasificadores MB

5.9.1. Propuesta

El *Markov blanket* (MB) de una variable, en una red Bayesiana, es el conjunto de padres, hijos, y padres de hijos de esa variable. De este modo, el objetivo es buscar individuos en los que se cumpla que el MB de la variable clase sea el conjunto de todas las variables predictoras del problema.

El enfoque MB presenta dos problemas principales. El primero de ellos es que el espacio de búsqueda no se reduce de forma considerable, lo que lleva a un proceso de aprendizaje estructural lento y muy costoso. El segundo problema consiste en que MB tiene la tendencia de sobreajustarse a los datos de entrenamiento, dando por ello peores resultados de los esperados.

Con el objetivo de paliar ambos problemas, se ha realizado un nuevo enfoque que relaja las condiciones que deben cumplirse para que la estructura sea considerada MB. Para ello, se añaden las siguientes restricciones:

- Un nodo variable predictor, padre de la clase, no puede ser a su vez padre de ninguna otra variable.
- Un nodo variable predictor, que no sea padre ni hijo de la clase, sólo puede ser padre de un hijo de la clase como máximo.
- Un nodo variable predictor, hijo de la clase, no puede ser padre de ninguna otra variable.

Con estas restricciones el espacio de búsqueda de clasificadores se reduce considerablemente, y, además, al igual que pasaba con BAN y TAN, al reducir el número posible de padres de una variable, se facilita el cálculo de las tablas de probabilidad condicional.

En esta sección se propone, teniendo en cuenta las anteriores restricciones, un algoritmo basado en EDAs que realiza la búsqueda de estructuras de tipo MB. Este algoritmo ha sido denominado MB-EDA.

5.9.2. Formato de los individuos

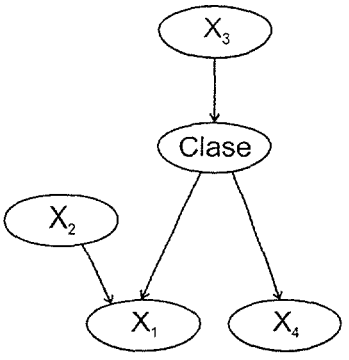
Los individuos MB muestran una pequeña diferencia respecto a los anteriores BAN y TAN. La diferencia radica en que ahora, la variable clase, no es forzosamente padre del resto de variables, sino que puede ser padre, hija o no tener relación alguna.

Por tanto, la matriz de representación tiene un rango igual al número de variables totales, incluida la variable a clasificar. La matriz sigue siendo cuadrada y no se permiten ciclos. En la figura 5.17 se muestra un ejemplo de una estructura MB junto al individuo que la representa.

Algoritmo MB-EDA

En la figura 5.18 se encuentra el pseudocódigo del algoritmo de corrección. Aunque a primera vista este algoritmo pueda parecer complicado de entender, a poco que se profundiza en su estudio se ve que es muy sencillo y fácil de aplicar. Lo importante es considerar tres grandes pasos:

- Paso 1. Estudio de los hijos de la variable clase.
- Paso 2. Estudio de los padres de la variable clase.
- Paso 3. Estudio del resto de variables.



	Clase	X_1	X_2	X_3	X_4
Clase	0	1	0	0	1
X_1	0	0	0	0	0
X_2	0	1	0	0	0
X_3	1	0	0	0	0
X_4	0	0	0	0	0

Figura 5.17: Representación de los individuos MB

Tal y como se puede observar, estos tres pasos están bien diferenciados en el pseudocódigo. A continuación se explica uno a uno cada paso.

En el primer paso se estudian los hijos de la variable clase C. Esto significa estudiar la primera fila de la matriz. Cuando se encuentra un 1 en una columna, por ejemplo, en la correspondiente a la variable X_j, se comprueba en el grafo del individuo si esa variable aparece también como padre de C. En caso negativo, X_j es hija de C, y siguiendo las condiciones de MB, ya no puede ser padre de nadie, y por tanto, se pone su fila a ceros. En caso afirmativo, se debe elegir aleatoriamente entre dejar a la variable como hija de C o como padre. Si se elige dejarla como hija, se hace lo mismo que en el caso anterior, y si se deja como padre, simplemente se pone un 0 donde estaba el 1, y ya se analizará la nueva situación cuando corresponda en el siguiente paso. Es importante destacar que las filas cuyas variables han resultado hijas de C, y que han sido puestas a 0, se bloquean con el fin de llevar la cuenta de los hijos de C y con el fin de que no puedan ser modificadas bajo ningún otro criterio.

En el segundo paso se estudian los padres de C. Ya no es necesario estudiar las variables que son hijas de C, por lo que dichas filas se pasan por alto directamente. En este caso, que una variable X_j sea padre de C, significa que hay un 1 en la primera columna de la fila correspondiente a X_j. En esta situación, se sabe de antemano que esa variable no es hija de C por el paso anterior, y ya que las condiciones de MB imponen que un padre de C ya no puede ser padre ni hijo de nadie más, se pone a 0 el resto de su fila y toda su columna. Además, se bloquea la columna con el mismo fin que en el caso de las filas.

Por último, quedan las variables predictoras que no tienen ninguna relación con la variable a clasificar C, es decir, las variables cuyas filas y columnas no han sido bloqueadas. Estas variables forzosamente tienen que quedar en alguna de estas situaciones:

- X_i es padre de un hijo de C: para que esto ocurra, debe haber al menos un 1 en alguna columna de la fila X_i, en posiciones que correspondan a variables hijas de C (ya que esto viene impuesto por las condiciones de MB). Aquellos unos que no correspondan a este tipo de variables se ponen a 0. Si hay más de un 1 de este tipo, se elige uno aleatoriamente. Ahora, puesto que X_i ya no puede relacionarse con nadie más, el resto de su fila y toda su columna deben ser 0, y además, se bloquea la columna.

- Padre o hija de C: en caso de que no haya unos en las columnas cuyas variables son hijas de C, se hace que sea padre o hija de C aleatoriamente. Si se elige como hija de C, se pone un 1 en la columna de la primera fila que corresponda, se pone a 0 toda la fila, y se bloquea la fila; si resulta ser padre, se pone un 1 en la primera columna de la fila correspondiente, y se pone a 0 el resto de la fila y toda la columna, bloqueando ésta como último paso.

De esta forma se aseguran todas las condiciones que debe cumplir un individuo para ser un clasificador MB según nuestros criterios.

5.9.3. Resultados

Resultados experimentales

El algoritmo propuesto, MB-EDA, presenta una mejora media de 5,80 % en los 11 conjuntos de datos en los que ha sido ejecutado. Además, las estructuras MB encontradas son, por lo general, muy sencillas, con un número medio de 10 arcos (hay que tener en cuenta que en este valor están incluidos los arcos entre la variable clase y las variables predictoras).

	MB-EDA	*Evaluaciones*	*Arcos*	*Mejora*
breast	97,85 ± 0,12 †	6300	10	0,71
cleve	90,76 ± 0,00 †	12900	13	6,94
corral	95,31 ± 0,00 †	2300	6	10,94
crx	90,72 ± 0,09 †	7200	15	4,49
flare	85,08 ± 0,00 †	3400	9	4,22
glass	85,05 ± 0,00 †	5600	9	10,28
glass2	88,34 ± 0,00 †	2500	9	6,13
iris	96,67 ± 0,00 †	500	4	2,00
mofn-3-7-10	95,31 ± 0,00 †	7300	10	8,98
pima	80,21 ± 0,00 †	5800	8	2,48
vote	96,78 ± 0,07 †	9200	16	6,67

Tabla 5.11: Resultados experimentales del algoritmo MB-EDA

También se ha utilizado el test no paramétrico de Mann-Whitney para validar la hipótesis nula de la misma distribución de densidad para cada uno de los conjuntos de prueba. Esta tarea se realizó con el paquete estadístico S.P.S.S. versión 11.50. Los resultados de estas pruebas se muestran a continuación.

- Pruebas del clasificador MB-EDA:
  - Naïve-Bayes vs. MB-EDA. Valor de ajuste en todos los conjuntos de datos: $p <$ 0,001.

Conclusiones

En esta sección se ha presentado el algoritmo MB-EDA, un algoritmo heurístico basado en EDAs para la búsqueda de clasificadores supervisados de tipo MB.

$listaFilasBloq = \emptyset$ y $listaColumnasBloq = \emptyset$
Poner a 0 la diagonal principal
Para cada A_i (Estudio de los hijos de C)
  Si $matriz(C, A_i) = 1$ y $matriz(A_i, C) = 1 \Rightarrow$
    Elegir aleatoriamente si A_i se queda como padre o hijo de C
    Si A_i se queda como hijo $\Rightarrow$
      Poner a 0 fila A_i y $A_i \rightarrow listaFilasBloq$
    Si A_i se queda como padre $\Rightarrow$
      $matriz(C, A_i) = 0$
  Si $matriz(C, A_i) = 1$ y $matriz(A_i, C) = 0 \Rightarrow$
    Poner a 0 fila A_i y $A_i \rightarrow listaFilasBloq$
Para cada $A_i \notin listaFilasBloq$ (Estudio de los padres de C)
  Si $matriz(A_i, C) = 1 \Rightarrow$
    Poner resto de fila a 0
    Poner a 0 columna A_i y $A_i \rightarrow listaColumnasBloq$
Para cada fila $A_i \notin listaFilasBloq$ y $A_i \notin listaColumnasBloq$
  $listaCandidatas = \emptyset$
  Para cada columna $A_j \notin listaColumnasBloq$
    Si $matriz(A_i, A_j) = 1$ y $A_j \in listaFilasBloq \Rightarrow$
      $A_j \rightarrow listaCandidatas$
    $matriz(A_i, A_j) = 0$
  Calcular $numUnos$ fila A_i
  Si $numUnos > 0 \Rightarrow$
    Elegir aleat. $A_j \in listaCandidatas$ y poner $matriz(A_i, A_j) = 1$
    Poner a 0 columna A_j elegida y $A_j \rightarrow listaColumnasBloq$
  Si $numUnos = 0 \Rightarrow$
    Elegir aleatoriamente si A_i se queda como padre o hijo de C
    Si A_i se queda como hijo $\Rightarrow$
      $matriz(C, A_i) = 1$ y $A_i \rightarrow listaFilasBloq$
    Si A_i se queda como padre $\Rightarrow$
      $matriz(A_i, C) = 1$ y $A_i \rightarrow listaColumnasBloq$

Figura 5.18: Pseudocódigo de la corrección de un individuo MB

Los resultados obtenidos son muy buenos, ya que se consigue mejorar sustancialmente los resultados del clasificador naïve-Bayes, a la vez que se mantiene una buena simplicidad en las estructuras encontradas.

Parte IV

APLICACIÓN EN BIOLOGÍA COMPUTACIONAL

Capítulo 6

Estado del arte de la predicción de la estructura secundaria de las proteínas

La **genómica** y la **proteómica** son actualmente dos de los campos más importantes de la biología computacional, en los que la minería de datos está siendo más utilizada. En este capítulo se analiza el estado del arte de uno de los problemas más conocidos de la proteómica: la predicción de la estructura secundaria de las proteínas (PSSP). Las primeras secciones del capítulo contienen una introducción a las proteínas para una mejor compresión de este problema.

El índice del capítulo es el siguiente:

- En la sección 6.1 se explica por qué es importante el estudio de las proteínas.
- En la sección 6.2 se realiza una introducción a las proteínas y se explican los cuatro niveles de estructura de las proteínas.
- En la sección 6.3 se analiza en profundidad la estructura secundaria de las proteínas, mostrando los posibles tipos de estructuras existentes.
- La sección 6.4 contiene el estado del arte en predicción de la estructura secundaria de las proteínas.
- Por último, en la sección 6.5, se analizan las matrices de sustitución y los métodos de alineamiento de secuencias, a través de los cuales se puede obtener la información evolutiva necesaria para solucionar el problema de la predicción de la estructura secundaria.

6.1. La importancia del estudio de las proteínas

6.1.1. Las proteínas y la vida

El término proteína, del griego $\pi\rho\omega\tau\varepsilon\upsilon\omega$ –ser el primero en influencia–, indica que todas las funciones básicas en biología dependen de proteínas específicas. Se puede decir

que no existe vida sin proteínas. Están presentes en cada célula y en cada organoide celular. La gran diversidad de estructuras de las proteínas nos indica la enorme cantidad de funciones que realizan:

- **Catalización biológica (enzimas).** La mayor parte de las reacciones químicas que suceden en los sistemas biológicos están catalizadas por enzimas, que son proteínas -aunque realmente no todos los enzimas son proteínas, como por ejemplo la *ribozima* que es una molécula de ARN-. Algunas reacciones sencillas, como la hidratación del dióxido de carbono o la modificación de pequeñas moléculas orgánicas, son catalizadas por enzimas. Por otra parte, y, a gran escala, también existen complejas transformaciones de grandes moléculas que son catalizadas por enzimas. Las enzimas están presentes en la lectura de la información genética almacenada en el ADN, el primer paso en la propia síntesis de las proteínas. Además, los componentes no protéicos de las células son sintetizados por enzimas. En resumen, las enzimas son un componente central del funcionamiento interno de las células.

- **Almacenamiento y transporte.** Las proteínas están involucradas en el almacenamiento y transporte de partículas, que pueden ir desde electrones hasta macromoléculas. Por ejemplo, el hierro es transportado por la *transferrina.* Otro ejemplo es la hemoglobina, que se encuentra en las células rojas de la sangre y que transporta oxígeno desde los pulmones u otros tejidos e interviene en el transporte de dióxido de carbono a los pulmones. Algunas proteínas forman poros en membranas celulares a través de los cuales pasan los iones. El transporte de las mismas proteínas a través de membranas también depende de otras proteínas.

- **Regulación biológica (hormonas).** Las proteínas están presentes en la transmisión de impulsos nerviosos, actuando como receptores de pequeñas moléculas que cruzan las uniones que separan las células nerviosas. Dentro de un organismo los procesos biológicos deben estar coordinados entre células del mismo tejido e, incluso, entre organismos diferentes. Esto se logra a través de las moléculas de señalización llamadas hormonas. Algunas hormonas son proteínas, como por ejemplo la insulina.

- **Función inmunológica (anticuerpos).** El sistema inmunológico depende de la producción de anticuerpos, que son proteínas capaces de acoplarse a partículas específicas exteriores tales como bacterias y virus.

- **Función regularizadora.** La información requerida para sintetizar proteínas es almacenada en genes (secuencias de ADN). La orquestación necesaria para la actividad celular necesita que varias proteínas estén presentes en las cantidades apropiadas en el momento correcto. Las enzimas sintetizan proteínas traduciendo secuencias de ADN. Esta producción puede ser estimulada o inhibida por otras proteínas en complejos mecanismos de retroalimentación.

- **Función estructural.** Algunas proteínas tienen un papel estructural, proporcionando mecanismos de soporte. El esqueleto de una célula consiste en una compleja red de filamentos protéicos. A gran escala, la contracción muscular depende de la acción de grandes cadenas de proteínas. Otro material orgánico, como el pelo y los huesos, están también basados en proteínas.

6.1.2. Estructura de las proteínas

Las proteínas son heteropolímeros lineales de longitud fija. Un determinado tipo de proteínas siempre tiene el mismo número y composición de monómeros, pero diferentes proteínas tienen diferente número de monómeros, desde unas pocas decenas hasta millares. Los monómeros son aminoácidos, y existen 20 diferentes, cada uno con sus propiedades químicas. Existe, por tanto, una gran diversidad de posibles secuencias de proteínas.

Las cadenas lineales se pliegan en específicas formas tridimensionales que vienen determinadas por la secuencia de aminoácidos. De esta forma, la estructura tridimensional de las proteínas es extremadamente diversa, desde totalmente fibrosas hasta globulares. La estructura de las proteínas puede ser determinada a nivel atómico por la difracción de rayos-X y los estudios de difracción de neutrones de proteínas cristalizadas. Más recientemente está siendo utilizada la resonancia magnética nuclear (RMN) espectroscópica de proteínas en disolución. Sin embargo, existen todavía muchas proteínas cuyas estructuras no han sido resueltas.

Las secuencias de proteínas están codificadas en el ADN, el contenedor de la información genética, que realmente es una molécula lineal compuesta de cuatro tipos de bases (monómeros que forman el alfabeto genético). En principio, debería ser posible traducir una secuencia de un gen a su cadena de aminoácidos y predecir la estructura tridimensional de la cadena resultante. Sin embargo, hay muchos problemas que hacen este cometido muy complicado, como se tendrá la oportunidad de comprobar.

6.1.3. Beneficios: pasado, presente y futuro

- **Medicina.** Comprender la función de los enzimas permite el diseño de fármacos que los puedan inhibir por propósito terapéutico. Las proteínas pueden ser diseñadas con intención terapéutica. Así, por ejemplo, un área de investigación actual se centra en la ingeniería de la insulina, intentando que se disocie en su forma activa más rápidamente, para, de esta forma, acelerar la respuesta si se inyecta en un paciente diabético.

- **Agricultura.** Así como se pueden producir proteínas terapéuticas y fármacos para propósitos médicos y veterinarios, se puede utilizar el conocimiento de la estructura y función de las proteínas para tratar enfermedades de las plantas y para modificar el crecimiento y desarrollo de las semillas. Como ejemplo se pueden citar las frutas de maduración rápida.

- **Industria proteica.** La industria presenta un gran potencial para la síntesis de enzimas, por ejemplo, para realizar procesos a escala masiva. Actualmente hay un campo de investigación dedicado a las lipasas para la descomposición de grasas. Un ejemplo doméstico de la aplicación de la ciencia proteica es la aparición de los detergentes biológicos que contienen enzimas.

6.2. Introducción a las proteínas

6.2.1. Dogma central de la biología molecular

Los ácidos nucleicos son macromoléculas de suma importancia biológica. Todos los organismos vivos contienen ácidos nucleicos en la forma de ácido desoxirribonucleico (ADN) y ribonucleico (ARN). Algunos virus sólo contienen ARN, mientras que otros sólo poseen ADN.

$$\text{ADN} \xrightarrow{\text{Transcripción}} \text{ARN} \xrightarrow{\text{Traducción}} \text{Proteínas}$$

Figura 6.1: Dogma central de la biología molecular

El ADN constituye el depósito fundamental de información genética. Esta información es copiada o trascrita en las moléculas de ARN, cuyas secuencias de nucleótidos contienen el código para las secuencias específicas de aminoácidos. Entonces se produce la síntesis de las proteínas, en un proceso en el que interviene la traducción del ARN. Es frecuente referirse a esta serie de fenómenos como el *dogma central de la biología molecular*, el cual puede resumirse tal como se ve en la figura 6.1. Además es importante saber que muchas veces en lugar de trascripción utilizamos el término *síntesis de ARN*, y en lugar de traducción *síntesis de proteínas*.

6.2.2. Los aminoácidos

Las unidades que constituyen las proteínas son los aminoácidos. Un aminoácido es un ácido orgánico en el cual el carbono próximo al grupo carboxilo ($COOH$) (llamado carbono alfa) está unido también a un grupo amino (NH_2). Además, el carbono alfa se une a una cadena lateral (R), que es diferente para cada aminoácido. Las cadenas laterales también se denominan radicales libres o simplemente radicales. La estructura de un aminoácido se puede ver en la figura 6.2.

Las propiedades de los aminoácidos dependen de la composición química de la cadena lateral. Así, la *lisina* y la *arginina* son básicos porque la cadena lateral contiene un grupo amino extra; en los aminoácidos ácidos hay un grupo carboxilo adicional (*ácidos aspártico* y *glutámico*). Dada la presencia simultánea de grupos ácidos y básicos, los aminoácidos tienen a la vez cargas positivas y negativas y, por tanto, son moléculas anfóteras. En los sistemas biológicos se pueden codificar 20 aminoácidos, de los cuales 2 son ácidos: *ácido aspártico* y *ácido glutámico*; tres son básicos: *lisina*, *arginina* e *histidina*; siete son neutros y al mismo tiempo polares (es decir, hidrofílicos): *serina*, *treonina*, *tirosina*, *triptófano*, *asparagina*, *glutamina* y *cisteína* y ocho son neutros no polares (es decir, hidrófobos): *glicina*, *alanina*, *valina*, *leucina*, *isoleucina*, *fenilalanina*, *prolina* y *metionina*. Es importante advertir que dos aminoácidos, *metionina* y *cisteína*, contienen un átomo de azufre. Entre dos *cisteínas* se puede formar fácilmente un puente de disulfuro ($-S-S-$) covalente porque los átomos de H de los grupos $-SH$ pueden ser eliminados.

Por lo común, se abrevian los nombres de los aminoácidos utilizando sus tres primeras letras, o incluso, cada aminoácido tiene una letra asignada, lo que hace mucho más sencillo

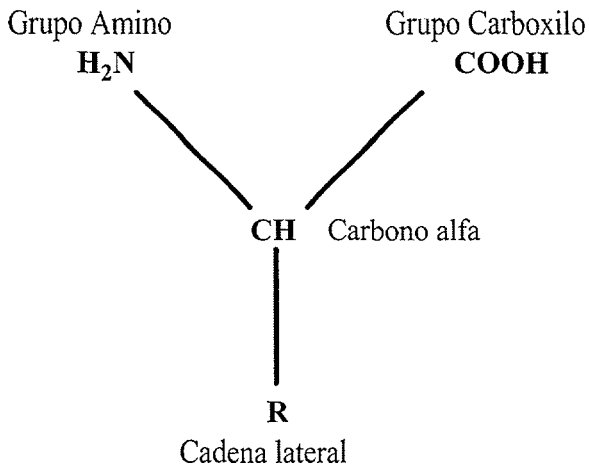


Figura 6.2: Estructura de un aminoácido

su manejo a nivel de programación. En la tabla 6.1 se pueden ver todos los aminoácidos con sus correspondientes abreviaturas.

6.2.3. Enlaces peptídicos

La condensación de aminoácidos para formar una molécula proteica se produce de tal modo que el grupo ácido de un aminoácido se combina con el grupo básico del adyacente, con la formación simultánea de una molécula de agua. El proceso de formación del enlace peptídico se puede observar en la figura 6.3. La unión $NH - CO-$ se conoce como unión peptídica o puente peptídico.

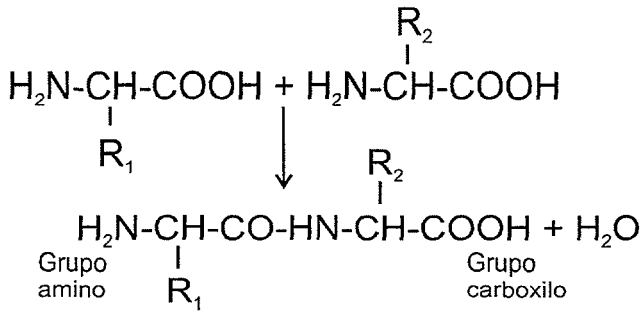


Figura 6.3: Formación de un enlace peptídico entre dos aminoácidos

En los polímeros lineales siempre existe en los extremos un grupo amino (aminoácido N-terminal) y un grupo carboxilo (aminoácido C-terminal) que también se pueden observar en la figura 6.3. La molécula formada mantiene su carácter anfótero, ya que siempre existe un grupo ácido en uno de sus extremos y uno básico en el otro, además de los residuos laterales que son ácidos o básicos.

Nombre español	Nombre inglés	Abreviatura	Letra	Clasificación
Alanina	Alanine	Ala	A	Neutro no polar
Cisteína	Cysteine	Cys	C	Neutro polar
Ácido Aspártico	Aspartic Acid	Asp	D	Ácido
Ácido Glutamico	Glutamic Acid	Glu	E	Ácido
Fenialanina	Phenylalanine	Phe	F	Neutro no polar
Glicina	Glycine	Gly	G	Neutro no polar
Histidina	Histidine	His	H	Básico
Isoleucina	Isoleucine	Ile	I	Neutro no polar
Lisina	Lysine	Lys	K	Básico
Leucina	Leucine	Leu	L	Neutro no polar
Metionina	Methionine	Met	M	Neutro no polar
Asparagina	AsparagiNe	Asn	N	Neutro no polar
Prolina	Proline	Pro	P	Neutro no polar
Glutamina	Glutamine	Gln	Q	Neutro no polar
Arginina	ARginine	Arg	R	Básico
Serina	Serine	Ser	S	Neutro polar
Treonina	Threonine	Thr	T	Neutro polar
Valina	Valine	Val	V	Neutro no polar
Triptófano	Tryptophan	Trp	W	Neutro polar
Tirosina	TYrosine	Try	Y	Neutro polar

Tabla 6.1: Los aminoácidos y sus abreviaturas

6.2.4. Cuatro niveles de estructura proteica

Por lo común se diferencian cuatro niveles de estructura de las proteínas. Estos cuatro niveles están representados en la figura 6.4.

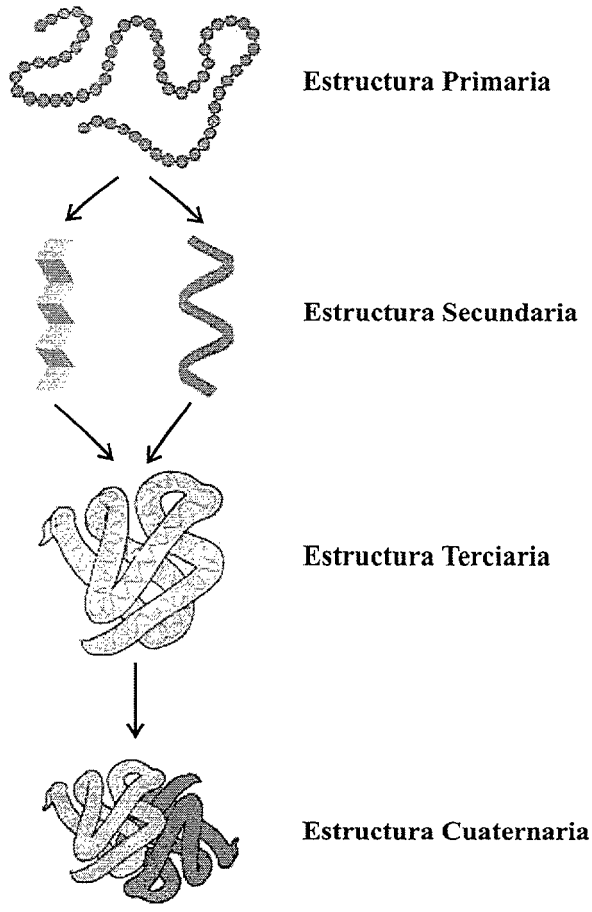


Figura 6.4: Cuatro niveles de estructura proteica

La estructura primaria es la secuencia de residuos que forman una cadena ligada por uniones peptídicas, donde cada residuo es uno de los 20 posibles aminoácidos. La secuencia de residuos de una proteína determina el nivel más importante en la estructura de la molécula.

La estructura secundaria es la organización espacial de los aminoácidos que se encuentran próximos en la cadena peptídica. A este tema se le dedica por entero la siguiente sección.

La estructura terciaria es la forma en que las distintas regiones se disponen entre sí, es decir, es la relación tridimensional de los segmentos de aminoácidos.

La estructura cuaternaria es la disposición de subunidades proteicas en proteínas complejas formadas por dos o más de dichas subunidades. Por ejemplo, la molécula de hemoglobina está compuesta por cuatro subunidades, dos de ellas denominadas α y dos

β.

6.2.5. Programas de visualización de proteínas

Existen tres programas (realmente existen muchos más, pero éstos son los más utilizados), que permiten visualizar proteínas en tres dimensiones. Son *Chime*, *Rasmol* y *Kinemage*. Como entrada cogen ficheros de extensión PDB (*Protein Data Bank*), que contienen las coordenadas en tres dimensiones de todos los átomos de la proteína. A través de estos programas se pueden visualizar imágenes como la que se ve en la figura 6.5, obtenida con el programa Rasmol.

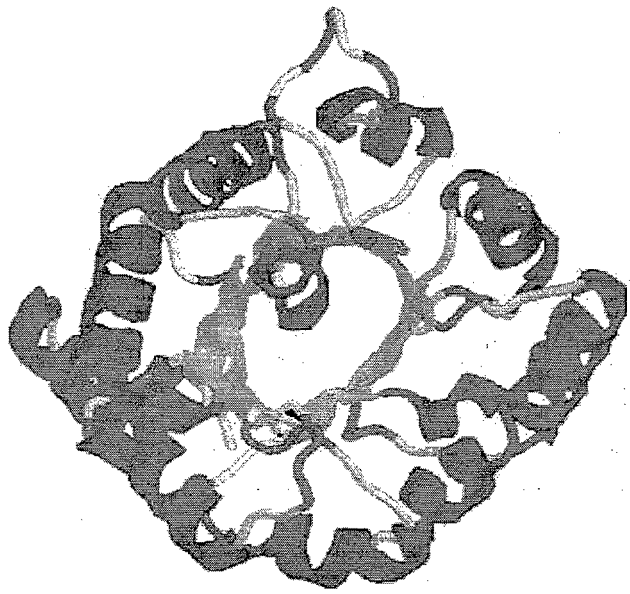


Figura 6.5: La proteína 1TIM visualizada en forma de dibujo (*cartoon*)

6.3. Estructura secundaria de las proteínas

6.3.1. Tipos de estructuras secundarias

En la estructura secundaria de las proteínas se pueden observar tres estructuras diferentes: hélices, β laminas y *coil* (resto). A continuación se describe cada una de ellas.

Hélices

Algunas regiones pueden tener una estructura en forma de cilindro, o estructura en hélice. En la hélices la cadena polipeptídica se enrolla alrededor de un cilindro imaginario. La estructura hélice es bastante común en las proteínas y se obtiene cuando se forman puentes de hidrógeno entre un aminoácido y otro aminoácido cercano. Existen 3 tipos de

Hélice	Frecuencia	Puentes H	Átomos/giro
α hélice	Abundante	i, i + 4	13
3_{10} hélice	Infrecuente	i, i + 3	10
π hélice	Rara	i, i + 5	16

Tabla 6.2: Los diferentes tipos de hélices y sus características

hélices: α, 3_{10} y π, dependiendo de cómo se establezcan los puentes de hidrógeno entre los aminoácidos. En la tabla 6.2 se muestran las características de cada una de estas hélices.

Los posibles tipos de hélices son:

- α **hélice.** Llamada α porque fue la primera descubierta por Pauling y Corey a comienzos de la década de 1950. Es la más abundante, dándose en la vida real en el 32-38 % de los residuos. Tiene una longitud media de 10 residuos. Su formación se debe al establecimiento de puentes de hidrógeno entre un aminoácido y otro que está 4 posiciones por debajo de éste. La α hélice presenta un total de 13 átomos por giro.
- 3_{10} **hélice.** Es infrecuente encontrarla, dándose en un 3.4 % de los residuos de una proteína. Su formación se debe al establecimiento de puentes de hidrógeno entre un aminoácido y otro que está 3 posiciones por debajo de éste. Las α hélices empiezan o terminan el 25 % de los casos con algunos residuos en forma de 3_{10} hélice.
- π **hélice.** De rara aparición en las proteínas, su formación se debe al establecimiento de puentes de hidrógeno entre un aminoácido y otro que está 5 posiciones por debajo de éste. A veces existe un giro de π hélice al inicio de una α hélice.

En la figura 6.6 se puede observar un dibujo de estas hélices.

Beta laminar

En la estructura β laminar los aminoácidos adoptan la conformación de una hoja de papel plegada. Representa el 20-28 % de los residuos de una proteína.

No existen ni puentes de hidrógeno ni interacciones de van der Waals entre residuos vecinos. Estas formaciones son estables como parte de una lámina, en donde las cintas (*strands*) alineadas crean la estabilización. Por tanto, la estructura β laminar se hace estable debido a la existencia de puentes de hidrógeno entre los residuos de dos cintas, ya sean estas paralelas o antiparalelas. En la figura 6.7 se puede observar la formación de puentes de hidrógeno en unas cintas antiparalelas. En la figura 6.8 se pueden observar los puentes de hidrógeno que se forman en unas cintas paralelas. Aunque al realizar una inspección visual no lo parezca, tanto las formaciones paralelas como las antiparalelas son exactamente igual de estables y de probables.

Coil

El resto de estructuras que se encuentran dentro de una proteína son denominadas de forma genérica *coil*.

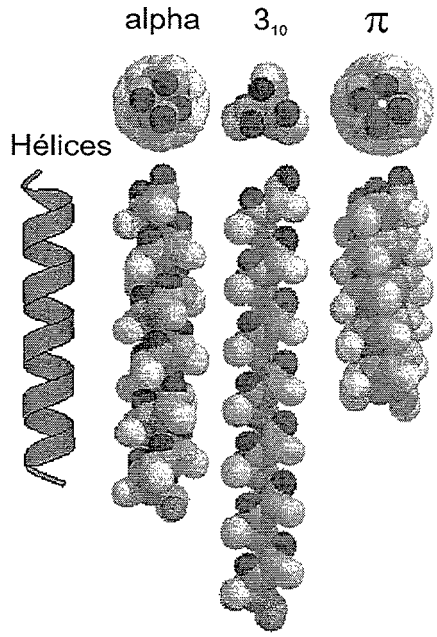


Figura 6.6: Las hélices alpha, 3_{10} y π

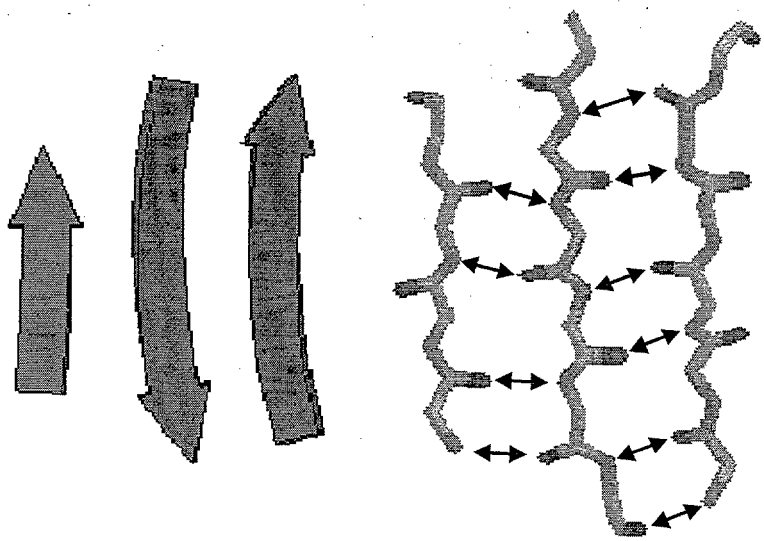


Figura 6.7: Formación de puentes de hidrógeno en cintas antiparalelas

6.3.2. Programas de obtención de la estructura secundaria de las proteínas

A la hora de poder realizar la predicción de la estructura secundaria de las proteínas se debe partir de una serie de proteínas cuya estructura terciaria es conocida. Una vez

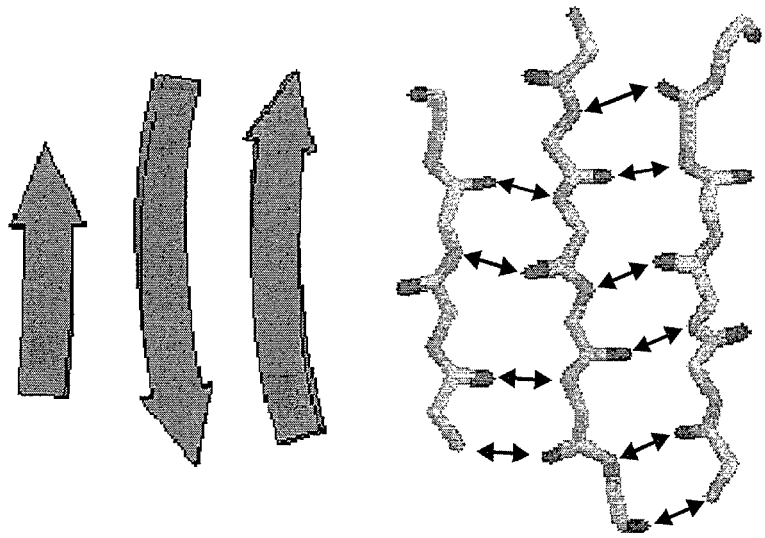


Figura 6.8: Formación de puentes de hidrógeno en cintas paralelos

seleccionadas las proteínas que formarán parte del conjunto de datos, se debe obtener la estructura secundaria de estas proteínas a partir de su estructura tridimensional. Para realizar este trabajo existen tres programas diferentes: DSSP, DEFINE y STRIDE. Estos programas cogen los ficheros en formato PDB y, a partir de ellos, calculan de forma bastante precisa la estructura secundaria. DSSP y DEFINE coinciden en un 95 % de las posiciones, mientras que DSSP y STRIDE sólo en un 71 % de las posiciones. Debido a su uso masivo en este tipo de técnicas, en esta tesis se usará el programa DSSP.

Nombre de la clase	Abreviatura
α hélice	H
3_{10} hélice	G
π hélice (extremadamente rara)	L
β-strand	E
β-bridge	B
β-turn	T
bend	S
resto	C

Tabla 6.3: Estados de la estructura secundaria

Estos programas calculan la estructura secundaria en ocho estados diferentes. De estos ocho estados se realiza una proyección a sólo tres estados, que son los que se utilizarán para la predicción de la estructura secundaria: hélice (H), β lámina (E) y *coil* (L). Existen tres métodos para realizar la proyeccción de los ocho estados a tres [CB99]. Los ocho estados diferentes de la estructura secundaria se muestran en la tabla 6.3 y en la tabla 6.4

se muestran los métodos para proyectar esos ocho estados en tres. De todas las posibles proyecciones que se pueden realizar, la que resulta más difícil de predecir y la que se su usa de forma más generalizada, es la realizada por el *método A*. Ésta será la proyección que se utilizará a lo largo de toda la tesis.

Método A	E y B ⇒ E G y H ⇒ H resto ⇒ L
Método B	E ⇒ E H ⇒ H resto ⇒ L EE y HHHH ⇒ L
Método C	GGGHHHH ⇒ HHHHHHH B y GGG ⇒ L H ⇒ H E ⇒ E

Tabla 6.4: Métodos para la proyección de las ocho clases en tres clases

6.4. Predicción de la estructura secundaria de las proteínas

El estudio de la estructura secundaria de las proteínas es de vital importancia, ya que [SLB00] proporciona un punto de partida para la predicción de la estructura tridimensional de las proteínas y puede mejorar significativamente el análisis de secuencias o las técnicas de *threading* [RCB96], que ayudan en la determinación de la estructura y función de las proteínas.

La predicción de la estructura secundaria ha sido ampliamente estudiada durante cuatro décadas y abordada por una amplia variedad de métodos. Los avances en la predicción de la estructura secundaria se han basado, en cierta medida, en desarrollos pertenecientes a la teoría del aprendizaje automático, comenzando con enfoques basados en reglas y pasando por enfoques basados en redes de neuronas.

A continuación se muestra cómo se ha desarrollado esta historia y las innovaciones que gran cantidad de investigadores han aportado al problema. Se explican las tres generaciones existentes de métodos de predicción de la estructura secundaria, y en cada una de estas generaciones, se explica en detalle alguno de los métodos más destacados.

6.4.1. Primera generación

Las primeras estructuras tridimensionales de proteínas, correspondientes a la hemoglobina y mioglobina, fueron publicadas en 1960 [KDS+60, PRC+60]. Casi una década antes, Pauling y Corey propusieron una explicación a la formación de ciertas configuraciones locales como α hélices y β láminas [PC51, PCB51]. Poco después, y todavía antes de que se determinara experimentalmente la primera estructura, se realizó el primer intento para relacionar el contenido de determinados aminoácidos con la formación de α hélices

[SGC57]. La idea se amplió para relacionar el contenido de todos los aminoácidos con las α hélices y las β láminas [BdLBF60, Blo62].

La principal característica de los métodos de primera generación es el uso de estadísticas de residuos individuales para realizar la predicción. A pesar de que los resultados inicialmente fueron positivos, en 1983 se hizo evidente que la exactitud de estos métodos había sido sobreestimada [RSS94a].

Reglas de Chou-Fasman

Un primer acercamiento al problema de la predicción de la estructura secundaria de las proteínas lo realizaron Chou y Fasman en 1974 [CF74], empleando las 15 estructuras disponibles en ese momento. Varios años después se actualizó el método con una mayor cantidad de estructuras. Su método implica asignar a cada aminoácido la probabilidad de que forme una hélice o una β lámina, basándose en sus frecuencias conocidas. Para predecir la estructura secundaria de una nueva secuencia se buscan regiones contiguas de residuos con una alta probabilidad de formar una determinada estructura secundaria. Es decir, si 4 de 6 residuos contiguos es probable que formen una hélice ó 3 de 5 es probable que formen una β lámina, entonces a esos residuos se les asigna esa estructura secundaria. Estas regiones son expandidas usando una regla similar.

6.4.2. Segunda generación

El incremento en el número de estructuras conocidas permitió el empleo de estadísticas más detalladas, pasándose a la segunda generación de predicción de la estructura secundaria. Normalmente se toman de 11 a 21 residuos contiguos de la proteína y se recopilan estadísticas con el fin de determinar con qué probabilidad el residuo que se encuentra en la posición central de ese segmento está en un determinado estado de estructura secundaria. En la figura 6.9 se muestra el proceso seguido con las ventanas de aminoácidos en la segunda generación.

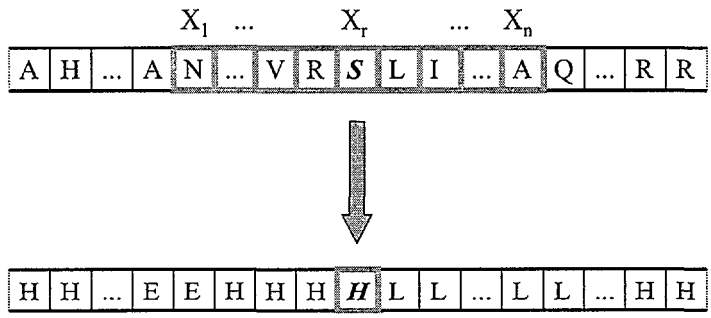


Figura 6.9: La segunda generación está basada en el uso de ventanas de aminoácidos para predecir la estructura secundaria

Los principales algoritmos de segunda generación se basan en:

- Información estadística [RS76, Nag77, GOR78, GGR87, BGL+88, GG88, LDS91, VDW91, JLTW93, MY93, DOB94, PF83, TT83].

- Propiedades físico-químicas [CK89].
- Motivos de secuencias [RKW91, BBB+88, QS88].
- Redes de neuronas [HK89, KCL90, SLX92, ZMW92, MS93, CK95, MARW92, GD95].
- Teoría de grafos [Kan88, MS97].
- Estadística multivariable [KMLS92, MKS92].
- Reglas expertas [FA95, ZB96, Aso97, MS97, YL93, SS94].
- Algoritmos del vecino más cercano [SS95, KS83].

La exactitud de los métodos de segunda generación es menor del 70 %.

Qian y Sejnowski

Qian y Sejnowski, en 1988, fueron los primeros que trataron de predecir la estructura secundaria empleando para ello redes de neuronas. La red de neuronas que emplearon es un *perceptrón multicapa*, completamente conectado, con una capa oculta [Bis95, Hay99].

La entrada a la red de neuronas es una pequeña ventana de aminoácidos, normalmente con un tamaño de 13 residuos. El objetivo de la red de neuronas es predecir la estructura secundaria del aminoácido que se encuentra en la posición central de la ventana que se le da como entrada a la red. La red tiene 3 nodos de salida que representan la probabilidad de que el aminoácido tenga como estructura secundaria hélice, β lámina o *coil*. De este modo, el problema de predicción de la estructura secundaria se convierte en un problema de clasificación.

Los aminoácidos se codifican con lo que se conoce como codificación ortonormal. A cada aminoácido se le asigna un vector binario, como por ejemplo (1, 0, 0,..., 0) ó (0, 1, 0,..., 0), cada uno de los cuales se elige de forma que sea único y ortonormal. De este modo, para codificar el alfabeto de los 20 aminoácidos protéicos se requiere que los aminoácidos se representen en un espacio de 20 dimensiones.

La red fue inicializada con pesos aleatorios y entrenada por medio del algoritmo de retropropagación del error [Bis95], usando un conjunto de datos seleccionado de la base de datos PDB (*Protein Data Bank*). El entrenamiento se realiza empleando instancias aleatorias de dicho conjunto de datos. Esto evita que se produzcan oscilaciones asociadas con correlaciones en los datos si son presentados de forma contigua [BBF+99b].

La red que construyeron superó a los métodos de predicción realizados hasta ese momento, aunque la falta de un conjunto de datos de prueba común hacía difícil la comparación. A pesar de esto, sus autores quedaron insatisfechos con el rendimiento de su red, en torno al 64 % de predicciones correctas, describiéndolo como "decepcionantemente bajo".

También sugirieron que no se podrían realizar mejoras significativas sobre los resultados obtenidos. La ausencia de diferencia en el rendimiento de la predicción entre una red de una sola capa y una red con capas ocultas, indica que no hay correlaciones de segundo orden para ser explotadas. Nuevos experimentos con estructuras artificiales indicaron que la red era capaz de usar información de segundo orden, por lo que los autores concluyeron que tal información no estaba presente en el conjunto de datos usado. Finalmente, los autores

sugirieron que hay un límite del 70 % en la exactitud de la predicción, demostrando la equivalencia entre el conjunto de datos y un conjunto de datos artificial que contiene sólo correlaciones de primer orden y un 30 % de ruido.

A pesar del pesimismo de los autores en cuanto a mejoras en la exactitud de la predicción, les siguieron una ola de artículos que utilizaban redes de neuronas para la predicción de la estructura secundaria (Holley y Karplus [HK89], Kneller y Cohen [KCL90], Stolorz et al. [SLX92]), esencialmente usando la misma técnica que Qian y Sejnowski, con la excepción de Maclin y Shalvik [MS93] que incorporaron las reglas de Chou-Fasman [CF74] en el diseño de la red.

6.4.3. Problemas de los métodos de primera y segunda generación

Todos los métodos de la primera y segunda generación comparten, al menos, dos de los siguientes problemas, estando presentes los tres problemas en la mayoría de los métodos:

- Exactitud en las predicciones inferior al 70 %.
- Las β láminas se predicen con una exactitud del 28-48 %, solamente un poco mejor que de forma aleatoria.
- Las hélices y las β láminas predichas son demasiado cortas.

El primer problema está comúnmente relacionado con dos características:

- La formación de la estructura secundaria está parcialmente determinada por un largo rango de interacciones, es decir, por contactos entre residuos que no son visibles por ningún método basado en las estadísticas de segmentos formados por 11 a 21 residuos contiguos.
- La asignación de la estructura secundaria varía entre un 5-12 % para una misma proteína, dependiendo de cómo cristalice.

El segundo problema se debe a que la formación de β láminas es determinada por más contactos no locales que la formación de hélices.

Estos problemas hacen que las predicciones sean muy difíciles de usar en la práctica.

6.4.4. Tercera generación

La principal característica de los métodos de tercera generación es el uso de información evolutiva para realizar las predicciones, lo que permite una mejora sustancial en los resultados obtenidos. La información evolutiva que se usa en la predicción, son los perfiles (*profiles*) de intercambio de residuos, extraídos de los alineamientos de la proteína de partida.

Las ventajas de estos métodos son:

- Exactitud en las predicciones en torno al 80 %.

- Exactitud en la predicción de las β láminas prácticamente igual que en las hélices y en *coil*.

Sin embargo, estos métodos siguen presentando una serie de problemas:

- Malos alineamientos llevan a malas predicciones.
- Se debe tener precaución al evaluar los resultados para proteínas con características inusuales.

Con el fin de entender la información evolutiva que usan los métodos de tercera generación, en la siguiente sección se analiza en profundidad el alineamiento de secuencias de proteínas. Los métodos actuales que mejor exactitud están obteniendo, PHD [RSS94a], PSIPRED [Jon99], SAM-T02 [KBH98], SSPro [BBF+99a, BBF+99b] y Prof [OK00], usan la información evolutiva obtenida con el programa PSI-BLAST que se analiza en la sección 6.5.2.

Método PHD de Rost y Sander

Uno de los métodos de predicción que más éxito obtuvo en su día, y que sigue de plena actualidad, es el denominado PHD, realizado por Rost y Sander [RS93, RSS93, RSS94a].

El método PHD está basado en tres niveles de redes neuronales:

1. **Primer nivel: secuencia-estructura.** Esencialmente tiene la misma arquitectura que el método desarrollado por Qian y Sejnowski, con la diferencia de que realiza las predicciones basándose en los perfiles de la proteína. En lugar de proporcionar como entrada a la red una simple secuencia, se le proporciona un alineamiento de secuencias de proteínas homólogas. Para realizar la predicción utiliza una ventana de 13 residuos. La salida de esta red neuronal es la estructura secundaria de la proteína.
2. **Segundo nivel: estructura-estructura.** La segunda red neuronal trata de mejorar la estructura secundaria obtenida por la primera red. La entrada de esta red neuronal es la salida de la red del primer nivel, cogiendo ventanas de 17 residuos consecutivos, con el objetivo de predecir la estructura secundaria del residuo central.
3. **Tercer nivel:** *jury decision.* El tercer nivel realiza una media aritmética de varias redes denominado *jury decision.* Como entrada toma la salida de los dos anteriores niveles, pero entrenados con diferente información y procedimientos de aprendizaje. La salida es una predicción combinada de los anteriores resultados.

El modo en que se extrae la información evolutiva de una proteína es el siguiente:

1. Buscar en una base de datos de estructuras conocidas secuencias similares, empleando métodos de alineamiento de secuencias.
2. Una vez obtenidas las secuencias similares a la secuencia cuya estructura se quiere predecir, se filtran aquellas que no superan un cierto umbral. Este umbral indica el número de identidades que tiene que haber entre las dos secuencias para que la secuencia se tenga en cuenta.

3. Se construye un perfil de secuencias de aminoácidos a partir de los homólogos encontrados.

4. Finalmente, este perfil se emplea para realizar la predicción.

6.4.5. Límite en la exactitud de la predicción de la estructura secundaria

La asignación de la estructura secundaria puede variar con una misma estructura primaria. Una razón es que las estructuras de las proteínas no son "rocas", sino objetos dinámicos con algunas regiones más móviles que otras. Otra razón es que cualquier método de asignación de la estructura secundaria debe seleccionar determinados umbrales (por ejemplo, DSSP elige un umbral en la energía de Coulomb de un puente de hidrógeno), dependiendo de los cuales pueden dar estructuras secundarias diferentes. Por lo tanto, las asignaciones varían en torno a un 5-15 % entre diferentes versiones de rayos-X o diferentes modelos RMN para la misma proteína (Andersen y Rost, resultado no publicado), y en torno a un 12 % entre homólogos estructurales [RS94].

Un valor en torno al 88 % constituye un límite superior operacional para la exactitud en la predicción de la estructura secundaria.

6.5. Alineamiento de secuencias

Cuando se analizan secuencias se suelen utilizar de manera indiscriminada los términos de similitud y homología. Sin embargo, estos términos se refieren a conceptos muy distintos.

Similitud es la característica resultante de la observación de que dos o más secuencias muestran algún grado de coincidencia en la secuencia de aminoácidos. La similitud, dado que es una observación, no puede ser indicador a priori de ninguna relación biológica entre las secuencias, ya que ésta se podría deber a cambios que se hayan dado al azar.

En cambio, se habla de **homología** cuando la similitud se puede atribuir a verdaderas razones evolutivas y no simplemente al azar. En este caso, se afirma que hay regiones de la secuencia conservadas en el tiempo.

La similitud es producto de una medida, mientras que la homología es una hipótesis que se postula en base a la similitud de las secuencias estudiadas y otras características adicionales. Se puede hablar de un porcentaje de similitud entre dos secuencias pero no de un porcentaje de homología. Ya que la homología es una característica cualitativa, no es susceptible de ser medida, por lo que dos secuencias simplemente o son o no son homólogas.

El alineamiento es el procedimiento que permite dar los primeros pasos hacia la conclusión de que dos o más secuencias son homólogas. Consiste en establecer un segmento entre ellas, donde el número de coincidencias sea máximo. Una coincidencia se presenta cuando el aminoácido de la secuencia A es igual al de la secuencia B o bien si sus características físico-químicas (hidrofobicidad, tamaño, carga, etc.) son similares. Los programas de alineamiento de secuencias utilizan matrices de sustitución, en las que a cada combinación posible de aminoácidos se le asigna un valor. Estas matrices de sustitución –que se verán en detalle en la siguiente sección– varían desde modelos simples que asignan el valor 1 si los aminoácidos son iguales y 0 si son distintos, hasta modelos más complejos evolutivos,

estructurales y/o funcionales, que fijan un determinado coste por sustituir un aminoácido por otro dependiendo de ambos aminoácidos.

Esta técnica, aparentemente sencilla, se hace más compleja en la medida en que el tamaño de las secuencias a comparar se hace mayor y, más aún, cuando se comparan más de dos secuencias. Para realizar esta tarea, se emplean distintos programas que dadas dos secuencias, generan el mejor alineamiento.

El alineamiento se puede clasificar de las siguientes maneras:

- Nivel de análisis.
  - Alineamiento global.
  - Alineamiento local.
- Número de secuencias analizadas.
  - Alineamiento de un par de secuencias.
  - Alineamiento múltiple.
- Con interrupciones (*gaps*) y sin interrupciones.

El *alineamiento global* se extiende a lo largo de la longitud total de las secuencias, de forma que el número de coincidencias obtenido sea máximo. En la figura 6.10 se puede ver un ejemplo de este tipo de alineamiento. Aunque en este ejemplo hay una región de identidad obvia (la secuencia FGKG), un alineamiento global no alinea estas regiones para tratar de favorecer que se produzca un mayor número de coincidencias a lo largo de las longitudes de las secuencias.

```
LGPSTKQFGKGSSSRIWDN
|       | |    || |
LQSIKFGKGNGIMRLIWPN
```

Figura 6.10: Alineamiento global

El *alineamiento local* encuentra el mejor segmento alineado existente entre dos secuencias. En este caso, el alineamiento tiende a detenerse al final de las regiones de identidad o de fuerte similitud. Se da mayor prioridad a encontrar estas regiones locales que a extender el alineamiento para incluir más pares de aminoácidos. Este tipo de alineamiento favorece encontrar motivos (*motifs*) de aminoácidos conservados en secuencias de proteínas relacionadas. En la figura 6.11 se puede observar un ejemplo de alineamiento local, en donde los guiones indican la parte de la secuencia no incluida en el alineamiento.

```
-------FGKG-------
-------FGKG-------
```

Figura 6.11: Alineamiento local

El alineamiento local, además de ser más efectivo desde el punto de vista computacional, tiene mayor sentido biológico, pues es sabido que sólo una región relativamente

pequeña de la secuencia de aminoácidos de una proteína es la que constituye su sitio activo, y por tanto, la que le confiere su función.

El *alineamiento de un par de secuencias* recibe dos secuencias y encuentra el segmento mejor alineado entre ellas. En cambio, el *alineamiento múltiple* trabaja sobre muchas secuencias y el resultado que obtiene es una secuencia consenso. Esta secuencia consenso tiene en cada posición el aminoácido que más se ha conservado en esa posición en todas las secuencias estudiadas.

En la figura 6.12 se muestra un fragmento de un alineamiento múltiple, en donde los asteriscos señalan las regiones de secuencia consenso de todas las secuencias.

```
MSDN----------KKQQALELALKQIEKQFGKGSIMKLGDG-ADHSIEAIPSGSIALDI
M------AINTDTSGKQKALTMVLNQIERSFGKGAIMRLGDA-TRMRVETISTGALTLDL
M------------DRQKALEAAVSQIERAFGKGSIMKLGGKDQVVETEVVSTRILGLDV
M--------DE---NKKRALAAALGQIEKQFGKGAVMRMGDHE-RQAIPAISTGSLGLDI
MD-----------------------KIEKSFGKGSIMKMGEE-VVEQVEVIPTGSIALNA
MDDKTSKAAAA---EKAKALAAALSQIEKQFGKGSIMRYGDNEVEHDIQVVSTGSLGLDI
M------AIDE---NKAKALAAALGQIEKQFGKGSIMRLGEDR-SMNVETISTGSLSLDV
MDD----------NKSKALAAALSQIEKQFGKGSIMRMGDADIGEDLQVVSTGSLGLDI
MDE----------NRSKALAAALSQIEKQFGKGSIMRMGDTDVAADIQAVSTGSLGLDI
MSD----------DKSKALAAALAQIEKSFGKGAIMKMDGSQQEENLEVISTGSLGLDL
M------AIDE---NKQKALAAALGQIEKQFGKGSIMRLGEDR-SMNVETISTGSLSLDV
M--------DE---NKKRALAAALGQIERQFGKCAVMRMGDHE-RQAIPAISTGSLGLDI
M--------TA---EKSKALAAALAQIEKQFGKGSIMRMGDGEAAEDIQVVSTGSLGLDI
MSQNSLRLVEDKSVDKSKALEAALSQIERSFGKGSIMKLGSNENVIEIETISTGSLGLDI
M------AIDE---NKQKALAAALGQIEKQFGKGSIMRLGEDR-SMDVETISTGSLSLDI
M------SAISNNPDKEKALNLVLNQIERNFGKGAIMRLGDA-AQMKVATIPSGALTLDQ
MDE----------QRSKGLSAALSQIDKQFGKGAVMRLGDHNAIKDIEVYSTGSLGLDL
M--------DE---NKQKALAAALGQIEKQFGKGSIMRLGDNR-TMDVETISTGSLSLDI
*                          *   ****  *                    *
```

Figura 6.12: Alineamiento múltiple

La complejidad en la realización de un alineamiento múltiple es función geométrica del número de secuencias sumadas al proceso. Por lo tanto, se requieren programas especializados más exigentes, en cuanto a capacidad de cómputo, que aquellos que sólo realizan alineamientos por pares.

```
LGPSTKQFGKGSSRSIWDN
|       | |    || |
LQSIKFGKGNGIMRLIWPN
```

Figura 6.13: Alineamiento sin interrupciones

```
LGPSTKQFGKG--SSRSIWDN
|   | | ||||    | || |
LQ-SIK-FGKGNGIMRLIWPN
```

Figura 6.14: Alineamiento con interrupciones

La introducción de interrupciones en las secuencias permite incrementar el número de

coincidencias del alineamiento. Este intento de incrementar el número de coincidencias, puede llegar a deformar las secuencias hasta tal punto que pierdan su señal biológica. Para evitar la deformación se penaliza la introducción de interrupciones. La forma más empleada penaliza por iniciar interrupciones (iG) y por extender interrupciones (eG), de la siguiente forma:

$$\alpha_g = iG + eG(g-1)$$

donde g es la longitud de la interrupción. A mayor penalización, menor será el número de interrupciones.

Para realizar alineamientos de secuencias existen tres métodos:

- DotPlots (matrices de puntos).
- Métodos de programación dinámica.
  - Needleman-Wunsch [NW70].
  - Smith-Waterman [SW81].
- Métodos heurísticos.
  - FASTA [LP85, LP88].
  - BLAST [AGM+90].

En secciones posteriores se explican en detalle estos métodos así como las matrices de sustitución de aminoácidos, que son empleadas para obtener alineamientos óptimos.

6.5.1. Matrices de sustitución

Cuando se busca el mejor alineamiento entre dos secuencias de aminoácidos, la matriz de sustitución empleada puede afectar en gran medida a los resultados obtenidos. Idealmente, los pesos de estas matrices deben reflejar los fenómenos biológicos de evolución subyacentes que el alineamiento pretende mostrar.

Dos ejemplos de matrices de sustitución sencillas, que no hacen uso de fenómenos biológicos, son las siguientes:

- **Matriz identidad.**

$$M_{ij} = \begin{cases} 1 & si\ i = j \\ 0 & en\ otro\ caso \end{cases}$$

- **Matriz con base en el código genético.** M_{ij} es igual al número mínimo de sustituciones necesarias que hay que realizar en las bases del codón del aminoácido i para convertirlo en el codón del aminoácido j.

  Es importante notar que el único caso que requiere una sustitución en cada una de las posiciones del codón es de metionina a tirosina y viceversa.

Sin embargo, en el problema de la predicción de la estructura secundaria de las proteínas, las matrices más utilizadas son las denominadas PAM y BLOSUM, que a continuación se definen.

	A	S	G	L	K	V	T	P	E	D	N	I	Q	R	F	Y	C	H	M	W	Z	B	X
A(ala)	0	1	1	2	2	1	1	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2
S(ser)	1	0	1	1	2	2	1	1	2	2	1	1	2	1	1	1	1	2	2	1	2	2	2
G(gly)	1	1	0	2	2	1	2	2	1	1	2	2	2	1	2	2	1	2	2	1	2	2	2
L(leu)	2	1	2	0	2	1	2	1	2	2	2	1	1	1	1	2	2	1	1	1	2	2	2
K(lys)	2	2	2	2	0	2	1	2	1	2	1	1	1	1	2	2	2	2	1	2	1	2	2
V(val)	1	2	1	1	2	0	2	2	1	1	2	1	2	2	1	2	2	2	1	2	2	2	2
T(thr)	1	1	2	2	1	2	0	1	2	2	1	1	2	1	2	2	2	2	1	2	2	2	2
P(pro)	1	1	2	1	2	2	1	0	2	2	2	2	1	1	2	2	2	1	2	2	2	2	2
E(glu)	1	2	1	2	1	1	2	2	0	1	2	2	1	2	2	2	2	2	2	2	1	2	2
D(asp)	1	2	1	2	2	1	2	2	1	0	1	2	2	2	2	1	2	1	2	2	2	1	2
N(asn)	2	1	2	2	1	2	1	2	2	1	0	1	2	2	2	1	2	1	2	2	2	1	2
I(ile)	2	1	2	1	1	1	1	2	2	2	1	0	2	1	1	2	2	2	1	2	2	2	2
Q(gln)	2	2	2	1	1	2	2	1	1	2	2	2	0	1	2	2	2	1	2	2	1	2	1
R(arg)	2	1	1	1	1	2	1	1	2	2	2	1	1	0	2	2	1	1	1	1	2	2	2
F(phe)	2	1	2	1	2	1	2	2	2	2	2	1	2	2	0	1	1	2	2	2	2	2	2
Y(try)	2	1	2	2	2	2	2	2	2	1	1	2	2	2	1	0	1	1	3	2	2	1	2
C(cys)	2	1	1	2	2	2	2	2	2	2	2	2	2	1	1	1	0	2	2	1	2	2	2
H(his)	2	2	2	1	2	2	2	1	2	1	1	2	1	1	2	1	2	0	2	2	2	1	2
M(met)	2	2	2	1	1	1	1	2	2	2	2	1	2	1	2	3	2	2	0	2	2	2	2
W(trp)	2	1	1	1	2	2	2	2	2	2	2	2	2	1	2	2	1	2	2	0	2	2	2
Z(glx)	2	2	2	2	1	2	2	2	1	2	2	2	1	2	2	2	2	2	2	2	1	2	2
B(asx)	2	2	2	2	2	2	2	2	2	1	1	2	2	2	2	1	2	1	2	2	2	1	2
X(???)	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

Figura 6.15: Matriz con base en el código genético

Matrices PAM

Las primeras matrices PAM específicas fueron desarrolladas por Margaret Dayhoff et al. en 1978 [SDO78, DS79]. Examinaron 1.572 cambios en 71 familias de proteínas estrechamente relacionadas. Durante este proceso se dieron cuenta de que las sustituciones que se habían producido en grupos de proteínas estrechamente relacionadas, no eran al azar. Llegaron a la conclusión de que la sustitución de algunos aminoácidos, se produce con más frecuencia que la de otros, probablemente porque no tienen un gran efecto en la estructura y función de la proteína. Esto significa que proteínas relacionadas evolutivamente no necesitan tener los mismos aminoácidos en cada posición y que podrían tener aminoácidos con características similares. De estas observaciones nació la matriz PAM.

Las unidades PAM se emplean para medir la distancia evolutiva entre dos secuencias de aminoácidos. Dos secuencias S_1 y S_2 están a una distancia evolutiva de una unidad PAM, si una serie de mutaciones puntuales aceptadas han convertido S_1 en S_2 con una media de una mutación puntual aceptada por cada 100 aminoácidos.

Las matrices PAM codifican los cambios evolutivos esperados a nivel de aminoácido. Cada matriz PAM se diseña para comparar secuencias que estén a un determinado número de unidades PAM de distancia evolutiva. Por ejemplo, la matriz PAM 120 está diseñada para comparaciones entre secuencias que están a 120 unidades PAM de distancia evolutiva. La entrada (i, j) de la matriz PAM N refleja la frecuencia a la que se espera que el aminoácido i sustituya al aminoácido j en dos secuencias, que están a una distancia evolutiva de N unidades PAM. Estas frecuencias deben ser estimadas recogiendo estadísticas de los aminoácidos sustituidos.

Matrices BLOSUM - *BLOcks SUbstitution Matrices*

Las matrices BLOSUM fueron calculadas por Steven Henikoff y Jorja G. Henikoff en el año 1992 [HH92]. La diferencia entre las matrices PAM y las BLOSUM es que las primeras se derivan de alineamientos globales de proteínas, mientras que las segundas vienen de alineamientos sin interrupciones de secuencias más cortas. Esto es, de bloques

de secuencias que son idénticas en un tanto por ciento definido de sus aminoácidos. Este tanto por ciento viene dado por el número asociado a la matriz BLOSUM. Por tanto, la matriz BLOSUM N se calcula usando bloques de secuencias que son idénticas en un N % de sus aminoácidos. De este modo, el método BLOSUM incorpora muchos más datos en sus matrices y, por tanto, es supuestamente más exacto.

Las matrices BLOSUM62 y BLOSUM50 son ampliamente utilizadas para alineamientos de pares de secuencias y para búsquedas de similitud en bases de datos. BLOSUM62 es estándar para alineamientos sin interrupciones, mientras que la BLOSUM50 normalmente es mejor para alineamientos con interrupciones.

En el caso de las matrices PAM, las matrices para distancias evolutivas grandes son extrapoladas a partir de las de distancias evolutivas pequeñas. En cambio, cada matriz BLOSUM es calculada directamente, es decir, sin emplear extrapolaciones.

BLOSUM actúa, generalmente, mejor que las matrices PAM para búsquedas de similitud local.

6.5.2. Métodos de alineamiento de secuencias

Métodos de programación dinámica

La idea de la programación dinámica es plantear la solución del problema en términos de un problema más sencillo, éste en términos de otro más sencillo y así sucesivamente. De este modo, en el caso en que el problema a resolver es la construcción de un alineamiento óptimo entre dos secuencias, éste se puede construir empleando para ello alineamientos óptimos de subsecuencias más pequeñas.

El primer método de aplicación de la programación dinámica al alineamiento de secuencias fue desarrollado por Needleman y Wunsch en 1970 [NW70]. En 1981 Smith y Waterman desarrollaron un nuevo método [SW81]. La diferencia entre ellos reside en que el primero proporciona un alineamiento global, mientras que el alineamiento proporcionado por el segundo es local.

A pesar de que los métodos de programación dinámica son la forma más exhaustiva para la búsqueda de similitud entre dos secuencias, la complejidad de estos métodos es $O(N^2)$, siendo N la longitud de las secuencias. En el caso de búsqueda de similitudes en bases de datos, el tiempo de computación requerido para este tipo de algoritmos es muy elevado.

Métodos heurísticos

Los métodos heurísticos buscan velocidad a cambio de sensibilidad y selectividad. Para reducir el espacio de búsqueda y, por tanto, el tiempo de computación, estos métodos intentan localizar las posiciones donde es más probable que se encuentre el mejor alineamiento.

FASTA fue el primer algoritmo ampliamente utilizado para búsqueda de similitud en bases de datos. Fue desarrollado por David Lipman y William Pearson en 1985 [LP85] y mejorado en 1988 [LP88]. Es empleado principalmente por el EMBL – EBI (*European Molecular Biology Laboratories – European Biology Institute*). Compara una secuencia de ADN o una proteína, contra cada una de las secuencias de una base de datos, y devuelve los

mejores segmentos alineados. Es una herramienta de alineamiento local por pares. Basa su estrategia en identificar diagonales con el mayor número de identidades (residuos iguales) sin incluir interrupciones, aunque posteriormente reevalúa estas regiones usando matrices de sustitución para permitir que los reemplazos conservativos también sean tenidos en cuenta.

Por otra parte, el algoritmo BLAST (*Basic Local Alignment Search Tool*) fue desarrollado por Altschul, Gish, Miller, Myers y Lipman en 1990 [AGM+90] y es el algoritmo más empleado actualmente. La razón por la que se desarrolló BLAST fue la necesidad de incrementar la velocidad de FASTA.

La principal característica de BLAST es su velocidad, empleando pocos minutos para cualquier búsqueda en una base de datos. En cambio, FASTA es notablemente más lento, empleando para búsquedas equivalentes hasta varias horas.

BLAST es una herramienta ampliamente utilizada para búsqueda de similitudes entre una secuencia incógnita y las secuencias de una base de datos de proteínas o de ADN. Altschul et al. propusieron en 1997 [AMS+97] una serie de refinamientos para la comparación de proteínas, que permiten que el tiempo de ejecución de los programas BLAST decrezca sustancialmente mientras que aumenta su sensibilidad en la búsqueda de relaciones débiles. Estos refinamientos son:

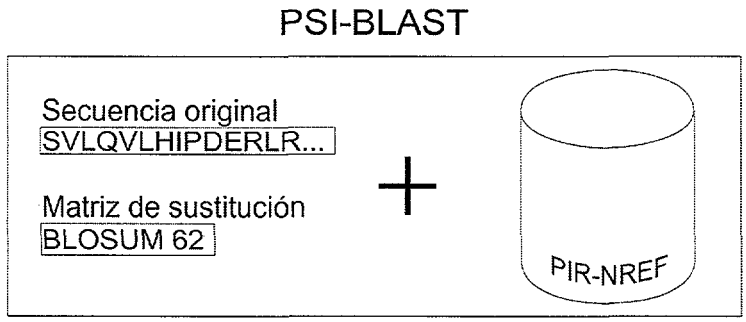
- **Gapped BLAST**. El uso de un nuevo criterio en la búsqueda inicial, combinado con una nueva heurística para generar alineamientos con interrupciones, da lugar al programa *gapped* BLAST, que es aproximadamente tres veces más rápido que el programa BLAST original.

- **PSI-BLAST**. Se introdujo un nuevo método que combina automáticamente los alineamientos estadísticamente significativos producidos por BLAST en una *position-specific score matrix*, y realiza la búsqueda en la base de datos con esta matriz. El programa resultante PSI-BLAST (*Position-Specific Iterated BLAST*) ejecuta por iteración a la misma velocidad que *gapped* BLAST, pero en muchos casos es mucho más sensible a similitudes débiles pero biológicamente relevantes entre secuencias.

  El funcionamiento de la herramienta PSI-BLAST se puede observar en la figura 6.16.

  Las búsquedas de similitudes en bases de datos empleando *position-specific score matrix* (PSSM), también llamados perfiles, a menudo son mucho más capaces de detectar relaciones débiles que las búsquedas que emplean como *query* una simple secuencia [McL83, Sta84, Tay86, BvH87, DE87, GME87, Pat87, SH89, TAK94, YL93]. El empleo de estos métodos, sin embargo, implica frecuentemente el uso de diversos programas diferentes y un grado de experiencia considerable. En consecuencia, para dejar el poder de la búsqueda con perfiles fácilmente disponible, Altschul et al. [AMS+97] crearon un procedimiento para construir PSSMs automáticamente a partir de la salida de la ejecución de BLAST, y modificaron BLAST para que opere con tales matrices en lugar de con una simple secuencia. El programa PSI-BLAST que obtuvieron como resultado, es a menudo considerablemente más sensible que BLAST. Los objetivos para el desarrollo de PSI-BLAST fueron: operación automática, velocidad y sencillez.

  Los pasos del algoritmo PSI-BLAST son:



	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
S	76	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	7	0	0	0
V	0	0	0	0	0	0	0	0	0	45	17	1	1	0	2	0	0	0	0	34
L	1	35	0	0	0	0	0	0	1	0	38	16	2	1	0	0	0	0	4	1
Q	0	3	5	18	0	7	19	0	0	0	1	12	0	0	31	1	1	0	1	0
V	0	0	0	0	0	0	0	0	0	75	10	0	2	2	0	0	0	0	0	11
L	2	8	1	0	1	0	2	1	2	16	26	2	0	1	0	0	5	0	1	32
H	2	17	0	0	1	14	6	0	3	8	4	13	2	5	3	0	11	0	6	4
I	13	0	1	6	0	0	7	0	3	6	15	0	7	7	0	2	0	1	24	6
P	0	0	0	3	0	0	1	50	0	0	0	0	0	0	40	3	0	0	0	0
D	1	0	17	54	0	0	7	0	10	0	0	2	1	0	0	4	0	0	1	0
E	3	3	0	7	0	2	12	0	2	1	0	8	0	0	59	2	0	0	0	0
R	2	23	0	0	1	1	0	1	1	17	3	2	0	2	0	1	7	0	0	37
L	0	0	0	0	0	0	0	0	0	0	96	0	1	2	0	0	0	0	0	0
R	2	52	0	0	0	0	1	0	8	0	8	14	0	2	0	7	5	0	0	1
...																				

Figura 6.16: Obtención del perfil de una secuencia con la herramienta PSI-BLAST

1. Toma como entrada una simple proteína y la compara con las proteínas de la base de datos, usando el programa *gapped* BLAST [AMS+97].
2. Construye un alineamiento múltiple y, a partir de éste, un perfil empleando alineamientos locales significativos encontrados. La secuencia que toma como entrada sirve como plantilla para el alineamiento múltiple y para el perfil.
3. El perfil se compara con las secuencias de la base de datos, buscando de nuevo alineamientos locales.
4. PSI-BLAST estima la significancia estadística de los alineamientos locales encontrados. Puesto que las puntuaciones del perfil se construyen a una escala fija [KA90], y las penalizaciones por la inserción de interrupciones siguen siendo independientes de la posición, la teoría estadística y los parámetros para obtener alineamientos con *gapped* BLAST [AG96] siguen siendo aplicables a los alineamientos del perfil [AMS+97].
5. Finalmente, PSI-BLAST itera, volviendo al paso 2, un número arbitrario de veces o hasta que converge. La convergencia se alcanza cuando no se detectan nuevos alineamientos significativos.

Capítulo 7

Estudio del problema y propuestas

En este capítulo realizamos tres propuestas dentro del problema de la predicción de la estructura secundaria de las proteínas.

El índice del capítulo es el siguiente:

- En la sección 7.1 se realiza un estudio del problema y se plantean las diferentes propuestas.
- Para la realización de todas estas propuestas es necesario partir de unos conjuntos de datos de proteínas ya conocidos, de tal forma que los resultados obtenidos puedan ser comparados de forma más ecuánime. En la sección 7.2 se enumeran los conjuntos de datos que serán utilizados.
- La sección 7.3 explica las estadísticas que habitualmente se obtienen en este dominio para la validación de los resultados logrados por los diferentes métodos de predicción de la estructura secundaria.
- En la sección 7.4, y dentro de las nuestras propuestas, se presenta una modelización matemática de las proteínas y del problema de la predicción de la estructura secundaria de las proteínas.
- En la sección 7.5, se desarrolla un clasificador para la predicción de la estructura secundaria basado en redes Bayesianas.
- Por último, en la sección 7.6, se desarrolla la propuesta de un multiclasificador basado en redes Bayesianas para el problema de la predicción de la estructura secundaria de las proteínas.

7.1. Estudio del problema y propuestas

A continuación se detallan las tres propuestas que son realizadas dentro del problema de la predicción de la estructura secundaria de las proteínas y el porqué de las mismas.

7.1.1. Modelización matemática de las proteínas y del problema PSSP

Hemos realizado una modelización matemática de las proteínas y del problema de la predicción de la estructura secundaria.

Esta modelización matemática resulta de gran utilidad, ya que permite acercar el campo de la proteómica y de la predicción de la estructura secundaria a los científicos de la comunidad del aprendizaje automático, consiguiendo superar la barrera que supone la biología para los investigadores en inteligencia artificial.

7.1.2. Desarrollo de un clasificador para la PSSP con redes Bayesianas

El problema de la predicción de la estructura secundaria de las proteínas (PSSP) ya ha sido abordado desde diversos enfoques metodológicos siendo, como ya se vio, muchos de ellos pertenecientes al mundo del aprendizaje automático. Sin embargo, nunca se ha intentado plantear este problema con el uso de redes Bayesianas, por lo que nos hemos propuesto su implementación [RLPP02, RPH+03].

El uso de las redes Bayesianas puede suponer dos ventajas principales:

- Mejoras en el proceso de aprendizaje, ya que uno de los principales problemas que presentan las redes neuronales es el tiempo que se debe emplear en el mismo, que puede llegar a ser de semanas.
- Mejorar la transparencia del aprendizaje, ya que los clasificadores basados en redes Bayesianas generan estructuras comprensibles, en contraposición con los modelos obtenidos con las redes neuronales.

La principal dificultad en este punto radica en conseguir introducir la información evolutiva de las proteínas en las redes Bayesianas, información que actualmente están utilizando todos los algoritmos de predicción de tercera generación. Esto ha propiciado que sólo naïve-Bayes e IENB hayan podido ser utilizados.

7.1.3. Multiclasificador basado en redes Bayesianas para el problema PSSP

Combinar un conjunto de clasificadores ha demostrado ser una forma eficiente de mejorar la exactitud de los clasificadores que se combinan.

Lo que proponemos en este apartado es la realización de un multiclasificador –un clasificador de clasificadores– basado en redes Bayesianas, para la predicción de la estructura secundaria de las proteínas [RLP+03c] basado en el paradigma *stacked generalization* [Wol92].

Con este objetivo hemos desarrollado una aplicación Web que contacta con todos los servidores Web de predicción de la estructura secundaria existentes en Internet. A través de esta aplicación se persiguen tres objetivos:

- Comparar los algoritmos de predicción actuales, extrayendo estadísticas sobre la exactitud de sus predicciones. Sin embargo, esto no es novedoso, ya que el proyecto

EVA (*EValuation of Automatic protein structure prediction*) [RE01] fue creado con el mismo objetivo y está actualmente en funcionamiento.

- Tener un punto central en Internet a través de cual, y con una sola petición, se puedan conseguir los resultados de todos los servidores de predicción de estructura secundaria de Internet.
- Obtener el conjunto de datos necesario para la realización del multiclasificador de predicción de estructura secundaria.

7.2. Conjuntos de datos para PSSP

Para comparar dos métodos de predicción, se debe usar el mismo conjunto de datos de aprendizaje y validación, ya que si no, la comparación no es significativa. De esta forma, el problema de la predicción de la estructura secundaria de las proteínas se ha estandarizado sobre varios conjuntos de datos.

La elección de un conjunto de datos es un problema difícil, que requiere tanto de conocimientos de aprendizaje automático como de conocimientos específicos del dominio. La idea es escoger un conjunto representativo de instancias con soluciones conocidas que se puedan utilizar en el proceso de aprendizaje. Sin embargo, existen tres posibles problemas:

- La elección de un conjunto de datos que no refleje la distribución de probabilidad subyacente del mundo real.
- La selección de instancias que contienen información contradictoria.
- La elección de un conjunto de datos que contiene correlaciones artificiales.

Los conjuntos de datos que se utilizarán en las pruebas de este capítulo son: RS126 (Rost y Sander - 1993) [RS93], CB513 (Cuff y Barton - 1999) [CB99], HS1771 (Hobohm et al. - 1992) [HSSS92] y los conjuntos de datos obtenidos de la página del proyecto EVA [RE01].

Uno de los conjuntos de datos utilizados en las pruebas es el propuesto por Rost y Sander (1993) [RS93], quienes seleccionaron 126 proteínas con las que realizar el aprendizaje y la validación de los algoritmos de predicción de estructura secundaria. Definieron la "no-redundancia" para decir que dos proteínas en el conjunto de aprendizaje no comparten una identidad de secuencia de más del 25 %. Desafortunadamente, el conjunto de datos RS126 contiene pares de proteínas que son claramente secuencias similares cuando son comparados por métodos más sofisticados que el porcentaje de identidad. Hace mucho que se sabe que el porcentaje de identidad es una medida pobre de similitud de secuencias.

En 3Dee [SB98] (*Database of Protein Domain Definitions*) se creó un conjunto de secuencias [CB99] no redundantes, usando un algoritmo de comparación de secuencias sensible y por análisis de grupo. Esto proporcionó un conjunto de 1.233 secuencias donde ningún par compartió similitud de secuencia obvia. Las secuencias fueron entonces filtradas para permitir sólo estructuras obtenidas a través de la técnica de Rayos-X con resoluciones menor o igual de 2,5 Angstrom. Esto dejó un conjunto representativo de 554 secuencias de dominio, designado como CB554. Juntando el conjunto de datos CB554 y RS126, y

	Predichos			
Observados	H	E	L	
H				obs_H
E				obs_E
L				obs_L
	prd_H	prd_E	prd_L	N_{res}

Tabla 7.1: Matriz de confusión para el problema PSSP

quitando las secuencias con cierto grado de similitud, se consiguió el conjunto de datos CB513.

Otro conjunto de datos utilizado para nuestras pruebas es el proporcionado por U. Hobohm, M. Scharf, R. Schneider y C. Sander (1992), denominado HS1771. Son los subconjuntos de la base de datos PDB (*Protein Data Bank*) [BWF+00] no redundantes del EMBL (*European Molecular Biology Laboratory*) a los que se puede acceder vía ftp anónimo de EMBL [1]. En el conjunto de datos se cumple que dos proteínas no tienen una identidad de secuencia mayor del 25 % (para alineaciones de 80 o más residuos de longitud). Cada cierto tiempo, este conjunto de datos es actualizado con las últimas proteínas introducidas en la base de datos PDB. El conjunto de datos utilizado en este capítulo pertenece al mes de Marzo de 2002.

Además de los conjuntos de datos RS126, CB513 y HS1771, se han utilizado seis conjuntos de datos más, obtenidos del proyecto EVA [RE01]. Actualmente, EVA utiliza sólo proteínas con estructuras nuevas para evaluar la predicción de la estructura secundaria. Del proyecto EVA se han seleccionado seis conjuntos de datos: EVA1 con 114 secuencias, EVA2 con 116, EVA3 con 107, EVA4 con 121, EVA5 con 200 y EVA6 con 507 secuencias.

7.3. Estadísticas para el problema PSSP

Para la validación de los resultados obtenidos por los métodos de predicción de estructura secundaria, se han definido una serie de estadísticas particulares.

Para poder realizar las estadísticas se ha utilizado una matriz de confusión de tamaño 3x3, en donde las filas reflejan los estados de la estructura secundaria real (obtenida con el programa DSSP [KS83]) y las columnas los estados de la estructura secundaria predicha por el clasificador. En la tabla 7.1 se pueden ver los elementos de la matriz de confusión.

De la matriz de confusión se pueden obtener los siguientes valores:

- obs_H: representa el número de estados hélice (H) observados, es decir, estados H que aparecen en la estructura real,
- obs_E: representa el número de estados β laminar (E) observados,
- obs_L: representa el número de estados *coil* (L) observados,
- prd_H: representa el número de estados de la secuencia observada que se han predicho como hélice (H),

[1] ftp://ftp.embl-heidelberg.de/pub/databases/protein_extras/pdb_select

- prd_E: representa el número de estados de la secuencia observada que se han predicho como β laminar (E),
- prd_L: representa el número de estados de la secuencia observada que se han predicho como *coil*,
- N_{res}: representa el número total de estados de la cadena observada (longitud de la secuencia).

Si se representa matemáticamente, se tiene que:

M_{ij} denota el número de residuos observados en el estado i y predichos con el estado j, con $i, j \in \{H, E, L\}$

El número total de residuos observados en el estado i es:

$$obs_i = \sum_{j=1}^{3} M_{ij} \mid i, j \in \{H, E, L\} \tag{7.1}$$

El número total de residuos predichos con el estado i es:

$$prd_j = \sum_{i=1}^{3} M_{ij} \mid i, j \in \{H, E, L\} \tag{7.2}$$

y el número total de estados en la secuencia es:

$$N_{res} = \sum_i obs_i = \sum_j prd_j = \sum_{i,j} M_{ij} \tag{7.3}$$

7.3.1. Exactitud de la predicción para los tres estados: Q_3

Es la medida que se utiliza tradicionalmente para evaluar la exactitud de la predicción de la estructura secundaria. Obtiene el número total de residuos predichos correctamente. Para ello, suma los estados hélice, β laminar y *coil* que se han predicho correctamente (suma de los M_{ii}), dividiendo el resultado por el número total de residuos de la secuencia observada (N_{res}) y mostrando el resultado en forma de porcentaje. El Q_3 se obtiene como:

$$Q_3 = 100 \frac{1}{N_{res}} \sum_{i=1}^{3} M_{ii} \tag{7.4}$$

7.3.2. Porcentajes de exactitud por estados

Para definir la exactitud de un estado particular hay dos posibles variantes:

- Porcentaje de estados que se han predicho correctamente (%obs).

$$Q_i^{\%obs} = 100 \frac{M_{ii}}{obs_i} \tag{7.5}$$

De esta forma, por ejemplo, se tendrá para el estado i=H (hélice) el porcentaje de H observadas que se han predicho correctamente.

- Porcentaje de estados en la secuencia predicha que se han predicho correctamente (%prd).

$$Q_i^{\%prd} = 100\frac{M_{ii}}{prd_i} \tag{7.6}$$

Por ejemplo, para el estado i=H, se obtendrá el porcentaje de estados H que hay en la secuencia predicha, y que se han predicho correctamente.

7.3.3. Índice de información

El índice de información viene dado por:

$$info = \ln\left(\frac{P_{prd}}{P_{obs}}\right) \tag{7.7}$$

donde P_{obs} describe la probabilidad de encontrar una cadena particular de N_{res} residuos con obs_i residuos en la estructura i de todas las combinaciones posibles, y P_{prd} es la probabilidad de una realización particular de la matriz de confusión M.

El índice de información se puede calcular como:

$$info = \frac{info^{\%obs} + info^{\%prd}}{2} \tag{7.8}$$

con:

$$info^{\%obs} = 1 - \frac{\sum_{i=1}^{3} prd_i \ln prd_i - \sum_{i,j=1}^{3} M_{ij} \ln M_{ij}}{N_{res} \ln N_{res} - \sum_{i=1}^{3} obs_i \ln obs_i} \tag{7.9}$$

$$info^{\%prd} = 1 - \frac{\sum_{i=1}^{3} obs_i \ln obs_i - \sum_{i,j=1}^{3} M_{ij} \ln M_{ij}}{N_{res} \ln N_{res} - \sum_{i=1}^{3} prd_i \ln prd_i} \tag{7.10}$$

7.3.4. Coeficiente de correlación de Matthew

El coeficiente de correlación de Matthew [Mat85] es una medida que no se ve influida por el porcentaje de verdaderos positivos (cantidad de la estructura i predicha correctamente dividida por la cantidad de la estructura i que hay en la secuencia observada) en una muestra y es la manera más exacta de evaluar diferentes métodos. El resultado es un número comprendido entre -1 y 1, representando el 1 una coincidencia perfecta, el -1 una desigualdad total y 0 para indicar que la predicción no tiene correlación con los resultados.

Aunque el coeficiente de correlación sea una medida útil de exactitud de la predicción, no evalúa cómo de parecida es la predicción a una proteína. Para saber cómo de realista es una predicción, se deben tener en cuenta las longitudes de los segmentos estructurales secundarios predichos (SOV) que se verán más adelante. El coeficiente de correlación de Matthew se obtiene de la siguiente formula:

$$C_i = \frac{p_i n_i - u_i o_i}{\sqrt{(p_i + u_i)(p_i + o_i)(n_i + u_i)(n_i + o_i)}} \tag{7.11}$$

con:

$$p_i = M_{ii}, n_i = \sum_{j \neq i}^{3} \sum_{k \neq i}^{3} M_{jk}, o_i = \sum_{j \neq i}^{3} M_{ji}, u_i = \sum_{j \neq i}^{3} M_{ij} \tag{7.12}$$

donde:

- n_i: contiene el número de estados distintos de i que se han observado y que se han predicho como estado j, siendo j distinto de i. Por ejemplo, para el estado $i = H$, se tiene la cantidad de estados L y E observados, que se han predicho como L o E.
- u_i: contiene el número de residuos en estado i que se encuentran en la secuencia observada y que se han predicho como un estado distinto de i. Por ejemplo, para el estado $i = H$, contiene el número de estados H observados que se han predicho como E o L.
- p_i: representa el número de residuos observados en el estado i predichos correctamente.
- o_i: representa el número de residuos observados con estado distinto de i, y que se han predicho como estado i.

7.3.5. SOV: medida de superposición de segmento

Toda la estadística anterior es la estadística general que se puede aplicar a cualquier problema de clasificación. Sin embargo, la superposición de segmento –*Segment OVerlap*– (*SOV*) es una medida, desarrollada por Rost [RSS94b] y modificada por Zemla [ZVFR99], que refleja los objetivos específicos de la predicción de la estructura secundaria.

Al contrario que la medida Q_3, que considera los residuos de forma individual, *SOV* mide la exactitud considerando los distintos segmentos de una secuencia. *SOV* proporciona la medida de la cantidad de superposición de segmentos para un solo estado (H, E ó L) o para los tres estados.

Tomando como ejemplo el estado *coil* (L), la medida *SOV* calcula la exactitud de la predicción de los segmentos de la secuencia observada en dicho estado. Por segmento se entiende las partes de la secuencia donde aparece de forma consecutiva el estado i (en este caso L). Así, se obtendrá un 100 % cuando los segmentos de la secuencia observada sean iguales en la secuencia predicha. Cuando se calcule el *SOV* para los tres estados, se tendrán en cuenta los segmentos de los tres estados (hélice, β lámina y *coil*).

SOV para un estado conformacional

Viene dado por:

$$SOV(i) = 100\frac{1}{N(i)}\sum_{S(i)}\frac{minov(s_1,s_2)+\delta(s_1,s_2)}{maxov(s_1,s_2)}len(s_1) \tag{7.13}$$

donde:

- s_1 y s_2: son los segmentos observados y predichos de la estructura secundaria en el estado i –el cual puede ser H, E ó L–,
- $len(s_1)$: es el número de residuos en el segmento s_1,
- $minov(s_1,s_2)$: es la longitud del solapamiento actual de s_1 y s_2, es decir, la extensión para la que ambos segmentos tienen residuos en el estado i,
- $maxov(s_1,s_2)$: es la longitud de la extensión total para la cual los segmentos s_1 o s_2 tienen un residuo en el estado i,
- $\delta(s_1,s_2)$: es el valor entero definido como:

$$\delta(s_1,s_2) = min\left\{\begin{array}{c} maxov(s_1,s_2) - minov(s_1,s_2) \\ minov(s_1,s_2) \\ int(0{,}5 * len(s_1)) \\ int(0{,}5 * len(s_2)) \end{array}\right\} \tag{7.14}$$

- $\sum$: se toma sobre todos los pares de segmentos (s_1,s_2), donde s_1 y s_2 tienen al menos un residuo en estado i en común,
- $N(i)$: es el número de residuos en el estado i. Se define como:

$$N(i) = \sum_{S(i)} len(s_1) + \sum_{S'(i)} len(s_2) \tag{7.15}$$

Las dos sumas se realizan sobre S y S'. $S(i)$ es el número de todos los pares de segmentos (s_1,s_2), en los que s_1 y s_2 tienen en común al menos un residuo en estado i. $S'(i)$ es el número de segmentos S_1 que no producen ningún segmento asociado.

En la figura 7.1 se muestra un fragmento de secuencia observada secuencia predicha, en donde se pueden ver claramente los elementos de la fórmula de SOV.

SOV: para tres estados

La medida SOV para los tres estados se obtiene como:

$$SOV = 100\frac{1}{\sum_i N(i)}\sum_i\sum_{S(i)}\frac{minov(s_1,s_2)+\delta(s_1,s_2)}{maxov(s_1,s_2)}len(s_1) \tag{7.16}$$

donde $\sum_i N(i)$ es la suma de $N(i)$ sobre los tres estados conformacionales.

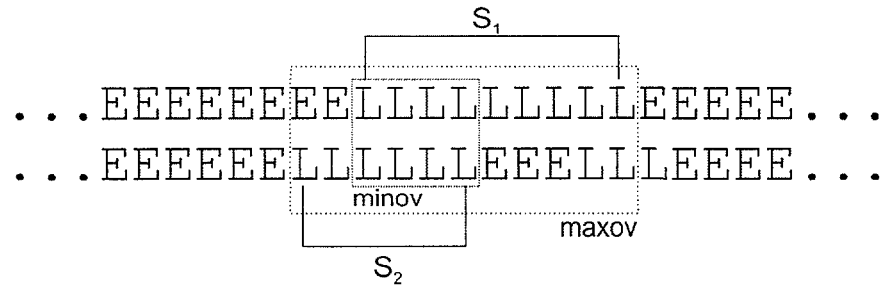


Figura 7.1: Fragmento de secuencia observada secuencia predicha con los elementos de la medida *SOV*

7.4. Modelización matemática de las proteínas y de PSSP

A continuación se presenta una modelización matemática de las proteínas y del problema de la predicción de la estructura secundaria. Consideramos que esta modelización matemática puede resultar de gran utilidad, ya que permite acercar este campo a los científicos de la comunidad del aprendizaje automático, consiguiendo que superen la barrera que normalmente impone la biología en estos casos.

7.4.1. Modelización matemática

Estructura primaria

Se define $\mathcal{A} = \{A, C, D, E, F, G, H, I, J, K, L, M, N, P, Q, R, S, T, V, W, Y\}$ como el conjunto de todos los símbolos que denotan los aminoácidos.

De esta forma, $\mathcal{A}^*$ es el conjunto de todas las secuencias finitas formadas por los elementos del conjunto $\mathcal{A}$. Los elementos de $\mathcal{A}^*$ se denotan por $x, y, z, \ldots$, por ejemplo, $x \in \mathcal{A}^*$.

Estructura primaria de la proteínas: cualquier $x \in \mathcal{A}^*$ se denomina sub-unidad de proteína (hay que recordar que una proteína está formada por una o más sub-unidades).

Estructura secundaria

Sea $S = \{H, E, L\}$ el conjunto de símbolos de los estados secundarios (H para hélices, E para β láminas y L para *coil*) y S^* el conjunto de todas las secuencias finitas de elementos de S.

Denotamos los elementos de S^* por e, con subíndices en caso necesario. Por ejemplo, $e_1 \in S^*$.

Cualquier función f de 1 a n

$$f : \mathcal{A}^* \rightarrow S^* \tag{7.17}$$

en donde,

$$f(x) = e \mid \|x\| = \|e\| \tag{7.18}$$

se denomina *función de identificación de estructura secundaria.* f es una función de 1 a n, ya que para la misma estructura primaria pueden existir diversas estructuras secundarias, dependiendo de la forma de cristalización de la proteína.

Los elementos $(x, e) \in f$ se denominan *estructura secundaria de sub-unidad*, donde el elemento e es la estructura secundaria de la sub-unidad s.

Cualquier conjunto de datos (CD) utilizado en la predicción de la estructura secundaria de la proteínas, define su propia función de identificación f_{CD} de forma empírica –a través del programa DSSP, que toma la estructura tridimensional y devuelve la estructura secundaria–. De esta forma se tiene:

$$CD = \left\{ \begin{pmatrix} x \\ e \end{pmatrix} | f_{CD}(x) = e \right\}, x \in \mathcal{A}^*, e \in S^* \tag{7.19}$$

Por ejemplo, si un conjunto de datos tiene la sub-unidad $ARNVSTVVLA$ con estructura secundaria observada $HHHEEELLLH$, se puede definir como:

$$f_{CD}(ARNVSTVVLA) = HHHEEELLLH \tag{7.20}$$

o

$$\begin{pmatrix} ARNVSTVVLA \\ HHHEEELLLH \end{pmatrix} \in CD \tag{7.21}$$

Estructura terciaria

Sea $x \in A^*$ la secuencia de una sub-unidad y $(x, e) \in f$ la estructura secundaria de x. El elemento:

$$\varphi_x = ((x, e), t_x), x \in \mathcal{A}^*, e \in S^* \tag{7.22}$$

denota la estructura secundaria de x, donde $t_x : \mathcal{A}^* \times S^* \rightarrow \tau$ (siendo τ el espacio tridimensional) es la *función de plegamiento terciaria* de la secuencia x.

Estructura cuaternaria

La estructura cuaternaria es el par

$$(Q, f_Q) \tag{7.23}$$

donde Q es el conjunto de todas las sub-unidades de la proteína y f_Q es la *función de plegamiento cuaternaria* de Q.

Definición simbólica de las proteínas

Una proteína se puede definir como una cuaterna:

$$\begin{aligned} \textit{Proteina } P = \{&\textit{subunidades de } P;\ \textit{sus estructuras secundarias};\\ &\textit{sus estructuras terciarias};\ \textit{su estructura cuaternaria}\} \end{aligned} \tag{7.24}$$

es decir,

$$\begin{aligned} \textit{Proteina } P = \{&x_1, x_2, \ldots, x_n; (x_1, e_1), (x_2, e_2), \ldots, (x_n, e_n);\\ &\alpha_{x_1} = ((x_1, e_1), t_{x_1}), \alpha_{x_2} = ((x_2, e_2), t_{x_2}), \ldots, \alpha_{x_n} = ((x_n, e_n), t_{x_n});\\ &([\alpha_{x_1}, \alpha_{x_2}, \ldots, \alpha_{x_n}], f_{\alpha_{x_1}, \alpha_{x_2}, \ldots, \alpha_{x_n}})\} \end{aligned} \tag{7.25}$$

donde x_i son las sub-unidades de la proteína P, t_{x_i} son las funciones de plegamiento terciario y $f_{\alpha_{x_1}, \alpha_{x_2}, \ldots, \alpha_{x_n}}$ es la función de plegamiento cuaternario.

Por ejemplo, la *hemoglobina*, que está formada por cuatro sub-unidades, dos denominadas α y dos denominadas β, se podría definir simbólicamente como

$$\begin{aligned} \textit{Hemoglobina } P = \{&x, y; (x, e_x), (y, e_y); \alpha = ((x, e_x), t_x), \beta = ((y, e_y), t_y);\\ &([\alpha, \alpha, \beta, \beta], f_{\alpha,\alpha,\beta,\beta})\} \end{aligned} \tag{7.26}$$

Por último, se puede destacar que en la predicción de la estructura secundaria de las proteínas se trata con sub-unidades y no con la proteína completa. De esta forma, se define $P|_{x_i}$ para referirse sólo a la sub-unidad x_i, $P|_{x_i,e_i}$, para referirse a la sub-unidad x_i y a su estructura secundaria y $P|_{\alpha_{x_i}}$, para referirse a la sub-unidad x_i, a su estructura secundaria y a su estructura terciaria.

Definición simbólica de las ventanas de residuos

Sea $f : \mathcal{A}^* \rightarrow S^*$ una función de identificación de la estructura secundaria de una sub-unidad. Dado $x = (x_1, x_2, \ldots, x_n)$ y $e = (e_1, e_2, \ldots, e_n)$ con $f(x) = e$, se define:

$$f(x, e)|_{x_i} : \mathcal{A}^* \rightarrow S \tag{7.27}$$

en donde,

$$f(x, e)|_{x_i} = e_i \tag{7.28}$$

Dado un conjunto de datos (CD) para la predicción de la estructura secundaria de las proteínas, hay que recorrer secuencia por secuencia este conjunto de datos, generando las instancias para el aprendizaje y la validación de los diferentes clasificadores. La generación de estas instancias se realiza a través de una ventana de tamaño definido, que recorre todas las secuencias.

La ventana tiene los siguientes parámetros:

1. $i \in \mathbb{N}^+$ (número natural > 0), indica el punto de comienzo de la ventana según se desplaza a lo largo de la secuencia. Por ejemplo, el valor $i = 1$ indica que la ventana comienza en el residuo x_1, ó $i = 5$ que comienza en el residuo x_5.

2. $k \in \mathbb{N}^+$ indica el tamaño de la ventana. Por ejemplo, en PHD [RSS94a] se utilizan ventanas de tamaño 13.

3. $p \in \{1, 2, \ldots, k\}$ es una posición especial dentro de la ventana, e indica el residuo para el cual se debe aprender la estructura secundaria.

De esta forma se define la función W, que recibe los parámetros de la ventana y una estructura secundaria (x, e), y devuelve una subsecuencia de x y un elemento de e. Simbólicamente:

$$W : \mathbb{N}^+ \times \mathbb{N}^+ \times \mathbb{N}^+ \times (\mathcal{A}^*, S^*) \rightarrow \mathcal{A}^* \times S \tag{7.29}$$

en donde,

$$W(i, k, p, (x, e)) = (x_i, x_{i+1}, \ldots, x_{i+k-1},\ f(x)|_{x_{i+p}}) \tag{7.30}$$

con $(x, e) = (x_1, x_2, \ldots, x_n,\ e_1, e_2, \ldots, e_n)$.

7.4.2. Resultados

Líneas de trabajo futuro

Es necesario seguir desarrollando la presente modelización matemática. El siguiente paso es, sin duda, modelizar la información evolutiva de las proteínas. Este paso intermedio es imprescindible a la hora de poner definir de forma totalmente simbólica el problema de la predicción de la estructura secundaria de las proteínas.

7.5. Desarrollo de un clasificador para PSSP con redes Bayesianas

7.5.1. Objetivos

En esta sección proponemos el uso de los clasificadores naïve-Bayes e Interval Estimation naïve-Bayes (ambos métodos basados en redes bayesianas) para la predicción de la estructura secundaria de las proteínas [RLPP02, RPH+03].

El método que proponemos se basa en dos capas de predicción, la primera de tipo secuencia-estructura y la segunda del tipo estructura-estructura. Por supuesto, la primera capa tiene que ser capaz de trabajar con la información evolutiva de las proteínas, como cualquier otro método de tercera generación.

De esta forma, se explicará la obtención del conjunto de datos para el aprendizaje y la validación, el cambio que ha sido necesario hacer al clasificador naïve-Bayes para que

pueda trabajar con información evolutiva en la primera capa de predicción y el desarrollo de la segunda capa.

Por último, se describen las pruebas que han sido realizadas a fin de obtener la matriz de sustitución más adecuada para el cálculo de la información evolutiva y para el cálculo del tamaño de ventana óptimo. También se describen las pruebas realizadas para la segunda capa de predicción.

7.5.2. Obtención del conjunto de datos

Para obtener los casos con los que realizar los procesos de aprendizaje y validación de ambos métodos, se ha partido del conjunto de datos CB513 [CB99]. Este conjunto de datos, que contiene 513 proteínas, fue construido por Cuff y Barton en 1999.

Para cada una de las proteínas de este conjunto de datos se ejecuta el programa PSI-BLAST [AMS+97] contra la base de datos de proteínas PIR-NREF y se obtiene su perfil. Este perfil es una tabla que lista las frecuencias de cada aminoácido en cada posición de la proteína. El perfil tiene tantas filas como aminoácidos tenga la proteína y 20 columnas, una para cada aminoácido protéico. De esta forma, la posición (i, j) del perfil indica la frecuencia con que el aminoácido j aparece en la posición i de la proteína.

A partir de los perfiles se extraen los casos con los que se entrenan y validan los métodos. Estos casos están formados por una ventana de aminoácidos de tamaño impar y la clase correspondiente al aminoácido que se encuentra en la posición central de esa ventana. Se debe sacar un caso para cada uno de los residuos de las distintas proteínas que componen el conjunto de datos, con el objetivo de realizar la predicción de la estructura secundaria de cada uno de ellos.

7.5.3. Modificación del clasificador naïve-Bayes para el uso de información evolutiva

En los métodos de predicción de tercera generación es necesario el uso de la información evolutiva contenida en los perfiles de las proteínas. Como comentamos previamente, un perfil tiene tantas filas como aminoácidos tiene la proteína y 20 columnas, una para cada aminoácido protéico.

Por tanto, para cada variable predictora X_i necesitamos 20 valores, $p_i^{(1)}, \ldots, p_i^{(20)}$, correspondientes a cada uno de los 20 aminoácidos $a^{(1)}, \ldots, a^{(20)}$.

De esta forma, la ecuación estándar del naïve-Bayes

$$P(C = c_i | X_1 = x_1, \ldots, X_n = x_n) \propto P(C = c_i) \prod_{k=1}^{n} P(X_k = x_k | C = c_i) \tag{7.31}$$

la podemos calcular como,

$$P(C = c_i | X_1 = p_1^{(1)}, \ldots, p_1^{(20)}, \ldots, X_n = p_n^{(1)}, \ldots, p_n^{(20)}) \propto$$
$$P(C = c_i) \prod_{k=1}^{n} \sum_{j=1}^{20} p_k^{(j)} P(X_k = a^{(j)} | C = c_i) \tag{7.32}$$

7.5.4. Resultados experimentales de la primera capa de predicción

Experimentación con naïve-Bayes

Dentro de la primera capa de predicción se ha realizado una batería de pruebas con el objetivo de encontrar, tanto el tamaño de ventana óptimo, como la matriz de sustitución más adecuada para la obtención de los perfiles de las proteínas. Los resultados se pueden observar en la figura 7.2.

Como se puede apreciar, el mejor resultado obtenido corresponde a un tamaño de ventana de 13 residuos y a la matriz de sustitución BLOSUM62.

Mientras que la mejor de las pruebas realizadas con información evolutiva dio un 67,58 % de bien clasificados, las mismas pruebas realizadas sin información evolutiva dieron sólo un 61,35 %. Esto significa que el uso de la información evolutiva supone una mejora de 6,23 %.

Experimentación con Interval Estimation naïve-Bayes

Dado que el tiempo de computación requerido por este método es muy elevado, se realizó sólo una prueba con los parámetros encontrados con el naïve-Bayes: ventana de 13 residuos y matriz de sustitución BLOSUM62.

El resultado obtenido es de un 70,33 % de bien clasificados, por lo que se supera al clasificador naïve-Bayes en un 2,75 %.

7.5.5. Segunda capa de predicción

La información de entrada a la segunda capa de predicción son las estructuras observadas de las proteínas y las estructuras predichas por los métodos empleados en la primera capa de predicción, naïve-Bayes o *Interval Estimation naïve-Bayes.*

A partir de estas estructuras se obtiene un conjunto de datos con los casos que servirán para el proceso de aprendizaje y de validación de esta segunda capa.

Los procesos de aprendizaje y validación son idénticos a los de la primera capa con naïve-Bayes. La única diferencia es que aquí cada variable predictora tiene un único estado para cada caso (ya que no hay información evolutiva), por lo que se puede utilizar la fórmula original del clasificador naïve-Bayes.

7.5.6. Resultados experimentales de la segunda capa de predicción

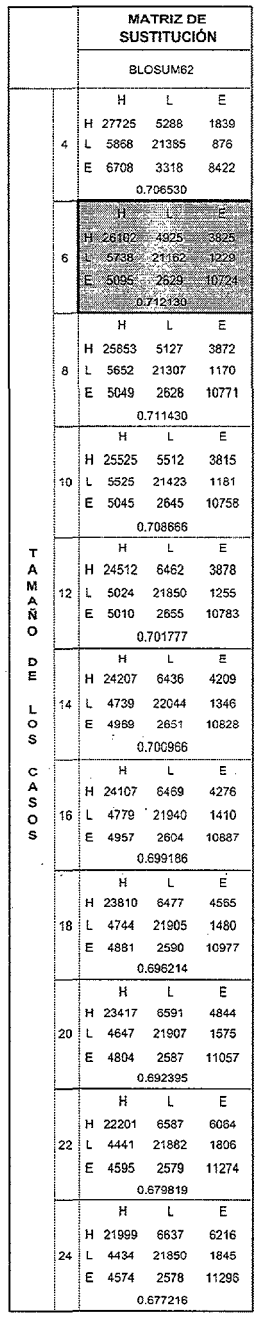
Para realizar las pruebas experimentales con la segunda capa de predicción se ha partido del mejor de los resultados obtenido con la primera capa, es decir, de los resultados obtenidos con *Interval Estimation naïve-Bayes.*

MATRIZ DE SUSTITUCIÓN

TAMAÑO DE LOS CASOS		BLOSUM45			BLOSUM62			BLOSUM80			PAM30			PAM70		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	4	H 28848	5588	738	H 29159	5660	716	H 28927	5713	790	H 27476	6316	995	H 28577	5923	874
		L 9429	18268	618	L 9488	18520	612	L 9304	18520	689	L 8819	18330	998	L 9089	18586	797
		E 8318	6767	3708	E 8355	6914	3594	E 8147	6929	3752	E 7683	6807	4071	E 7933	6914	3940
		0.617681			0.617613			0.618562			0.612025			0.618433		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	6	H 28648	5776	757	H 29015	5814	715	H 28800	5867	778	H 27506	6280	1026	H 28511	5991	888
		L 7629	20176	512	L 7604	20527	492	L 7440	20532	542	L 7257	20088	808	L 7427	20391	657
		E 7934	6434	4426	E 7967	6580	4317	E 7795	6567	4468	E 7491	6324	4752	E 7655	6498	4638
		0.647086			0.648661			0.649545			0.642030			0.647745		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	8	H 28141	6141	902	H 28542	6136	870	H 28301	6238	912	H 27147	6479	1204	H 27991	6377	1030
		L 6636	21178	504	L 6668	21450	507	L 6486	21480	550	L 6449	20917	791	L 6474	21359	645
		E 7766	5970	5058	E 7826	6098	4940	E 7704	6038	5089	E 7336	5966	5269	E 7514	6022	5258
		0.660749			0.661536			0.662697			0.653927			0.660554		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	10	H 27459	6577	1151	H 27845	6603	1104	H 27617	6662	1157	H 26520	6857	1458	H 27336	6777	1291
		L 6091	21711	517	L 6139	21965	523	L 6029	21898	591	L 6030	21311	819	L 6036	21773	670
		E 7501	5443	5850	E 7540	5565	5759	E 7437	5466	5928	E 7082	5439	6054	E 7272	5449	6078
		0.668530			0.669159			0.669561			0.660517			0.667461		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	12	H 26851	6975	1364	H 27208	7045	1303	H 26971	7093	1396	H 26032	7142	1684	H 26764	7112	1535
		L 5751	22003	566	L 5785	22274	570	L 5715	22189	616	L 5764	21546	854	L 5718	22054	698
		E 7099	5187	6508	E 135	5276	6453	E 7017	5173	6641	E 6818	5107	6652	E 6881	5163	6758
		0.672653			0.673518			0.673836			0.664591			0.672197		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	14	H 26329	7350	1512	H 26652	7463	1433	H 26464	7480	1518	H 25606	7474	1788	H 26302	7498	1615
		L 5455	22238	629	L 5435	22565	631	L 5445	22398	679	L 5516	21750	902	L 5452	22275	755
		E 6827	4982	6986	E 6933	5028	6905	E 6771	4984	7077	E 6607	4931	7042	E 6681	4963	7161
		0.674940			0.675841			0.675461			0.666511			0.673962		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	16	H 25923	7663	1606	H 26237	7762	1561	H 26052	7801	1611	H 22188	6731	1757	H 25939	7742	1738
		L 5256	22420	648	L 5262	22745	627	L 5245	22586	693	L 4681	19163	869	L 5251	22468	765
		E 6697	4903	7196	E 6703	5078	7086	E 6592	4955	7286	E 5720	4213	6791	E 6505	4925	7378
		0.674738			0.675022			0.675239			0.667406			0.674457		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	18	H 25596	7946	1651	H 25881	8109	1572	H 25780	8019	1666	H 25059	7910	1915	H 25671	7972	1780
		L 5099	22604	623	L 5073	22924	640	L 5080	22769	678	L 5216	22057	904	L 5084	22642	761
		E 6561	5015	7221	E 6554	5131	7183	E 6485	5019	7330	E 6383	4943	7260	E 6402	5008	7400
		0.673271			0.674010			0.674655			0.665989			0.673513		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	20	H 25309	8245	1639	H 25611	8400	1553	H 25512	8300	1654	H 24846	8140	1906	H 25376	8287	1764
		L 4992	22721	615	L 4939	23082	619	L 4944	22923	663	L 5117	22182	883	L 4990	22749	750
		E 6544	5077	7177	E 6507	5225	7137	E 6430	5123	7282	E 6301	4997	7290	E 6365	5071	7376
		0.670647			0.672060			0.672659			0.665156			0.670885		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	22	H 25051	8499	1643	H 25358	8646	1561	H 25290	8531	1645	H 24605	8372	1921	H 25232	8458	1739
		L 4886	22836	608	L 4860	23176	607	L 4871	23012	650	L 5025	22274	888	L 4939	22810	742
		E 6491	5202	7106	E 6427	5349	7094	E 6365	5209	7262	E 6252	5073	7267	E 6299	5165	7350
		0.668023			0.669588			0.670779			0.662928			0.669519		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	24	H 24859	8677	1658	H 25147	8865	1555	H 25098	8727	1641	H 24499	8489	1917	H 25016	8673	1742
		L 4809	22917	605	L 4785	23277	583	L 4804	23093	639	L 4991	22308	893	L 4861	22896	736
		E 6409	5290	7101	E 6361	5430	7080	E 6305	5283	7249	E 6199	5161	7235	E 6233	5255	7328
		0.666590			0.668055			0.669250			0.661534			0.667634		
		H	L	E	H	L	E	H	L	E	H	L	E	H	L	E
	26	H 24696	8866	1632	H 24961	9057	1550	H 24873	8954	1639	H 24336	8650	1927	H 24885	8812	1735
		L 4718	23013	601	L 4708	23348	591	L 4745	23143	651	L 4919	22378	899	L 4823	22940	732
		E 6336	5377	7089	E 6270	5535	7068	E 6221	5374	7243	E 6143	5228	7227	E 6174	5321	7324
		0.665505			0.666486			0.667033			0.660176			0.666485		

Figura 7.2: Resultados para la primera capa de predicción con naïve-Bayes

En la figura 7.3 se muestran los resultados obtenidos considerando diversos tamaños de ventana de predicción.



TAMAÑO DE LOS CASOS	MATRIZ DE SUSTITUCIÓN: BLOSUM62			
4		H	L	E
	H	27725	5288	1839
	L	5868	21385	876
	E	6708	3318	8422
	0.706530			
6		H	L	E
	H	26102	4925	3825
	L	5738	21162	1229
	E	5095	2629	10724
	0.712130			
8		H	L	E
	H	25853	5127	3872
	L	5652	21307	1170
	E	5049	2628	10771
	0.711430			
10		H	L	E
	H	25525	5512	3815
	L	5525	21423	1181
	E	5045	2645	10758
	0.708666			
12		H	L	E
	H	24512	6462	3878
	L	5024	21850	1255
	E	5010	2655	10783
	0.701777			
14		H	L	E
	H	24207	6436	4209
	L	4739	22044	1346
	E	4989	2651	10828
	0.700966			
16		H	L	E
	H	24107	6469	4276
	L	4779	21940	1410
	E	4957	2604	10887
	0.699186			
18		H	L	E
	H	23810	6477	4565
	L	4744	21905	1480
	E	4881	2590	10977
	0.696214			
20		H	L	E
	H	23417	6591	4844
	L	4647	21907	1575
	E	4804	2587	11057
	0.692395			
22		H	L	E
	H	22201	6587	6064
	L	4441	21882	1806
	E	4595	2579	11274
	0.679819			
24		H	L	E
	H	21999	6637	6216
	L	4434	21850	1845
	E	4574	2578	11296
	0.677216			

Figura 7.3: Pruebas de la segunda capa de predicción empleando los resultados obtenidos por *Interval Estimation naïve-Bayes*

Los mejores resultados se han obtenido con un tamaño de ventana de 5 residuos. La exactitud de las predicciones ha mejorado de un 70,33 % en la primera capa de predicción con *Interval Estimation naïve-Bayes*, a un 71,21 %. Esto supone una mejora de 0,88 %.

7.5.7. Conclusiones

El problema de la predicción de la estructura secundaria de las proteínas (PSSP) ya había sido abordado desde diversos enfoques metodológicos, muchos de ellos pertenecientes al mundo del aprendizaje automático. Sin embargo, nunca se había intentado solucionar este problema con el uso de redes Bayesianas.

La principal dificultad en este desarrollo ha sido la de conseguir introducir la información evolutiva de las proteínas en las redes Bayesianas. Esta información es de vital importancia en todos los métodos de predicción actuales.

En cuanto a los resultados obtenidos, los mejores métodos de predicción actuales llegan a un 80 % de bien clasificados, mientras que con las redes Bayesianas se ha obtenido un 71,21 %.

Sin embargo, todavía quedan por desarrollar varias extensiones que pueden mejorar estos resultados, por lo que somos optimistas con estos resultados.

7.5.8. Líneas de trabajo futuro

Prácticamente todos los predictores de estructura secundaria actuales realizan, en su primera capa de predicción, una multiclasificación. El clasificador básico es entrenado con diferentes tamaños de ventana y diferentes conjuntos de datos, realizándose una multiclasificación de todos estos resultados. Esta multiclasificación supone, normalmente, un 4 % de mejora aproximada. Es posible usar esta misma idea con el clasificador basado en redes Bayesianas.

Por otra parte, la segunda capa de predicción que proponemos sólo mejora un 0,88 %, muy por debajo de las mejoras obtenidas por otros métodos. De esta forma, nos planteamos sustituir la segunda capa de predicción por una basada en redes de neuronas.

Creemos que con estos cambios la exactitud de este método puede llegar a ser similar a la de los métodos actuales.

7.6. Multiclasificador basado en redes Bayesianas para PSSP

7.6.1. Objetivos

Como se ha podido ver en el capítulo anterior, son muchos los algoritmos de predicción ya existentes para el problema PSSP. Lo que proponemos en este apartado es la realización de un multiclasificador basado en redes Bayesianas, es decir, un clasificador de clasificadores basado en el paradigma *stacked generalization* [RLP+03c].

Durante los últimos años, los investigadores de la comunidad del aprendizaje automático, han intentado averiguar cómo combinar un conjunto de clasificadores en una gran cantidad de dominios de aplicación. Se ha demostrado que con la combinación de las predicciones se pueden obtener mejores exactitudes [HS94, SSL+99].

En esta sección presentamos un multiclasificador para la predicción de la estructura secundaria de las proteínas basado en el conocido paradigma *stacked generalization* [Wol92]. En este paradigma el multiclasificador se diseña como un conjunto de capas de clasificadores, en donde las capas superiores reciben las predicciones de la capa inmediatamente

inferior.

Para el problema de la predicción de la estructura secundaria hemos diseñado un sistema de clasificación basado en dos niveles. En el *nivel* − 0 hemos situado todos los servidores de predicción encontrados en Internet, mientras que en el *nivel* − 1 hemos situado una red Bayesiana que actúa como sistema de consenso. Una vez construido el multiclasificador, y dada una nueva instancia a clasificar, se ejecutan los clasificadores del *nivel* − 0 con dicha instancia y se pasan sus predicciones a la red Bayesiana del *nivel* − 1. Como se verá, los resultados experimentales demuestran que el multiclasificador mejora los resultados de todos los clasificadores utilizados. En la figura 7.4 se puede ver el esquema utilizado.

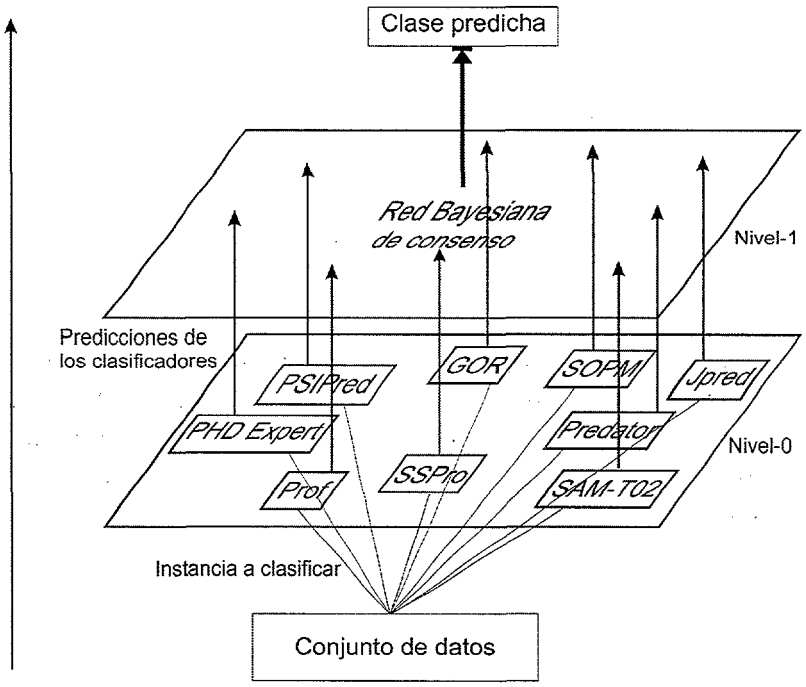


Figura 7.4: Esquema del multiclasificador

Con el objetivo de realizar el multiclasificador se ha desarrollado una aplicación Web que contacta con los servidores Web de predicción de la estructura secundaria. A través de esta aplicación se han logrado dos objetivos:

- Comparar los algoritmos de predicción actuales, extrayendo estadísticas sobre la exactitud de sus predicciones.
- Conseguir el conjunto de datos necesario para la realización del multiclasificador.

La aplicación Web ha sido desarrollada de forma flexible, por lo que resulta muy sencilla la incorporación de nuevos servidores de predicción en caso necesario. Teniendo en cuenta los temas de portabilidad, la aplicación ha sido desarrollada en JSP y clases Java.

En el desarrollo del multiclasificador se han diferenciado cinco fases, las cuales representan los diferentes objetivos que se pretenden conseguir:

- Obtener los servidores de PSSP existentes en Internet. De cada servidor se almacenará la información necesaria para el posterior envío automático de los distintos conjuntos de datos.
- Enviar los conjuntos de datos a los servidores obtenidos en la fase anterior. Además, se deberán recoger los resultados de la predicción de la estructura secundaria de dichos servidores, almacenándolos para estudios posteriores. Los servidores pueden devolver la información vía correo electrónico o a través de página Web.
- Creación del conjunto de datos para el multiclasificador, a partir de los resultados obtenidos de los servidores de predicción.
- Desarrollo de diferentes multiclasificadores, todos ellos basados en redes Bayesianas.
- Validar los resultados obtenidos, tanto de los servidores de predicción contactados, como del multiclasificador desarrollado.

7.6.2. Clasificadores de nivel-0

Tras una búsqueda exhaustiva por Internet se encontraron 9 servidores de predicción de PSSP en funcionamiento. Estos servidores son los que forman parte del *nivel* − 0.

En la tabla 7.2 se muestran, a modo de resumen, los servidores contactados. Por cada servidor se muestra su localización y su método de predicción. Además, en la figura 7.5, se puede observar la localización geográfica de los servidores.

Servidores Web de predicción de estructura secundaria		
Servidor	*Localización*	*Método predicción*
GOR	Universidad de Southampton, Reino Unido	Teoría de la info.
JPred	Universidad Dundee, Escocia	Consenso
PHD	Universidad de Columbia, Estados Unidos	Redes de neuronas
PREDATOR	Instituto Pasteur, Francia	Alg. vecino más cerc.
Prof	Universidad de Gales, UK	Redes de neuronas
PSIPRED	Universidad College London, Reino Unido	Redes de neuronas
SAM-T02	Universidad de California, Santa Cruz, USA	Homologías
SOPM	Instituto de Biología de Lyon, Francia	Homologías
SSPro	Universidad de California Irvine, Estados Unidos	Redes de neuronas

Tabla 7.2: Servidores Web de predicción de estructura secundaria

A continuación se detallan los servidores de predicción que han sido utilizados:

1. **Servidor GOR**

   GOR [GOR78] es un método de predicción de la estructura secundaria basado en la teoría de la información. Su nombre está formado por la iniciales de los nombres de sus desarrolladores originales (Garnier, Osguthorpe, y Robson). Utilizaron ventanas deslizantes de 17 residuos para obtener los estados conformacionales.

Figura 7.5: Disposición geográfica de los servidores de predicción de estructura secundaria

2. **Servidor JPred**

   JPred [CCS+98] es un servidor de Internet interactivo para la predicción de la estructura secundaria de las proteínas. El servidor permite enviar una secuencia o un alineamiento múltiple, y devuelve las predicciones de seis algoritmos de predicción de la estructura secundaria que emplean la información evolutiva de secuencias múltiples. Se devuelve además una predicción de consenso, la cual mejora la exactitud media Q_3 de la predicción en un 1 %, obteniendo de esta forma una exactitud de 72,9 % (según sus autores).

   De esta forma, JPred ejecuta seis métodos de predicción diferentes (DSC [KS96], PHD [RS93], NNSSP [SS95], PREDATOR [FA97], ZPRED [ZBTS87] y MULPRED [BT88]), y combina los resultados de cada uno de ellos.

   Los métodos NNSSP, DSC, PREDATOR, MULPRED, ZPRED y PHD se eligieron como representantes de los métodos de predicción de la estructura secundaria, que emplean la información evolutiva de secuencias múltiples. Cada uno obtiene su predicción utilizando una heurística diferente, basada en el vecino más próximo (NNSSP), redes de neuronas (PHD), discriminación lineal (DSC), método de combinación de secuencias simples por consenso (MULPRED), propensiones de enlaces de hidrógeno (PREDATOR), o conservación de predicción ponderada (ZPRED).

   A lo largo del desarrollo de la tesis, se han observado diferentes problemas con este servidor, ya que de las 3515 secuencias enviadas, 72 no han recibido resultado.

3. **Servidor PHD**

   PHD [RSS94a] es un servidor de predicción de estructura secundaria al que se envía una secuencia de aminoácidos y que devuelve, vía correo electrónico, un alineamiento

múltiple de la secuencia y la predicción de la estructura secundaria. Está basado en una red de neuronas multicapa.

PHD presenta tres modos de funcionamiento diferentes:

- Modo *por defecto*. Muestra una página Web con una interfaz muy simple que sólo admite la secuencia de aminoácidos, tomando el resto de parámetros los valores por defecto.
- Modo *avanzado*. Muestra los mismos campos que en el modo *por defecto* pero da la posibilidad de establecer parámetros tales como *tipo de predicción*, *PSI-BLAST*, *formato de salida para el alineamiento múltiple de la secuencia*, *formato de la secuencia de entrada*, etc.
- Modo *experto*. En este caso se da la posibilidad de establecer todos los parámetros posibles para realizar la predicción (base de datos donde buscar proteínas similares, parámetros para el alineamiento, etc.).

Para la realización del envío de los conjuntos de datos, se escogió el modo *experto*. Al realizar los envíos se encontró el problema de saturar el servidor al enviarle gran cantidad de secuencias. Para solucionar este problema se han tenido que enviar bloques de aproximadamente 20 secuencias.

4. **Servidor PREDATOR**

Frishman y Argos crearon PREDATOR [FA96, FA97] en un esfuerzo de combinar el algoritmo estándar del vecino más cercano, con otros resultados que reflejan interacciones no locales de puentes de hidrógeno.

El comportamiento del servidor PREDATOR con los envíos de los conjuntos de datos ha sido bastante satisfactorio. Las respuestas de las predicccciones han sido muy rápidas durante el envío de las secuencias. El único inconveniente encontrado en este servidor, es el de no predecir secuencias de menos de 30 aminoácidos. De esta forma de las 3515 secuencia enviadas, la estructura secundaria no fue predicha para 19 de ellas (13 de la cuales pertenecen al conjunto de datos CB513) por ser su longitud menor que 30.

5. **Servidor Prof**

El servidor de predicción de la estructura secundaria de las proteínas Prof [OK00] se ha utilizado para predecir todas las secuencias. Todas las predicciones se han realizado automáticamente usando los valores por defecto del servidor. Prof predice la estructura, conectando en cascada diversos tipos de clasificadores basados en redes de neuronas y discriminación lineal [OK00]. Los autores estimaron la exactitud de Prof en 76.7 % con el conjunto de datos CB513 [CB99].

En cuanto a su funcionamiento al enviar los conjuntos de datos, al contrario que los anteriores servidores, no se satura a causa de la cola de trabajos. La desventaja que presenta es el tiempo de respuesta de la predicción, que puede ser de hasta 10 horas. Además, algunos envíos no obtuvieron respuesta. De las 3515 secuencias enviadas, 70 no obtuvieron resultado.

6. **Servidor PSIPRED**

El servidor de predicción de estructura secundaria PSIPRED [Jon99] permite al usuario enviar una secuencia de aminoácidos, realizar la predicción de su estructura secundaria y recibir los resultados de la predicción a través de correo electrónico.

PSIPRED es un método simple y fiable de predicción de la estructura secundaria. Basa sus predicciones en un clasificador de dos capas implementadas con redes de neuronas. La primera red neuronal toma como entrada la matriz PSSM obtenida al ejecutar PSI-BLAST sobre secuencia de partida. La segunda red neuronal es del tipo estructura-estructura, similar a la que se describió en el algoritmo PHD [RSS94a].

La versión 2.0 de PSIPRED incluye una mejora que realiza un promedio de la salida de hasta 4 redes de neuronas independientes usadas en el proceso de predicción de la primera capa. De esta forma se consigue aumentar la exactitud.

En cuanto a su funcionamiento, este es uno de los servidores que mejor comportamiento ha tenido. El único inconveniente que se ha encontrado es que su cola de trabajos es muy pequeña (alrededor de 10 trabajos). Sin embargo, teniendo en cuenta que la predicción se obtiene en aproximadamente 2 minutos, esta desventaja se atenúa.

7. **Servidor SAM-T02**

SAM-T02 [KBH98] es un método iterativo de búsqueda que crea *modelos ocultos de Markov* de una secuencia de proteína, a través de búsquedas iterativas en una base de datos de proteínas. Actualmente, es el algoritmo de detección de homología remota más sensible.

Respecto a su funcionamiento, es el servidor más lento de todos los utilizados. No permite enviar secuencias muy continuadas ya que se bloquea el acceso. Sólo permite enviar unas 35 secuencias por día.

8. **Servidor SOPM**

El método SOPM [GD94] está basado en la base de datos DSSP [KS83] que contiene las proteínas y sus estructuras secundarias conocidas. De esta forma, dada una nueva secuencia, utiliza un algoritmo predictivo basado en similitudes de secuencia.

Los resultados de las predicciones, para los conjuntos de datos enviados, se han obtenido de forma rápida y sencilla, sin encontrar ningún problema en el funcionamiento del servidor.

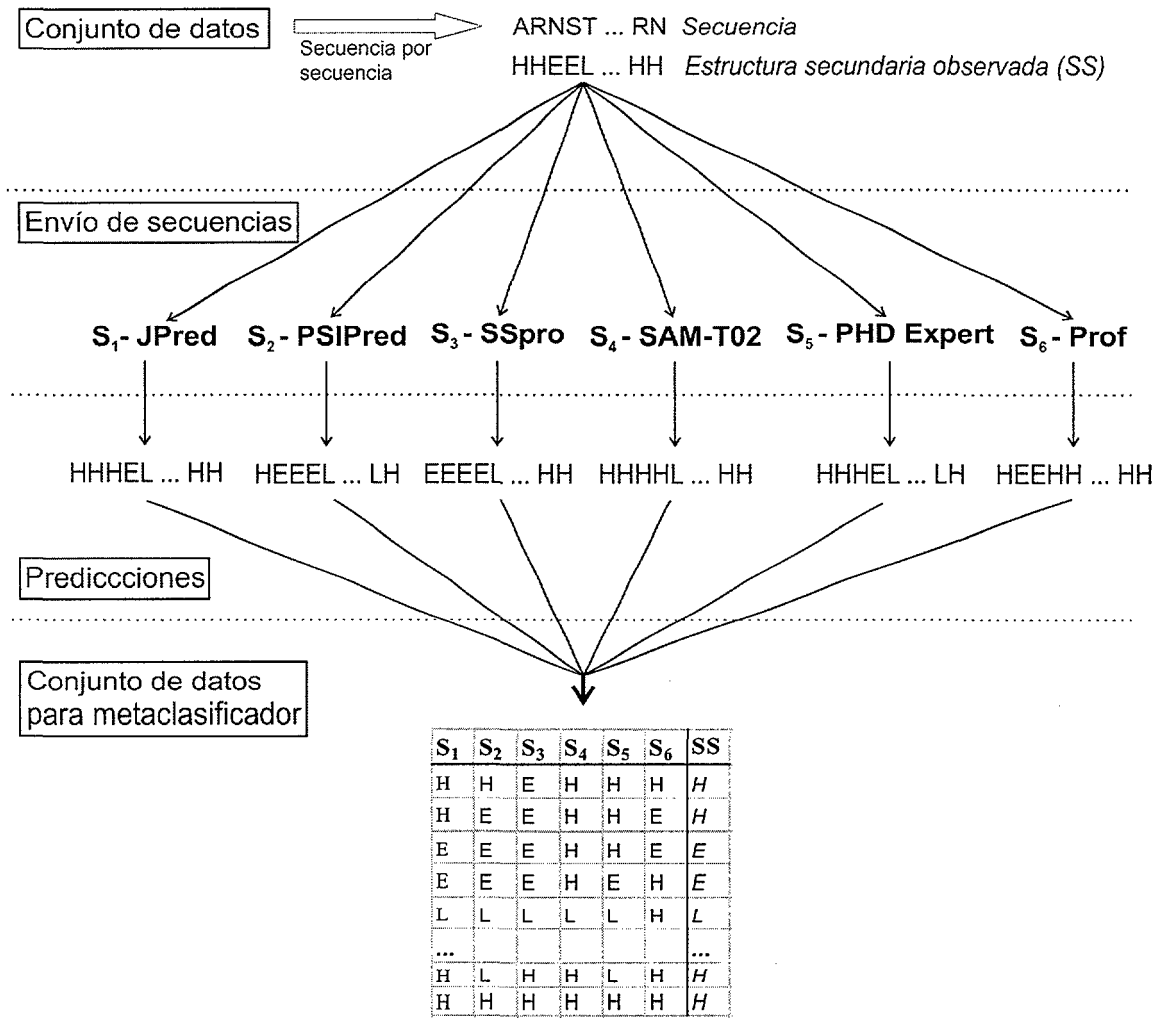
9. **Servidor SSPro**

SSPro es un sistema totalmente automatizado para la predicción de la estructura secundaria de las proteínas. El sistema se basa en un conjunto de redes de neuronas recurrentes bidireccionales (BRNNs) [BBF$^+$99a, BBF$^+$99b].

Un conjunto de 11 redes de neuronas recurrentes bidireccionales realizan el aprendizaje. Las redes contienen aproximadamente 70.000 pesos ajustables, y realizan el aprendizaje utilizando una entropía relativa entre las distribuciones objetivo y de salida.

Este servidor ha sido el único al que no se han podido enviar automáticamente todas las secuencias de los conjuntos de datos. Después de enviar los conjuntos de datos CB513, RS126 y HS1771, el servidor bloqueó el acceso por no poder servir tantas

predicciones. Por esta razón, el resto de los conjuntos de datos tuvieron que ser enviados por correo electrónico al administrador, que devolvió las predicciones por correo electrónico.



S_1	S_2	S_3	S_4	S_5	S_6	SS
H	H	E	H	H	H	*H*
H	E	E	H	H	E	*H*
E	E	E	H	H	E	*E*
E	E	E	H	E	H	*E*
L	L	L	L	L	H	*L*
...						...
H	L	H	H	L	H	*H*
H	H	H	H	H	H	*H*

Figura 7.6: Obtención del conjunto de datos para el multiclasificador

7.6.3. Obtención del conjunto de datos para el multiclasificador

Para la obtención del conjunto de datos del multiclasificador, tal y como se puede observar en la figura 7.6 –aunque en total han sido usados nueve servidores, en la figura sólo aparecen los seis mejores–, han sido necesarios varios pasos:

1. Seleccionar un conjunto de proteínas con estructura secundaria conocida. Se han utilizado todos los conjuntos de datos de proteínas descritos en la sección 7.2. Por tanto, se usarán los conjuntos HS1771, CB513, RS126 y los seis conjuntos del datos

del proyecto EVA [RE01]. De ellos, sólo el conjunto HS1771 será utilizado para el aprendizaje (por ser considerado el más completo de ellos) y el resto de conjuntos serán utilizados para la validación.

2. Mandar estas secuencias de proteínas a los nueve servidores de Internet y esperar sus respectivas respuestas por correo electrónico o vía Web.

3. Procesar los correos electrónicos o las páginas Web, y extraer la información de la estructura secundaria predicha por cada uno de ellos.

4. Con estos nueve resultados obtenidos se construye el conjunto de datos del multiclasificador. Por cada aminoácido de la proteína se construye una instancia del conjunto de datos, que contiene las predicciones de cada servidor y la estructura real.

7.6.4. Clasificadores de nivel-1

Una vez recibidas las predicciones de los nueve servidores, se calcularon las estadísticas descritas en el apartado 7.3. Las estadísticas obtenidas se muestran en la siguiente sección de resultados experimentales.

Con las estadísticas en mano, se realizaron varios multiclasificadores basados en redes Bayesianas. Se han hecho un total de cinco multiclasificadores:

- Naïve-Bayes con los cinco mejores servidores (MC-NB-5).
- Naïve-Bayes con los seis mejores servidores (MC-NB-6).
- Interval Estimation Naïve-Bayes con los cinco mejores servidores (MC-IENB-5).
- Interval Estimation Naïve-Bayes con los seis mejores servidores (MC-IENB-6).
- Pazzani-EDA con los seis mejores servidores (MC-Pazzani-EDA). Pazzani-EDA realiza una selección de variables, por lo que no tendría sentido ejecutarlo con los cinco mejores servidores.

Los cinco mejores servidores han sido: SSPro, SAM-T02, Prof, PHD Expert y PSIPRED. El sexto mejor servidor fue JPred.

7.6.5. Resultados

Resultados experimentales

Se han extraído las estadísticas de los nueves servidores Web de predicción de la estructura secundaria descritos en la sección 7.6.2, y de los cinco multiclasificadores descritos en la sección 7.6.4. Las tablas 7.6.5 a 7.6.5 contienen las estadísticas para los conjuntos de datos de proteínas seleccionados. Son nueve tablas en total, para los conjuntos: HS1771, CB513, RS126, EVA1, EVA2, EVA3, EVA4, EVA5 y EVA6.

Los resultados experimentales muestran que el servidor de predicción que mejores resultados obtiene es el PSIPRED. Además, también se demuestra que es posible mejorar estos resultados a través de los multiclasificadores.

En la tabla 7.3 se muestran las mejoras obtenidas por los multiclasificadores en los diferentes conjuntos de datos respecto al algoritmo PSIPRED, que ha demostrado ser el mejor de los clasificadores. La mayor de las mejoras se ha conseguido con el clasificador MC-Pazzani-EDA que porcentualmente supera al método PSIPRED en 1,21 %. Sin embargo, y teniendo en cuenta que al máximo valor que se puede obtener se estima en un 88 %, la mejora de error relativa es de 13,30 %. MC-Pazzani-EDA mejora los resultados de PSIPRED en todos los conjuntos de datos menos uno, el EVA6. En la figura 7.7 se puede observar la estructura final obtenida por Pazzani-EDA para el conjunto de datos del multiclasificador. Como se puede apreciar, los cuatro mejores servidores de predicción han sido introducimos en un único grupo.

Comparación de los resultados experimentales de los multiclasificadores					
Conjunto	*MC-NB-5*	*MC-NB-6*	*MC-IENB-5*	*MC-IENB-6*	*MC-Pazzani-EDA*
HS1771	1,02	0,82	1,10	0,84	1,35
CB513	0,66	0,58	0,73	0,57	1,04
RS126	-0,62	-0,80	-0,46	-0,79	0,64
EVA1	1,51	1,48	1,47	1,51	2,03
EVA2	1,55	1,50	1,52	1,53	2,11
EVA3	1,21	0,86	1,23	0,91	1,9
EVA4	1,12	0,75	1,16	0,79	1,79
EVA5	1,14	0,82	1,19	0,84	1,72
EVA6	-0,09	0,09	-0,06	0,10	-1,70
Mejora media	0,83	0,68	0,88	0,70	1,21

Tabla 7.3: Comparación de los resultados experimentales obtenidos con los multiclasificadores en relación con PSIPRED

El análisis que se realiza a continuación está basado en el conjunto de datos HS1771.

Las mejoras se han obtenido, sobre todo, en la predicción de las β láminas, ya que su predicción ha pasado de un 69,18 % en el PSIPRED a un 79,05 % en MC-Pazzani-EDA (es importante recordar que la estructura β laminar es la más complicada de predecir). Este espectacular incremento conlleva un empeoramiento de la predicción de la estructura *coil* que pasa del 80,29 % al 76,76 % (aunque sus predicciones son de mejor calidad, ya que pasan de un 75,72 % a un 80,94 %).

Respecto al índice de información, todos los multiclasificadores son superiores al método PSIPRED. Los multiclasificadores presentan unos índices de información de 0,43 ó 0,44, mientras que PSIPRED tiene un índice de información de 0,41.

En los coeficientes de correlación de Matthews los multiclasificadores también son superiores, sea cual sea la estructura secundaria analizada.

Por último, en las estadísticas por segmentos, se mejora en las hélices y en las β láminas, pero se empeora en las *coil*.

Conclusiones

Es esta sección se ha propuesto la realización de un multiclasificador basado en redes Bayesianas, para la predicción de la estructura secundaria de las proteínas.

Para poder llegar a este objetivo se ha desarrollado una aplicación Web en JSP y clases

de Java, que ha cumplido con dos importantes objetivos:

- Comparar los algoritmos de predicción actuales, extrayendo estadísticas sobre la exactitud de sus predicciones.
- Obtener el conjunto de datos necesario para la realización del multiclasificador de predicción de estructura secundaria.

Una vez recopilado el conjunto de datos para el multiclasificador, se experimentó con varios de los algoritmos de clasificación desarrollados a lo largo de la tesis.

Los multiclasificadores han demostrado un excelente resultado, ya que se ha conseguido superar en un 1,21 % de media al mejor de los clasificadores actuales. Además, se ha logrado una mejora espectacular en la predicción de las β láminas, la estructura más complicada de predecir.

Líneas de trabajo futuro

Todavía queda mucho trabajo a desarrollar basado en el multiclasificador. Algunas ideas son:

- Probar más algoritmos de clasificación como multiclasificadores. En este punto se está ya avanzando en el uso de *rough sets* como una posible alternativa.
- Poner el multiclasificador a disposición pública. En este sentido, nos queremos poner en contacto con los responsables del proyecto EVA, para una posible integración del multiclasificador en su sistema.
- Crear un punto central en Internet a través del cual, y con una sola petición, se puedan conseguir los resultados de todos los servidores de predicción de estructura secundaria de Internet.

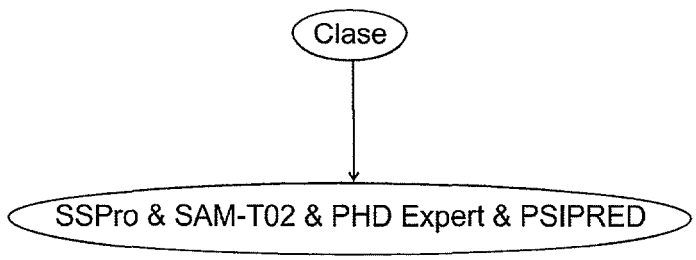


Figura 7.7: Estructura de Pazzani encontrada para el multiclasificador

Servidores	Conjunto de datos HS1771														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	78,90	83,40	69,18	80,29	83,34	78,19	75,72	0,41	0,74	0,67	0,61	77,86	71,53	71,65	74,62
PHD Expert	77,37	80,01	72,87	77,55	84,73	71,58	74,81	0,38	0,73	0,64	0,58	72,81	72,92	70,22	73,44
Prof	74,57	71,56	71,01	78,95	86,50	67,27	70,93	0,34	0,69	0,60	0,55	65,32	69,95	69,43	69,74
GOR	54,31	56,67	45,28	57,13	54,43	42,34	61,49	0,08	0,31	0,27	0,31	52,15	54,41	49,60	50,21
SOPM	65,66	70,34	54,85	67,48	67,40	57,52	68,28	0,19	0,51	0,44	0,45	64,38	64,28	60,81	62,52
SSPro	78,04	82,65	66,26	80,44	83,84	76,83	74,17	0,40	0,74	0,64	0,59	74,70	70,09	69,81	71,78
JPred	71,68	63,56	54,57	87,52	87,04	73,13	64,40	0,30	0,64	0,54	0,52	62,78	63,23	67,82	65,91
SAM-T02	77,54	84,39	74,97	73,20	81,41	72,16	77,14	0,39	0,73	0,66	0,58	75,54	73,56	68,15	73,05
Predator	66,29	65,42	45,83	77,83	71,00	64,42	63,90	0,20	0,52	0,44	0,45	63,64	54,73	61,95	61,07
MC-NB-5	79,92	87,09	77,47	75,81	82,50	72,66	81,62	0,43	0,77	0,68	0,62	83,47	76,07	66,38	72,19
MC-NB-6	79,72	86,40	76,77	76,09	82,87	73,01	80,64	0,43	0,77	0,68	0,62	82,28	75,29	66,51	72,11
MC-IENB-5	80,00	86,96	77,19	76,16	82,65	73,49	81,24	0,44	0,77	0,68	0,62	82,67	75,97	66,70	72,37
MC-IENB-6	79,74	86,47	76,79	76,09	82,87	73,11	80,65	0,43	0,77	0,68	0,62	82,39	75,37	66,51	72,15
MC-Pazzani-EDA	80,25	85,41	79,05	76,76	84,80	71,80	80,94	0,44	0,77	0,69	0,63	80,16	77,30	67,00	72,28

Tabla 7.4: Estadísticas para el conjunto de datos HS1771

Servidores	Conjunto de datos CB513														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	79,95	83,52	70,31	82,16	84,67	80,14	76,37	0,43	0,76	0,68	0,63	78,98	69,81	73,42	76,48
PHD Expert	77,61	78,92	73,32	78,82	85,58	72,00	74,84	0,39	0,73	0,65	0,59	73,36	69,83	71,27	74,98
Prof	77,13	74,19	74,23	81,03	88,65	71,14	73,10	0,39	0,73	0,64	0,58	68,58	69,83	72,24	73,74
GOR	55,36	56,20	47,53	58,82	55,91	43,02	62,55	0,08	0,33	0,28	0,33	51,45	51,87	50,89	51,60
SOPM	66,84	69,78	57,24	69,56	69,29	58,81	68,98	0,20	0,53	0,46	0,46	65,31	62,70	62,39	64,35
SSPro	79,07	82,72	66,90	82,56	85,90	78,43	74,53	0,42	0,76	0,65	0,61	76,22	68,11	72,27	74,39
JPred	73,37	65,21	56,18	89,07	89,41	77,01	65,40	0,34	0,67	0,58	0,54	64,65	61,00	69,16	68,03
SAM-T02	78,17	83,99	75,37	74,96	82,82	72,90	77,23	0,40	0,75	0,66	0,59	75,47	71,11	69,35	74,01
Predator	80,04	78,18	71,88	85,87	84,37	83,40	75,83	0,42	0,72	0,71	0,65	73,49	68,47	72,55	74,88
MC-NB-5	80,61	88,13	78,93	76,20	82,06	73,62	83,15	0,45	0,78	0,70	0,63	85,42	77,35	66,18	73,25
MC-NB-6	80,53	87,58	78,49	76,49	82,48	74,12	82,34	0,45	0,77	0,70	0,63	84,64	77,15	66,61	73,39
MC-IENB-5	80,68	87,97	78,67	76,53	82,19	74,42	82,79	0,45	0,78	0,70	0,63	84,07	77,20	66,45	73,41
MC-IENB-6	80,52	87,62	78,44	76,49	82,46	74,15	82,34	0,45	0,77	0,70	0,63	84,65	77,14	66,61	73,39
MC-Pazzani-EDA	80,99	86,24	80,29	77,37	84,97	72,48	82,27	0,45	0,78	0,70	0,64	82,64	78,57	66,74	73,37

Tabla 7.5: Estadísticas para el conjunto de datos CB513

Servidores	Conjunto de datos RS126														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	81,01	84,35	72,62	83,01	84,58	80,44	78,88	0,45	0,77	0,7	0,65	81,47	69,42	74,34	76,24
PHD Expert	76,92	78,33	73,65	77,62	84,77	69,46	75,93	0,38	0,73	0,63	0,57	71,42	68,86	70,32	72,57
Prof	76,95	74,18	75,20	79,82	88,07	69,30	74,79	0,38	0,73	0,63	0,58	69,44	68,66	70,23	71,70
GOR	55,39	56,21	48,46	58,41	53,75	44,17	63,68	0,08	0,33	0,29	0,32	51,02	51,47	51,39	50,89
SOPM	66,03	67,98	57,30	69,17	66,37	57,80	70,06	0,19	0,52	0,45	0,45	64,14	60,96	62,65	62,43
SSPro	77,01	80,84	64,38	80,85	82,81	74,62	74,32	0,38	0,74	0,61	0,58	75,63	64,87	68,83	70,24
JPred	73,82	65,53	56,46	88,78	89,32	77,56	66,68	0,34	0,68	0,58	0,54	65,66	59,88	67,33	66,55
SAM-T02	78,81	84,93	77,11	75,36	82,94	72,56	79,28	0,42	0,76	0,67	0,60	77,83	72,62	68,58	73,30
Predator	80,06	79,71	69,38	85,85	83,67	82,62	76,89	0,42	0,73	0,69	0,64	71,48	65,29	69,86	71,42
MC-NB-5	80,39	87,05	77,09	77,77	81,83	74,73	82,31	0,44	0,77	0,69	0,63	84,84	77,30	64,41	70,53
MC-NB-6	80,21	86,46	76,54	77,98	82,23	75,12	81,43	0,44	0,77	0,69	0,62	84,23	77,37	64,66	70,50
MC-IENB-5	80,55	86,88	76,89	78,31	81,91	76,05	81,91	0,45	0,77	0,69	0,63	84,61	77,29	64,98	70,96
MC-IENB-6	80,22	86,55	76,47	77,98	82,23	75,16	81,43	0,44	0,77	0,69	0,62	84,22	77,38	64,43	70,37
MC-Pazzani-EDA	81,65	85,36	82,43	78,85	85,67	71,96	83,85	0,47	0,79	0,71	0,65	80,95	83,36	64,64	70,67

Tabla 7.6: Estadísticas para el conjunto de datos CB513

Servidores	conjunto de datos EVA1														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	75,48	80,83	66,10	75,46	79,72	72,85	73,03	0,35	0,69	0,62	0,55	73,52	69,94	66,88	70,17
PHD Expert	75,26	79,37	73,36	72,64	80,58	68,53	74,23	0,35	0,69	0,63	0,54	67,78	74,05	66,84	69,81
Prof	72,11	71,88	68,67	74,00	81,42	63,82	69,56	0,30	0,64	0,57	0,50	61,84	69,98	65,84	66,59
GOR	54,44	62,23	43,03	53,30	55,07	41,01	61,69	0,08	0,32	0,26	0,30	50,63	56,06	48,92	49,49
SOPM	63,11	70,41	52,29	62,10	63,58	54,55	66,94	0,16	0,46	0,42	0,40	57,84	64,63	57,91	57,89
SSPro	74,48	81,14	63,51	74,10	77,49	72,54	72,64	0,33	0,67	0,60	0,53	67,90	71,07	65,80	67,03
JPred	68,82	59,23	54,07	84,37	83,32	67,67	62,57	0,25	0,58	0,51	0,47	54,99	64,78	65,11	61,79
SAM-T02	74,65	82,03	73,33	68,95	78,00	68,65	74,77	0,34	0,68	0,63	0,52	70,12	72,98	63,55	67,70
Predator	61,72	62,72	37,08	72,91	64,24	56,28	61,41	0,13	0,43	0,35	0,39	54,12	52,55	58,24	55,09
MC-NB-5	76,99	82,33	73,60	74,05	81,62	69,66	76,52	0,38	0,72	0,64	0,56	75,28	71,93	62,58	67,84
MC-NB-6	76,96	81,65	73,11	74,69	82,17	70,48	75,58	0,38	0,71	0,65	0,57	74,34	73,06	63,56	68,22
MC-IENB-5	76,95	82,15	73,31	74,23	81,71	70,21	76,10	0,38	0,71	0,65	0,56	73,99	71,30	62,32	67,61
MC-IENB-6	76,99	81,68	73,18	74,70	82,16	70,65	75,58	0,38	0,71	0,65	0,57	74,32	73,17	63,57	68,26
MC-Pazzani-EDA	77,51	80,94	75,67	75,27	83,85	70,37	75,48	0,39	0,72	0,66	0,57	74,69	77,57	64,80	69,50

Tabla 7.7: Estadísticas para el conjunto de datos EVA1

Servidores	Conjunto de datos EVA2														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	75,49	80,91	65,80	75,55	79,61	72,88	73,12	0,35	0,69	0,62	0,55	74,16	68,96	67,26	70,40
PHD Expert	75,27	79,42	73,29	72,65	80,62	68,47	74,25	0,35	0,69	0,63	0,54	68,31	73,15	66,75	69,91
Prof	72,11	72,00	68,45	73,99	81,47	63,72	69,56	0,30	0,64	0,57	0,50	62,41	69,08	65,66	66,65
GOR	54,40	61,97	43,15	53,36	55,14	40,68	61,77	0,08	0,32	0,26	0,30	50,60	55,48	48,97	49,53
SOPM	63,03	70,28	52,12	62,09	63,58	54,15	66,94	0,16	0,46	0,41	0,40	57,47	63,77	57,74	57,55
SSPro	74,45	81,21	63,33	74,05	77,50	72,38	72,62	0,33	0,67	0,60	0,53	68,25	70,18	65,52	66,98
JPred	68,86	59,35	54,02	84,34	83,40	67,54	62,63	0,25	0,58	0,51	0,47	55,25	64,08	65,41	61,86
SAM-T02	74,77	82,34	73,29	68,98	77,99	68,76	74,98	0,34	0,68	0,63	0,53	70,34	72,12	63,38	67,61
Predator	61,71	62,56	37,04	73,01	64,29	55,98	61,46	0,13	0,43	0,35	0,39	54,25	51,90	58,57	55,29
MC-NB-5	77,04	82,35	73,61	74,13	81,81	69,51	76,55	0,38	0,72	0,64	0,57	76,04	71,34	62,19	67,57
MC-NB-6	76,99	81,68	73,23	74,67	82,32	70,24	75,64	0,38	0,71	0,65	0,57	75,15	73,10	63,03	67,84
MC-IENB-5	77,01	82,19	73,35	74,30	81,88	70,08	76,15	0,38	0,72	0,65	0,56	74,79	70,84	62,08	67,48
MC-IENB-6	77,02	81,74	73,26	74,68	82,30	70,41	75,64	0,38	0,71	0,65	0,57	75,12	73,18	63,03	67,88
MC-Pazzani-EDA	77,60	81,62	75,38	75,09	83,13	70,92	76,03	0,39	0,72	0,66	0,57	75,07	78,28	64,45	69,61

Tabla 7.8: Estadísticas para el conjunto de datos EVA2

Servidores	Conjunto de datos EVA3														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	77,65	83,58	67,72	76,70	81,45	75,13	75,09	0,38	0,71	0,65	0,58	79,27	69,03	69,19	72,65
PHD Expert	76,50	80,82	71,86	74,60	82,23	70,99	73,83	0,36	0,70	0,64	0,56	74,88	73,35	67,79	72,28
Prof	73,78	72,87	69,72	76,54	83,61	65,37	70,23	0,32	0,66	0,59	0,53	64,88	70,23	65,90	66,74
GOR	54,76	57,06	47,45	56,02	58,95	38,96	60,34	0,08	0,32	0,28	0,30	50,63	57,38	48,90	50,31
SOPM	66,61	71,56	58,42	65,79	70,85	55,55	68,08	0,20	0,53	0,46	0,44	65,00	67,17	59,30	62,64
SSPro	76,38	82,06	64,84	76,41	80,92	73,47	73,36	0,35	0,69	0,62	0,56	74,57	69,71	66,63	69,74
JPred	71,16	65,33	53,89	84,78	83,93	71,79	63,89	0,28	0,61	0,55	0,50	62,81	64,18	66,35	65,15
SAM-T02	77,21	85,07	74,89	70,88	80,22	71,56	76,93	0,38	0,71	0,67	0,57	76,96	73,83	65,27	71,69
Predator	63,88	63,27	42,73	74,34	71,67	56,40	60,75	0,16	0,48	0,39	0,40	60,54	52,75	57,89	57,19
APSSP	92,49	94,78	93,26	90,43	88,66	96,14	93,58	0,72	0,88	0,93	0,86	79,79	84,15	79,25	74,74
MC-NB-5	78,86	84,72	75,81	75,00	82,81	71,41	78,63	0,41	0,74	0,67	0,60	81,36	72,37	65,30	71,77
MC-NB-6	78,51	83,65	74,97	75,35	83,33	71,39	77,31	0,40	0,73	0,67	0,50	80,21	73,27	65,47	71,40
MC-IENB-5	78,88	84,59	75,34	75,34	82,92	72,31	78,15	0,41	0,73	0,68	0,60	80,06	72,15	65,60	71,90
MC-IENB-6	78,56	83,74	75,03	75,36	83,37	71,54	77,31	0,40	0,73	0,67	0,59	80,23	73,34	65,47	71,44
MC-Pazzani-EDA	79,55	85,06	76,38	76,01	83,39	73,54	78,74	0,42	0,74	0,69	0,61	80,33	73,19	66,28	72,27

Tabla 7.9: Estadísticas para el conjunto de datos EVA3

Servidores	Conjunto de datos EVA4														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	77,92	83,74	68,71	76,98	81,82	76,24	75,00	0,39	0,72	0,66	0,59	79,29	70,42	70,02	73,56
PHD Expert	76,53	80,85	72,09	74,64	82,54	71,86	73,34	0,37	0,70	0,65	0,56	74,65	73,79	68,19	72,57
Prof	73,93	72,98	70,23	76,65	84,01	66,21	70,13	0,32	0,66	0,60	0,53	65,39	70,91	66,79	67,70
GOR	54,70	57,86	46,82	55,61	58,19	40,20	60,14	0,08	0,32	0,28	0,30	51,22	57,26	48,56	50,36
SOPM	66,50	71,82	57,95	65,70	70,11	57,14	67,72	0,20	0,52	0,47	0,44	64,81	67,44	59,38	62,61
SSPro	76,67	82,26	65,59	76,87	81,54	74,49	73,18	0,36	0,70	0,63	0,57	74,75	70,47	67,65	70,48
JPred	71,29	65,61	54,60	84,88	84,14	72,97	63,71	0,29	0,61	0,55	0,51	62,94	64,65	67,05	65,63
SAM-T02	77,40	85,03	75,73	71,04	80,75	72,15	76,73	0,39	0,72	0,67	0,57	76,71	74,77	65,94	72,24
Predator	63,61	63,59	42,36	74,06	70,87	57,49	60,42	0,16	0,48	0,39	0,40	60,40	52,97	57,72	57,31
MC-NB-5	79,04	85,19	76,48	74,83	82,88	72,29	78,75	0,41	0,74	0,68	0,60	82,19	72,91	65,49	72,08
MC-NB-6	78,67	84,06	75,75	75,12	83,32	72,36	77,39	0,41	0,73	0,68	0,59	81,07	73,72	65,74	71,99
MC-IENB-5	79,08	85,06	76,09	75,17	82,99	73,15	78,32	0,41	0,74	0,68	0,60	81,04	72,73	65,84	72,26
MC-IENB-6	78,71	84,14	75,78	75,12	83,36	72,48	77,39	0,41	0,74	0,68	0,59	81,10	73,80	65,74	72,02
MC-Pazzani-EDA	79,71	85,12	78,53	75,44	84,07	71,86	79,46	0,43	0,75	0,69	0,61	81,39	73,69	65,54	71,88

Tabla 7.10: Estadísticas para el conjunto de datos EVA4

Servidores	Conjunto de datos EVA5														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	77,95	83,03	67,68	77,86	82,95	74,80	74,57	0,38	0,72	0,65	0,59	79,33	70,64	70,37	73,32
PHD Expert	76,60	80,42	70,22	75,89	83,68	70,36	73,05	0,36	0,71	0,63	0,56	74,91	73,63	68,67	72,91
Prof	73,61	72,07	68,94	77,31	85,12	64,24	69,40	0,32	0,66	0,58	0,53	66,00	70,98	67,32	68,21
GOR	54,40	56,87	47,04	55,49	58,72	38,10	60,36	0,08	0,31	0,27	0,31	51,08	57,11	49,19	49,86
SOPM	65,64	70,77	56,18	65,12	70,15	53,89	67,06	0,18	0,51	0,44	0,43	64,60	66,16	58,14	61,32
SSPro	76,74	81,64	64,43	77,81	82,72	73,47	72,64	0,36	0,71	0,62	0,57	74,95	70,10	68,76	70,46
JPred	70,79	63,69	53,53	85,98	85,71	69,67	63,12	0,28	0,61	0,53	0,51	62,32	65,15	66,97	65,17
SAM-T02	77,40	84,92	74,59	71,40	81,49	71,05	76,35	0,38	0,72	0,66	0,57	75,80	74,96	66,54	72,00
Predator	63,00	62,76	41,39	73,50	70,77	54,25	60,10	0,15	0,47	0,37	0,39	60,82	53,03	57,70	56,89
MC-NB-5	79,09	86,20	75,54	74,42	82,14	71,16	79,88	0,41	0,74	0,67	0,60	82,29	72,08	65,16	71,42
MC-NB-6	78,77	85,32	74,60	74,69	82,59	71,05	78,71	0,40	0,74	0,66	0,60	81,22	71,87	65,11	71,08
MC-IENB-5	79,14	86,08	75,22	74,73	82,30	71,92	79,48	0,41	0,74	0,67	0,60	81,59	71,83	65,46	71,56
MC-IENB-6	78,79	85,38	74,61	74,69	82,60	71,16	78,71	0,40	0,74	0,67	0,60	81,28	71,93	65,13	71,11
MC-Pazzani-EDA	79,67	85,65	76,43	75,60	83,87	72,12	79,16	0,42	0,75	0,68	0,61	81,67	73,30	65,88	71,70

Tabla 7.11: Estadísticas para el conjunto de datos EVA5

Servidores	Conjunto de datos EVA6														
	Q_3	$Q_h^{\%obs}$	$Q_e^{\%obs}$	$Q_l^{\%obs}$	$Q_h^{\%prd}$	$Q_e^{\%prd}$	$Q_l^{\%prd}$	$info$	C_h	C_e	C_l	SOV_h	SOV_e	SOV_l	SOV
PSIPRED	77,56	83,50	66,79	77,20	81,88	75,67	74,31	0,38	0,71	0,64	0,58	78,75	71,15	69,42	72,93
PHD Expert	76,28	80,58	70,16	75,21	82,81	70,38	73,22	0,36	0,70	0,63	0,56	74,07	73,32	67,71	72,03
Prof	73,40	72,51	68,15	76,82	84,43	64,64	69,49	0,31	0,66	0,58	0,53	65,18	69,82	66,88	67,58
GOR	54,70	59,37	45,43	54,8	57,75	39,35	61,07	0,08	0,31	0,26	0,31	52,12	56,46	48,14	50,04
SOPM	65,05	71,23	54,17	64,53	68,29	54,73	66,93	0,18	0,50	0,43	0,43	63,60	65,39	57,85	60,69
SSPro	76,30	82,16	63,85	76,83	81,31	73,69	72,79	0,35	0,7	0,61	0,57	74,36	70,39	67,63	69,96
JPred	70,51	63,45	53,55	85,62	85,30	69,96	62,95	0,28	0,61	0,53	0,50	61,8	64,96	66,24	64,53
SAM-T02	76,76	84,62	73,65	70,81	80,57	70,69	76,00	0,37	0,71	0,65	0,56	75,88	74,35	65,87	71,40
Predator	62,73	63,30	39,72	73,49	69,23	55,26	60,25	0,15	0,46	0,36	0,39	60,82	52,40	58,40	57,24
MC-IENB-5	77,50	85,11	75,45	72,09	79,88	68,90	79,41	0,39	0,72	0,66	0,57	80,37	73,63	64,54	70,70
MC-IENB-6	77,66	84,63	75,21	72,75	80,99	68,98	78,69	0,39	0,72	0,65	0,58	79,98	73,82	64,36	70,74
MC-NB-5	77,47	85,23	75,74	71,85	79,78	68,15	79,79	0,39	0,72	0,65	0,57	81,13	74,10	64,31	70,66
MC-NB-6	77,65	84,60	75,17	72,75	80,99	68,91	78,69	0,39	0,72	0,65	0,58	79,97	73,77	64,36	70,71
MC-Pazzani-EDA	75,86	84,36	76,47	69,22	76,48	66,12	80,00	0,36	0,69	0,65	0,55	79,81	76,38	63,62	69,53

Tabla 7.12: Estadísticas para el conjunto de datos EVA6

Parte V

CONCLUSIONES Y LÍNEAS FUTURAS

Capítulo 8

Conclusiones y líneas futuras

La realización de la presente tesis ha alcanzado de una forma muy satisfactoria los objetivos fijados inicialmente. En este capítulo se hace una revisión de todas las aportaciones realizadas, que ya han sido presentadas en diversas secciones de la tesis. Además, se señalan las principales líneas de trabajo futuro.

El índice del capítulo es el siguiente:

- En la sección 8.1 se detallan las aportaciones realizadas en la presente tesis.
- En la sección 8.2 se detallan las líneas de trabajo futuro, que permiten extender las metas logradas.

8.1. Aportaciones

En la presente tesis se ha realizado una importante apuesta, que ha resultado muy acertada, por el uso de los algoritmos heurísticos de optimización EDA en el campo de la clasificación supervisada. A continuación se detallan las aportaciones realizadas.

8.1.1. Realización y paralelización de un nuevo algoritmo de clasificación semi naïve-Bayes

Se ha propuesto un nuevo algoritmo semi naïve-Bayes denominado *Interval Estimation naïve-Bayes* (IENB) [RLP+03b, RLP+03a, RLM+03]. Este algoritmo se puede encuadrar dentro de las variantes semi naïve-Bayes que corrigen las probabilidades producidas por dicho clasificador.

En IENB, en lugar de estimar probabilidades puntuales de los datos, tal y como se hace en naïve-Bayes, se realizan estimaciones por intervalo. A continuación, a través de algoritmos de optimización heurísticos, se consigue la combinación de valores, cada uno de ellos dentro de los intervalos correspondientes, que maximiza el tanto por ciento de bien clasificados.

Los resultados obtenidos son datos excelentes, tanto si se comparan con naïve-Bayes, como si se comparan con otros enfoques semi naïve-Bayes similares como *Iterative Bayes*. Teniendo en cuenta los 21 conjuntos de datos utilizados, se ha conseguido mejorar al naïve

Bayes en un 4,61 % de media.

En la fase experimental se ha demostrado que el algoritmo IENB puede llegar a ser muy costoso desde el punto de vista computacional, debido a la búsqueda heurística que realiza.

Para paliar ese coste computacional se realizó una paralelización del algoritmo basada en islas [RPP+03], en donde cada isla contiene una población diferente y, cada cierto tiempo y con un esquema de migración predeterminado, las islas se intercambian los mejores individuos entre sí. Este modelo de paralelización por islas ya fue utilizado con éxito en los algoritmos genéticos paralelos.

Los resultados obtenidos en la paralelización han sido sorprendentemente excelentes, ya que comparativamente se obtienen mejores resultados que con la versión secuencial y, además, se obtiene un *speedup* superlineal.

8.1.2. Aprendizaje de clasificadores en el espacio de estructuras

Las variantes semi naïve-Bayes están normalmente basadas en la realización de búsquedas de determinadas estructuras o determinados valores. Estas búsquedas se suelen realizar a través de algoritmos de búsqueda voraz (*greedy*).

En esta tesis se ha propuesto el uso de los algoritmos de optimización EDA, como herramienta para la realización de las búsquedas de estructuras semi naïve-Bayes:

- Se ha propuesto una extensión del algoritmo APNBC, denominada APNBC-EDA [RLP+03d], que realiza la búsqueda de los valores de ajuste de las probabilidades de las clases por medio de los algoritmos heurísticos EDAs.

  Los resultados obtenidos son muy satisfactorios, ya que se ha conseguido mejorar en gran medida los resultados de APNBC. Sobre los 21 conjuntos de datos utilizados, APNBC-EDA consigue mejorar un 2,46 %.

- También se ha presentado una extensión a los algoritmos heurísticos FSSJ y BSEJ de Pazzani, que permite realizar la búsqueda del mejor producto cartesiano entre variables con los algoritmos heurísticos de optimización EDA [RLP+03d].

  Los resultados obtenidos por Pazzani-EDA son muy buenos. Por un lado se ha conseguido mejorar los resultados de los algoritmos FSSJ y BSEJ de Pazzani. Por otra parte, las estructuras encontradas por Pazzani-EDA son más sencillas que las encontradas por los algoritmos voraces FSSJ y BSEJ. Respecto a naïve Bayes, el algoritmo FSSJ consigue una mejora media de 1,25 %, el algoritmo BSEJ de 3,61 % y el algoritmo EDA de 5,51 %.

Por otra parte, como se vio en el capítulo 3, existen varias estructuras de red Bayesiana que pueden ser utilizadas para realizar clasificación supervisada: TAN, BAN y MB. En todas estas estructuras se ha tenido en cuenta el hecho de que existe una variable especial, la variable a clasificar.

Frente a los métodos de búsqueda propuestos actualmente en la literatura, que se basan en la cantidad de información mutua entre las variables predictoras, en esta tesis se ha

propuesto el desarrollo de algoritmos voraces y heurísticos guiados por el tanto por ciento de bien clasificados para la búsqueda de estructuras TAN, BAN y MB.

Se han presentado dos algoritmos para la búsqueda de estructuras tipo BAN. El primer algoritmo, denominado BAN-voraz, está basado en una búsqueda voraz, mientras que el segundo, denominado BAN-EDA, está basado en una búsqueda heurística con EDAs. Ambos algoritmos han sido comparados con las búsquedas realizadas por Friedman y col. [FG96], que están basadas en la cantidad de información mutua entre las variables predictoras.

Los resultados obtenidos muestran que ambos algoritmos desarrollados son muy superiores al enfoque de Friedman y col. Esta vez, la complejidad de las estructuras encontradas y los valores del tanto por ciento de bien clasificados obtenidos, hacen que la recomendación se decante por BAN-voraz. BAN-voraz obtiene una mejora media de 5,80 % respecto a naïve Bayes.

En relación a los clasificadores TAN, se han presentado dos algoritmos para la búsqueda de estas estructuras. El primer algoritmo, denominado TAN-voraz, está basado en una búsqueda voraz, mientras que el segundo, denominado TAN-EDA, está basado en una búsqueda heurística con EDAs. Ambos algoritmos han sido comparados con las búsquedas realizadas por Friedman.

Los resultados obtenidos muestran que ambos algoritmos desarrollados son muy superiores al enfoque de Friedman. TAN-voraz presenta una media de 5,74 % respecto a naïve Bayes, mientras que TAN-EDA sólo eleva esta mejora a 6,09 %.

Por último, se ha desarrollado el algoritmo MB-EDA, para la búsqueda heurística de clasificadores tipo MB. Los resultados obtenidos también han sido muy buenos, ya que se consigue mejorar sustancialmente los resultados del clasificador naïve-Bayes a la vez que se mantiene una buena simplicidad en las estructuras encontradas. La mejora media obtenida por el algoritmo MB-EDA es de 5,80 %.

8.1.3. Modelización matemática de las proteínas y del problema PSSP

Por primera vez en este campo, se ha realizado una modelización matemática de las proteínas y del problema de la predicción de la estructura secundaria.

Esta modelización matemática puede resultar de gran utilidad, ya que permite acercar el campo de la proteómica y de la predicción de la estructura secundaria a los científicos de la comunidad del aprendizaje automático.

8.1.4. Desarrollo de un clasificador para la predicción de la estructura secundaria de las proteínas con redes Bayesianas

El problema de la predicción de la estructura secundaria de las proteínas (PSSP) ya había sido abordado desde diversos enfoques metodológicos, muchos de ellos pertenecientes al mundo del aprendizaje automático. Sin embargo, nunca se había intentado solucionar este problema con el uso de redes Bayesianas.

La principal dificultad en este desarrollo [RLPP02, RPH+03] ha sido la de introducir la información evolutiva de las proteínas en las redes Bayesianas. Esta información es de vital importancia en todos los métodos de predicción actuales.

Se han creado dos posibles adaptaciones de naïve-Bayes e IENB para la incorporación de la información evolutiva. Una de las adaptaciones no ha sido exitosa, mientras que la otra ha presentado buenos resultados.

En cuanto a los resultados obtenidos, los mejores métodos de predicción actuales llegan aproximadamente a un 80 % de bien clasificados, mientras que con las redes Bayesianas se ha obtenido un 71,21 %. Sin embargo, todavía quedan por desarrollar varias extensiones que pueden mejorar estos resultados. Pensamos que sería factible alcanzar un valor de 76-78 % de bien clasificados con las redes Bayesianas.

8.1.5. Multiclasificador basado en redes Bayesianas para el problema PSSP

Se ha propuesto un multiclasificador –un clasificador de clasificadores– basado en redes Bayesianas, para la predicción de la estructura secundaria de las proteínas [RLP+03c]. Este multiclasificador está basado en el conocido paradigma *stacked generalization* [Wol92].

Para poder llegar a este objetivo se ha desarrollado una aplicación Web en JSP y clases de Java, que ha cumplido con dos importantes objetivos:

- Poder comparar los algoritmos de predicción actuales, extrayendo estadísticas sobre la exactitud de sus predicciones.
- Obtener el conjunto de datos necesario para la realización del multiclasificador de predicción de estructura secundaria.

Una vez recopilado el conjunto de datos para el multiclasificador, se experimentó con varios de los algoritmos de clasificación desarrollados a lo largo de la tesis.

Los multiclasificadores desarrollados han demostrado un excelente resultado, ya que se ha conseguido superar hasta en un 1,21 % de media al mejor de los clasificadores actuales.

Consideramos que los resultados que hemos obtenido en esta propuesta proporcionan un gran avance dentro de este campo.

8.2. Líneas de trabajo futuro

La gran cantidad de desarrollos realizados en la presente tesis han permitido dejar muchos campos abiertos para futuras investigaciones. A continuación se detallan los principales.

8.2.1. Líneas abiertas en IENB

IENB es un algoritmo de nueva creación, por lo que quedan muchos campos abiertos para futuros desarrollos:

- Adaptación del clasificador IENB para su uso con variables continuas.
- Realización de implementaciones de IENB que cambien la función objetivo del tanto por ciento de bien clasificados. Por ejemplo, se podría maximizar el área bajo la curva ROC del clasificador.

- Otra posible idea, sería realizar una selección de variables previa a la ejecución de IENB.

Por otra parte, la paralelización de IENB realmente ha supuesto paralelizar los algoritmos heurísticos EDAs. Dentro de este campo queda mucho trabajo por desarrollar. Algunas ideas son:

- Realizar un estudio detallado sobre los ratios de migración entre islas.
- Probar diferentes topologías de migración entre islas.
- Estudio de algoritmos heurísticos paralelos híbridos AGs-EDAs. En este sentido sería posible tener unas islas que ejecuten una optimización con AGs y otras que lo hagan con EDAs.
- Probar otro tipo de paralelizaciones de EDAs, similares a las de grano fino que se realizan con los AGs.

8.2.2. Líneas abiertas en el aprendizaje de clasificadores supervisados basados en redes Bayesianas

En este campo también han quedado diversas líneas abiertas. Las principales son:

- Usar la idea de estimaciones por intervalo expuesta en el IENB, con algoritmos tales como Pazzani, BAN o TAN.
- Realizar búsquedas de estructuras de tipo BAN en donde se penalice la complejidad de las estructuras encontradas.

8.2.3. Líneas abiertas en la modelización matemática de las proteínas y del PSSP

Es necesario seguir desarrollando la modelización matemática realizada. El siguiente paso es, sin duda, modelizar la información evolutiva de las proteínas. Este paso intermedio es imprescindible a la hora de poner definir de forma totalmente simbólica el problema de la predicción de la estructura secundaria de las proteínas.

8.2.4. Líneas abiertas en el desarrollo de un clasificador para PSSP con redes Bayesianas

Prácticamente todos los predictores de estructura secundaria actuales realizan, en su primera capa de predicción, una multiclasificación. El clasificador básico es entrenado con diferentes tamaños de ventana y diferentes conjuntos de datos, realizándose una multiclasificación de todos estos resultados. Esta multiclasificación supone, normalmente, un 4 % de mejora aproximada. Es posible usar esta misma idea con el clasificador basado en redes Bayesianas.

Por otra parte, la segunda capa de predicción que proponemos sólo mejora un 0,88 %, muy por debajo de las mejoras obtenidas por otros métodos. De esta forma, nos planteamos sustituir la segunda capa de predicción por una basada en redes de neuronas.

Creemos que con estos cambios la exactitud de este método puede llegar a ser similar a la de los métodos actuales.

8.2.5. Líneas abiertas en el multiclasificador basado en redes Bayesianas para PSSP

Todavía queda mucho trabajo a desarrollar basado en el multiclasificador. Algunas ideas son:

- Probar más algoritmos de clasificación como multiclasificadores. En este punto se está ya avanzando en el uso de *rough sets* como una posible alternativa.
- Poner el multiclasificador a disposición pública. En este sentido, nos queremos poner en contacto con los responsables del proyecto EVA, para una posible integración del multiclasificador en su sistema.
- Crear un punto central en Internet a través del cual, y con una sola petición, se puedan conseguir los resultados de todos los servidores de predicción de estructura secundaria de Internet.

Bibliografía

[AG96] S. F. Altschul and W. Gish. Local alignment statistics. *Meth. Enzymol*, 266:460–480, 1996.

[AGM+90] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[AMS+97] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, 25(17):3389–3402, 1997.

[And73] M.R. Anderberg. *Cluster Analysis for Applications.* Academic Press, New York, 1973.

[Aso97] M. Asogawa. Beta-sheet prediction using inter-strand residue pairs and refinement with Hopfield neural network. In T. Gaasterland, P. Karp, K. Karplus, C. Ouzounis, and C. Sander et al., editors, *In Fifth International Conference on Intelligent Systems for Molecular Biology*, pages 48–51. AAAI Press, Halkidiki, Greece., 1997.

[BBB+88] H. Bohr, J. Bohr, S. Brunak, R.M.J. Cotterill, and B. Lautrup et al. Protein secondary structure and homology by neural networks. *FEBS Letters*, 241:223–228, 1988.

[BBF+99a] P. Baldi, S. Brunak, P. Frasconi, G. Pollastri, and G. Soda. Bidirectional dynamics for protein secondary structure prediction. Stockholm, Sweden, 1999. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI99).

[BBF+99b] P. Baldi, S. Brunak, P. Frasconi, G. Pollastri, and G. Soda. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, 15:937–946, 1999.

[BdLBF60] E.R. Blout, C. de Lozé, S.M. Bloom, and G.D. Fasman. Dependence of the conformation of synthetic polypeptides on amino acid composition. *J. Am. Chem. Soc.*, 82:3787–3789, 1960.

[Bel95] T.C. Belding. The distributed genetic algorithm revisited. In *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 114–121, 1995.

[Ben02] E. Bengoetxea. *Inexact Graph Matching using Estimation of Distribution Algorithms.* PhD thesis, Escuela Nacional de Telecomunicaciones, París, Octubre 2002.

[BFOS84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.

[BGL+88] V. Biou, J.F. Gibrat, J.M. Levin, B. Robson, and J. Garnier. Secondary structure prediction: combination of three different methods. *Protein Engineering*, 2:185–191, 1988.

[Bis95] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[Blo62] E.R. Blout. The dependence of the conformation of polypetides and proteins upon amino acid composition. pages 275–279. Univ. of Wisconsin Press, Madison, 1962.

[Bre94] L. Breiman. Baggings predictors. Technical report, Technical Report Statistic Department, University of California at Berkeley, 1994.

[BS88] B. Buchanan and R. Smith. Fundamentals of expert systems. *Annual Review of Computer Science*, 3:23–58, 1988.

[BT88] G. Barton and W.R. Taylor. Prediction of protein secondary structure and active sites using the alignment of homologous sequences. *Journal of Molecular Biology*, 195:957–961, 1988.

[Bun91] W. Buntine. Classifiers: a theoretical and empirical study. In *Proceedings of the IJCAI*, pages 638–655, 1991.

[BvH87] O. G. Berg and P.H. von Hippel. Selection of dna binding sites by regulatory proteins. *Journal of Molecular Biology*, 193:723–750, 1987.

[BW03] P. E. Bourne and H. Weissig, editors. *Structural Bioinformatics.* John Wiley & Sons, Inc., 2003.

[BWF+00] H.M. Berman, J. Westbrook, Z. Feng, G. Gilliand, T.N. Bhat, H. Weissig, I.N. Shindyalov, and P.E. Bourne. The protein data bank. *Nucleid Acid Research*, 28:235–242, 2000.

[CB99] J.A. Cuff and G.J. Barton. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins*, 34:508–519, 1999.

[CCS+98] J.A. Cuff, M.E. Clamp, A.S. Siddiqui, M. Finlay, and G.J. Barton. JPRED: A consensus secondary structure prediction server. *Bioinformatics*, 14:892–893, 1998.

[Ces90] B. Cestnik. Estimating probabilities: a crucial task in machine learning. In *Proceedings of the European Conference on Artificial Intelligence*, pages 147–149, 1990.

[CF74] P.Y. Chou and U.D. Fasman. Prediction of protein conformation. *Biochemistry*, 13:222–245, 1974.

[CG83] P. Cohen and M. Grinberg. A framework for heuristic reasoning about uncertainty. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence (IJCAI–83)*, pages 355–357, Karlsruhe, Germany, 1983.

[CG99] J. Cheng and R. Greiner. Comparing Bayesian network classifiers. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 101–107, 1999.

[CGH97] E. Castillo, J.M. Gutierrez, and A.S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer, 1997.

[CGK+02] J. Cheng, R. Greiner, J. Kelly, D.A. Bell, and W. Liu. Learning Bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137:43–90, 2002.

[CK89] F.E. Cohen and I.D. Kuntz. Tertiary structure prediction. In G.D. Fasman, editor, *In Prediction of Protein Structure and the Principles of Protein Conformation*, pages 647–706. Plenum Press, New York, London, 1989.

[CK95] J.-M. Chandonia and M. Karplus. Neural networks for secondary structure and structural class predictions. *Prot. Sci.*, 4:275–285, 1995.

[CKB87] B. Cestnik, I. Kononenko, and I. Bratko. ASSISTANT-86: a knowledge elicitation tool for sophisticated users. In I. Bratko and N. Lavrac, editors, *Progress in Machine Learning*. Sigma Press, 1987.

[CL68] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

[CN89] P. Clark and T.Ñiblett. The CN2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

[Coh95] W.W. Cohen. Fast effective rule induction. In *Machine Learning: Proceedings of the Twentieth International Conference*, Lake Tahoe, California, 1995.

[Cow01] R. Cowell. On searching for optimal classifiers among Bayesian networks. In T. Jaakkola and T. Richardson, editors, *Proceedings of the 8th International Conference on Artificial Intelligence and Statistics*, pages 175–180, 2001.

[CP01] E. Cantú-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, 2001.

[Das91] B.V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Recognition Classification Techniques*. IEEE Computer Society Press, 1991.

[DE87] I. B. Dodd and J. B. Egan. Systematic method for the detection of potential lambda cro-like DNA-binding regions in proteins. *Journal of Molecular Biology*, 194:557–564, 1987.

[DH73] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, 1973.

[Die86] T.G. Dietterich. Learning at the knowledge level. *Machine Learning*, 1(3):287–315, 1986.

[DIV97] J. S. De Bonet, C. L. Isbell, and P. Viola. MIMIC: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, Vol. 9:424, 1997.

[DKS95] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proceedings of the 12th International Conference on Machine Learning*, pages 194–202, 1995.

[DOB94] D. Donnelly, J.P. Overington, and T.L. Blundell. The prediction and orientation of α-helices from sequence alignments: the combined use of environment-dependent substitution tables, Fourier transform methods and helix capping rules. *Protein Engineering*, 7:645–653, 1994.

[Doy79] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

[DP96] P. Domingos and M. Pazzani. Beyond independence: conditions for the optimality of the simple Bayesian classifier. In *Proceedings of the 13th International Conference on Machine Learning*, pages 105–112, 1996.

[DP97] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.

[DS79] M. Dayhoff and R. Schwartz. Matrices for detecting distant relationship. *Atlas of Protein Sequences*, pages 353–358, 1979.

[DS81] N.R. Draper and H. Smith. *Applied Regression Analysis.* John Wiley & Sons, 2nd edition, 1981.

[EL99] R. Etxeberria and P. Larrañaga. Global optimization with Bayesian networks. In *II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization*, pages 332–339, 1999.

[FA95] D. Frishman and P. Argos. Knowledge-based protein secondary structure assignment. *Proteins*, 23:566–579, 1995.

[FA96] D. Frishman and P. Argos. Incorporation of long-distance interactions into a secondary structure prediction algorithm. *Protein Engineering*, 9:133–142, 1996.

[FA97] D. Frishman and P. Argos. 75% accuracy in protein secondary structure prediction. *Proteins*, 27:329–335, 1997.

[FDH01] J.T.A.S. Ferreira, D.G.T. Denison, and D.J. Hand. Weighted naive Bayes modelling for data mining. Technical report, Department of Mathematics, Imperial College, 180 Queen's Gate, London SW7 2BZ, UK, 2001.

[FG96] N. Friedman and M. Goldszmidt. Building classifiers using Bayesian networks. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 1277–1284, 1996.

[FGG97] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[FI93] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Conference on Artificial Intelligence*, pages 1022–1027, 1993.

[FPSS96] U.M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: an overview. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*. American Association for Artificial Intelligence, Menlo Park, CA, 1996.

[Gam00] J. Gama. A cost-sensitive iterative Bayes. In *Proceedings of Workshop on Cost-Sensitive Learning at the 17th International Conference on Machine Learning*, Stanford University, June 2000.

[GC02] J. Gama and G. Castillo. Adaptive Bayes. In F. Garijo, J. Riquelme, and M. Toro, editors, *Advances in Artificial Intelligence - Iberamia 2002*, pages 765–774. LNAI 2527, Springer Verlag, 2002.

[GD94] C. Geourjon and G. Deleage. SOPM: a self optimised prediction method for protein secondary structure prediction. *Protein Engineering*, 7:157–164, 1994.

[GD95] C. Geourjon and G. Deléage. SOPMA: significant improvements in protein secondary structure prediction by consensus prediction from multiple alignments. *CABIOS*, 11:681–684, 1995.

[GG88] O. Gascuel and J.L. Golmard. A simple method for predicting the secondary structure of globular proteins: implications and accuracy. *CABIOS*, 4:357–365, 1988.

[GGR87] J.-F. Gibrat, J. Garnier, and B. Robson. Further developments of protein secondary structure prediction using information theory. New parameters and consideration of residue pairs. *Journal of Molecular Biology*, 198:425–443, 1987.

[GME87] M. Gribskov, A. D. McLachlan, and D. Eisenberg. *Proc. Natl. Acad. Sci. USA*, 84:4355–4358, 1987.

[Goo65] I.J. Good. *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. MIT Press, 1965.

[GOR78] J. Garnier, D.J. Osguthorpe, and B. Robson. Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins. *Journal of Molecular Biology*, 120:97–120, 1978.

[GZ02] R. Greiner and W. Zhou. Structural extension to logistic regression: discriminant parameter learning of belief net classifiers. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 167–173, 2002.

[Hay99] S. Haykin. *Neural Networks. A Comprehensive Foundation.* New Jersey, 1999. Prentice Hall.

[Hen88] M. Henrion. Propagating uncertainty in Bayesian networks by probabilistic logic sampling. *Uncertainty in Artificial Intelligence*, (2):149–163, 1988.

[HH92] S. Henikoff and J. G. Henikoff. Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Science USA*, 89(22):10915–10919, November 1992.

[HK89] H.L. Holley and M. Karplus. Protein secondary structure prediction with a neural network. *Proceedings of the National Academy of Science U.S.A.*, 86:152–156, 1989.

[Hol94] R.C. Holte. Very simple classification rules perform well on most commonly used databases. *Machine Learning*, 11:63–90, 1994.

[HS94] T.K. Ho and S.N. Srihati. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:66–75, 1994.

[HSSS92] U. Hobohm, M. Scharf, R. Schneider, and C. Sander. Selection of a representative set of structures from the brookhaven protein data bank. *Protein Science*, 1:409–417, 1992.

[HT01] D.J. Hand and R.J. Till. A simple generalization of the area under the ROC curve for multiple class classification problems. *Machine Learning*, 45:171–186, 2001.

[HY01] D.J. Hand and K. Yu. Idiot's Bayes - not so stupid after all? *International Statistical Review*, 69(3):385–398, 2001.

[ILES00] I. Inza, P. Larrañaga, R. Etxeberría, and B. Sierra. Feature subset selection by Bayesian network-based optimization. *Artificial Intelligence*, 123:157–184, 2000.

[Jen01] F.V. Jensen. *Bayesian Networks and Decision Graphs.* Springer-Verlag, 2001.

[JL95] G. John and P. Langley. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 338–345, 1995.

[JLTW93] D. Juretic, B. Lee, N. Trinajstic, and R.W. Williams. Conformational preference functions for predicting helices in membrane proteins. *Biopolymers*, 33:255–273, 1993.

[Jon99] D.T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, 292:195–202, 1999.

[KA90] S. Karlin and S. F. Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, 87:2264–2268, 1990.

[Kan88] M. Kanehisa. A multivariate analysis method for discriminating protein secondary structural segments. *Protein Engineering*, 2:87–92, 1988.

[KBH98] K. Karplus, C. Barrett, and R. Hughey. Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, 14:846–856, 1998.

[KBS97] J. Kohavi, B. Becker, and D. Sommerfield. Improving simple Bayes. Technical report, Data Mining and Visualization Group, Silicon Graphics, 1997.

[KCL90] D.G. Kneller, F.E. Cohen, and R. Langridge. Improvements in protein secondary ptructure prediction by an enhanced neural network. *Journal of Molecular Biology*, 214(1):171–182, 1990.

[KDS+60] J.C. Kendrew, R.E. Dickerson, B.E. Strandberg, R.J. Hart, and D.R. Davies et al. Structure of myoglobin: a three-dimensional Fourier synthesis at 2 Amstrongs resolution. *Nature*, 185:422–427, 1960.

[KJ97] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[KJL+94] R. Kohavi, G. John, R. Long, D. Manley, and K.Pfleger. Mlc++: A machine learning library in c++. *Tools with Artificial Intelligence. IEEE Computer Society Press*, pages 740–743, 1994.

[KMLS92] R.D. King, S. Muggleton, R.A. Lewis, and M.J.E. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proceedings of the National Academy of Science U.S.A.*, 89:11322–11326, 1992.

[Koh95] R. Kohavi. *Wrappers for Performance Enhancement and Oblivious Decision Graphs.* PhD thesis, Stanford University, Computer Science Department, 1995.

[Koh96] R. Kohavi. Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 202–207, 1996.

[Kon93] I. Kononenko. Inductive and Bayesian learning in medical diagnosis. *Applied Artificial Intelligence*, 7:317–337, 1993.

[Kon94] I. Kononenko. Estimating attributes: Analysis and extensions of Relief. In F. Bergadano and L.D. Raedt, editors, *Proceedings of the European Conference on Machine Learning*, 1994.

[KP99] E.J. Keogh and M. Pazzani. Learning augmented Bayesian classifiers: a comparison of distribution-based and non distribution-based approaches. In *Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics*, pages 225–230, 1999.

[KS83] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

[KS96] R.D. King and M.J.E. Sternberg. Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Protein Science*, 5:2298–2310, 1996.

[Lan93] P. Langley. Induction of recursive Bayesian classifiers. In *Proceedings of the 8th European Conference on Machine Learning*, pages 153–164, 1993.

[LBL+01] P. Larrañaga, E. Bengoetxea, J.A. Lozano, V. Robles, A. Mendiburu, and P. de Miguel. Searching for the best permutation with estimation of distribution algorithms. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 7–14, Seattle, USA, 2001. Workshop on Stochastic Search Algorithms.

[LDS91] A. Lupas, M. Van Dyke, and J. Stock. Predicting coiled coils from protein sequences. *Science*, 252:1162–1164, 1991.

[LELP00] P. Larrañaga, R. Etxeberria, J.A. Lozano, and J.M. Peña. Optimization in continuous domains by learning and simulation of Gaussian networks. In *Proceedings of the Workshop in Optimization by Building and Using Probabilistic Models. A Workshop within the 2000 Genetic and Evolutionary Computation Conference, GECCO 2000*, pages 201–204, 2000. Las Vegas, Nevada, USA.

[Lev94] D. Levine. *A Parallel Genetic Algorithm for the Set Partitioning Problem.* PhD thesis, Illinois Institute of Technology, Mathematics and Computer Science Division, Argonne National Laboratory, 1994.

[LIT92] P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 223–228, 1992.

[LL02] P. Larrañaga and J.A. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publisher, 2002.

[LLD01] J.N.K. Liu, B.N.L. Li, and T.S. Dillon. An improved naive Bayesian classifier technique coupled with a novel input solution method. *IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews*, 31(2):249–256, 2001.

[LP85] D. Lipman and W. Pearson. Rapid and sensitive protein similarity searches. *Science*, 227:1435–1441, 1985.

[LP88] D. Lipman and W. Pearson. Improved tools for biological sequence comparison. *Proceedings of the National Academy of Science USA*, 85:2444–2448, 1988.

[LS94] P. Langley and S. Sage. Induction of selective Bayesian classifiers. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, pages 399–406, 1994.

[LSL02] J.A. Lozano, R. Sagarna, and P. Larrañaga. Parallel estimation of distribution algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publisher, 2002.

[MA95] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/, 1995.

[MARW92] E.M. Mitchell, P.J. Artymiuk, D.W. Rice, and P. Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *Journal of Molecular Biology*, 212:151–166, 1992.

[Mat85] F.S. Matthews. The structure, function and evolution of cytochromes. *Prog. Biophys. Mol. Biol.*, 45:1–56, 1985.

[McL83] A. D. McLachlan. Rapid comparison of protein structures. *Journal of Molecular Biology*, 169:15–30, 1983.

[Mic87] D. Michie. Current developments in expert systems. In J.R. Quinlan, editor, *Applications of Expert Systems*, pages 137–156. Turing Institute Press, 1987.

[MKKC86] T.M. Mitchell, R.M. Keller, and S.T. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1):47–80, 1986.

[MKS92] S. Muggleton, R.D. King, and M.J.E. Sternberg. Protein secondary structure prediction using logic-based machine learning. *Protein Engineering*, 5:647–657, 1992.

[MP96] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions I. Binary parameters. In *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV*, pages 178–187, 1996.

[MPI94] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard*, May 1994.

[MPW97] S. Mani, M. Pazzani, and J. West. Knowledge discovery from a breast cancer database. In *Lecture Notes in Artificial Intelligence, 1211*, pages 130–133, 1997.

[MS93] R. Maclin and J.W. Shavlik. Using knowledge-based neural networks to improve algorithms: refining the Chou-Fasman algorithm for protein folding. *Machine Learning*, 11:195–215, 1993.

[MS94] S.K. Murthy and S. Salzberg. A system for the induction of oblique decision trees. *Journal of Artificial Intelligence*, 2:1–33, 1994.

[MS97] P.J. Munson and R.K. Singh. Multi-body interactions within the graph of protein structure. In T. Gaasterland, P. Karp, K. Karplus, C. Ouzounis, and C. Sander et al., editors, *In Fifth International Conference on Intelligent Systems for Molecular Biology*, pages 198–201. AAAI Press, Halkidiki, Greece, 1997.

[MS00] G. Michaelson and N. Scaife. Parallel functional island model genetic algorithms through nested skeletons. In *Proceedings of 12th International Workshop on the Implementation of Functional Languages*, pages 307–313, September 2000.

[Müh98] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346, 1998.

[MY93] H. Mamitsuka and K. Yamanishi. Protein helix region prediction based on stochastic-rule learning. In *In 26th Annual Hawaii International Conference on System Sciences eds.*, pages 659–668, Maui, HI, U.S.A., 1993. IEEE Computer Society.

[Nag77] K.Ñagano. Triplet information in helix prediction applied to the analysis of super-secondary structures. *Journal of Molecular Biology*, 109:251–274, 1977.

[NB03] NIH-BISTIC. National institute of health – biomedical information science and technology initiative consortium. http://www.bisti.nih.gov/bistic2.cfm, 2003.

[New82] A.Ñewell. The knowledge level. *Artificial Intelligance*, (18):87–127, 1982.

[NW70] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[OK00] M. Ouali and R.D. King. Cascaded multiple classifiers for secondary structure prediction. *Protein Science*, 9:1162–1176, 2000.

[Pat87] L. Patthy. Detecting homology of distantly related proteins with consensus sequences. *Journal of Molecular Biology*, 198:567–577, 1987.

[Paz97] M. Pazzani. Searching for dependencies in Bayesian classifiers. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 239–248, 1997.

[PB97] M. Pazzani and D. Billsus. Learning and revising user profiles: the identification of interesting web sites. *Machine Learning*, 27:313–331, 1997.

[PC51] L. Pauling and R.B. Corey. Configurations of polypeptide chains with favored orientations around single bonds: two new pleated sheets. *Proc. Natl. Acad. Sci. USA*, 37:729–740, 1951.

[PCB51] L. Pauling, R.B. Corey, and H.R. Branson. The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain. *Proc. Natl. Acad. Sci. USA*, 37:205–234, 1951.

[Pea88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*. Morgan Kaufmann Publisher, Inc., 1988.

[PF83] O.B. Ptitsyn and A.V. Finkelstein. Theory of protein secondary structure and algorithm of its prediction. *Biopolymers*, 22:15–25, 1983.

[PLL02] J.M. Peña, J.A. Lozano, and P. Larrañaga. Learning recursive Bayesian multinets for data clustering by means of constructive induction. *Machine Learning*, 47:63–89, 2002.

[PMB96] M. Pazzani, J. Murumatsu, and D. Billsus. Syskil and Webert: identifying interesting web sites. In *Proceedings of the 13th National Conference on Artificial Intelligence*, pages 54–61, 1996.

[PPRB01] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi. Improving the prediction of protein secondary strucure in three and eight classes using recurrent neural networks and profiles. *Proteins*, pages 228–235, 2001.

[PRC+60] M.F. Perutz, M.G. Rossmann, A.F. Cullis, G. Muirhead, and G. Will et al. Structure of haemoglobin: a three-dimensional Fourier synthesis at 5.5 Amstrongs resolution, obtained by X-ray analysis. *Nature*, 185:416–422, 1960.

[Pun98] W.F. Punch. How effective are multiple populations in genetic programming. In *Genetic Programming, Proceedings of the Third Annual Conference*, pages 308–313, 1998.

[QS88] N. Qian and T.J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:865–884, 1988.

[Qui86] J.R. Quilan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[Qui93] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, Los Altos, California, 1993.

[RCB96] R.B. Russell, R.R. Copley, and G.J. Barton. Protein fold recognition by mapping predicted secondary structures. *Journal of Molecular Biology*, 259:349–365, 1996.

[RdML02] V. Robles, P. de Miguel, and P. Larrañaga. Solving the travelling salesman problem with estimation of distribution algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*, pages 211–229. Kluwer Academic Publisher, 2002.

[RE01] B. Rost and V.A. Eyrich. EVA: large-scale analysis of secondary structure prediction. *Proteins*, 5:192–199, 2001.

[Rei80] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980.

[Ris86] J. Rissanen. Stochastic complexity and modelling. *Ann. Statist.*, (14):1080–1100, 1986.

[RKF83] E. Russek, R.A. Kronmal, and L.D. Fisher. The effect of assuming independence in applying Bayes theorem to risk estimation and classification in diagnosis. *Computers and Biomedical Research*, 16:537–552, 1983.

[RKW91] M.J. Rooman, J.P. Kocher, and S.J. Wodak. Prediction of protein backbone conformation based on seven structure assignments: influence of local interactions. *Journal of Molecular Biology*, 221:961–979, 1991.

[RLM+03] V. Robles, P. Larrañaga, E. Menasalvas, M.S. Pérez, and V. Herves. Improvements of naïve Bayes collaborative filtering using interval estimation. In *IEEE Computer Society Press*. Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence (WI 2003), 2003.

[RLP+03a] V. Robles, P. Larrañaga, J.M. Peña, O. Marbán, J. Crespo, and M.S. Pérez. Collaborative filtering using interval estimation naïve Bayes. In *Lecture Notes in Artificial Intelligence (Advances in Web Intelligence)*, volume 2663, pages 46–53, Madrid, Spain, May 2003. 1st International Atlantic Web Intelligence Conference AWIC 2003.

[RLP+03b] V. Robles, P. Larrañaga, J.M. Peña, E. Menasalvas, and M.S. Pérez. Interval estimation naïve Bayes. In *Lecture Notes in Computer Science (Advances in Intelligent Data Analysis)*, Berlin, Germany, 2003. The 5th International Symposium on Intelligent Data Analysis.

[RLP+03c] V. Robles, P. Larrañaga, J.M. Peña, M.S. Pérez, E. Menasalvas, and V. Herves. Bayesian netwoks as consensed voting system in the construction of a multi-classifier for protein secondary structure prediction. *Artificial Intelligence in Medicine*, 2003.

[RLP+03d] V. Robles, P. Larrañaga, J.M. Peña, M.S. Pérez, E. Menasalvas, and V. Herves. Learning semi naïve Bayes estructures by estimation of distribution algorihtms. In *Proceedings of the 11th Portuguese Conference on Artificial Intelligence*, Beja, Portugal, 2003. Pendiente de aceptación.

[RLPP02] V. Robles, P. Larrañaga, J.M. Peña, and M.S. Pérez. Protein secondary structure prediction with naïve bayes classifiers. In *Proceedings of the Symposium on Bioinformatics and Computational Biology*, Salamanca, Spain, 2002.

[Ros98] B. Rost. Neural networks for protein structure prediction: hype or hit? *TICS*, page in press, 1998.

[RPH+03] V. Robles, M.S. Pérez, V. Herves, J.M. Peña, and P. Larrañaga. Parallel stochastic search for protein secondary structure prediction. In *Lecture Notes in Computer Science*, Czestochowa, Poland, 2003. 5th International Conference on Parallel Processing and Applied Mathematics. Pendiente de aceptación.

[RPP+03] V. Robles, M.S. Pérez, J.M. Peña, V. Herves, and P. Larrañaga. Parallel interval estimation naïve Bayes. In *Proceedings de las XIV Jornadas de Paralelismo*, Madrid, 2003. Pendiente de aceptación.

[RS76] B. Robson and E. Suzuki. Conformational properties of amino acid residues in globular proteins. *Journal of Molecular Biology*, 107:327–356, 1976.

[RS93] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70 % accuracy. *Journal of Molecular Biology*, 232:584–599, 1993.

[RS94] B. Rost and C. Sander. Evaluation of accuracy. *Proteins*, 19:55–72, 1994.

[RS01] M. Ramoni and P. Sebastiani. Robust learning with missing data. *Machine Learning*, 45(2):147–170, 2001.

[RSL02] J. Roure, R. Sangüesa, and P. Larrañaga. Partitional clustering by means of estimation of distribution algorithms. In P. Larrañaga and J. A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, 2002.

[RSS93] B. Rost, C. Sander, and R. Schneider. Progress in protein structure prediction. *Trends Biochem. Sci.*, 18:120–123, Abril 1993.

[RSS94a] B. Rost, C. Sander, and R. Schneider. PHD – an automatic mail server for protein secondary structure prediction. *Computer Applications in the Biosciences*, 10:53–60, 1994.

[RSS94b] B. Rost, C. Sander, and R. Schneider. Redefining the goals of protein secondary structure prediction. *Journal of Molecular Biology*, 235:13–26, 1994.

[Sah96] M. Sahami. Learning limited dependence Bayesian classifiers. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 335–338, 1996.

[SB98] A. Siddiqui and G.J. Barton. 3Dee-database of protein domain definitions. *Bioinformatics*, 17:200–201, 1998.

[Sch83] C. Schaffer. Selecting a classification method by cross-validation. *Machine Learning*, 13(1):135–143, 1983.

[Sch95] R. Schalkoff. *Pattern Recognition: Statistical, Structural, and Neural Approaches.* Wiley, 1995.

[SDO78] R. Schwartz, M. Dayhoff, and B. Orcutt. A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 5:345–352, 1978.

[SGC57] A.G. Szent-Gyorgyi and C. Cohen. Role of proline in polypeptide chain configuration of proteins. *Science*, 126:697, 1957.

[SH89] G. D. Stormo and G. W. Hartzell. *Proc. Natl. Acad. Sci. USA*, 86:1183–1187, 1989.

[Sim83] H. Simon. Why should machines learn? In R.S. Mitchalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 1, pages 25–37. Morgan Kaufmann Publishers, Inc., 1983.

[SL98] B. Sierra and P. Larrañaga. Predicting the survival in malignant skin melanoma using Bayesian networks automatically induced by genetic algorithms. An empirical comparison between different approaches. *Artificial Intelligence in Medicine*, 14:215–230, 1998.

[SLB00] S.C. Schmidler, J.S. Liu, and D.L. Brutlag. Bayesian segmentation of protein secondary structure. *Journal of Computational Biology*, 2(1/2):233–248, 2000.

[SLX92] P. Stolorz, A. Lapedes, and Y. Xia. Predicting protein secondary structure using neural net and statistical methods. *Journal of Molecular Biology*, 225(2):363–377, 1992.

[SP82] J.A. Swets and R.M. Pickett. *Evaluation of Diagnostic Systems: Methods from Signal Detection Theory.* Nueva York: Academic Press, 1982.

[Spa88] A.K. Spackman. Learning categorical criteria in biomedical domains. In *Proceedings of the Fifth International Machine Learning Conference*, pages 36–46. Morgan Kaufmann Publisher, Inc., 1988.

[SS94] V.V. Solovyev and A.A. Salamov. Predicting a-helix and b-strand segments of globular proteins. *CABIOS*, 10:661–669, 1994.

[SS95] A.A. Salamov and V.V. Solovyev. Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignment. *Journal of Molecular Biology*, 247:11–15, 1995.

[SSL+99] B. Sierra, N. Serrano, P. Larranaga, E.J. Plasencia, I. Inza, J.J. Jimenez, J.M. De la Rosa, and J.M. Mora. Machine learning approaches to combine standard medical measures at an intensive care unit. In *Joint European Conference on Artificial Intelligence and Medical Decision Making*, pages 932–939, Aalborg, Denmark, 1999.

[Sta84] R. Staden. Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res.*, 12:505–519, 1984.

[Ste93] J. Stender. *Parallel Genetic Algorithms: Theory and Applications.* IOS Press, 1993.

[SW81] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[TAK94] R. L. Tatusov, S. F. Altschul, and E. V. Koonin. Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 91:12091–12095, 1994.

[Tay86] W. R. Taylor. Identification of protein sequence homology by consensus template alignment. *Journal of Molecular Biology*, 188:233–258, 1986.

[Tin94] K.M. Ting. Discretization of continuous-valued attributes and instance-based learning. Technical Report 491, University of Sydney, 1994.

[TMS+81] D.M. Titterington, G.D. Murray, L.S. Spiegelhalter, A.M. Skene, J.D.F. Habbema, and G.J. Gelpke. Comparison of discrimination techniques applied to a comples data set of head injured patients (with discussion). *Journal of the Royal Statistical Society Series A*, 144(2):145–175, 1981.

[TT83] W.R. Taylor and J.M. Thornton. Prediction of super-secondary structure in proteins. *Nature*, 301:540–542, 1983.

[VDW91] V.N. Viswanadhan, B. Denckla, and J.N. Weinstein. New joint prediction algorithm (Q7-JASEP) improves the prediction of protein secondary structure. *Biochem*, 30:11164–11172, 1991.

[WH60] B. Widrow and M.E. Hoff. Adaptive switching circuits. In *IRE WESCON, New-York*, pages 96–104, 1960.

[Wid87] B. Widrow. ADALINE and MADALIBE. In *Proceedings IEEE in First International Conference on Neural Networks*, pages 143–158, 1987.

[Wol92] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

[WP98] G.I. Webb and M.J. Pazzani. Adjusted probability naive Bayesian induction. In *Proceedings of the 11th Australian Joint Conference on Artificial Intelligence*, pages 285–295, 1998.

[WRH97] D. Whitley, S.B. Rana, and R.B. Heckendorn. Island model genetic algorithms and linearly separable problems. In *Evolutionary Computing, AISB Workshop*, pages 109–125, 1997.

[YL93] T.-M. Yi and E.S. Lander. Protein secondary structure prediction using nearest-neigbor methods. *Journal of Molecular Biology*, 232:1117–1129, 1993.

[Zaf02] M. Zaffalon. The naive credal classifier. *Journal of Statistical Planning and Inference*, 105(1):5–21, 2002.

[ZB96] Z.-Y. Zhu and T.L. Blundell. The use of amino acid patterns of classified helices and strands in secondary structure prediction. *Journal of Molecular Biology*, 260:261–276, 1996.

[ZBTS87] M. Zvelebil, G. Barton, W. Taylor, and M. Sternberg. Prediction of protein secondary structure and active sites using the alignment of homologous sequences. *Journal of Molecular Biology*, 195:957–961, 1987.

[ZHS97] K.H. Zou, W.J. Hall, and D.E. Shapiro. Smooth non-parametric receiver operating characteristic (roc) curves for continuous diagnostic tests. *Statistics in Medicine*, (16):2143–2156, 1997.

[ZL01] H. Zhang and C.X. Ling. An improved learning algorithm for augmented naive Bayes. In *In Pacific-Asia Conference on Knowledge Discovery and Data Mining*, number 45, pages 581–586, 2001.

[ZMW92] X. Zhang, J.P. Mesirov, and D.L. Waltz. Hybrid system for protein secondary structure prediction. *Journal of Molecular Biology*, 225:1049–1063, 1992.

[ZVFR99] A. Zemla, C. Venclovas, K. Fidelis, and B. Rost. A modified definition of SOV, a segment-based measure for protein secondary structure prediction assessment. *Proteins*, 34:220–223, 1999.

[ZW00] Z. Zheng and G.I. Webb. Lazy learning of Bayesian rules. *Machine Learning*, 41:53–84, 2000.