

REDES BAYESIANAS

María Oliveira Pérez

Silvia Suárez Crespo

15 de enero de 2010

Índice general

1. Introducción	5
1.1. Motivación	5
1.2. Introducción	6
2. Conceptos previos	9
2.1. Prerrequisitos de Teoría de la Probabilidad	9
2.2. Teoría de Grafos	15
2.3. Redes causales	17
3. Estudio teórico	21
3.1. Redes bayesianas. Definición.	21
3.2. Redes bayesianas como técnica del <i>Aprendizaje Automático</i>	27
3.3. Inferencia sobre redes bayesianas	28
3.4. Aprendizaje de redes bayesianas	46
3.4.1. Aprendizaje estructural	47
3.4.2. Aprendizaje paramétrico	61
3.5. Clasificadores basados en redes bayesianas	65
3.5.1. Clasificador Naive Bayes	65
3.5.2. Extensiones del clasificador Naive Bayes	66
3.5.3. Redes bayesianas como clasificadores	68

3.5.4. Evaluación de los clasificadores	69
4. Estudio práctico	71
4.1. Análisis de datos con Weka	71
4.1.1. Pequeña introducción a Weka	71
4.1.2. Clasificadores basados en redes bayesianas con Weka	76
4.1.3. Trabajando con los datos	82
4.2. Análisis de datos con GeNIe	101
4.3. Análisis de datos con R	112
4.3.1. Aprendizaje de redes bayesianas	112
4.3.2. Clasificadores	113

Capítulo 1

Introducción

1.1. Motivación

Los mecanismos de representación de conocimiento más utilizados desde hace muchos años están basados en terminología lógica y se rigen por los siguientes supuestos:

- Todo hecho sobre el que razonemos debe poder ser evaluado como cierto o falso.
- Para poder razonar necesitamos tener todos los hechos a nuestra disposición.

Pero en la práctica nos encontramos con estos problemas:

- Representar el conocimiento para cubrir todos los hechos que son relevantes para un problema es difícil.
- Existen dominios en los que se desconocen todos los hechos y reglas necesarias para resolver el problema.
- Existen problemas en los que aún teniendo las reglas para resolverlos no disponemos de toda la información necesaria.

Por este motivo se plantea el cambio a un nuevo tipo de modelos: los modelos probabilistas.

1.2. Introducción

Uno de los principales problemas a los que se enfrentan las técnicas de minería de datos es cómo trabajar con incertidumbre. Este hecho, que es un problema para muchas de las técnicas de minería de datos en general, no lo es en absoluto para los métodos y técnicas bayesianas, puesto que una de sus principales características es el uso de la teoría de la probabilidad para cuantificar la incertidumbre.

Las redes bayesianas modelan un fenómeno mediante un conjunto de variables y las relaciones de dependencia entre ellas (representado mediante un grafo). Se han vuelto extremadamente populares en la última década y han sido utilizadas para distintas aplicaciones en distintas áreas, como en el aprendizaje automático, minería de textos, procesamiento de lenguaje natural, reconocimiento de velocidades, procesamiento de señales, bioinformática, códigos de control de errores, diagnosis médica, predicción meteorológica y redes celulares.

Inicialmente, estos modelos eran contruidos “a mano” basados en un conocimiento experto, pero en los últimos años se han desarrollado diversas técnicas para aprender a partir de datos, tanto la estructura como los parámetros asociados al modelo. También es posible combinar conocimiento experto con los datos para aprender el modelo.

Por otra parte, en una forma general del modelo, los nodos pueden representar, además de variables aleatorias, hipótesis, creencias o variables latentes. Una estructura como ésta puede resultar conveniente y apetecible para representar tanto la semántica causal como la probabilística y es ideal para combinar información a priori y datos observados. En resumen, las redes bayesianas pueden, incluso en el caso en el que existan datos faltantes, utilizarse para el aprendizaje de las relaciones causales e incluso para predecir eventos futuros.

El nombre de redes bayesianas puede llevar a malentendidos. Aunque el uso de la estadística bayesiana en conjunto con los modelos de redes bayesianas proporciona aproximaciones eficientes que evitan la infrasuavización de los datos, la utilización de redes bayesianas no se encuentra vinculada a la estadística bayesiana en exclusiva. De hecho, muchos usuarios utilizan métodos frecuentistas para estimar los parámetros de las redes bayesianas.

Nuestro objetivo final del presente trabajo será el estudio de dichas redes bayesianas (como clasificadores que utilizan relaciones de dependencia) y su aplicación a unos datos reales.

Para ello comenzaremos con un recorrido teórico que nos describa pormenorizadamente en qué consisten las redes bayesianas y finalizaremos realizando un estudio práctico y obteniendo las consiguientes conclusiones.

Podemos dividir el documento en las siguientes secciones:

- Esta introducción, en la que se encauza la motivación que lleva a la creación de clasificadores bayesianos que tomen en cuenta las relaciones de dependencia.
- La siguiente sección la dedicaremos a conceptos previos que serán necesarios conocer para comprender los razonamientos y resultados que se siguen en los capítulos posteriores.
- En la siguiente sección trataremos los primeros aspectos teóricos de las redes bayesianas: definición, aplicación al aprendizaje automático y métodos de inferencia.
- Dedicaremos una sección completa al aprendizaje de redes bayesianas, tanto paramétrico como estructural.
- Para finalizar con el estudio teórico, concluimos con el estudio de las redes bayesianas como clasificadores.
- Proseguiremos con el estudio práctico en el que se tratará de comparar el desempeño de los distintos clasificadores estudiados, así como introducir todos los conceptos anteriormente estudiados referentes a las demás secciones (inferencia, aprendizaje,...).
- Finalmete, dedicamos una última sección a las conclusiones sobre el estudio realizado.

Capítulo 2

Conceptos previos

De forma concisa y clara se tiene que las redes bayesiana se definen como redes causales para las cuales sus arcos representan probabilidades condicionales. Por este motivo será de interés, antes de dirigirnos hacia el estudio teórico de la redes bayesianas, hacer un recorrido sobre los conceptos más importantes dentro de los campos de la probabilidad condicional y las redes causales. Para ello comenzaremos redactando una serie de resultados importantes relacionados con la probabilidad condicional y términos relacionados. A continuación dedicaremos parte de la sección a la descripción de los elementos más importantes de un grafo para terminar aplicándolo a un breve estudio de las redes causales.

2.1. Prerrequisitos de Teoría de la Probabilidad

En esta subsección haremos un recordatorio de los resultados y definiciones más importantes de la Teoría de la Probabilidad que utilizaremos para el tratamiento de las redes bayesianas. Se suponen conocidos dichos conceptos y el objetivo será tan sólo recordar dichos conocimientos a lo largo del texto.

Consideremos un experimento cualquiera. Nos referiremos al conjunto de posibles resultados de un experimento como el *espacio muestral* de dicho experimento. Entenderemos como *experimento* cualquier tipo de proceso para el cual el resultado sea incierto. Asumiremos que el espacio muestral del experimento contiene todos los posibles resultados del mismo, y que cada par de resultados son mutuamente excluyentes. Esto garantiza que experimento finaliza al obtener exactamete uno de los resultados especificados en el espacio muestral. Un subconjunto del espacio muestral se denomina *evento* y si el evento contiene tan sólo

un resultado, nos referiremos al evento simplemente como *resultado*.

Para medir nuestro grado de incertidumbre sobre un experimento, asignaremos la probabilidad $P(A)$ a cada evento $A \subseteq S$, denotando por S al espacio muestral. Estas probabilidades deben obedecer los siguientes tres axiomas:

Axioma 1.

$$P(S) = 1. \quad (2.1)$$

Es decir, el evento S del que obtendremos un resultado en el espacio muestral ocurrirá con toda seguridad y por tanto le asignamos probabilidad 1.

Axioma 2. Para todo $A \subseteq S$ se tiene que $P(A) \geq 0$.

Esto es, todo evento A debe tener probabilidad no negativa.

Axioma 3. Si $A \subseteq S$, $B \subseteq S$ y $A \cap B = \emptyset$ entonces

$$P(A \cup B) = P(A) + P(B). \quad (2.2)$$

Equivalentemente, si dos eventos A y B son disjuntos, entonces la probabilidad de que ocurran los dos eventos de forma combinada es la suma de la probabilidad individual de cada uno de los eventos. Por otra parte, si los eventos no son disjuntos se obtiene que:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B).$$

Cuando se especifica una afirmación de la probabilidad $P(A)$ del evento A , esta puede encontrarse implícitamente condicionada por otros factores. Este tipo de probabilidades se denominan *probabilidades condicionadas* y generalmente son afirmaciones del siguiente tipo:

“Dado el evento B , la probabilidad del evento A es p .”

La notación para la afirmación anterior es $P(A|B) = p$. Debe recalarse que $P(A|B) = p$ no significa que si B es verdadero, entonces la probabilidad de A es p ; significa que si B es verdadero y todo lo demás es irrelevante para A , entonces la probabilidad de A es p .

Propiedad 1 (Probabilidad condicionada). Dados dos eventos A y B , con $P(B) > 0$, la probabilidad condicionada de A dado B es:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}. \quad (2.3)$$

Obviamente, cuando se trabaja con probabilidades condicionadas podemos condicionar en más de un evento; en este caso la definición de la probabilidad condicionada se generaliza a:

$$P(A|B \cap C) = \frac{P(A \cap B \cap C)}{P(B \cap C)}.$$

Podemos reescribir la expresión (2.3) para obtener la regla fundamental de cálculo de probabilidades:

Teorema 1 (Regla fundamental).

$$P(A|B)P(B) = P(A \cap B). \quad (2.4)$$

A través de razonamientos simples sobre (2.4) llegamos a la Regla de Bayes:

Teorema 2 (Regla de Bayes).

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (2.5)$$

La regla de Bayes nos proporciona un método para actualizar nuestras creencias sobre un evento A dada información sobre otro evento B . Por esta razón se denomina usualmente a $P(A)$ como *probabilidad a priori* (indica el grado de creencia en A a falta de otra información) mientras que $P(A|B)$ se denomina *probabilidad a posteriori* de A dado B (indica el grado de creencia en un evento tras la observación de otros eventos asociados a ella); la probabilidad $P(B|A)$ se denomina la *verosimilitud de A dado B*.

Finalmente, al igual que para la regla fundamental, podemos generalizar (2.5) para otro evento C :

$$P(A|B, C) = \frac{P(B|A, C)P(A|C)}{P(B|C)}.$$

Algunas veces la información sobre un evento B no cambia para nada la probabilidad de ocurrencia de otro evento A , y en ese caso decimos que A y B son *independientes*.

Definición 1 (Independencia). Los eventos A y B son independientes si:

$$P(A|B) = P(A).$$

Esta noción de independencia es simétrica, por lo que si A es independiente de B entonces B es independiente de A , es decir,

$$P(A|B) = P(A) \Rightarrow P(B|A) = P(B).$$

Cuando dos eventos son independientes (2.4) puede reescribirse como sigue:

$$P(A \cap B) = P(A|B)P(B) = P(A)P(B). \quad (2.6)$$

Esto es, podemos calcular la probabilidad de que ambos eventos ocurran sin más que multiplicar las probabilidades para los eventos individuales.

El concepto de independencia aparece también cuando condicionamos a más de un evento. Específicamente, si la información del evento B no cambia la probabilidad de ocurrencia de A cuando conocemos de antemano el evento C , entonces se dice que A y B son condicionalmente independientes dado C .

Definición 2 (Independencia condicional). Los eventos A y B son condicionalmente independientes dado el evento C si:

$$P(A|B \cap C) = P(A|C). \quad (2.7)$$

De la misma forma se puede probar que la independencia condicional es simétrica.

Cuando dos eventos son condicionalmente independientes, entonces podemos utilizar una regla multiplicativa de la forma:

$$P(A \cap B|C) = P(A|C)P(B|C).$$

Hasta ahora nos hemos referido a probabilidades de eventos simples. Nos interesará sin embargo trabajar con colecciones de espacios muestrales, también llamadas *variables*, y extenderemos los conceptos anteriores para probabilidades sobre variables. Una variable aleatoria tiene un dominio de valores, y dependiendo del mismo podemos clasificar las variables aleatorias en booleanas, discretas o continuas.

Una variable puede ser considerada un experimento, y para cada resultado del experimento la variable tiene un estado correspondiente (supondremos que los estados serán discretos, y por tanto la variable será discreta). El conjunto de estados asociados a la variable A se denota por $\text{sp}(A) = (a_1, \dots, a_n)$ y de forma similar al espacio muestral estos estados deber ser tales que la variable deba encontrarse por lo menos en algún estado (aunque no conozcamos cuál) y además mutuamente excluyentes. Esta última afirmación asegura que la variable se encuentre sólo en un estado. Utilizaremos letras mayúsculas para las variables y minúsculas para los estados y, a excepción de que se establezca lo contrario, la variable

posee un número finito de estados.

Para una variable A con estados a_1, \dots, a_n , expresamos nuestra incertidumbre sobre sus estados mediante una distribución de probabilidad $P(A)$ sobre estos estados:

$$P(A) = (x_1, \dots, x_n) ; x_i \geq 0 ; \sum_{i=1}^n x_i = x_1 + \dots + x_n = 1,$$

donde x_i representa la probabilidad de que la variable A se encuentre en el estado a_i . La distribución se dice *uniforme* (o *par*) si todas las probabilidades son iguales.

Al igual que hablamos de probabilidades condicionadas para eventos, podemos hablar de probabilidades condicionadas para variables: si la variable B tiene los estados b_1, \dots, b_m , entonces $P(A|B)$ contiene $n \cdot m$ probabilidades condicionales $P(a_i|b_j)$ que especifican la probabilidad de estar en a_i dado b_j . Esto es, la probabilidad condicional para una variable dada otra variable es un conjunto de probabilidades (normalmente organizado en una tabla de dimensión $n \cdot m$) con una probabilidad para cada combinación de los estados de las variables involucradas. Además, dado que $P(A|B)$ especifica la distribución de probabilidad para cada evento $B = b_j$, sabemos por (2.1) que la probabilidad sobre A debe sumar 1 para cada estado de B :

$$\sum_{i=1}^n P(A = a_i | B = b_j) = 1 \text{ para cada } b_j.$$

La probabilidad de ver resultados conjuntos para diferentes experimentos puede ser expresada por la *probabilidad conjunta* para dos o más variables: para cada configuración (a_i, b_j) de las variables A y B , $P(A, B)$ especifica la probabilidad de ver $A = a_i$ y $B = b_j$ al mismo tiempo. Por tanto, $P(A, B)$ consiste en $n \cdot m$ números, y, de forma similar a $P(A|B)$, $P(A, B)$ se representa normalmente en una tabla de dimensión $n \cdot m$. Notemos que dadas las condiciones supuestas sobre los espacios de estados de A y B se sigue que todas las combinaciones de sus estados (el producto cartesiano) verifican las mismas condiciones que los primeros, y por tanto pueden ser consideradas un espacio muestral. Consecuentemente, por (2.1) se tiene:

$$P(A, B) = \sum_{i=1}^n \sum_{j=1}^m P(A = a_i | B = b_j) = 1.$$

Cuando la Regla Fundamental (2.4) se utiliza sobre variables A y B , el proceso consiste en aplicar la regla sobre cada una de las $n \cdot m$ combinaciones (a_i, b_j) de las dos variables:

$$P(a_i|b_j)P(b_j) = P(a_i, b_j).$$

Teorema 3 (Regla fundamental para variables).

$$P(A, B) = P(A|B)P(B), \quad (2.8)$$

y condicionada en otra variable C tenemos:

$$P(A, B|C) = P(A|B, C)P(B|C). \quad (2.9)$$

Utilizando una tabla de probabilidades conjuntas $P(A, B)$, la distribución de probabilidad $P(A)$ puede calcularse considerando los resultados de B que pueden ocurrir con cada estado a_i de A . Existen exactamente m resultados diferentes para los cuales A se encuentra en el estado a_i , concretamente los resultados mutuamente excluyentes $(a_i, b_1), \dots, (a_i, b_m)$. Por tanto, por (2.2)

$$P(a_i) = \sum_{j=1}^m P(a_i, b_j), \quad i = 1, \dots, n.$$

Este cálculo se denomina cálculo de *marginales*. La notación es:

$$P(A) = \sum_B P(A, B).$$

La regla de Bayes para los eventos (2.5) puede extenderse igualmente a variables:

Teorema 4 (Regla de Bayes para variables).

$$P(B|A) = \frac{P(A \cap B)P(B)}{P(A)} = \frac{P(A, B)}{\sum_B P(A, B)}, \quad (2.10)$$

y condicionado en otra variable C tenemos:

$$P(B|A, C) = \frac{P(A \cap B, C)P(B|C)}{P(A|C)} = \frac{P(A, B|C)}{\sum_B P(A, B|C)}. \quad (2.11)$$

Notemos que las dos igualdades en las ecuaciones (2.10) y (2.11) se tienen, la primera, por aplicación de la Regla Fundamental y la segunda, por el cálculo de marginales explicado anteriormente.

El concepto de independencia condicional se generaliza también para variables:

Definición 3 (Independencia condicional para variables). Dos variables A y B se dicen condicionalmente independientes dada la variable C si:

$$P(a_i|b_j, c_k) = P(a_i|c_k) \quad (2.12)$$

para cada $a_i \in \text{sp}(A)$, $b_j \in \text{sp}(B)$ y $c_k \in \text{sp}(C)$.

Utilizaremos la notación $P(A|B, C) = P(A|C)$ para referirnos a la expresión (2.12).

Esto significa que cuando se conoce el estado de C , entonces ningún tipo de conocimiento sobre B puede alterar la probabilidad de A . Observemos que estamos requiriendo que la condición de independencia se mantenga para todos y cada uno de los estados de C ; cuando el conjunto condicionante es vacío, es decir, $C = \emptyset$, entonces decimos que A y C son marginalmente independientes (es decir, $P(A|B) = P(A)$).

Cuando dos variables A y B son condicionalmente independientes dado C , entonces la Regla Fundamental para variables (2.9) se simplifica:

$$P(A, B|C) = P(A|B, C)P(B|C) = P(A|C)P(B|C).$$

Estamos suponiendo en todo caso que las variables con las que trabajamos son discretas. Para variables continuas existen conceptos equivalentes (función de densidad), pero dado que para las redes bayesianas veremos que se utilizan variables discretas (y en caso de existir variables continuas se discretizan), no profundizaremos en estos conceptos.

2.2. Teoría de Grafos

Un modelo probabilístico puede definirse usando un grafo que describa las relaciones existentes entre las variables. Supóngase un conjunto de variables $X = \{X_1, \dots, X_n\}$, $i = 1, \dots, n$, que pueden relacionarse entre sí. El conjunto anterior puede representarse gráficamente por una colección de nodos o vértices, asociando un nodo a cada elemento de X . Estos nodos pueden conectarse por arcos, indicando las relaciones existentes entre los mismos. Un arco entre los nodos X_i y X_j se denotará mediante L_{ij} . Así mismo, el conjunto de todos los arcos se denotará por $L = \{L_{ij}/X_i \text{ y } X_j \text{ están conectados}\}$. Por tanto, un grafo puede definirse mediante el conjunto de nodos, \mathbf{X} , cuyos elementos (nodos) denotaremos con el mismo nombre que las variables, y las relaciones entre los mismos, L .

Definición 1 (Grafo). Un grafo es un par de conjuntos $G = (\mathbf{X}, L)$ donde $\mathbf{X} = \{X_1, \dots, X_n\}$ es un conjunto finito de elementos (nodos) y L es un conjunto de arcos, es decir, un subconjunto de pares ordenados de elementos distintos de \mathbf{X} .

El concepto de grafo puede definirse de forma más general. Por ejemplo, puede permitirse que dos nodos estén conectados por más de un arco o, incluso, que un nodo esté conectado consigo mismo. Sin embargo, al utilizar los grafos para representar un conjunto de variables

(nodos), y unas relaciones de dependencia entre ellas (arcos), no será necesario que dos nodos estén unidos por más de un arco, o que un arco una a un nodo consigo mismo. Los arcos de un grafo pueden ser dirigidos o no dirigidos, dependiendo de si se considera o no el orden de los nodos.

Definición 2 (Arco dirigido). Dado un grafo $G = (\mathbf{X}, L)$, si $L_{ij} \in L$ y $L_{ji} \notin L$, el arco L_{ij} entre los nodos X_i y X_j se denomina dirigido y se denota mediante $X_i \rightarrow X_j$.

Definición 3 (Arco no dirigido). Dado un grafo $G = (\mathbf{X}, L)$, si $L_{ij} \in L$ y $L_{ji} \in L$, el arco L_{ij} entre los nodos X_i y X_j se denomina no dirigido y se denota mediante $X_i - X_j$ o $X_j - X_i$.

Definición 4 (Grafo dirigido y no dirigido). Un grafo en el cual todos los arcos son dirigidos se denomina grafo dirigido, y un grafo en el que todos sus arcos son no dirigidos se denomina no dirigido. Por tanto, en un grafo dirigido es importante el orden del par de nodos que definen cada arco, mientras que en un grafo no dirigido, el orden carece de importancia.

Definición 5 (Camino entre dos nodos). Un camino del nodo X_i al nodo X_j es una sucesión de nodos X_{i_1}, \dots, X_{i_r} comenzando en $X_{i_1} = X_i$ y finalizando en $X_{i_r} = X_j$, de forma que existe un arco del nodo X_{i_k} al nodo $X_{i_{k+1}}$, $k = 1, \dots, r - 1$. La longitud del camino, $(r - 1)$, se define como el número de arcos que contiene.

Definición 6 (Camino cerrado). Un camino X_{i_1}, \dots, X_{i_r} se dice cerrado si el nodo inicial coincide con el final, es decir, si $X_{i_1} = X_{i_r}$.

Definición 7 (Padre e hijo). Cuando existe un arco dirigido, $X_i \rightarrow X_j$, del nodo X_i al nodo X_j , entonces se dice que el nodo X_i es un padre del nodo X_j , y que el nodo X_j es un hijo de X_i . El conjunto de los padres de un nodo X_i se denota por $Pa(X_i)$.

Definición 8 (Ciclo). Un ciclo es un camino cerrado en un grafo dirigido.

Definición 9 (Grafos cíclicos y acíclicos). Un grafo dirigido se denomina cíclico si contiene al menos un ciclo; en caso contrario se denomina grafo dirigido acíclico.

Definición 10 (Grafo conexo). Un grafo conexo es aquél que verifica que entre cualquier par de nodos existe al menos un camino.

Definición 11 (Grafo simplemente conexo o poliárbol). Un grafo simplemente conexo o poliárbol es aquél que verifica que entre cualquier par de nodos existe un único camino.

Definición 12 (Grafo múltiplemente conexo). Un grafo múltiplemente conexo es aquél que contiene bucles o ciclos.

Definición 13 (Árbol). Un árbol es un poliárbol en el que cada nodo tiene un sólo padre, menos el nodo raíz que no tiene ninguno.

2.3. Redes causales

Definición 1 (Red causal). Una red causal consiste en un conjunto de variables y un conjunto de arcos dirigidos entre las variables. Matemáticamente, la estructura se denomina un grafo dirigido.

Una variable puede tener cualquier número de estados (o salidas). Las variables deben tener un conjunto de estados continuo o numerable, pero consideraremos tan sólo variables con un número finito de estados. En una red causal, una variable representa un conjunto de posibles estados de determinados sucesos. Las redes causales pueden utilizarse para controlar cómo un cambio de incertidumbre en una variable puede cambiar la probabilidad de ocurrencia de otras. Presentamos aquí una serie de reglas para este tipo de razonamientos (estas reglas son independientes de los cálculos probabilísticos particulares).

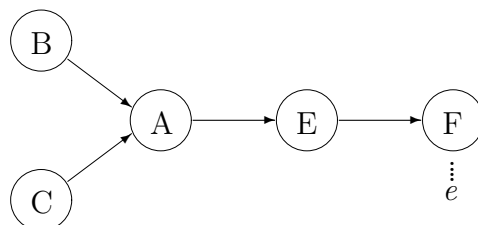


Figura 2.1: Ejemplo de red causal en la que se le aplica evidencia a uno de sus nodos.

Se dice que tenemos *evidencia dura* (hard) sobre una variable A cuando poseemos conocimiento determinista sobre ella, es decir, $P(A) = 1$ o $P(A) = 0$. Al asignar evidencia dura al nodo se le llama *instanciación*. Se dice que tenemos *evidencia parcial* (soft) sobre la variable A cuando el conocimiento que tenemos es probabilístico (ente 0 y 1). En este último caso se incluyen las probabilidades a priori y las probabilidades actualizadas al instanciarse alguna variable. Por ejemplo, en la Figura (2.1) lo que hacemos es aplicarle evidencia al nodo F (se denota mediante una e unida por puntos suspensivos al nodo). Como veremos cuando profundicemos en las posibles estructuras de la red, se verifica que el hecho anterior influye en el nodo A y se produce una evidencia parcial en el mismo.

Las conexiones entre los nodos pueden clasificarse en:

- **Conexión en serie:** Consideremos el caso en que una variable A tiene influencia

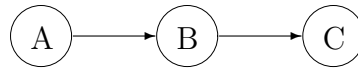


Figura 2.2: *Conexión en serie.*

en una variable B , que a su vez la tiene sobre C (Figura (2.2)). Obviamente, la evidencia que tengamos de A influirá la certidumbre que tengamos sobre B , que a su vez influirá en C . De forma similar, una evidencia en C influirá en A a través de B .

- **Conexión divergente:** Consideremos una variable A con hijos B, C, D y E (Figura

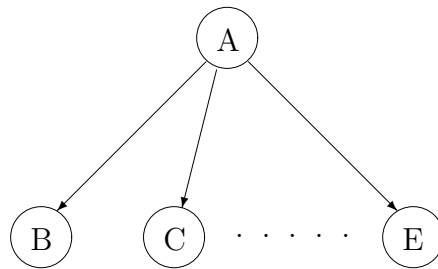


Figura 2.3: *Conexión divergente.*

(2.3)). Entonces la información se transmite desde A a los hijos y también entre los hijos a través de A .

- **Conexión convergente:** Por último, sea el caso en el que contamos con varias va-

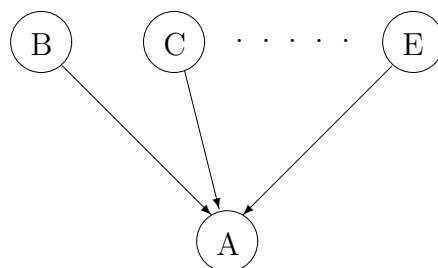


Figura 2.4: *Conexión convergente.*

riables (sean B, C, D y E) no relacionadas directamente por un arco entre sí y que comparten el mismo hijo A (Figura (2.4)). Entonces, la información se transmite tan sólo de padres a hijos.

Introduciremos ahora la definición de un algoritmo muy simple (y además rápido: el tiempo de ejecución aumenta de forma lineal al ir añadiendo variables) para determinar si dos variables son condicionalmente independientes o no:

Definición 2 (d-separación). Dos variables distintas A y B en una red causal se dice que están d-separadas (“d” de “grafo dirigido”) si para cualquier camino entre A y B existe una variable intermedia V (distinta de A o B) tal que se verifica alguna de la dos posibilidades siguientes:

- La conexión es en serie o divergente y V está instanciado.
- La conexión es convergente y se verifica que o bien V o alguno de sus descendientes están instanciados.

Si A y B no están d-separadas se dice que están *d-conectadas*.

Se tiene que si A y B están d-separadas entonces cualquier cambio en A no produce ningún impacto sobre la certidumbre de B y viceversa. De esta forma se tiene que si las variables están d-separadas, entonces las variables son condicionalmente independientes, mientras que si las variables están d-conectadas entonces son condicionalmente dependientes.

Volvamos ahora a los tipos de conexiones definidas anteriormente. En la Figura (2.2) se tiene que, si el estado de B es conocido, entonces el canal entre A y C se bloquea y A y

C están entonces d-separadas dada B (son independientes). Concluimos entonces que para este caso, la evidencia se transmitirá de una variable A a otra B a través de la conexión en serie a no ser que se conozca el estado de una variable que se encuentre en el camino entre ambas variables. Para la Figura (2.3), la influencia puede fluir entre los hijos de A a no ser que se conozca el estado de A , es decir, que B, C, D y E estén d-separadas dado A . Finalmente, para el último caso (Figura (2.4)) el razonamiento es más delicado. Si no conocemos absolutamente nada de A excepto lo que se infiere de lo que se sabe de sus padres B, \dots, E , entonces los padres son independientes: la evidencia de alguno de ellos no nos dice nada de lo que ocurre con las restantes (el conocimiento de una posible causa no implica la ocurrencia o no de las demás). Sin embargo, si se conoce algo sobre la consecuencia (A), entonces sí ocurre que la información acerca de una causa puede proporcionar información acerca de las demás. Es decir, la evidencia se transmitirá a través de una conexión divergente de A a B sólo si o bien la variable de la conexión o bien alguno de sus descendientes ha recibido evidencia.

Definición 3 (Manto de Markow). La manto de Markov de una variable A es el conjunto consistente en los padres de A , los hijos de A y las variables que comparten hijos con A .

Para finalizar esta sección conviene tener en cuenta que las relaciones causales que se establecen tienen también un lado cuantitativo al que denominamos *intensidad*. Esta puede expresarse adjuntando los números a las uniones de la red. En la sección posterior veremos que para el caso de las redes bayesianas las intensidades no son otra cosa que probabilidades condicionadas.

Capítulo 3

Estudio teórico

3.1. Redes bayesianas. Definición.

Partamos de que nuestro objetivo es, dado un conjunto de datos, estructurar el mismo para poder representar el conocimiento incierto. Consideremos una red causal cualquiera. Consideremos dos variables A y B entre las cuales existe una relación de padre e hijo, respectivamente. Usando cálculo probabilístico parece natural la idea de que la intensidad de la relación causal sea $P(B|A)$. Sin embargo, si C es otra variable que es también padre de B , entonces las dos probabilidades condicionadas $P(B|A)$ y $P(B|C)$ por sí solas no proporcionan información sobre cómo interactúan los impactos de A y C . Ambos impactos deben sumarse o contrarrestarse en varios aspectos, por lo que necesitaremos la especificación de $P(B|A, C)$.

Definición 1 (Red bayesiana). Una red bayesiana consiste en lo siguiente:

- Un conjunto de variables (representadas por nodos) y un conjunto de arcos dirigidos entre las variables (si existe un arco dirigido de un nodo A a un nodo B entonces A tiene influencia sobre B).
- Cada variable tiene un conjunto finito de estados mutuamente excluyentes.
- Las variables y los arcos forma conjuntamente un grafo acíclico direccionado (abreviadamente DAG); un grafo direccionado es acíclico si no existe ningún camino direccionado $A_1 \rightarrow \dots, A_n$ tal que $A_1 = A_n$.

- Para cada variable A con padres B_1, \dots, B_n se adjunta la tabla de probabilidades condicionadas $P(A|B_1, \dots, B_n)$.

Equivalentemente: Sea $X = \{X_1, X_2, \dots, X_n\}$ un conjunto de variables aleatorias. Formalmente, una red bayesiana para X es un par $U = \langle G, T \rangle$ en el que:

- G es un grafo acíclico dirigido en el que cada nodo representa una de las variables X_1, X_2, \dots, X_n , y cada arco representa relaciones de dependencia directas entre las variables. La dirección de los arcos indica que la variable “apuntada” por el arco depende de la variable situada en su origen.
- T es un conjunto de parámetros que cuantifica la red. Contiene las probabilidades $P(x_i|Pa(x_i))$ para cada posible valor x_i de cada variable X_i y cada posible valor $Pa(x_i)$ de $Pa(X_i)$, donde éste último denota al conjunto de padres de X_i en G .

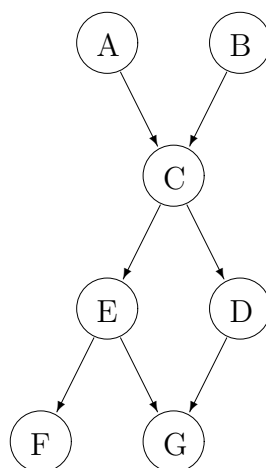


Figura 3.1: *Grafo acíclico direccionado (DAG).*

Consideremos el DAG que aparece en la Figura (3.1). Se tiene que el hecho de introducir probabilidades a priori sobre un modelo es una aportación indeseada de sesgo sobre el mismo. Sin embargo, será necesario especificar las probabilidades a priori sobre A y B , y no por razones matemáticas, sino porque para el razonamiento humano sobre la posible ocurrencia de un suceso se basa en la adjudicación de probabilidades a priori a sucesos anteriores y que pueden influir en nuestro objeto de estudio.

En la definición de las redes bayesianas no nos referimos en ningún momento al concepto de causalidad, y no se requiere que los arcos representen ningún impacto causal. Sin embargo,

en ese caso tendremos la necesidad de comprobar las propiedades de d-separación sobre el modelo y asegurarnos que estas se corresponden con nuestra percepción de las propiedades de la independencia condicional naturales, es decir, que si A y B están d-separadas dada la evidencia e , entonces el cálculo de probabilidad que utilicemos en nuestra red debe ser fiel a que $P(A|e) = P(A|B, e)$. En muchas ocasiones la comprobación de todo lo que acabamos de mencionar resulta tedioso y será muy fácil equivocarse en la creación de la red en la que se representa la dependencia de las distintas variables. Por esa razón supondremos que se utilizan redes causales en todo caso.

Nos centraremos ahora en describir el proceso de construcción de las redes bayesianas. Para ello debemos introducir el concepto de regla de la cadena para redes bayesianas que detallaremos a continuación.

Sea $X = \{X_1, \dots, X_n\}$ un universo de variables. Si tenemos acceso a la tabla de probabilidades conjuntas $P(X) = P(X_1, \dots, X_n)$ entonces podemos calcular también $P(X_i)$, así como $P(X_i|e)$ donde e es una evidencia sobre alguna de las variables en la red bayesiana. Sin embargo, $P(X)$ crece de forma exponencial con el número de variables, y por tanto si X es muy grande la tabla se volverá intratablemente larga. Por tanto, necesitamos buscar una forma de representación más compacta de $P(X)$, es decir, una forma de almacenamiento de información de forma que $P(X)$ pueda ser calculada en caso de necesitarse.

Sea BN una red bayesiana sobre X y sea $P(X)$ su distribución de probabilidad que refleja las propiedades especificadas para BN : las probabilidades condicionadas para una variable dados sus padres en $P(X)$ deben ser especificadas de igual forma que en BN y si las variables A y B están d-separadas en BN dado el conjunto C , entonces A y B son independientes dado C en $P(X)$.

Para distribuciones de probabilidad sobre conjuntos de variables tenemos una ecuación denominada regla de la cadena. Para redes bayesianas esta ecuación tiene una forma especial:

Propiedad 1 (Regla de la cadena (general)). Sea $X = \{X_1, \dots, X_n\}$ un conjunto de variables. Entonces para cualquier distribución de probabilidad $P(X)$ se tiene que

$$P(X) = P(X_n|X_1, \dots, X_{n-1})P(X_{n-1}|X_1, \dots, X_{n-2}) \cdots P(X_2|X_1)P(X_1). \quad (3.1)$$

Propiedad 2 (Regla de la cadena para redes bayesianas). Sea BN una red bayesiana sobre $S = \{X_1, \dots, X_n\}$. Entonces BN define una única distribución de probabilidad conjunta $P(X)$ que viene dada por el producto de todas las tablas de probabilidad condicionada

especificadas en BN:

$$P(X) = \prod_{i=1}^n P(X_i | pa(X_i)) \quad (3.2)$$

donde $pa(X_i)$ son las distintas configuraciones de los padres de X_i , $Pa(X_i)$, en BN, y $P(X)$ refleja las propiedades de BN.

Se tiene que (3.2) implica por tanto que una red bayesiana es una representación compacta de la distribución de probabilidad conjunta. Por tanto, podemos ver las redes bayesianas, además de como una representación de un conjunto de aseveraciones de independencia condicional (útil para la inferencia), como una representación de distribución de probabilidad conjunta (útil para entender cómo se contruyen las redes).

Por otra parte, debemos asumir que en el modelo de las redes bayesianas se verifica la propiedad de Markov: *Conocido el estado del proceso en un momento dado, su comportamiento futuro no depende del pasado. Dicho de otro modo, “dado el presente, el futuro es independiente del pasado”*

En términos de d-separación, la verificación de esta propiedad es equivalente a que el modelo refleje las d-separaciones existentes entre las variables correspondientes. De la anterior definición se sigue que para que una red bayesiana pueda modelar una distribución de probabilidad, debe verificarse que “*cada variable sea condicionalmente independiente de todos sus no-descendientes en el grafo dado el valor de sus padres.*”

En resumen, es importante observar que la topología o estructura de la red no sólo proporciona información sobre las dependencias probabilísticas entre las variables, sino también sobre las independencias condicionales de una variable o conjunto de ellas dada otra u otras variables. Cada variable es independiente de las variables que no son descendientes suyas en el grafo dado el estado de sus variables padre. La inclusión de las relaciones de independencia en la propia estructura del grafo hace de las redes bayesianas una buena herramienta para representar conocimiento de forma compacta (se reduce el número de parámetros necesarios). Además, proporcionan métodos flexibles de razonamiento basados en la propagación de las probabilidades a lo largo de la red de acuerdo con las leyes de la teoría de la probabilidad.

Los objetivos para la construcción de redes bayesianas serán la determinación de la topología de la red, la especificación de probabilidades condicionadas de los nodos con dependencias

directas y el cálculo de estimaciones de cualquier otro valor de probabilidad posible asociado a un evento no observable directamente (o observable bajo un cierto coste). Podemos reducir el proceso a siguientes pasos:

1. Escoger el conjunto de variables.
2. Definir un orden parcial para el conjunto de variables; primero los nodos causales (variables mediante las cuales se proporciona información) y luego los nodos efecto (los eventos que deseamos estimar). Debemos tener en cuenta que si escogemos el orden de forma inapropiada, las consecuencias son graves pues la estructura de la red resultante expande una independencia condicional a lo largo de las variables que realmente es falsa.
3. Mientras queden variables:
 - Escoger siguiente variable X_i y añadir nodo a la RB .
 - Asignar $\text{Pa}(X_i)$ a un conjunto mínimo de nodos presente en la red, de manera que sea satisfecha la propiedad de independencia condicional con los restantes nodos.
 - Elaborar la tabla de probabilidad condicionada de X_i .

Siguiendo este proceso se garantiza la obtención de redes acíclicas, evita la redundancia en la definición de probabilidades y la violación de los axiomas de probabilidad. Sin embargo, aunque aparentemente puede parecer tarea fácil el establecer las uniones entre variables y las direcciones de las mismas, en una gran cantidad de casos puede complicarse bastante. En primer lugar, porque las relaciones causales no son siempre obvias y en segundo lugar porque el concepto de causalidad no se encuentra totalmente definido, pues en algunos casos se dice que se rige por leyes naturales mientras que en otros se asocia a puntos de vista particulares. El razonamiento que haremos será el siguiente: supongamos que tenemos dos variables correladas A y B a las que no podemos asignar una dirección causa-efecto. Supongamos que agentes externos fijan el estado de A . Si esto no influye en la certidumbre de B , entonces A no es una causa de B . Por otra parte, si al realizarlo de forma contraria (para ver si B es causa de A) no obtenemos nada, entonces debemos buscar un evento que pueda tener un impacto causal en ambos A y B . Si C es el candidato, se debe verificar si A y B se vuelven independientes dado C .

Coste de representación

En general, las leyes de la probabilidad permiten establecer diferentes métodos de inferencia. En este caso, el método que se utiliza es el de las probabilidades condicionadas (se calcula el valor de la probabilidad de una variable dados unos valores para algunas variables e independiente del resto de variables), que se construye a partir de la regla del producto:

$$P(X|E) = \alpha \sum_y P(X, e, y)$$

donde el valor α es un factor de normalización que corresponde a factores comunes que hacen que las probabilidades sumen 1. Pero hacer estos procesos de inferencia requiere almacenar y recorrer la distribución de probabilidad conjunta de todas las variables. En el mejor de los casos, suponiendo que las variables son binarias, el coste en espacio y tiempo es del orden $O(2^n)$ siendo n el número de variables y para cualquier problema real estas condiciones son impracticables. Necesitamos por tanto mecanismos que nos simplifiquen el coste del razonamiento.

Sin embargo, como ya hemos comentado, la representación de redes bayesianas nos permite una representación más compacta gracias a la factorización de la distribución conjunta. De hecho, suponiendo que cada nodo de la red tenga como máximo k padres ($k \ll n$), un nodo necesitará 2^k para representar la influencia de sus padres, por lo tanto el espacio necesario será $O(n2^k)$.

Variables continuas

Estamos suponiendo en todo caso que las variables que estamos tratando son discretas. Si existen variables de tipo continuo la estrategia más habitual es discretizarlas antes de construir el modelo estructural. Existen algunos modelos de redes bayesianas con variables continuas, pero están limitados a variables gaussianas relacionadas linealmente. Es posible también efectuar la discretización mientras se construye el grafo de la red, si éste se aprende utilizando el principio MDL como medida de ajuste.

3.2. Redes bayesianas como técnica del *Aprendizaje Automático*

Desde un punto de vista matemático, la propiedad básica de las redes bayesianas es la regla de la cadena: una red bayesiana es una representación compacta de la tabla de probabilidad conjunta sobre su universo. Sin embargo, las redes bayesianas son más que eso: son un tipo de modelo gráfico. La estructura de la red se formula con una semántica muy simple que se denomina causalidad (las redes bayesianas son en particular redes causales). Además, la especificación gráfica también especifica los requerimientos para la parte cuantitativa del modelo (las intensidades - que para este tipo de redes se denominan potenciales- son en este caso las probabilidades condicionadas).

Se sabe que los modelos gráficos son lenguajes de comunicación. Los modelos gráficos pueden ser utilizados para comunicación interpersonal: la especificación gráfica resulta fácil de leer y comprender y ayuda a concentrar nuestra atención. Sin embargo, el siguiente paso en el uso de modelos gráficos tiene que ver con la comunicación con el ordenador, por lo que podemos asociar las redes bayesianas al conjunto de técnicas asociadas al Aprendizaje Automático.

El Aprendizaje Automático o Máquinas de Aprendizaje es una rama de la Inteligencia Artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender. De forma más concreta, se trata de crear programas capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento. En muchas ocasiones el campo de actuación del Aprendizaje Automático se solapa con el de la Estadística, ya que las dos disciplinas se basan en el análisis de datos. Sin embargo, el Aprendizaje Automático se centra más en el estudio de la complejidad computacional de los problemas y puede ser visto como un intento de automatizar algunas partes del método científico mediante métodos matemáticos.

En las redes bayesianas se pretende comunicar un modelo gráfico al ordenador y el ordenador debe ser capaz de procesar el modelo y dar respuesta a diversos requerimientos. Para conseguir esto, la especificación del lenguaje debe ser definido formalmente con una sintaxis y semántica precisa.

Lo primero que debemos tener en cuenta cuando construimos el lenguaje de modelización gráfica es asegurar que se encuentra suficientemente bien definida de tal forma que pueda

ser comunicado a un ordenador. Esto cubre la parte gráfica así como la especificación de los potenciales. Debemos tener en cuenta además el ámbito del lenguaje y la tratabilidad del mismo. Es difícil definir el ámbito del lenguaje para redes bayesianas y en cuanto a la tratabilidad, dependerá del caso, pues, aunque existen algoritmos que sirven para la actualización de probabilidades, esta actividad es NP-dura (es decir, algunos modelos tienen un tiempo de actualización que crece de forma exponencial según se aumentan los nodos). Todas estas cuestiones sobre los algoritmos y el ámbito del lenguaje serán objeto de estudio en secciones posteriores.

3.3. Inferencia sobre redes bayesianas

El razonamiento probabilístico o propagación de probabilidades consiste en propagar los efectos de la evidencia a través de la red para conocer la probabilidad *a posteriori* de las variables. Es decir, se le dan valores a ciertas variables (evidencia), y se obtiene la probabilidad posterior de las demás variables dadas las variables conocidas (el conjunto de variables conocidas puede ser vacío, en este caso se obtienen las probabilidades *a priori*). Por otra parte, las redes bayesianas permiten, además de estimar la probabilidad de ciertos eventos, estimar qué variables de evidencia hay que observar para obtener información útil, hacer análisis de sensibilidad (determinar que variables tienen más peso en las probabilidades de las variables consultadas), explicar al usuario los resultados de una inferencia probabilista, ...

A la hora de hacer inferencia tendremos que tener en cuenta muchos aspectos, como por ejemplo el tipo de grafo y de si se desea obtener la probabilidad de una variable a la vez o de todas (es decir, hacer una única consulta o una consulta múltiple). Dependiendo de en qué situación nos encontremos, nos decantaremos por la inferencia exacta o por el contrario por la aproximada, y dentro de ellas por el algoritmo que más nos convenga.

Denotaremos como B la variable con estados b_1, \dots, b_n sobre la que queremos conocer la distribución y sea \mathbf{e} el conjunto de variables de las que conocemos su valor e_1, \dots, e_r . \mathbf{Y} será el conjunto de variable que no hemos observado Y_1, \dots, Y_m de manera que $\mathbf{X} = \{B\} \cup \mathbf{e} \cup \mathbf{Y}$ es el conjunto total de variables. Nos planteamos por tanto el cálculo de $P(B|\mathbf{e})$.

Inferencia exacta

Hablaremos en primer lugar de dos grandes tipos de inferencia: inferencia por enumeración e inferencia por eliminación de variables, siendo la segunda una mejora de la primera.

Se verifica, sin embargo, que la complejidad de la inferencia exacta es NP-dura en el caso general (redes multiconectadas). Aún así, existen algoritmos específicos cuando la red bayesiana cumple que para cada par de nodos hay un único camino no dirigido (poliárbol), o cuando de forma más general es una estructura sencillamente conectada (árbol), o incluso cuando se trata de redes multiconectadas de pequeño tamaño. Estos algoritmos serán los que se detallarán en la última parte de esta subsección.

Inferencia por enumeración

Cualquier probabilidad condicionada se puede calcular como la suma de todos los posibles casos a partir de la distribución de probabilidad conjunta

$$P(B|\mathbf{e}) = \alpha P(B, \mathbf{e}) = \alpha \sum_y P(B, \mathbf{e}, y).$$

La red bayesiana nos permite factorizar la distribución de probabilidad conjunta y obtener una expresión más fácil de evaluar. Para ver el funcionamiento de la inferencia por enumeración en redes bayesianas introduzcamos en este punto el siguiente ejemplo de red bayesiana: supongamos que queremos determinar la probabilidad de que una persona sufra un infarto. Sabemos que esta probabilidad está determinada por cuatro variables: la práctica de deporte, lo equilibrada de la alimentación, la presión sanguínea y si se es fumador. También sabemos que la presión sanguínea depende directamente del deporte y la alimentación, que son variables independientes, y que el ser fumador es independiente del resto de variables (ver Figura (3.2)).

Supongamos que queremos calcular la probabilidad de ser fumador si se ha tenido infarto y no se hace deporte:

$$P(\text{Fumador} | \text{Infarto} = \text{si}, \text{Deporte} = \text{no}).$$

La distribución de probabilidad conjunta de la red sería:

$$P(D, A, S, F, I) = P(I | S, F)P(F)P(S | D, A)P(D)P(A).$$

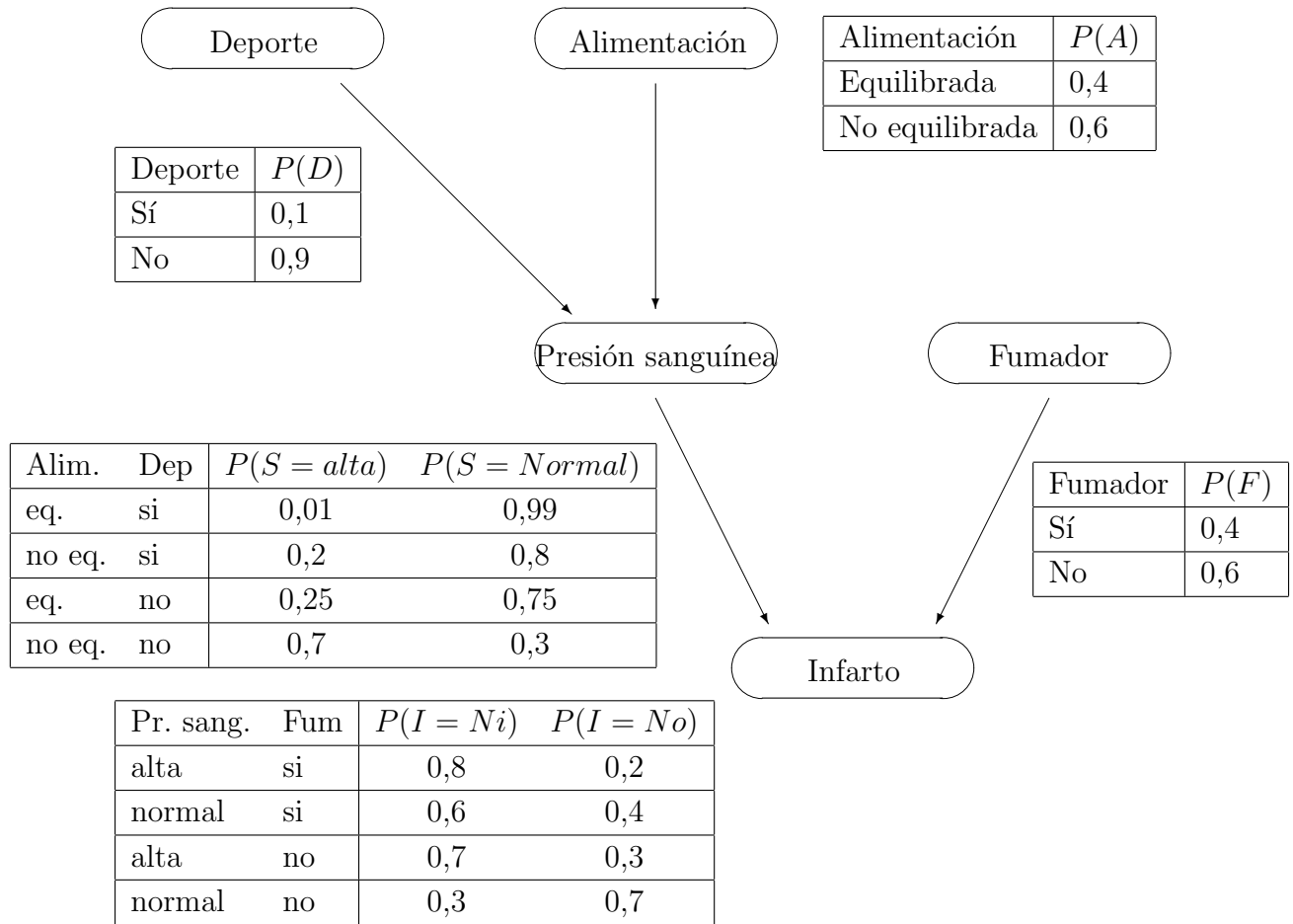


Figura 3.2: Red bayesiana asociada al problema de infarto.

Debemos calcular $P(F | I = si, D = no)$, por lo tanto tenemos:

$$\begin{aligned}
 P(F | I = si, D = no) &= \\
 &= \alpha P(F, I = si, D = no) = \alpha \sum_{A \in \{e, \neg e\}} \sum_{S \in \{a, n\}} P(D = no, A, S, F, I = si) = \\
 &= \alpha P(D = no) P(F) P(I = si) \sum_{A \in \{e, \neg e\}} \sum_{S \in \{a, n\}} P(S | D = no, A) P(I = si | S, F).
 \end{aligned}$$

Si enumeramos todas las posibilidades y las sumamos de acuerdo con la distribución de probabilidad conjunta tenemos que:

$$\begin{aligned}
 P(F | I = si, D = no) &= \\
 &= < 0,9 \cdot 0,4 \cdot (0,4 \cdot (0,25 \cdot 0,8 + 0,75 \cdot 0,6) + 0,6 \cdot (0,7 \cdot 0,8 + 0,3 \cdot 0,6)), \\
 &= 0,9 \cdot 0,6 \cdot (0,4 \cdot (0,25 \cdot 0,7 + 0,75 \cdot 0,3) + 0,6 \cdot (0,7 \cdot 0,7 + 0,3 \cdot 0,3)) > \\
 &= < 0,48, 0,52 > .
 \end{aligned}$$

Es decir, la probabilidad de que sea fumador es 0,48 contra probabilidad 0,52 de que no lo sea.

Podríamos incluir aquí la implementación del algoritmo, pero no vale la pena pues se trata de un algoritmo ineficiente: se repite el cálculo de probabilidades intermedias en el proceso. Una mejora es el algoritmo de eliminación de variables.

Inferencia por eliminación de variables

El algoritmo de eliminación de variables intenta evitar la repetición de cálculos que realiza la inferencia por enumeración. El algoritmo utiliza técnicas de programación dinámica de manera que se guardan cálculos intermedios para cada variable para reutilizarlos (factores) y el cálculo de la probabilidad requerida se realiza evaluando la expresión de la distribución de probabilidad conjunta de izquierda a derecha, acumulando los factores según se necesita.

La ventaja de este algoritmo es que las variables no relevantes desaparecen al ser factores constantes. Dada la variable B y el conjunto \mathbf{e} , una variable Y será irrelevante si no pertenece a los antecesores de B y \mathbf{e} .

La complejidad del algoritmo de eliminación de variables depende del tamaño del mayor factor, que depende del orden en el que se evalúan las variables y la topología de la red (el

orden de evaluación que escogeremos será el topológico según el grafo).

Algoritmo de eliminación de variables:

1. Entrada:
 - Una variable aleatoria B de consulta, un conjunto de valores observados \mathbf{e} para las variables de evidencia y una red bayesiana.
2. Sea REDE el resultado de eliminar de la red las variables irrelevantes para la consulta realizada.
3. Sea FACTORES igual a vacío.
4. Sea VARIABLES el conjunto de variables de REDE.
5. Sea VARORD el conjunto de VARIABLES ordenado según un orden de eliminación.
6. Para cada VAR en VARORD hacer:
 - Sea FACTOR el factor correspondiente a VAR (respecto de \mathbf{e}).
 - Añadir FACTOR a FACTORES.
 - Si VAR es una variable oculta hacer FACTORES=PRODUCTOYSUMA(VAR, FACTORES) donde la función PRODUCTOYSUMA se encarga de multiplicar los factores y sumarlos respecto a la variable oculta.
7. Devolver NORMALIZA(MULTIPLICA(FACTORES)) donde NORMALIZA es una función que normaliza el resultado.
8. Salida: $P(B | \mathbf{e})$.

Propagación en árboles

Supongamos entonces que estamos ante una estructura gráfica de árbol o poliárbol. Una de las ventajas de disponer de una estructura gráfica sobre relaciones entre las variables es que se puede utilizar esta información para reducir el número de operaciones necesarias de cara a obtener las probabilidades a posteriori. Existen varios métodos computacionales que aprovechan la estructura gráfica para propagar los efectos que las observaciones del

mundo real tienen sobre el resto de las variables de la red. Las diferencias entre ellos radican principalmente en la precisión de los resultados y en el consumo de recursos durante el tiempo de ejecución. Los algoritmos de propagación se dividen inicialmente en “exactos” o “aproximados” según cómo calculen los valores de las probabilidades. Los métodos exactos calculan los valores por medio del teorema de Bayes mientras que los métodos aproximados utilizan técnicas iterativas de muestreo en las que los valores se aproximarán más o menos a los exactos dependiendo del punto en que se detenga el proceso.

El algoritmo que detallamos a continuación se aplica a estructuras de tipo árbol y puede ser extendido muy fácilmente a poliárboles.

Supongamos dado el árbol que aparece en la Figura (3.3). Dada cierta evidencia e , representada por la instanciación de ciertas variables, la probabilidad posterior de la variable B , por el teorema de Bayes:

$$P(b_i | E) = \frac{P(e | b_i)P(b_i)}{P(e)}.$$

Debido a que la estructura de la red es un árbol, el nodo B la separa en dos subárboles; de esta manera podemos dividir la evidencia en dos grupos (ver Figura (3.3)):

- e^+ : Datos en el árbol que cuya raíz es B .
- e^- : Datos en el resto del árbol.

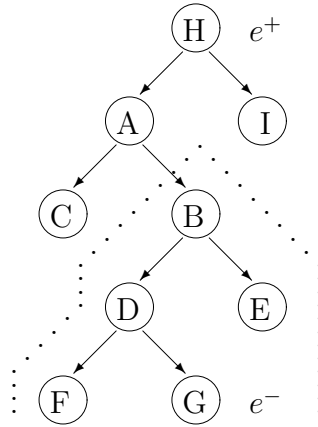


Figura 3.3: *Propagación en árboles. En un árbol, cualquier nodo (B) divide la red en dos subgrafos condicionalmente independientes, e^+ y e^- .*

Entonces:

$$P(b_i | e) = \frac{P(b_i)P(e^-, e^+ | b_i)}{P(e)}.$$

Pero, dado que ambos son independientes y aplicando nuevamente Bayes:

$$P(b_i | e) = \alpha P(b_i | e^+) P(e^- | b_i),$$

donde α es una constante de normalización. Si definimos los siguientes términos:

$$\lambda(b_i) = P(e^- | b_i),$$

$$\pi(b_i) = P(b_i | e^+),$$

entonces:

$$P(b_i | e) = \alpha \pi(b_i) \lambda(b_i). \quad (3.3)$$

En base a la ecuación (3.3), se puede integrar un algoritmo distribuido para obtener la probabilidad de un nodo dada cierta evidencia. Para ello, se descompone el cálculo en dos partes:

- i) Evidencia de los hijos (λ).
- ii) Evidencia de los demás nodos (π).

Cada nodo guarda los valores de los vectores π y λ , así como las matrices de probabilidad P . La propagación se hace por un mecanismo de paso de mensajes, en donde cada nodo envía los mensajes correspondientes a su padre e hijos:

- Mensaje al padre; nodo B a su padre A :

$$\lambda_B(a_i) = \sum_j P(b_j | a_i) \lambda(b_j).$$

- Mensaje a los hijos; nodo B a su hijo S_k :

$$\pi_k(b_i) = \alpha \pi(b_j) \prod_{i \neq k} \lambda_i(b_j).$$

Al instanciarse ciertos nodos, éstos envían mensajes a sus padres e hijos, y se propagan hasta llegar a la raíz u hojas, o hasta encontrar un nodo instanciado. Al final de la propagación, cada nodo tiene un vector π y un vector λ . Entonces se obtiene la probabilidad simplemente multiplicando ambos (término por término) de acuerdo a la ecuación (3.3).

Algoritmo

1. Inicialización:

- A. Inicializar todos los λ – mensajes y λ – valores a uno.
- B. Si la raíz A tiene m posibles valores entonces para $j = 1, \dots, m$ hacemos $\pi(a_j) = P(a_j)$.
- C. Para todos los hijos B de la raíz A , enviar un nuevo π – mensaje a B .

(En ese momento comenzará un flujo de propagación debido al procedimiento de actualización C).

Cuando una variable se instancia o una variable recibe un λ o π – mensaje, se usa uno de los siguientes procedimientos de actualización.

2. Actualización:

- A. Si una variable B se instancia a un valor b_j , entonces:
 - A.1. Inicializar $P^*(b_j) = 1$ y $P^*(b_i) = 0$, para todo $i \neq j$.
 - A.2. Calcular $\lambda(B)$.
 - A.3. Enviar un nuevo λ – mensaje al padre de B .
 - A.4. Enviar nuevos π – mensaje a los hijos de B .
- B. Si una variable B recibe un nuevo λ – mensaje de uno de sus hijos y la variable B no ha sido instanciada todavía, entonces,
 - B.1. Calcular el nuevo valor de $\lambda(B)$.
 - B.2. Calcular el nuevo valor de $P^*(B)$.
 - B.3. Enviar un nuevo λ – mensaje al padre de B .
 - B.4. Enviar nuevos π – mensaje a los hijos de B .
- C. Si una variable B recibe un nuevo π – mensaje de su padre y la variable B no ha sido instanciada todavía, entonces,
 - C.1. Calcular el nuevo valor de $\pi(B)$.
 - C.2. Calcular el nuevo valor de $P^*(B)$.
 - C.3. Enviar nuevos π – mensajes a los hijos de B .

Para ilustrar el algoritmo de propagación de probabilidades, vamos a utilizar el siguiente ejemplo:

Ejemplo:

Supongamos que un señor piensa que su esposa le está siendo infiel. La red bayesiana que se construye para evaluar esta posibilidad puede verse en la Figura (3.4); en ella la variable A se refiere a si la esposa está engañando al marido o no, la variable B se refiere a si la esposa cena con otro hombre o no, la variable C se refiere a si la esposa es vista cenando con otro hombre o no, y la variable D se refiere a si en el domicilio se reciben llamadas telefónicas sospechosas o no. Supondremos que la letra minúscula con el subíndice uno representa a la afirmación del hecho, y la minúscula con el subíndice dos, a la negación.

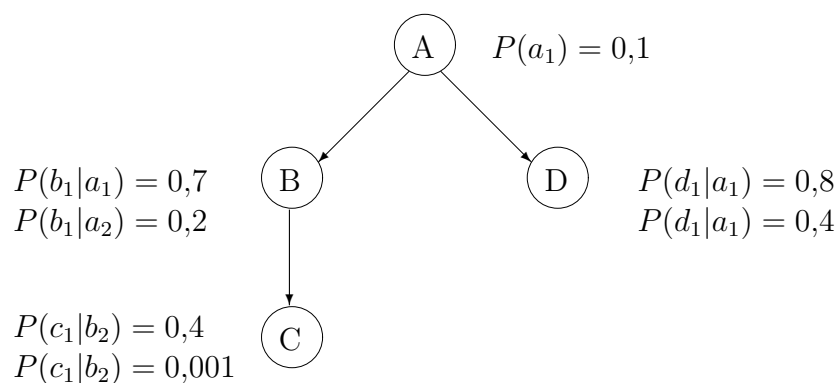


Figura 3.4: Red bayesiana asociada al problema de la infidelidad.

En primer lugar, vamos a calcular las probabilidades a priori de cada una de las variables de la red:

Inicialización:

- A. Ponemos todos los λ – mensajes y λ – valores a 1.
- B. Hacemos $\pi(a_j) = P(a_j)$, para $j = 1, 2$: $\pi(A) = (0,1, 0,9)$.
- C. A envía un mensaje a su hijo, B ,

$$\begin{aligned}\pi_B(a_1) &= \pi(a_1)\lambda_D(a_1) = 0,1, \\ \pi_B(a_2) &= \pi(a_2)\lambda_D(a_2) = 0,9.\end{aligned}$$

B toma entonces nuevos π – valores:

$$\begin{aligned}\pi(b_1) &= P(b_1|a_1)\pi_B(a_1) + P(b_1|a_2)\pi_B(a_2) = 0,7 \cdot 0,1 + 0,2 \cdot 0,9 = 0,75, \\ \pi(b_2) &= P(b_2|a_1)\pi_B(a_1) + P(b_2|a_2)\pi_B(a_2) = 0,25.\end{aligned}$$

Y con ellos y con los λ – valores de B , se obtienen las probabilidades:

$$\begin{aligned}P(b_1) &= \alpha \cdot 0,25 \cdot 1 = 0,25, \\ P(b_2) &= \alpha \cdot 0,75 \cdot 1 = 0,75.\end{aligned}$$

Ahora, C recibe un π – mensaje por ser hijo de B :

$$\begin{aligned}\pi_C(b_1) &= \pi(b_1) = 0,25, \\ \pi_C(b_2) &= \pi(b_2) = 0,75.\end{aligned}$$

Y actualiza su π – valor:

$$\begin{aligned}\pi(c_1) &= P(c_1|a_1)\pi_C(a_1) + P(c_1|a_2)\pi_C(a_2) = 0,4 \cdot 0,25 + 0,001 \cdot 0,75 = 0,10075, \\ \pi(c_2) &= P(c_2|a_1)\pi_C(a_1) + P(c_2|a_2)\pi_C(a_2) = 0,89925.\end{aligned}$$

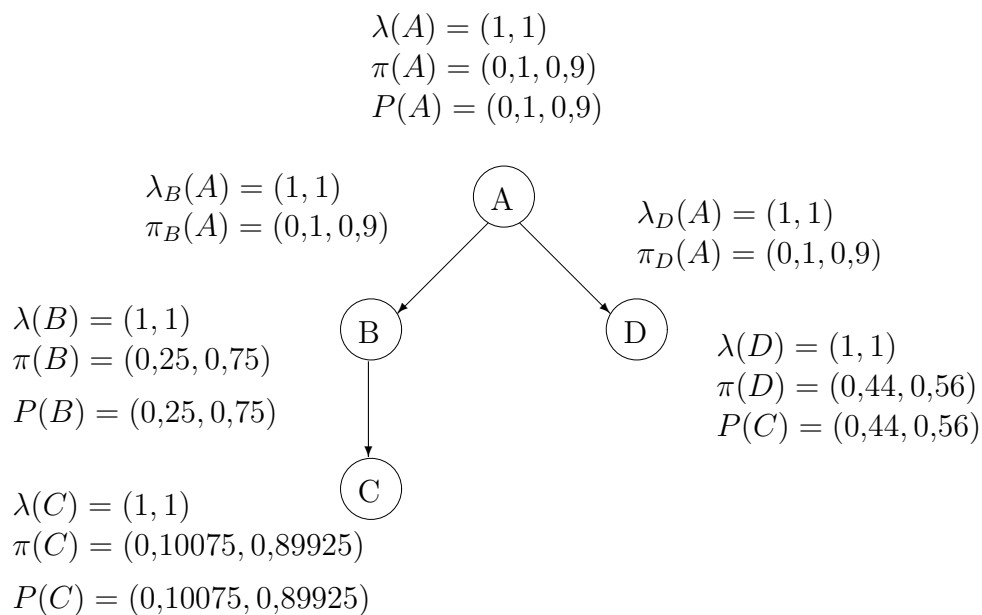
A partir de ellos, calculamos las probabilidades de C , multiplicando por los λ – valores y normalizando:

$$\begin{aligned}P(c_1) &= 0,10075, \\ P(c_2) &= 0,89925.\end{aligned}$$

El mismo procedimiento se repite para D , y obtenemos el estado inicial S_0 de la red causal que viene dada por la Figura (3.5).

Supongamos ahora que nos informan de que la esposa ha cenado con otro, es decir, conocemos ahora con certeza que $B = b_1$. Esta información se irá transmitiendo por la red, haciendo que las probabilidades a priori de los nodos, $P(X)$, cambien a las probabilidades a posteriori, $P^*(X) = P(X|B = b_1)$. En este caso, al ser la evidencia aportada a favor de la hipótesis que queremos probar, lo lógico será que todas estas probabilidades aumenten. En el momento que una variable se actualiza, comienza un flujo de propagación por la red, que en este caso es el siguiente:

- B informa a su padre mediante un λ – mensaje.

Figura 3.5: Red bayesiana asociada al estado S_0

- B informa a su hijo mediante un π – mensaje.
- A su vez, A va a informar a su hijo, D , mediante un π – mensaje.

Tras el paso de estos mensajes, todas las variables van a actualizar sus λ y π – valores y sus probabilidades.

Veamos entonces cómo se efectúa la actualización con el algoritmo:

Actualización:

Actualización de B :

- A.1. Calculamos ahora la probabilidad a posteriori de B , conocido que ha tomado el valor b_1 , que evidentemente será:

$$\begin{aligned}
 P^*(b_1) &= 1, \\
 P^*(b_2) &= 0.
 \end{aligned}$$

A.2. Calculamos $\lambda(B)$:

$$\begin{aligned}\lambda(b_1) &= 1, \\ \lambda(b_2) &= 0.\end{aligned}$$

A.3. Enviamos un λ – mensaje al padre de B , A :

$$\begin{aligned}\lambda_B(a_1) &= P(b_1 | a_1)\lambda(b_1) + P(b_2 | a_1)\lambda(b_2) = 0,7 \cdot 0,1 + 0,3 \cdot 0 = 0,7, \\ \lambda_B(a_2) &= 0,3.\end{aligned}$$

A.4. Enviamos un π – mensaje al hijo de B , C :

$$\begin{aligned}\pi_C(b_1) &= 1 \text{ puesto que } B \text{ ha sido instanciada a } b_1, \\ \pi_C(b_2) &= 0 \text{ puesto que } B \text{ ha sido instanciada a } b_1.\end{aligned}$$

Ahora, al haber recibido A y C nuevos mensajes, tienen que actualizar sus valores.

Actualización de C :

Al recibir C un π – mensaje, se dispara el procedimiento de actualización C :

C.1. El π – valor de C cambia,

$$\begin{aligned}\pi(c_1) &= P(c_1 | b_1)\pi_C(b_1) + P(c_1 | b_2)\pi_C(b_2) = 0,4, \\ \pi(c_2) &= 0,6.\end{aligned}$$

C.2. Calculamos la nueva probabilidad de C :

$$\begin{aligned}P^*(c_1) &= 0,4\alpha = 0,4, \\ P^*(c_2) &= 0,6\alpha = 0,6.\end{aligned}$$

C.3. No es necesario puesto que C no tiene hijos.

Actualización de A :

Al recibir A un λ – mensaje, se dispara el procedimiento de actualización B ;

B.1. Actualizamos el λ – valor:

$$\begin{aligned}\lambda(a_1) &= \lambda_B(a_1)\lambda_D(a_1) = 0,7, \\ \lambda(a_2) &= \lambda_B(a_2)\lambda_D(a_2) = 0,2.\end{aligned}$$

B.2. En base al λ – valor, calculamos la probabilidad a posteriori:

$$\begin{aligned}P^*(a_1) &= \alpha \cdot 0,7 \cdot 0,1 = \alpha \cdot 0,07 = 0,28, \\ P^*(a_2) &= \alpha \cdot 0,2 \cdot 0,9 = \alpha \cdot 0,18 = 0,72.\end{aligned}$$

Como era de esperar, la probabilidad de que el marido sea infiel ha aumentado, porque la evidencia aportada es a favor de la hipótesis.

B.3. A no tiene padre.

B.4. A envía un π – mensaje a su hijo, D ,

$$\begin{aligned}\pi_D(a_1) &= \pi(a_1)\lambda_B(a_1) = 0,1 \cdot 0,7 = 0,07, \\ \pi_D(a_2) &= \pi(a_2)\lambda_B(a_2) = 0,0 \cdot 0,2 = 0,18.\end{aligned}$$

Actualización de D :

Ahora la variable D , debido a la recepción del π – mensaje, comienza el proceso de actualización C.

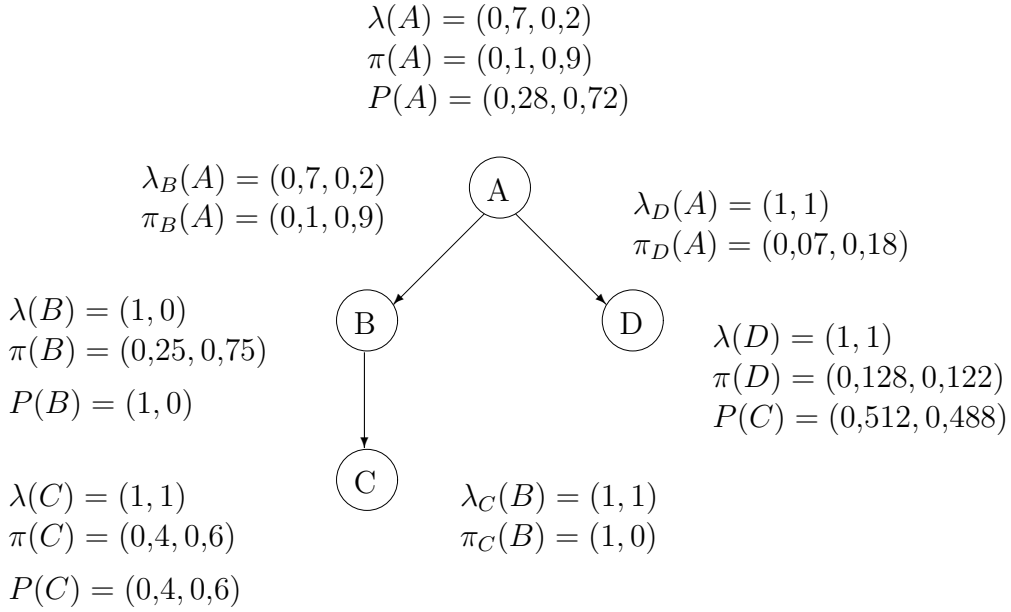
C.1. El π – valor de D cambia,

$$\begin{aligned}\pi(d_1) &= 0,128, \\ \pi(d_2) &= 0,122.\end{aligned}$$

C.2. Calculamos la nueva probabilidad de D :

$$\begin{aligned}P^*(d_1) &= 0,512, \\ P^*(d_2) &= 0,488.\end{aligned}$$

C.3. No es necesario puesto que D no tiene hijos.

Figura 3.6: Red bayesiana asociada al estado S_1

Así, tras la instanciación de B a b_1 , la red queda como aparece en la Figura (3.6).

Supongamos ahora que tenemos la información de que no se han recibido llamadas telefónicas extrañas en el domicilio, es decir, que sabemos que D ha tomado el valor d_2 . Nuevamente se iniciará el algoritmo que propagará esta información por la red:

- D enviará un λ – mensaje a su padre, A .
- A enviará un π – mensaje a su hijo, B .

Pero ahora, al estar B inicializada, el algoritmo se parará ahí, puesto que $P(B) = (1,0)$, y no podemos permitir que nada cambie ya estos valores. Así, en la ejecución del algoritmo, las variables que ya han sido inicializadas son extremos muertos, donde la propagación se para (en el caso de la propagación en árboles).

Hacemos pues el paso A de actualización para la variable D .

Actualización:

Actualización de D :

A.1. Calculamos ahora la probabilidad a posteriori de D :

$$\begin{aligned} P^*(d_1) &= 0, \\ P^*(d_2) &= 1. \end{aligned}$$

A.2. Calculamos $\lambda(D)$:

$$\begin{aligned} \lambda(d_1) &= 0, \\ \lambda(d_2) &= 1. \end{aligned}$$

A.3. Enviamos un λ – mensaje, al padre de D , A :

$$\begin{aligned} \lambda(a_1) &= P(d_1 | a_1) \lambda(d_1) + P(d_2 | a_1) \lambda(d_2) = 0,7 \cdot 0 + 0,2 \cdot 1 = 0,2, \\ \lambda(a_2) &= 0,6. \end{aligned}$$

A.4. No se hace puesto que D no tiene hijos.

Actualización de A :

B.1.

$$\begin{aligned} \lambda(a_1) &= \lambda_B(a_1) \lambda_C(a_1) = 0,7 \cdot 0,2 = 0,14, \\ \lambda(a_2) &= \lambda_B(a_2) \lambda_C(a_2) = 0,2 \cdot 0,6 = 0,12. \end{aligned}$$

B.2.

$$\begin{aligned} P^*(a_1) &= \alpha 0,014 = 0,1148, \\ P^*(a_2) &= \alpha 0,108 = 0,8852. \end{aligned}$$

Ahora la probabilidad de a_1 se ha reducido, puesto que la evidencia aportada es en contra de a_1 .

B.3. A no tiene padres.

B.4. Este paso no se realiza pues B está ya instanciado.

Tras estos cálculos se obtiene un estado de la red, S_2 . Este estado es el mismo que tendríamos si procesásemos la información al revés, es decir, si instanciásemos primero la variable D al valor d_2 , y después la variable B al valor b_1 .

Propagación en redes multiconectadas: árbol de uniones

El algoritmo general más común en redes bayesianas es el de agrupamiento o “árbol de uniones” (*junction tree*). Este algoritmo es una versión del método de eliminación de variables. El método de agrupamiento consiste en transformar la estructura de la red para obtener un árbol de uniones (grupos de nodos), mediante la agrupación de nodos aplicando teoría de grafos. El procedimiento es el que aparece a continuación:

1. Eliminar la direccionalidad de los arcos.
2. Ordenar los nodos por máxima cardinalidad.
3. Moralizar el grafo (insertar un arco entre todos los nodos con hijos comunes).
4. Triangular el grafo.
5. Obtener los cliques y ordenar. (Definiremos a continuación qué es un clique)
6. Construir árbol de cliques.

Una vez transformado el grafo, la propagación se produce mediante el envío de mensajes a lo largo árbol de uniones o cliques (de forma similar a cómo se realiza en los árboles). Inicialmente se calcula la probabilidad conjunta (potencial) de cada clique, y la condicional dado el padre. Dada cierta evidencia se recalculan las probabilidades de cada clique. La probabilidad individual de cada variable se obtiene de la del clique por marginalización.

Vamos a ver ahora como llevar a cabo los pasos 4-6:

Notación: Sea B un nodo en un grafo no dirigido. El conjunto de vecinos de B lo denotaremos por $nb(B)$, y el conjunto de vecinos incluyendo a B lo denotaremos por $fa(B)$, la

familia de B . Si los nodos están numerados, escribiremos N_i mejor que N_{B_i} . Los nodos con un conjunto completo de vecinos se llaman *nodos simplificados*. Un vecino del nodo B se dice que es *adyacente* a B . Notemos que B es un nodo simplificado si y sólo si $fa(B)$ es un clique.

Definición 1 (Clique). *Un clique es un subconjunto máximo de nodos completamente conectados, de forma que hay un arco entre cada par de nodos, y no existe un conjunto completamente conectado del que éste sea subconjunto.*

Definición 2 (Grafo triagulado). Un grafo no dirigido con un perfecto orden de eliminación se llama grafo triagulado.

Un nodo se puede eliminar insertando un arco entre cada par de sus nodos vecinos no eliminados.

Definición 3 (Orden de eliminación perfecto). Un orden de eliminación es perfecto si todos los nodos pueden ser eliminados sin que la secuencia inserte un arco entre un par de variables no eliminadas.

Hay distintos órdenes de eliminación, y muchos de ellos producen diferentes grafos triagulados. Nosotros proponemos el que produce el dominio más pequeño:

Sea ν un conjunto de variables. Para $B \in \nu$, $|sp(B)|$ denota el número de estados de B . El tamaño de ν , $sz(\nu)$, es el producto $\prod_{B \in \nu} |sp(B)|$. Sea BN una red bayesiana, sea G el grafo triagulado extendido al moralizar la red BN , y sean V_1, \dots, V_n los cliques de G . El tamaño de G es la suma $size(G) = \sum_i sz(V_i)$.

Desafortunadamente, es un problema NP-duro determinar un orden de eliminación que proporcione una triangulación de mínimo tamaño. Sin embargo, hay algoritmos heurísticos que dan resultados bastante buenos. Por ejemplo:

Algoritmo:

Eliminar repetidamente un nodo simplificado, y si esto no es posible, eliminar un nodo B del $sz(fa(B))$ mínimo.

Se verifican los siguientes resultados:

Proposición: 1. *Sea G un grafo triangulado, y sea B un nodo simplificado. Sea G' el grafo resultante de eliminar B de G . Entonces G' es un grafo triangulado.*

Proposición: 2. *Un grafo triangulado con al menos dos nodos tiene al menos dos nodos simplificados.*

En general, es un problema NP-duro determinar el conjunto de cliques en un grafo. Para grafos triangulados, las proposiciones anteriores nos dan un procedimiento:

Algoritmo

1. Eliminar un nodo simplificado B ; $fa(B)$ es un clique candidato.
2. Si $fa(B)$ no incluye todos los nodos restantes, ir a 1.
3. Reducir el conjunto de cliques candidatos por supresión de los conjuntos que son subconjuntos de otros cliques candidatos.

En el peor caso, la propagación en redes bayesianas es un problema NP-duro. En la práctica, en muchas aplicaciones se tienen redes muy grandes (en función del mayor clique). Para redes muy complejas (muchas conexiones), la mejor alternativa son técnicas de simulación estocástica o técnicas aproximadas.

Inferencia aproximada

Como ya hemos visto en el caso de redes multiconectadas, el problema se convierte en un problema NP-duro en muchos casos. Por ese motivo será necesario aplicar en dichos casos métodos de aproximación en lugar de métodos exactos.

Simulación estocástica

Se asignan valores aleatorios a las variables no instanciadas, se calcula la distribución de probabilidad y se obtienen valores de cada variable obteniéndose una muestra; se repite el procedimiento para obtener un número apreciable de muestras y en base al número de

ocurrencias de cada valor se determina la probabilidad de dicha variable.

El proceso sería el siguiente: si queremos estimar la probabilidad $P(A = a_i | B = b_j)$, realizamos un gran número de muestras aleatorias y contamos el número de ocurrencias de los sucesos:

- N_C : número de muestras en las que ocurre $B = b_j$.
- N_S : número de muestras en las que ocurre $A = a_i$ y $B = b_j$.
- N : número de muestras aleatorias.

Entonces, si N es suficientemente grande,

- N_C/N es una buena estimación de $P(B = b_j)$.
- N_S/N es una buena estimación de $P(A = a_i, B = b_j)$.

Dado que $P(A = a_i | b_j) = \frac{P((A=a_i) \cap B=b_j)}{P(B=b_j)}$, entonces $\frac{N_S}{N_C}$ puede ser una buena estimación de $P(A = a_i | b_j)$.

3.4. Aprendizaje de redes bayesianas

El aprendizaje en la redes bayesianas consiste en definir la red probabilística a partir de datos almacenados en bases de datos en lugar de obtener el conocimiento del experto. Este tipo de aprendizaje ofrece la posibilidad de inducir la estructura gráfica de la red a partir de los datos observados y de definir las relaciones entre los nodos basándose también en dichos casos.

Más concretamente, el problema del aprendizaje bayesiano puede describirse informalmente como: dado un conjunto de entrenamiento $D = \{u_1, u_2, \dots, u_N\}$ de instancias de U , encuéntrase la red BN que se ajuste mejor a D . Típicamente, este problema se divide en dos partes:

- **Aprendizaje estructural** : obtiene la estructura de la red bayesiana a partir de bases de datos, es decir, las relaciones de dependencia e independencia entre las variables involucradas. Otra alternativa es combinar conocimiento subjetivo del experto

con aprendizaje, para lo cual se parte de la estructura dada por el experto y se la valida y mejora utilizando datos estadísticos. Las técnicas de aprendizaje estructural dependen del tipo de estructura o topología de la red (árboles, poliárboles o redes multiconectadas).

- **Aprendizaje paramétrico** : conocida la estructura del grafo, obtener las probabilidades correspondientes a cada nodo.

3.4.1. Aprendizaje estructural

El aprendizaje estructural consiste en encontrar las relaciones de dependencia e independencia entre las variables involucradas, de forma que se pueda determinar la topología o estructura de la red bayesiana. De acuerdo al tipo de estructura, podemos dividir los métodos de aprendizaje estructural en:

- Aprendizaje de árboles.
- Aprendizaje de poliárboles.
- Aprendizaje de redes multiconectadas.

Para el caso más general, que es el de redes multiconectadas, existen dos clases de métodos:

- Métodos basados en pruebas de independencia.
- Métodos basados en medidas de ajuste y búsqueda.

A continuación veremos el método para aprendizaje de árboles y su extensión a poliárboles, para después ver los dos enfoques para aprender redes multiconectadas.

Aprendizaje de árboles

El aprendizaje de árboles se basa en el algoritmo desarrollado por Chow y Liu para aproximar una distribución de probabilidad por un producto de probabilidades de segundo orden (árbol). La probabilidad conjunta de n variables se puede representar como:

$$P(X_1, \dots, X_n) = \prod_{i=1}^n P(X_i | pa(X_i)),$$

con $pa(X_i)$ cualquier combinación de $Pa(X_i)$.

Para obtener el árbol se plantea el problema como un problema de optimización: obtener la estructura del árbol que más se aproxime a la distribución “real”. Esto se basa en una medida de la diferencia de información entre la distribución real (P) y la aproximada (P^*):

$$DI(P, P^*) = \sum_X P(X) \log \frac{P(X)}{P^*(X)}.$$

El objetivo es minimizar $DI(\cdot, \cdot)$. Se puede definir dicha diferencia en función de la información mutua entre pares de variables, que se define como:

$$I(X_i, X_j) = \sum_{X_i, X_j} P(X_i, X_j) \log \frac{P(X_i, X_j)}{P(X_i)P(X_j)}.$$

Se puede demostrar que la diferencia de información es una función del negativo de la suma de las informaciones mutuas (pesos) de todos los pares de variables que constituyen el árbol. Entonces, encontrar el árbol más próximo equivale a encontrar el árbol con mayor peso. Podemos entonces encontrar el árbol óptimo mediante el siguiente algoritmo, que es equivalente al conocido problema del árbol de expansión de peso máximo:

1. Calcular la información mutua entre todos los pares de variables (que para n variables, son $n(\frac{n-1}{2})$).
2. Ordenar las informaciones mutuas de mayor a menor.
3. Seleccionar la rama de mayor valor como árbol inicial.
4. Agregar la siguiente rama mientras no forme un ciclo; en otro caso desechar.
5. Repetir 4 hasta que se cubran todas las variables ($n - 1$ ramas).

El algoritmo no provee la dirección de los arcos, por lo que ésta se puede asignar de forma arbitraria o utilizando semántica externa (experto).

Ejemplo:

Para ilustrar este algoritmo consideremos el clásico ejemplo dle jugador de golf, en el cual se tienen cuatro ariables: juega, ambiente, humedad y temperatura. Obtenemos entonces

Var. 1	Var. 2	Info. mutua
temp.	ambiente	0.2856
juega	ambiente	0.0743
juega	humedad	0.0456
juega	viento	0.0074
humedad	ambiente	0.0060
viento	temp.	0.0052
viento	ambiente	0.0017
juega	temp.	0.0003
humedad	temp.	0
viento	humedad	0

Cuadro 3.1: Tabla correspondiente al ejemplo de un jugador de golf.

las informaciones mutuas de cada par de variables, que se muestran en la Tabla (3.1).

Seleccionamos las cuatro primeras ramas y generamos el árbol que se muestra en la Figura (3.7), donde se toma la variable “juega” como raíz.

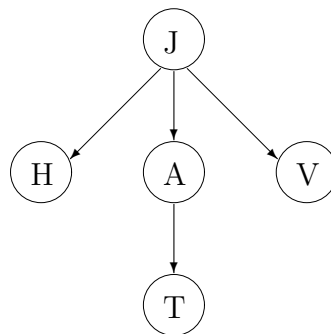


Figura 3.7: Árbol obtenido mediante el algoritmo de aprendizaje de árboles.

Aprendizaje de poliárboles

Una forma de darle direcciones al “esqueleto” aprendido con el algoritmo de Chow y Liu es mediante pruebas de independencia, no sólo entre dos variables, sino entre grupos de tres variables o tripletas. Mediante este esquema se genera un algoritmo que aprende poliárboles, ya que al asignar las direcciones puede ser que la estructura generada sea un árbol o

un poliárbol (en realidad, un árbol es un caso especial de poliárbol).

El algoritmo parte del esqueleto (estructura sin direcciones) obtenido con el algoritmo de Chow y Liu. Después se determinan las direcciones de los arcos utilizando pruebas de dependencia entre tripletas.

Notación: $I(A, B, \chi)$ denota que A es d-separada de B dado χ ; pondremos $I(A, B)$ cuando $\chi = \emptyset$, y si χ consta sólo de un elemento C , pondremos $I(A, B, C)$.

Direccionar los arcos:

Regla 1:[introducción de v-estructuras]: Si hay tres nodos, A, B, C , tal que $A - C$ y $B - C$, pero no $A - B$, entonces introducimos la v-estructura $A \rightarrow C \leftarrow B$ si $I(A, B)$ o existe un nodo V distinto de C tal que $I(A, B, V)$.

Ahora daremos dirección a los restantes arcos usando las siguientes reglas:

Regla 2:[evitar nuevas v-estructuras]: Cuando la regla 1 se agotó, y se tiene $A \rightarrow C - B$ (y no hay un arco entre A y B), entonces hacer $C \rightarrow B$.

Regla 2:[elegir aleatoriamente]: Si ninguna de las reglas anteriores se puede aplicar, tomar un arco no dirigido y darle una dirección arbitrariamente.

Aprendizaje de redes:

Como comentamos previamente, existen dos clases de métodos para el aprendizaje genérico de redes bayesianas, que incluyen redes multiconectadas. Estos son:

1. Métodos basados en pruebas de independencia.
2. Métodos basados en medidas de ajuste y búsqueda.

A continuación veremos ambos enfoques.

Aprendizaje basado en pruebas de independencia

Este enfoque se basa en medidas de dependencia local entre subconjuntos de variables. El caso más sencillo es el algoritmo de Chow y Liu, en el cual se mide la información mutua entre pares de variables. A partir de estas medidas, como se vio previamente, se genera una red bayesiana en forma de árbol. Analizando dependencias entre tripletas de variables el método se extiende a poliárboles.

Este enfoque se puede generalizar para el aprendizaje de redes multiconectadas, haciendo pruebas de dependencia entre subconjuntos de variables.

En primer lugar determinaremos el esqueleto y luego daremos dirección a los arcos.

Determinar el esqueleto

Para determinar el esqueleto utilizaremos el algoritmo PC. Pero antes vamos a ver un teorema encaminado a reducir el número de tests de independencia a realizar, ya que estos tienen un coste.

Teorema 1. *Los nodos A y B no están unidos si y sólo si $I(A, B, pa(A))$ o $I(A, B, pa(B))$.*

El teorema asegura que es suficiente con preguntarnos si $I(A, B, \chi)$, donde χ es un subconjunto de los vecinos de A o de B . Este hecho se utiliza en el algoritmo PC.

Algoritmo PC

1. Empezar con el grafo completo.
2. $i = 0$;
3. Mientras un nodo tenga $i + 1$ vecinos;
 - para todos los nodos A con al menos $i + 1$ vecinos;
 - para todos los vecinos B de A ;
 - para todos los conjuntos vecinos χ tal que $|\chi| = i$ y $\chi \subseteq (nb(A) \setminus \{B\})$;

- ◊ si $I(A, B, \chi)$ entonces borrar el arco $A - B$ y guardar $I(A, B, \chi)$.
- $i = i + 1$

Este algoritmo tiene las siguientes propiedades::

Propiedad 1. *Si el conjunto de casos es una muestra fiel de una red bayesiana, entonces el grafo resultante del algoritmo PC es el esqueleto de la red.*

Propiedad 2. *Las independencias condicionales encontradas por el algoritmo PC son suficientes para determinar las v-estructuras.*

Sea $A - C - B$ una cadena y supongamos que el algoritmo PC encontró $I(A, B, \chi)$. Sabemos que los dos arcos son parte del esqueleto, y $C \notin \chi$ entonces la única forma de dirigir los arcos será introducir la v-estructura $A \rightarrow C \rightarrow B$. Por otra parte, si $C \in \chi$ no tenemos un v-estructura.

El número de tests a realizar aún se puede reducir más: solamente haremos $I(A, B, \chi)$ para conjuntos χ , donde todos los miembros de χ se encuentran en un camino entre A y B (necessary path condition).

Direccionar los arcos

Regla 1:[introducción de v-estructuras]: Si hay tres nodos, A, B, C , tal que $A - C$ y $B - C$, pero no $A - B$, entonces introducimos la v-estructura $A \rightarrow C \leftarrow B$ si existe un χ (posiblemente vacío) tal que $I(A, B, \chi)$ y C no pertenece a χ .

Una vez introducidas las v-estructuras, daremos dirección a los restantes arcos usando las siguientes reglas:

Regla 2:[evitar nuevas v-estructuras]: Cuando la regla 1 se agotó, y se tiene $A \rightarrow C - B$ (y no hay un arco entre A y B), entonces hacer $C \rightarrow B$.

Regla 3:[evitar ciclos]: Si $A \rightarrow B$ introduce un ciclo dirigido en el grafo, entonces hacer $A \leftarrow B$.

Regla 4:[elegir aleatoriamente]: Si ninguna de las reglas anteriores se puede aplicar, tomar un arco no dirigido y darle una dirección arbitrariamente.

Ejemplo:

Supongamos que los casos son una muestra fiel de la red bayesiana dada por el primer gráfico que aparece en la Figura (3.8). Vamos a aplicar el algoritmo PC para obtener el esqueleto. Empezamos con el grafo completo (segundo gráfico de la Figura (3.8)).

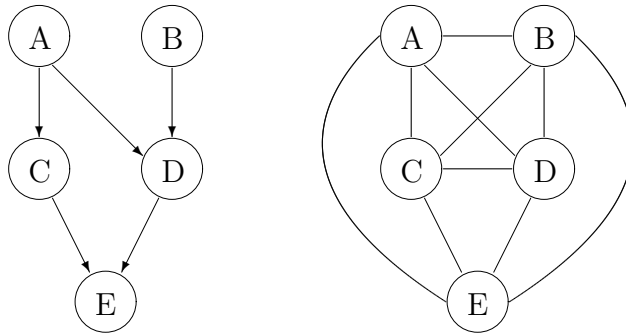


Figura 3.8: De izquierda a derecha: red bayesiana y grafo completo de la red bayesiana.

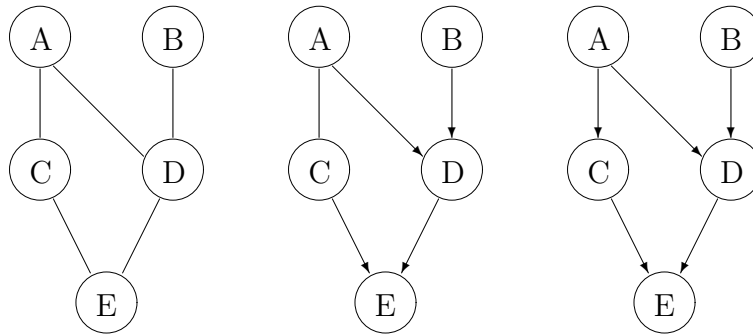


Figura 3.9: Aspecto de la red bayesiana para $i=0$, $i=1$ e $i=2$ respectivamente

Para $i = 0$, obtenemos “sí” para $I(A, B)$ y $I(B, C)$ entonces los arcos $A - B$ y $B - C$ se suprimen. Resulta entonces el primer gráfico correspondiente a la Figura (3.9).

Para $i = 1$, preguntamos $\downarrow I(A, C, E)?$, $\downarrow I(A, C, E)?$, $\downarrow I(B, C, D)?$, $\downarrow I(B, C, E)?$, $\downarrow I(B, D, C)?$, $\downarrow I(B, D, E)?$, $\downarrow I(B, E, C)?$, $\downarrow I(B, E, D)?$, $\downarrow I(C, B, A)?$, $\downarrow I(C, D, B)?$. La última pregunta tiene como respuesta “sí”, entonces borramos el arco $C - D$ y continuamos; $\downarrow I(C, E, A)?$, $\downarrow I(C, E, B)?$, $\downarrow I(D, B, E)?$, $\downarrow I(D, E, B)?$, $\downarrow I(E, A, B)?$, $\downarrow I(E, A, D)?$, $\downarrow I(E, B, A)?$, $\downarrow I(E, C, B)?$,

$\dot{I}(E, C, D)?$, $\dot{I}(E, D, A)?$, $\dot{I}(E, D, C)?$. No hay ninguna respuesta afirmativa entonces el segundo gráfico de la Figura (3.9) es la que resulta de hacer los test con una sola variable condicional.

Para $i = 2$, obtenemos respuesta afirmativa para las preguntas $\dot{I}(B, E, \{C, D\})?$ y $\dot{I}(A, E, \{C, D\})?$. Entonces borramos los arcos $B - E$ y $A - E$, resultando el tercer gráfico de la Figura (3.9).

Para $i = 3$, ya no hay ningún nodo que tenga 4 vecinos, por tanto el algoritmo termina.

Entonces de la aplicación del algoritmo obtenemos el esqueleto, (primer gráfico de la Figura (3.9)), y las independencias condicionales $I(A, B)$, $I(B, C)$, $I(C, D, A)$, $I(A, E, \{C, D\})$, $I(B, E, \{C, D\})$. Con esto ya podemos aplicar las reglas 1-4:

Regla 1 Introducimos las v-estructuras $A \rightarrow D \leftarrow B$ y $C \rightarrow E \leftarrow D$ (segundo gráfico de la Figura (3.9)).

Regla 2 No se puede aplicar.

Regla 3 No se puede aplicar.

Regla 4 Al arco $A - C$ le damos cualquier dirección (tercer gráfico de la Figura (3.9)).

Bases de datos

Vamos a ver ahora cómo respondemos a preguntas del tipo $I(A, B, \chi)$ a partir de una base de datos DB . Utilizaremos la notación $I_{DB}(A, B, \chi)$ para la independencia condicional en la distribución determinada por DB .

Definición 1 (Muestra fiable). DB es una muestra fiable de una red bayesiana N si se cumple: A y B son d-separadas en N dado χ si y sólo si $I_{DB}(A, B, \chi)$.

Si DB es fiel a N , podemos usar un test de independencia en DB para responder a nuestras preguntas. Para ello, se puede usar la condición de información mutua.

$$CMI(A, B | \chi) = \sum_{\chi} P^*(\chi) \sum_{A, B} P^*(A, B | \chi) \log_2 \frac{P^*(A, B | \chi)}{P^*(A)P^*(B)}.$$

De ahí se tiene,

$$I_{DB}(A, B, \chi) \leftrightarrow CMI(A, B | \chi) = 0$$

Basándose en la base de datos, se calculará una estimación de $CMI(A, B | \chi)$; luego actúa un test χ^2 sobre la hipótesis $CMI(A, B | \chi) = 0$ del que el usuario decide el nivel de significación. Un alto nivel de significación significa que pocos arcos son suprimidos. Como en todos los test, hay un riesgo de que los arcos que deberían ser suprimidos no lo sean y viceversa. El ratio de error está estrechamente relacionado con el tamaño de la muestra. Cuánto más pequeña sea la muestra, más independencias serán aceptadas y pocos arcos serán insertados.

Puede ocurrir que no sea posible direccionar un arco sin violar algunas de las independencias dadas por los test. Además, hay que mencionar que aunque un test estadístico sea completamente correcto para la independencia puede no proporcionar propiedades de d-separación correctas (aunque tengamos una base de datos grande); las probabilidades condicionales pueden ocultar dependencias.

Aprendizaje basado en medidas de ajuste y búsqueda

Cuando se hace aprendizaje estructural, buscamos una estructura de red bayesiana que por una parte pueda representar nuestra base de datos suficientemente bien y por otra parte que no sea demasiado compleja. En la sección anterior vimos como llevar a cabo el aprendizaje estructural basándonos en tests de independencia, ahora vamos a centrarnos en otro tipo de aprendizaje llamado aprendizaje por búsqueda y score. Se trata de asignar un número (una puntuación) a cada estructura de red bayesiana. La puntuación refleja la adecuación de una red bayesiana a un conjunto de datos.

Si tenemos una función score que lleva una red bayesiana como argumento y devuelve un valor, entonces la tarea del aprendizaje basado en el score puede considerarse un problema de búsqueda: simplemente buscar la estructura con la puntuación más alta. Esto significa que este aprendizaje puede describirse especificando dos componentes, una función score y un procedimiento de búsqueda de la mejor estructura.

Funciones score

La función score debería tener (al menos) las siguientes propiedades:

- Debería compensar la precisión de una estructura con la complejidad de la misma.
- Debería ser computacionalmente manejable para evaluar.

Un ejemplo de función score satisfaciendo las propiedades anteriores es el **criterio de información bayesiana (BIC)**, el cual contiene un término que mide como los datos se ajustan al modelo y otro término que mide la complejidad del modelo:

$$BIC(S|D) = \log_2 P(D|\hat{\theta}_S, S) - \frac{\text{size}(S)}{2} \log_2(N),$$

donde $\hat{\theta}_s$ es una estimación del parámetro de máxima verosimilitud para la estructura S . Si además asumimos que los casos son independientes dado el modelo, entonces

$$BIC(S|D) = \sum_{i=1}^n \log_2 P(d_i|\hat{\theta}_S, S) - \frac{\text{size}(S)}{2} \log_2(N)$$

Para buscar un modelo utilizando BIC empezamos estimando los estimadores de máxima verosimilitud para el modelo. Si la base de datos está completa, este es un problema de contar frecuencias, pero si alguno de los casos tiene valores perdidos debemos recurrir al algoritmo EM. Si todos los casos están completos, el cálculo de la medida BIC se reduce a un problema de cómputo: denotemos por r_i el número de estados de la variable X_i , y denotemos $q_i = \prod_{X_i \in Pa(X_i)} r_i$ el número de configuraciones sobre los padres de X_i en S (si X_i no tiene padres entonces $q_i = 1$). Con esta notación se tiene:

$$BIC(S|D) = \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log_2 \left(\frac{N_{ijk}}{N_{ij}} \right) - \frac{\log_2 N}{2} \sum_{i=1}^n q_i (r_i - 1), \quad (3.4)$$

donde N_{ijk} denota el número de casos en la base de datos con X_i en su k -ésima configuración y $Pa(X_i)$ en la j -ésima configuración.

Ejemplo:

Consideremos las dos estructuras de red bayesiana que se muestran en la Figura (3.10):

donde X_1 y X_2 son variables binarias. Disponemos de la base de datos dada por Tabla (3.2).

Figura 3.10: *Redes Bayesianas.*

Casos	X_1	X_2
1	Si	Positivo
2	Si	Positivo
3	Si	Positivo
4	Si	Positivo
5	Si	Positivo
6	Si	Positivo
7	Si	Positivo
8	Si	Positivo
9	No	Negativo
10	No	Negativo

Cuadro 3.2: *Base de datos asociada al ejemplo dado por la Figura (3.10).*

Vamos a ver cuál de las dos estructuras es mejor; para ello calculamos el valor de la función BIC para ambas estructuras.

Antes de calcular el valor de la función BIC para la estructura (a) necesitamos calcular N_{11k} y N_{2jk} :

	Positivo	Negativo	$X_1 X_2$	Sí	No
X_1	8	2	Positivo	6	0
			Negativo	2	2

Entonces, substituyendo estos valores en la Ecuación (3.4) obtenemos:

$$\begin{aligned}
 BIC_{(a)}(S|D) = & \left[8 \cdot \log\left(\frac{8}{8+2}\right) + 2 \cdot \log\left(\frac{2}{8+2}\right) + 6 \cdot \log\left(\frac{6}{6+2}\right) + 2 \cdot \log\left(\frac{2}{6+2}\right) + \right. \\
 & \left. 0 \cdot \log\left(\frac{0}{0+2}\right) + 2 \cdot \log\left(\frac{2}{0+2}\right) + -\frac{1+2}{2} \log(10) \right] = -18,69.
 \end{aligned}$$

De la misma forma, para calcular el valor de BIC para la estructura (b) calculamos: $N'_{111} = 8$, $N'_{112} = 2$, $N'_{211} = N_{211} + N_{221} = 6$ y $N'_{212} = N_{212} + N_{222} = 4$. Entonces:

$$BIC_{(b)}(S|D) = \left[8 \cdot \log\left(\frac{8}{8+2}\right) + 2 \cdot \log\left(\frac{2}{8+2}\right) + 6 \cdot \log\left(\frac{6}{6+4}\right) + 4 \cdot \log\left(\frac{4}{6+4}\right) - \frac{1+1}{2} \log(10) \right] = -20,25.$$

Los cálculos nos dicen que debemos escoger la estructura (a).

En este apartado tan sólo nos hemos centrado en la medida BIC pero existen otros tipos de medidas de calidad para calcular la adecuación de una red bayesiana a un conjunto de datos. Entre otras:

- Medidas bayesianas.
- Medidas de mínima longitud.
- Criterio de información de Akaike.

Procedimiento de búsqueda

Como hemos dicho, una vez establecida la forma de medir la calidad de una estructura, se establece un método para hacer una búsqueda de la “mejor” estructura entre todas las estructuras posibles. Dado que el número de posibles estructuras es exponencial en el número de variables, este problema no es posible resolverlo eficientemente de forma exacta. A continuación veremos algunos algoritmos para la resolución aproximada del aprendizaje estructural:

Algoritmo B(Buntine)

Este algoritmo de Buntine se basa en un esquema voraz para la construcción de una solución aproximada a partir de la red vacía de enlaces (cada nodo posee inicialmente un conjunto vacío de padres). Este algoritmo trata de introducir el arco que mayor ganancia representa con respecto a la red anterior y a la función score utilizada. Se detiene cuando la inclusión

de un arco no representa ninguna ganancia.

Algoritmo HC

Es un algoritmo local de ascensión de colinas (hill climbing) por el máximo gradiente basado en la definición de vecindad. El algoritmo parte de una solución inicial, como podría ser la red vacía de enlaces, u otra cualquiera. A partir de esta solución se calcula el nuevo valor de la función score para todas las soluciones (grafos) vecinas a la solución actual y nos quedamos con la solución que tenga el mejor valor de la función.

La vecindad clásica que se maneja en este algoritmo es la siguiente: dado un grafo G (solución actual) se denominan grafos vecinos G' a aquellos resultantes de incluir un sólo arco a G o borrar un sólo arco presente en G o invertir la dirección de un arco presente en G , todo ello sin incluir ciclos dirigidos en G' . El algoritmo parará cuando no exista ningún vecino que pueda mejorar la solución actual.

El algoritmo no garantiza encontrar la estructura óptima, ya que puede llegar a un máximo local.

Estos algoritmos, al igual que el anterior, se aprovechan de la descomponibilidad de la función score, de esta forma sólo hay que calcular el beneficio de hacer un cambio.

Definición 4 (Descomponibilidad). Una función score se dice que es descomponible si se puede calcular de forma local para cada familia (conjunto formado por un nodo y sus padres) de nodos de la estructura:

$$score(D, S) = \sum_{i=1}^n score(X_i, Pa(X_i), D)$$

Reducir el espacio de búsqueda

Como dijimos antes, el número de estructuras posibles es exponencial, entonces podemos optar por reducir el espacio de búsqueda.

Simplicidad

Dado que la función BIC incorpora un término que controla la complejidad del modelo, otra forma de tratar el problema de búsqueda de estructura es poner restricciones en las

estructuras permitidas y de este modo no considerar las estructuras demasiado complejas. Una clase particular de estructuras de redes bayesianas y que además son simples son las estructuras en forma de árbol, de las que ya hablamos con anterioridad.

Incorporar información previa

- Utilizar el conocimiento de expertos.
- Explotar la causalidad.
- Especificar una relación de orden parcial, \leq , sobre las variables, de forma que permitimos un arco de X_i a X_j sólo si $X_i \leq X_j$; basándose en este principio tenemos el algoritmo K2.

Algoritmo K2

Este algoritmo parte de que las variables de entrada están ordenadas, de forma que los posibles padres de una variable aparecen en el orden antes que ella misma. El proporcionarle al algoritmo un orden entre las variables hace que éste tan “sólo” tenga que buscar el mejor conjunto de padres posibles de entre las variables predecesoras en el orden. La búsqueda de este conjunto se hace de forma voraz.

El algoritmo parte de que el conjunto de padres para cada variable es el conjunto vacío. Posteriormente, y siguiendo el orden establecido, pasa a procesar cada variable X_i , calcula la ganancia que se produce en la medida utilizada al introducir una variable X_j como padre de X_i de entre todas sus predecesoras, esto es, para todo $j < i$, y se queda con la que produce mejor ganancia. Desde un enfoque bayesiano, esta ganancia se calcula como la razón de la métrica vista en el punto anterior de una red bayesiana con la variable X_i sin padres y una red bayesiana donde X_i tenga como padre X_j . Una vez fijado el primer padre de X_i se prosigue de la misma forma, pero esta vez midiendo la razón de una red bayesiana con X_i y el primer padre fijado y una red bayesiana donde se le introduce un segundo padre no insertado previamente.

3.4.2. Aprendizaje paramétrico

Supongamos que conocemos la estructura de una red bayesiana modelando las variables de U , pero no tenemos ninguna estimación de las probabilidades condicionadas. Por otra parte, supongamos ahora que tenemos acceso a bases de datos de algunos casos, es decir, un conjunto de valores simultáneos para algunas de las variables de U . En este caso podemos utilizar lo anterior para estimar dichas probabilidades condicionadas. Los números (probabilidades condicionadas) que necesitamos especificar para una red bayesiana se denominan *parámetros* de la red.

Consideraremos dos tipos de aproximaciones posibles para tratar este problema. En primer lugar veremos como una base de datos de casos puede ser utilizada para estimar los parámetros de una vez por todas. Seguidamente, investigaremos la situación en la que los casos se acumulan secuencialmente, y nuestro objetivo será adaptar el modelo para posibles nuevos casos.

Datos completos

Sea $M = (S, \theta)$ una red bayesiana con estructura S y parámetros θ , y sean U las variables de M . Sea, además D un conjunto de casos, donde cada caso es una configuración sobre todas las variables de U (es decir, una muestra de entrenamiento). Por otra parte, para asegurarnos de que el aprendizaje sobre cada parámetro se pueda realizar de forma independiente, supondremos las siguientes hipótesis:

- **Independencia global:** los parámetros de las diferentes variables son independientes. Esto significa que podemos modificar las tablas de las variables de forma independiente.
- **Independencia local:** las incertidumbres para los parámetros que tengan diferentes configuraciones de padres son independientes.

Estimador de máxima verosimilitud

El aprendizaje de los parámetros es simple cuando todas las variables son completamente observables en el conjunto de entrenamiento. El método más común es el llamado estimador de máxima verosimilitud, que consiste sencillamente en estimar las probabilidades deseadas a partir de la frecuencia de los valores de los datos de entrenamiento:

Para cada caso $d \in D$, la probabilidad $P(d|M)$ se denomina verosimilitud de M dado d . Si suponemos que los casos de D son independientes dado el modelo, entonces la verosimilitud de M dado D es:

$$L(M|D) = \prod_{d \in D} P(d|M)$$

Generalmente se toma el logaritmo, y se denomina log-verosimilitud:

$$LL(M|D) = \prod_{d \in D} \log_2 P(d|M)$$

Si nuestro objetivo es escoger de entre un conjunto de modelos uno que nos explique lo mejor posible nuestros datos, el principio de la máxima verosimilitud nos aconseja escoger un modelo que maximiza la máxima verosimilitud dados los datos. Esto significa que si nuestro objetivo es estimar las probabilidades condicionadas, entonces nuestros posibles modelos M_θ tienen la misma estructura pero difieren respecto de los parámetros θ . De esta forma escogemos una estimación $\hat{\theta}$ que maximice la verosimilitud:

$$\hat{\theta} = \arg_{\theta} \max L(M_\theta|D) = \arg_{\theta} \max LL(M_\theta|D)$$

La calidad de estas estimaciones dependerá de que exista un número suficiente de datos en la muestra. Cuando tengamos una base de datos escasa, la estimación hecha por máxima verosimilitud no será muy precisa.

Estimación bayesiana

Una alternativa al principio de máxima verosimilitud es la estimación bayesiana: empezar con una distribución a priori y utilizar la experiencia para actualizar la distribución (ver V. Jensen y D. Nielsen (2007)). A los parámetros estimados mediante estimación bayesiana se les denomina máximos parámetros a posteriori.

En general se puede cuantificar la incertidumbre existente representándola mediante una distribución de probabilidad, para así considerarla explícitamente en la definición de las probabilidades. Habitualmente se emplean distribuciones Beta en el caso de variables binarias, y distribuciones Dirichlet para variables multivaluadas. Esta aproximación es útil cuando se cuenta con el apoyo de expertos en el dominio de la aplicación para concretar los valores de los parámetros de las distribuciones.

Datos incompletos

Aparecen mayores dificultades cuando los datos de entrenamiento no están completos. Pueden plantearse dos tipos de información incompleta:

- **Valores faltantes** : faltan algunos valores de una o varias variables en algunos ejemplos.
- **Nodo oculto** : faltan todos los valores de una variable.

El primer caso es más sencillo, y existen varias alternativas, entre ellas:

- Eliminar los ejemplos con valores ausentes.
- Considerar un nuevo valor adicional para la variable “desconocida”.
- Considerar el valor más probable a partir de los datos de la misma en las demás instancias.
- Considerar el valor más probable en base a las demás variables.

Las dos primeras opciones son habituales en problemas de aprendizaje, y válidas siempre y cuando se cuente con un número elevado de datos completos. La tercera opción viene a ignorar las posibles dependencias de la variable con las demás, cuando ya se cuenta con la estructura que las describe en el grafo; no suele proporcionar los mejores resultados. La cuarta técnica se sirve de la red ya conocida para inferir los valores desconocidos. Primero se rellenan las tablas de parámetros usando todos los ejemplos completos. Después, para cada instancia incompleta, se asignan los valores conocidos a las variables correspondientes en la red y se propaga su efecto para obtener las probabilidades a posteriori de las no observadas. Entonces se toma como valor observado el más probable y se actualizan todas las probabilidades del modelo antes de procesar la siguiente instancia incompleta.

La aparición de nodos ocultos requiere un tratamiento más complejo. Existen diferentes técnicas para estimar las probabilidades faltantes en este caso. Una habitual es la aplicación del algoritmo EM (*Expectation Maximization*).

Algoritmo EM

Uno de los algoritmos más populares para realizar la estimación de los parámetros es el algoritmo EM: es un algoritmo general que realiza la búsqueda de las estimaciones de máxima verosimilitud para un conjunto de parámetros θ cuando se tiene un conjunto de datos incompleto.

Su aplicación al aprendizaje de parámetros se traduce, de forma esquemática, en lo siguiente:

- Asignar valores aleatorios (o basados en conocimiento experto, si se dispone de él) a las probabilidades desconocidas de la red.
- Utilizar los datos conocidos para estimar desconocidos infiriéndolos sobre el modelo con las probabilidades actuales.
- Completar el conjunto de datos con los valores estimados y volver a calcular las probabilidades de la red a partir de ellos.
- Repetir los dos pasos anteriores hasta que no haya cambios significativos en las probabilidades.

Algoritmo del gradiente ascendente

La técnica del gradiente trata de maximizar la probabilidad de los datos de entrenamiento conocida la hipótesis h , $P(D|h)$, considerando como espacio de hipótesis el conjunto de todas las posibles combinaciones de valores para las probabilidades que parametrizan la red. Para ello, sigue el gradiente de $\ln P(D|h)$ con respecto a las probabilidades de la red, actualizando cada parámetro w_{ijk} desconocido de forma iterativa con el incremento:

$$w_{ijk} \leftarrow w_{ijk} + k \sum_{d \in D} \frac{P(X_{ij}, \pi_{ij}|d)}{w_{ijk}}$$

donde w_{ijk} es el parámetro desconocido correspondiente a la probabilidad condicional de que la variable X_i tome el valor x_{ij} cuando sus padres $Pa(X_i)$ toman los valores π_{ik} , y k es una tasa de aprendizaje. En cada iteración las probabilidades w_{ijk} se renormalizan tras el incremento.

Tanto el algoritmo EM como el de gradiente ascendente encuentran soluciones que son sólo óptimas localmente, por lo que en ambos casos la calidad del resultado dependerá de la asignación inicial de las probabilidades desconocidas.

3.5. Clasificadores basados en redes bayesianas

Un clasificador, en general, suministra una función que mapea (clasifica) un dato (instancia), especificado por una serie de características o atributos, en una o diferentes clases predeterminada.

Formalmente, se tiene un conjunto de variables, $\{F_1, \dots, F_n\}$, llamadas atributos y una variable clase, C , donde los estados de C corresponden a las posibles clases. Entonces un clasificador es una función de $F_1 \times \dots \times F_n$ en C . Los clasificadores bayesianos son ampliamente utilizados debido a que presentan ciertas ventajas:

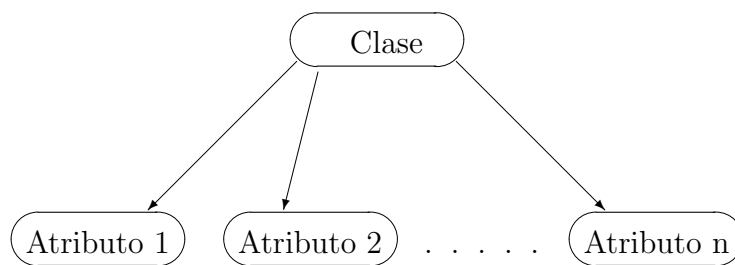
- Generalmente, son fáciles de construir y entender.
- Las inducciones de estos clasificadores son extremadamente rápidas, requiriendo sólo un paso para hacerlo.
- El proceso es muy robusto considerando atributos irrelevantes.
- Toma evidencia de muchos atributos para realizar la predicción final.

Un clasificador bayesiano se puede ver como un caso espacial de una red bayesiana en la cual hay una variable especial que es la clase y las demás variables son los atributos. La estructura de esta red depende del tipo de clasificador.

3.5.1. Clasificador Naive Bayes

El clasificador Naive Bayes y sus variantes se encuentran entre los algoritmos más conocidos para construir clasificadores de, por ejemplo, documentos de texto, filtración de correo electrónico, clasificación de galaxias y reconocimiento de emociones. A pesar de su simplicidad es comparable con clasificadores sofisticados como las redes neuronales y los árboles de decisión, ya que posee alta precisión y velocidad cuando es aplicado a conjuntos grandes de datos.

El clasificador Naive Bayes fue popularizado por Duda y Hart gracias a su simplicidad, eficiencia y bajo error de clasificación. Este clasificador supone que todas las variables son condicionalmente independientes dado el valor de la variable clase. La estructura de este clasificador puede representarse usando una red bayesiana en la que existe un nodo para la

Figura 3.11: *Clasificador Naive Bayes*

variable de clase C , que es padre de todas las variables, y en la que no existen arcos entre las variables (ver Figura (3.11)).

Esto significa que la estructura es fija, y lo único que hay que hacer es estimar los parámetros, los cuales se pueden determinar fácilmente utilizando los métodos que se presentaron en el apartado de estimación de parámetros.

El funcionamiento de este clasificador es algo sorprendente, ya que el supuesto de independencia no es realista. Considerérese un clasificador para la evaluación del riesgo en las solicitudes de crédito: es contra intuitivo ignorar las correlaciones entre edad, nivel de educación e ingreso. El ejemplo anterior lleva a la necesidad construir un clasificador que tome en consideración las relaciones de dependencia que existen en el conjunto de variables.

3.5.2. Extensiones del clasificador Naive Bayes

Ante el buen rendimiento ofrecido por el clasificador Naive Bayes (NB), a pesar de la fuerte suposición que realiza, cabe preguntarse si los resultados no serán mejores tras relajar o suprimir dicha suposición (independencia de los atributos dada la clase). Partiendo de que el clasificador NB constituye la red bayesiana más sencilla que podemos construir orientada a clasificación, en este apartado veremos otros modelos de redes bayesianas que han destacado por su aplicación en esta tarea.

Clasificador Naive Bayes Aumentado con un Árbol (TAN)

TAN (*Tree Augmented Naive Bayes*) constituye una extensión del clasificador NB. La idea es construir una red bayesiana un poco más compleja que el NB pero donde se da un tratamiento especial a la variable clase. Con TAN, los autores pretenden mantener la simplicidad computacional del clasificador NB pero intentando mejorar la tasa de acierto durante la clasificación. Para ello, en lugar de suponer todas las variables independientes (dada la clase), se admiten ciertas dependencias entre los atributos. En concreto, se supone que los atributos constituyen una red bayesiana con forma de árbol. La ventaja de restringir la topología de la red (entre los atributos) a un árbol, es que esta estructura puede aprenderse eficientemente. Así, los autores de TAN proponen usar una ligera modificación del algoritmo de Chow y Liu para realizar este aprendizaje. Este algoritmo está basado en el concepto de información mutua:

$$MI(X, Y | C) = \sum_{X, Y, C} P(X, Y | C) \log_2 \left(\frac{P(X, Y | C)}{P(X | C)P(Y | C)} \right),$$

que mide la cantidad de información que la variable Y nos proporciona sobre la variable X supuesto que el valor de la clase C es conocido.

Tras aprender la estructura de árbol entre los atributos, el algoritmo TAN añade la variable clase y la hace padre de todos ellos.

Algoritmo

1. Calcular $MI(F_i, F_j | C)$ para cada par de atributos (F_i, F_j) ($i \neq j$).
2. Crear un grafo no dirigido con todos los atributos como conjunto de nodos y añadir arcos entre cada par de nodos.
3. Asociar a cada arco (i, j) del grafo el peso $MI(F_i, F_j | C)$.
4. Construir un árbol expandido de máximo peso a partir del grafo anterior.
5. Elegir un nodo cualquiera del árbol anterior como raíz y direccionar a partir de él el resto de arcos.
6. Añadir la variable clase C y el conjunto de arcos dirigidos $(C \rightarrow F_i)$ para todo atributo F_i .
7. Aprender los parámetros.

Este algoritmo tiene dos claras ventajas: un bajo coste computacional $O(N^2 \cdot n)$, siendo N el número de instancias en el conjunto de entrenamiento y n el número de variables; y en segundo lugar, asegura que la estructura de red obtenida es la de máxima verosimilitud del conjunto de todas las posibles estructuras TAN.

Finalmente, indicar que existen distintas variaciones de este algoritmo; entre otras, podemos citar las siguientes:

- Métodos de aprendizaje de redes bayesianas que parten de la estructura del Naive Bayes como estado inicial y, progresivamente añaden arcos hasta llegar a una estructura TAN. Denotaremos esta variante como TANi- m , siendo m el método de aprendizaje utilizado.
- Métodos que simultáneamente al proceso de construcción realizan una selección de variables (s TAN).
- Métodos que distribuyen los atributos en un conjunto de árboles, un bosque, en lugar de un único árbol (FAN, Fores Augmented Naive Bayes).

Clasificador Naive Bayes Aumentado con una red bayesiana (BAN)

El algoritmo BAN se encuadra en la filosofía de TAN, es decir, aprender redes bayesianas orientadas a clasificación. En BAN (*Bayesian Network Augmented Naive Bayes*) se procede aprendiendo una red bayesiana para los atributos (excluyendo la clase) y posteriormente se aumenta el modelo añadiendo la variable clase C y aristas desde C hacia todos los atributos. Para aprender la red se puede usar cualquier algoritmo de aprendizaje de redes bayesianas.

Al igual que ocurre con TAN, existe la posibilidad de iniciar la estructura de red como un NB y a partir de ahí lanzar un algoritmo de aprendizaje que vaya añadiendo arcos. A estos modelos los notaremos como BANi- m , siendo m el algoritmo de aprendizaje de redes bayesianas utilizado.

3.5.3. Redes bayesianas como clasificadores

En los apartados anteriores también se aprenden redes bayesianas y se utilizan como clasificadores. La diferencia con este apartado es que ahora se cambia de filosofía. Ahora no se da un tratamiento especial a la variable clase, aprendiéndose así un modelo orientado a

clasificación, sino que se aprende una red incluyendo a todas las variables (clase y atributos) del problema, que posteriormente se usará para clasificar. A estos modelos los denotaremos como RB- m , siendo m el algoritmo de aprendizaje utilizado.

Puesto que en una red bayesiana se cumple que toda variable X es independiente del resto dado su envolvente (o manto) de Markov, podemos seleccionar sólo a dichas variables como variables predictoras “útiles” para la clasificación.

3.5.4. Evaluación de los clasificadores

Supongamos que se tiene un clasificador, $Clsf$, y una base de datos con los casos de las variables atributos y de la variable clase. Se quiere caracterizar la calidad de $Clsf$. Vamos a ver dos formas de hacerlo:

- **Precisión de la clasificación:** es la fracción de casos clasificados correctamente.
- **Pérdida esperada:** Para una descripción más detallada de un clasificador tendríamos que calcular la matriz de confusión, $P^*(\text{Valor clasificado} | \text{Valor correcto})$. Además de la matriz de confusión también se puede introducir un valor que cuantifique las malas clasificaciones, y así establecer una matriz de pérdida, que describa una sanción para los diferentes tipos de mala clasificación.

$$\text{Pérdida esperada} = \sum_{Clasificado, Correcto} P^*(Clasificado, Correcto) \text{Pérdida}(Clasificado, Correcto)$$

Capítulo 4

Estudio práctico

4.1. Análisis de datos con Weka

El objetivo de este estudio práctico es aplicar las redes bayesianas a un problema de clasificación con un conjunto de datos reales sobre accidentes.

4.1.1. Pequeña introducción a Weka

Weka es un conjunto de librerías JAVA para la extracción de conocimientos desde bases de datos. Weka contiene las herramientas necesarias para realizar transformaciones sobre los datos, tareas de clasificación (redes neuronales, reglas y árboles de decisión, aprendizaje Bayesiana), regresión (regresión lineal, SVM), clustering, asociación y visualización. Cuando abrimos el programa aparece la ventana de la Figura (4.1):

Program: registro de mensajes (LogWindow) y Exit.

Visualization: incluye un conjunto de programas para la visualización de información.

- *Plot*: visualización de ficheros de datos, en dos dimensiones eligiendo las variables a visualizar en cada dimensión y con la posibilidad de utilizar una tercera variable (categórica) para caracterizar los pares de puntos en el gráfico.
- *ROC*: visualización de curvas ROC (Receiver Operating Characteristics), herramienta



Figura 4.1: Ventana que aparece al iniciar el programa Weka.

de análisis de la eficacia de un clasificador.

- *TreeVisualizer*: visualizador de árboles de decisión.
- *GraphVisualizer*: visualizador de gráficos en diferentes formatos (XML, BIF, DOT).
- *BoundaryVisualizer*: visualizador de fronteras discriminantes en un problema de clasificación.

Tools:

- *ArffViewer*: visualizador y editor de ficheros de datos con formato *.arff* y otros.
- *SqlViewer*: interface para la conexión a bases de datos e interacción mediante comandos *SQL* (*Structured Query Language*), lenguaje estándar de consulta de bases de datos relacionales.
- *EnsembleLibrary*: módulo para la creación de modelos en formato *.xml* (*eXtensible Markup Language*).

Help: enlaces a diferentes páginas web con material de ayuda.

Applications:

- *Explorer*: Entorno visual que ofrece una interfaz gráfica para el uso de los paquetes. Este módulo permite utilizar cualquier técnica de minería de datos incluida en Weka, pero de manera individual, es decir, explorando sus capacidades y prestaciones de manera aislada, sin realizar tareas de comparación entre diferentes técnicas ni encadenar de manera automática tratamientos sucesivos.

- *Experimenter*: Este módulo permite realizar el proceso de selección del modelo, es decir, comparar el comportamiento de diferentes técnicas en distintas configuraciones de test.
- *KnowledgeFlow*: este módulo permite generar proyectos de minería de datos mediante la generación de flujos de información.
- *Simple CLI (Command Line Interface)*: Esta interfaz proporciona una consola para introducir comandos en línea. En principio, cualquier tarea realizable por los tres módulos anteriores puede ejecutarse desde aquí en modo de comandos.

Weka Explorer

Este modo cuenta con seis paneles distintos:

- *Preprocess*: Incluye las herramientas y filtros para cargar y manipular los datos.
- *Classify*: Alberga técnicas de clasificación y regresión.
- *Cluster*: Integra varios métodos de agrupamiento.
- *Associate*: Incluye algunas técnicas de reglas de asociación.
- *Select Attributes*: Permite aplicar diversas técnicas para la reducción del número de atributos.
- *Visualize*: Este panel permite estudiar el comportamiento de los datos mediante técnicas de visualización.

De estos, sólo nos pararemos en el panel Classify (ver Figura 4.2) ya que nuestro objetivo es aplicar redes bayesianas a un problema de clasificación.

En la ventana que aparece tenemos el botón choose de Classifier para configurar el método de clasificación o regresión que queramos utilizar. Si lo pulsamos aparece un menú con diferentes familias de métodos, como se puede ver en la ventana de la Figura (4.3).

Además, en la ventana que aparece en la Figura (4.3) podemos establecer como queremos efectuar la validación del modelo aprendido:

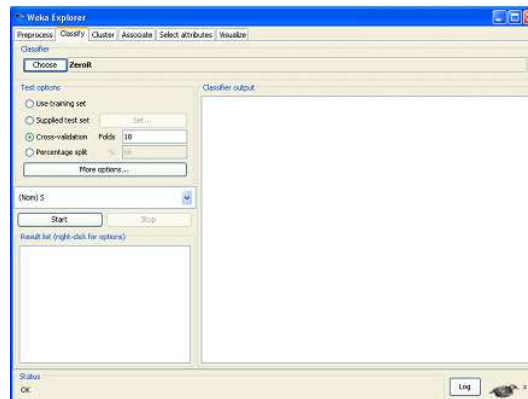


Figura 4.2: Ventana en la que se escoge el método de clasificación o regresión que se quiera utilizar.

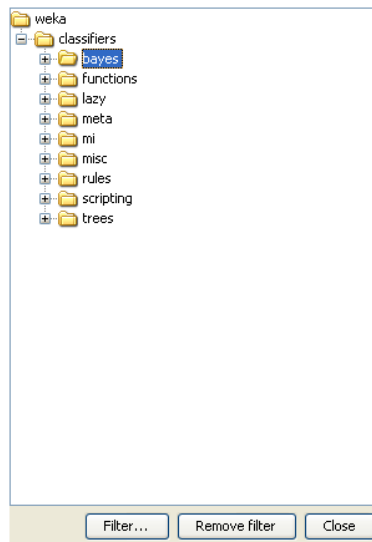


Figura 4.3: Ventana en la que aparecen las diferentes familias de métodos.

- *Use training set*: Con esta opción Weka entrenará el método con todos los datos disponibles y a posteriori realiza la evaluación sobre los mismos datos.
- *Supplied test set*: Con esta opción tendremos la oportunidad de seleccionar, pulsando el botón Set, un fichero de datos (normalmente diferentes a los de aprendizaje) con el que se probará el clasificador obtenido con el método de clasificación usado y los datos iniciales.
- *Cross-validation*: Mediante esta opción Weka realizará la evaluación mediante la técni-

ca de validación cruzada. La validación cruzada consiste en: dado un número n se divide los datos en n partes y, por cada parte, se construye el clasificador con las $n - 1$ partes restantes y se prueba con esa. Así por cada una de las n particiones. Una validación-cruzada es estratificada cuando cada una de las partes conserva las propiedades de la muestra original (porcentaje de elementos de cada clase).

- *Percentage splits*: Se define un porcentaje con el que se aprende el modelo y la evaluación se realiza con los datos restantes.

Una vez definido el método de prueba Weka nos permite seleccionar algunas opciones más con el botón *More Options* (ver Figura (4.4)). Las opciones que se nos presentan son entre otras:

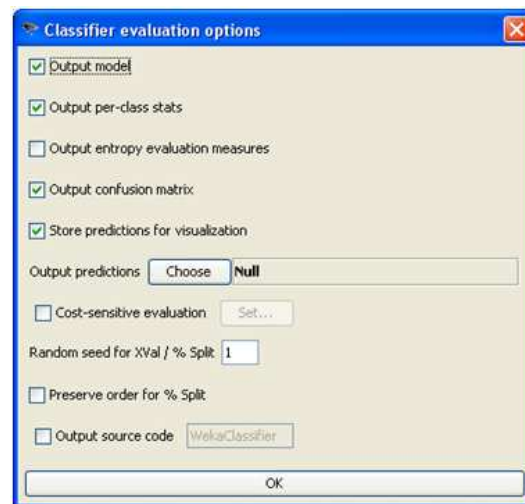


Figura 4.4: Ventana en la que se escogen opciones adicionales para el método de prueba.

- *Output Model*: Si la activamos una vez construido y probado el clasificador, nos mostrará en la salida del clasificador el modelo que ha construido.
- *Output per-class stats*: Activada muestra estadísticas referentes a cada clase.
- *Output entropy evaluation measures*: Muestra información de mediciones de la entropía en la clasificación.
- *Output confusion matrix*: Muestra la matriz de confusión del clasificador. Esta tabla cuyo número de columnas es el número de atributos muestra la clasificación de las instancias. Da una información muy útil porque no sólo refleja los errores producidos sino también informa del tipo de éstos.

Finalmente, se pulsará el botón *Start* que despierta al pájaro Weka de su letargo y realiza el modelo seleccionado.

4.1.2. Clasificadores basados en redes bayesianas con Weka

Sea $U = \{X_1, \dots, X_n\}$, $n \geq 1$ un conjunto de variables. La tarea de clasificación consiste en clasificar una variable llamada la variable clase dado un conjunto de variables llamadas variables atributos. El clasificador se aprende a partir de una base de datos D . La tarea de aprendizaje consiste en encontrar una red bayesiana apropiada dada una base de datos D sobre U .

Todos los algoritmos de redes bayesianas implementados en Weka asumen lo siguiente para la base de datos:

- Todas las variables son discretas y finitas. Si tenemos una base de datos con variables continuas, podemos utilizar un filtro para discretizarlas:
`weka.filters.unsupervised.attribute.Discretize`
- Las instancias no tienen valores perdidos. Si los hay, los valores son reemplazados utilizando el siguiente filtro:
`weka.filters.unsupervised.attribute.ReplaceMissingValues`

Como vimos en el estudio teórico hay varias formas de aprender la estructura de una red bayesiana. En Weka podemos llevar a cabo el aprendizaje estructural basándonos en:

- *Medidas de ajuste locales*: aprender la estructura de una red se puede considerar un problema de optimización donde se pretende maximizar la medida de calidad. Las medidas de calidad se pueden basar en enfoques bayesianos, MDL, información y otros criterios. Estas métricas tienen la propiedad de que son descomponibles. Esto tiene en cuenta puntuaciones locales y por tanto métodos de búsqueda locales.
- *Test de independencia condicional*: La suposición es que hay una estructura de la red que representa exactamente las independencias en la distribución que generaron los datos. Entonces de ahí se sigue que si se identifica una independencia (condicional) en los datos entre dos variables entonces no hay un arco entre estas dos variables. Una vez que se identifican las posiciones de los arcos, la dirección de los mismos se asigna de forma que las independencias en los datos estén correctamente representadas.

- *Medidas de ajuste globales*: Una forma natural de medir cómo actúa una red bayesiana sobre un conjunto de datos dado es predecir su futura actuación estimando utilidades esperadas, como la precisión de clasificación.
- *Estructuras fijas*: la estructura de antemano.

Para cada una de estas áreas, en Weka hay implementados diferentes algoritmos.

Una vez identificada una buena estructura, se pueden estimar las tablas de probabilidad condicional para cada una de las variables.

Podemos seleccionar un clasificador basado en una red bayesiana pinchando en el botón *Choose* de la pestaña *Classify* y seleccionando *BayesNet* (en la carpeta *bayes*). (Ver Figura (4.5))

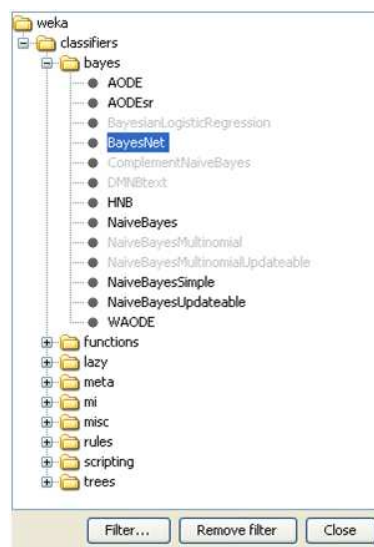


Figura 4.5: Ventana en la que se selecciona un clasificador basado en una red bayesiana.

El clasificador BayesNet tiene las siguientes opciones (ver Figura (4.6)):

- *BiFFile*: se puede usar para especificar una red bayesiana almacenada en un archivo en formato BIF.
- *Debug*: La opción debug no tiene efecto.

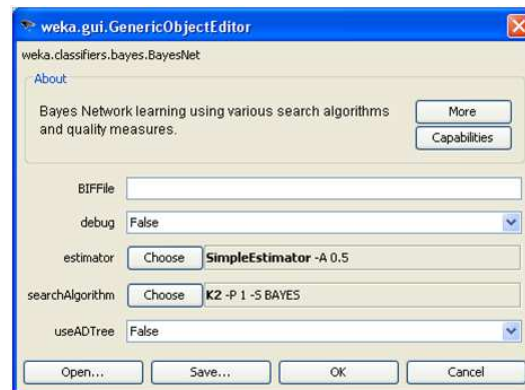


Figura 4.6: Ventana en la que se seleccionan las opciones del clasificar *BayesNet*.

- *Estimator*: sirve para seleccionar el método para estimar las distribuciones de probabilidad condicional, una vez que la estructura de la red está aprendida.
- *searchAlgorithm*: esta opción sirve para seleccionar un algoritmo de aprendizaje de la estructura y especificar sus opciones. El algoritmo puede estar basado en:
 - Medidas de ajuste locales.
 - Test de independencia.
 - Medidas de ajuste globales.
 - Una estructura fija.
- *useADTree*: cuando activamos esta opción, los cálculos se hacen usando el algoritmo *ADTree* de Moore (ver A. Moore, and M.S. Lee (1998)). Notar que este algoritmo es diferente del algoritmo clasificador de `weka.classifiers.tree.ADTree`.

Aprendizaje de la estructura basado en medidas de ajuste locales

Distinguimos medidas de calidad y algoritmos de búsqueda. Una medida local para el aprendizaje de la estructura se puede seleccionar eligiendo una en el paquete `weka.classifiers.bayes.net.search.local` (ver Figura (4.7)).

En Weka están implementados los siguientes algoritmos de búsqueda para medidas de ajuste locales: *K2*, *Hill Climber*, *Repeated Hill Climber*, *LAGD Hill Climbing*, *TAN*, *Simulated Annealing*, *Tabu search* y *Genetic search*.

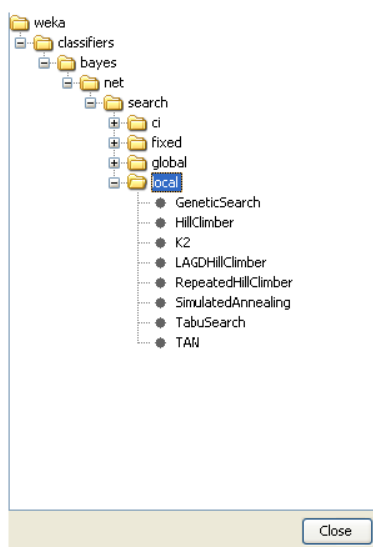


Figura 4.7: Ventana en la que se realiza el aprendizaje de la estructura basado en medidas de ajuste locales.

Vamos a pararnos en algunos de ellos (los que después aplicaremos a nuestro conjunto de datos):

- *Hill Climber*: busca la estructura utilizando un algoritmo hill climbing que va añadiendo, borrando o invirtiendo arcos. La búsqueda no está restringida por un orden de las variables.

Opciones:

- **initAsNaiveBayes**: si esta opción está activada (lo está por defecto), la estructura inicial de red bayesiana que se usa para empezar la búsqueda es una estructura naive Bayes. Si se desactiva, se usará una estructura de red vacía (es decir, sin ningún arco).
- **markovBlanketClassifier**: (*false* por defecto) si se cambia a *true*, al final de la travesía por el espacio de búsqueda, se usa un procedimiento heurístico para asegurar que cada uno de los atributos están en el manto de Markov del nodo clasificador. Si un nodo ya está en el manto de Markov no sucede nada, si no se añade un arco. Si dicha opción se deja desactivada tales arcos no son añadidos.
- **maxNrOfParents**: es el límite más alto en el número de padres de cada uno de los nodos en la estructura de la red.
- **scoreType**: determina la medida de ajuste usada. Están implementadas: Bayes, BDe, AIC, Entropy y MDL.

- **useArcReversal**: si esta opción está activada también se tiene en cuenta la inversión de arcos cuando se determina el próximo paso que hay que hacer.
- **K2**: Actúa como el anterior pero ahora la búsqueda está restringida por un orden de las variables.

Opciones:

- **initAsNaiveBayes**
- **markovBlanketClassifier**
- **maxNrOfParents**
- **scoreType**
- **randomorder**: dependiendo de si está activada o no, el orden de las variables es aleatorio o es el del conjunto de datos.
- **TAN**: Este algoritmo determina el árbol de expansión de peso máximo y devuelve una red Naive Bayes aumentada con un árbol.

Opciones:

- **markovBlanketClassifier**
- **scoreType**

Aprendizaje de la estructura basado en tests de independencia

En Weka están implementados los algoritmos *ICS* y el *CI*. Vamos a ver el *ICS*. (Ver Figura (4.8))

Este algoritmo hace dos pasos, primero encuentra el esqueleto y luego dirige los arcos para obtener un grafo acíclico dirigido.

Empieza con un grafo no dirigido completo y trata de encontrar independencias $I(A, B_\chi)$, donde χ es un subconjunto de nodos que son vecinos tanto de la variable A como de la variable B . Se empieza con un conjunto χ de cardinalidad 0 y va aumentando hasta llegar al tope que ha fijado el usuario. Si se identifica una independencia, el arco entre A y B se borra del esqueleto.

Una vez que ya se tiene el esqueleto se dirigen los arcos siguiendo una serie de reglas.

Opciones:

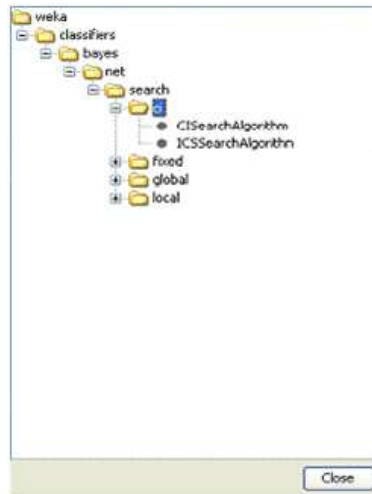


Figura 4.8: Ventana en la que se realiza el aprendizaje de la estructura basado en tests de independencia.

- `markovBlanketClassifier`
- `maxCardinality`: determina la máxima cardinalidad de χ que se considerará para hacer los test de independencia.
- `scoreType`

Aprendizaje de la estructura basado en medidas globales

La ventana en la que se realiza el aprendizaje de la estructura basado en medidas globales es la que aparece en la Figura (4.9). No vamos a entrar en detalles sobre este conjunto de métodos ya que tampoco lo tratamos en el estudio teórico.

Estructura fija

Los pasos para el aprendizaje de la estructura pueden ser omitidos seleccionando una estructura de red fija. Hay dos métodos de obtener una estructura fija: tomar la estructura *Naive Bayes*, o leyéndola de un archivo en el formato *XML*.

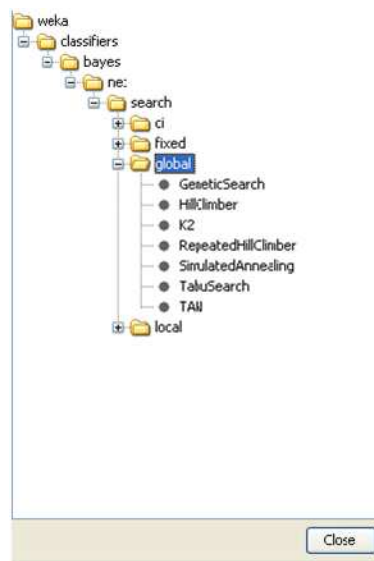


Figura 4.9: Ventana en la que se realiza el aprendizaje de la estructura basado en medidas globales.

4.1.3. Trabajando con los datos

Disponemos de un fichero con datos extraídos de la realización de una encuesta a diferentes empresas mineras, sobre las circunstancias que rodearon la ocurrencia de un *Suceso* (variable S) que puede ser Accidente (A) o Incidente (I) en función de su gravedad (ver Cuadro (4.1), en el que no se incluyen las variables continuas, que son *Horas de operación*, *Edad* y *Antigüedad*).

Disponemos de 62 observaciones de las variables recogidas mencionadas en el Cuadro (4.1), además de las variables continuas. Lo que vamos a hacer es, a partir de los datos de los que disponemos, aprender distintas estructuras de red bayesiana y posteriormente usarlas para clasificar.

Abrimos el Weka y lanzamos el *Explorer*. Lo primero que vamos a hacer es cargar los datos en el área de trabajo. Para ello, pinchamos en el botón *Open file* del entorno *Preprocess* y seleccionamos el fichero “*Encuesta Accidentes.arf*” y nos aparece la ventana que muestra la Figura (4.10). (En este fichero ya tenemos definida la variable *Suceso* (S) como la variable clase).

Variable	Valores
<i>Suceso</i>	Accidente, Incidente
<i>Hora</i>	Primera hora de la mañana, Después de comer, Dos últimas horas, Horas extra, Otras horas del día
<i>Día</i>	Lunes, Martes, Miércoles, Jueves, Viernes, Sábado, Domingo.
<i>Mes</i>	Enero, Febrero, Marzo, Abril, Mayo, Junio, Julio, Agosto, Septiembre, Octubre, Noviembre.
<i>Nacionalidad</i>	Africanos, Europa del Este, Latinoamericanos, Nacionales
<i>Tipo de contrato</i>	Obra y servicio, Temporal, Indefinido
<i>Tiempo en obra</i>	Menos de una semana, Entre una semana y un mes, Más de un mes
<i>Puesto de trabajo</i>	Puestos de trabajo que implican uso de maquinaria de gran tamaño, Puestos de trabajo en vehículo, Otros puestos de trabajo, Sin determinar
<i>Formación</i>	Formación genérica y específica, Formación genérica, Sin formación
<i>Reconocimiento previo del peligro</i>	Sí, No
<i>Factores personales</i>	Sí, No
<i>Comunidad Autónoma</i>	Castilla y León, Asturias, Comunidad Valenciana, Aragón
<i>Régimen</i>	Subcontrata, Contrata principal
<i>Evaluación del riesgo</i>	Sí, No
<i>Condiciones adecuadas</i>	Sí, No
<i>Dirección y supervisión</i>	Con supervisión y/o control de las condiciones de trabajo, Sólo recursos preventivos sin acciones de control y/o supervisión, Sin recursos preventivos

Cuadro 4.1: Tabla en la que se muestran las variables asociada al problema de accidentes e incidentes en diferentes empresas mineras

Desde esta ventana podemos conocer bastantes detalles del conjunto de datos que acabamos de cargar. Por ejemplo, el sistema nos indica que tenemos 62 registros con 19 atributos. Si seleccionamos cada uno de los atributos, conoceremos más información del atributo en cuestión: tipo (nominal o numérico), valores distintos, registros que no tienen información de ese atributo, el valor máximo y mínimo (sólo en atributos numéricos), y finalmente un histograma con información sobre la distribución de los ejemplos para ese atributo, reflejando con el uso de colores la distribución de clases de cada uno de los registros. Por ejemplo, en la Figura (4.10) podemos observar que el atributo *Na* (*Nacionalidad*) tiene cuatro valores diferentes (Nacionales (Nac), Africanos (Afr), Europa del Este (EE) y Latinoamericanos

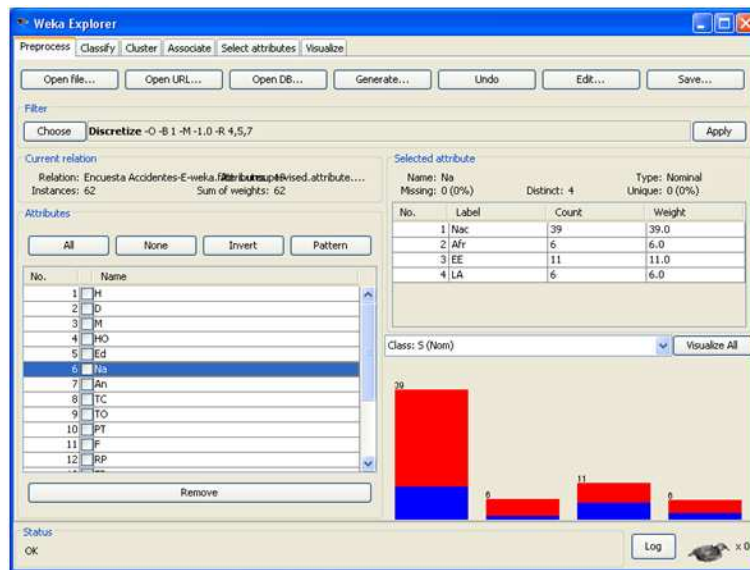


Figura 4.10: Ventana que aparece al cargar los datos.

(LA)) siendo la distribución de $[39, 6, 11, 6]$. En el caso de los 39 registros donde el atributo $Na = Nac$, tenemos 10 con clase Accidente (A) y 29 con clase Incidente (I); cuando $Na = Afr$ sólo 1 registro es A; cuando $Na = EE$, 5 registros son A y 6 son I; y finalmente cuando $Na = LA$ existen 2 con clase A, y 4 con clase I.

Pulsando en el botón *Choose* en *Filter*, hacemos los cambios necesarios en nuestro conjunto de datos para poder trabajar con ellos.

Dado que en nuestros datos tenemos variables continuas debemos aplicar un filtro para discretizarlas. Las variables continuas que tenemos son: *Horas de operación* (HO), *Edad* (Ed) y *Antigüedad* en meses (Am). (Con índices 4, 5 y 7 respectivamente).

Mediante el filtro no supervisado para atributos *Discretize* construimos variables discretas que sustituyan a las continuas, mediante intervalos de igual longitud determinados automáticamente (mediante *leave-one-out*; ver Figura (4.11)).

Después pulsamos *Apply* y guardamos el fichero mediante el botón *Save*.

A continuación, pasamos a la pestaña *Classify* para aplicar diferentes métodos de clasifi-

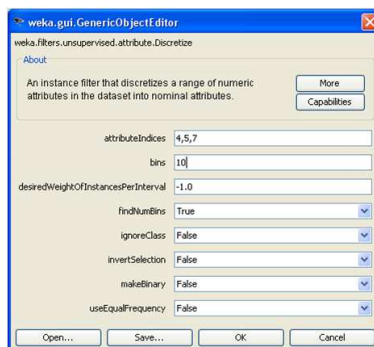


Figura 4.11: Ventana en la que se discretizan variables continuas.

cación a nuestros datos. Pinchamos en el botón *Choose* de *Classifier* y seleccionamos el método de clasificación *BayesNet* que se encuentra en la carpeta *bayes*. Pulsando sobre el recuadro que contiene el nombre del método (ver Figura (4.12)) nos aparece la ventana de la Figura (4.13). En esta ventana dejaremos las opciones que están por defecto, salvo la opción `searchAlgorithm`. Iremos mostrando los resultados que se producen según el algoritmo de búsqueda utilizado y las opciones específicas de este (en todos los casos se toma como opción de evaluación (*test options*) *Cross-validation (10 folds)*).



Figura 4.12: Ventana en la que se selecciona el método BayesNet.

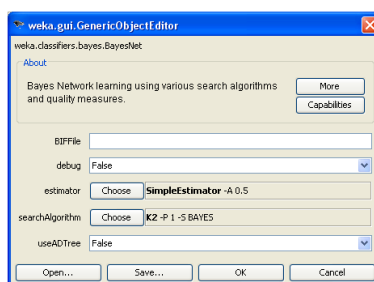


Figura 4.13: Ventana que aparece al seleccionar el recuadro de la Figura (4.12).

searchAlgorithm: Naive Bayes

```
Scheme: weka.classifiers.bayes.BayesNet -D -Q
weka.classifiers.bayes.net.search.fixed.NaiveBayes -- -E
weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
```

En las líneas anteriores dice cuales son las opciones para BayesNet.

```
Relation: Encuesta Accidentes-E-weka.filters.unsupervised.attribute.Reorder-
R2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,1-
weka.filters.unsupervised.attribute.Discretize-0-B10-M-1.0-R4,5,7
```

Instances: 62

Attributes: 19

H
D
M
H0
Ed
Na
An
TC
T0
PT
F
RP
FP
CA
R
ER
CT
DS
S

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

Bayes Network Classifier

not using ADTree

```
#attributes=19 #classindex=18
Network structure (nodes followed by parents)
H(5): S
D(6): S
M(7): S
H0(2): S
Ed(3): S
Na(4): S
An(2): S
TC(3): S
T0(3): S
PT(5): S
F(3): S
RP(2): S
FP(2): S
CA(4): S
R(2): S
ER(2): S
CT(2): S
DS(3): S
S(2):
```

Esta lista especifica la estructura de la red. Cada una de las variables está seguida por sus padres, en este caso vemos que todas las variables atributos tienen a la variable clase como padre. El número que está entre paréntesis es la cardinalidad de la variable.

```
LogScore Bayes: -992.347460178208
LogScore BDeu: -1137.4432117744402
LogScore MDL: -1141.3711783602444
LogScore ENTROPY: -965.9679669958279
LogScore AIC: -1050.967966995828
```

Esta lista muestra el logaritmo de las distintas funciones score para la estructura que se ha aprendido.

```
Time taken to build model: 0 seconds
```

```
=== Stratified cross-validation ===
```

```
=== Summary ===
```

```
Correctly Classified Instances 54 87.0968 %
Incorrectly Classified Instances 8 12.9032 %
Kappa statistic 0.6649
Mean absolute error 0.2
Root mean squared error 0.3682
Relative absolute error 48.0839 %
Root relative squared error 80.9322 %
Total Number of Instances 62
```

```
=== Detailed Accuracy By Class ===
```

```
TP Rate FP Rate Precision Recall F-Measure ROC Area Class}
0.667 0.045 0.857 0.667 0.75 0.822 A
0.955 0.333 0.875 0.955 0.913 0.822 I
Weighted Avg. 0.871 0.25 0.87 0.871 0.866 0.822
```

```
=== Confusion Matrix ===
```

```
a b <-- classified as
12 6 || a = A
2 42 || b = I
```

Por ultimo nos incluye información sobre la evaluación del modelo. En este problema, la red aprendida tiene una precisión del 87,0968 %.

También podemos visualizar la de una manera más atractiva si pulsamos el botón derecho sobre el texto *bayes.BayesNet* de la caja *Result-list*. Seleccionamos la opción *Visualize graph*, y obtendremos la red (aunque en este caso ya sabíamos de antemano como era, ver Figura (4.14)).

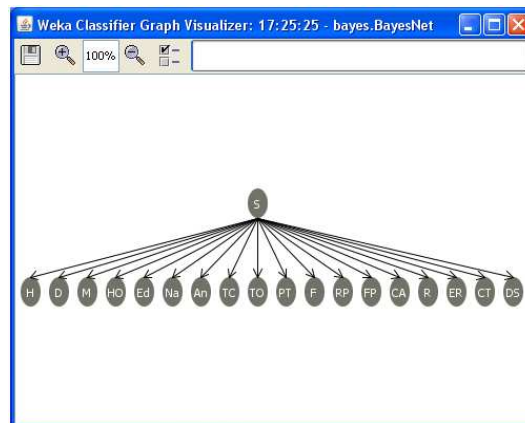


Figura 4.14: Ventana en la que se visualiza la red.

Si se pincha sobre un nodo de la red bayesiana, se abre una ventana con la tabla de probabilidad de ese nodo. A la izquierda aparecen los padres del nodo seleccionado junto con los valores que toman, a la derecha se muestra la probabilidad del nodo seleccionado condicionado a los valores de los padres (ver Figura (4.15)).

S	DU	DC	PH	HE	Ot
A	0,171	0,366	0,171	0,122	0,171
I	0,183	0,312	0,075	0,011	0,419

Figura 4.15: Ventana en la que aparecen las probabilidades condicionadas del nodo seleccionado.

searchAlgorithm: Hill Climber

El proceso es el mismo que el detallado para el algoritmo Naive Bayes (ver Figura (4.16) y (4.17)).

=== Run information ===

```
Scheme: weka.classifiers.bayes.BayesNet -D -Q
weka.classifiers.bayes.net.search.local.HillClimber -- -P 1 -S
BAYES -E weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
```

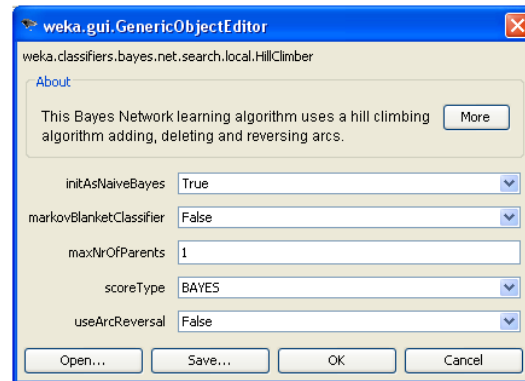


Figura 4.16: Ventana en la que se selecciona el algoritmo Hill Climber.

Relation: Encuesta

Accidentes-E-weka.filters.unsupervised.attribute.Reorder-
R2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,1-weka.filter
s.unsupervised.attribute.Discretize-0-B10-M-1.0-R4,5,7

Instances: 62

Attributes: 19

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

Bayes Network Classifier

not using ADTree

#attributes=19 #classindex=18

Network structure (nodes followed by parents)

H(5): S

D(6): TC

M(7): Ed

H0(2): S

Ed(3): S

Na(4): TC

An(2): TC

```
TC(3): Ed
TO(3): S
PT(5): An
F(3): CA
RP(2): ER
FP(2): S
CA(4): S
R(2): S
ER(2): An
CT(2): TO
DS(3): PT
S(2):
LogScore Bayes: -935.3868781576217
LogScore BDeu: -1168.370205641575
LogScore MDL: -1170.420196947319
LogScore ENTROPY: -937.2371041922713
LogScore AIC: -1050.2371041922713
```

```
Time taken to build model: 0.08 seconds
```

```
=== Stratified cross-validation ===
=== Summary ===
```

```
Correctly Classified Instances 52 83.871%
Incorrectly Classified Instances 10 16.129%
Kappa statistic 0.5658
Mean absolute error 0.217
Root mean squared error 0.3823
Relative absolute error 52.1721%
Root relative squared error 84.0445%
Total Number of Instances 62
```

```
=== Detailed Accuracy By Class ===
```

```
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
```

```

0.556 0.045 0.833 0.556 0.667 0.746 A
0.955 0.444 0.84 0.955 0.894 0.747 I
Weighted Avg. 0.839 0.329 0.838 0.839 0.828 0.747

```

```

=== Confusion Matrix ===

```

```

a b <-- classified as
10 8 | a = A
2 42 | b = I

```

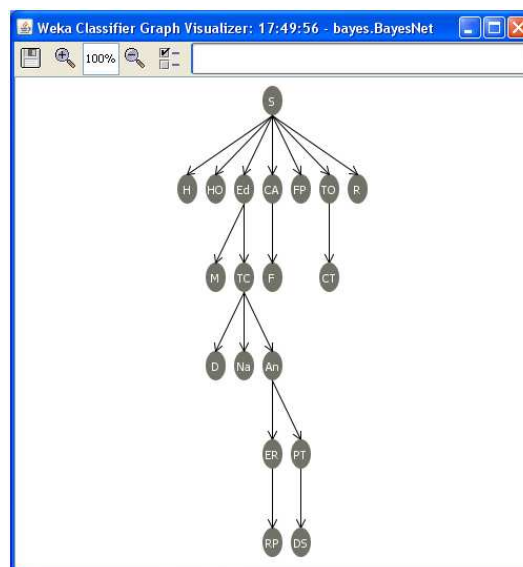


Figura 4.17: Ventana en la que se visualiza la red.

searchAlgorithm: K2

El proceso es el mismo que el detallado para el algoritmo Naive Bayes (ver Figura (4.18) y (4.19)).

```

=== Run information ===

```

```

Scheme: weka.classifiers.bayes.BayesNet -D -Q
weka.classifiers.bayes.net.search.local.K2 -- -P 2 -S MDL -E

```

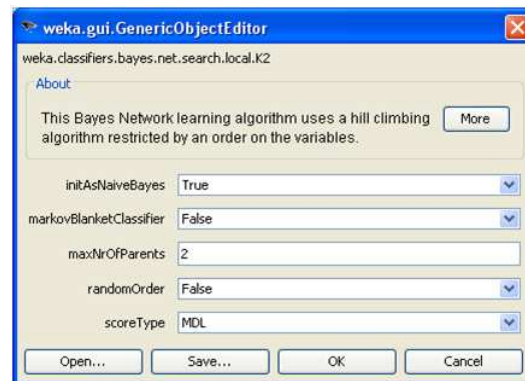


Figura 4.18: Ventana en la que se selecciona el algoritmo K2.

```
weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
```

Relation: Encuesta

```
Accidentes-E-weka.filters.unsupervised.attribute.Reorder-
R2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,1-weka.filter
s.unsupervised.attribute.Discretize-0-B10-M-1.0-R4,5,7
```

Instances: 62

Attributes: 19

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

Bayes Network Classifier

not using ADTree

#attributes=19 #classindex=18

Network structure (nodes followed by parents)

H(5): S

D(6): S

M(7): S

H0(2): S

Ed(3): S

Na(4): S
An(2): S
TC(3): S
T0(3): S
PT(5): S
F(3): S
RP(2): S
FP(2): S
CA(4): S
R(2): S CA
ER(2): S RP
CT(2): S RP
DS(3): S
S(2):

LogScore Bayes: -975.2416634593861
LogScore BDeu: -1146.5533081722808
LogScore MDL: -1148.50023320751
LogScore ENTROPY: -952.4613499178679
LogScore AIC: -1047.4613499178681

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances 51 82.2581%
Incorrectly Classified Instances 11 17.7419%
Kappa statistic 0.5136
Mean absolute error 0.2111
Root mean squared error 0.3785
Relative absolute error 50.7567%
Root relative squared error 83.1939%
Total Number of Instances 62

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.5	0.045	0.818	0.5	0.621	0.827	A
0.955	0.5	0.824	0.955	0.884	0.827	I
Weighted Avg.						0.823 0.368 0.822 0.823 0.808 0.827

=== Confusion Matrix ===

```
a b <-- classified as
9 9 | a = A
2 42 | b = I
```

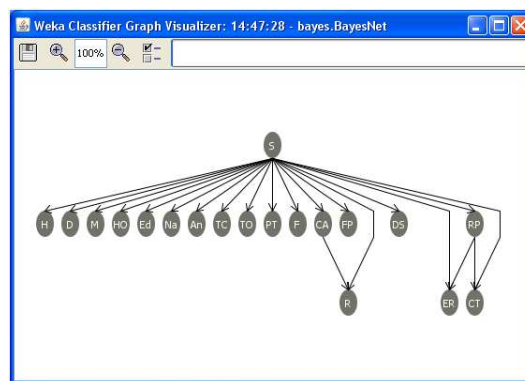


Figura 4.19: Ventana en la que se visualiza la red.

searchAlgorithm: TAN

El proceso es el mismo que el detallado para el algoritmo Naive Bayes (ver Figura (4.20) y (4.21)).

=== Run information ===

```
Scheme: weka.classifiers.bayes.BayesNet -D -Q
weka.classifiers.bayes.net.search.local.TAN -- -S BAYES -E
weka.classifiers.bayes.net.estimate.SimpleEstimator -- -A 0.5
```

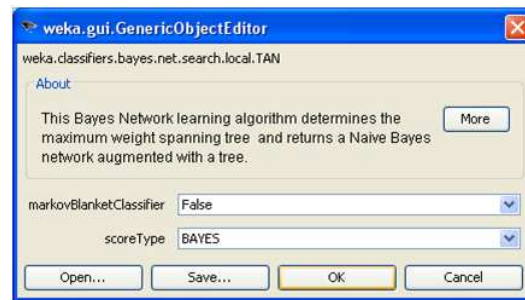


Figura 4.20: Ventana en la que se selecciona el algoritmo TAN.

Relation: Encuesta

Accidentes-E-weka.filters.unsupervised.attribute.Reorder-
R2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,1-weka.filter
s.unsupervised.attribute.Discretize-0-B10-M-1.0-R4,5,7

Instances: 62

Attributes: 19

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

Bayes Network Classifier

not using ADTree

#attributes=19 #classindex=18

Network structure (nodes followed by parents)

H(5): S M

D(6): S TC

M(7): S

H0(2): S CA

Ed(3): S M

Na(4): S TC

An(2): S TC

TC(3): S Ed

T0(3): S Na
 PT(5): S TC
 F(3): S M
 RP(2): S CT
 FP(2): S Na
 CA(4): S T0
 R(2): S CA
 ER(2): S RP
 CT(2): S T0
 DS(3): S PT
 S(2):

LogScore Bayes: -955.8933895537903
 LogScore BDeu: -1907.5410660279242
 LogScore MDL: -1709.6164010566124
 LogScore ENTROPY: -1076.1012729521908
 LogScore AIC: -1383.101272952191

Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
 === Summary ===

Correctly Classified Instances 48 77.4194%
 Incorrectly Classified Instances 14 22.5806%
 Kappa statistic 0.4135
 Mean absolute error 0.2617
 Root mean squared error 0.4203
 Relative absolute error 62.9135%
 Root relative squared error 92.3956%
 Total Number of Instances 62

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure ROC Area Class

```

0.5 0.114 0.643 0.5 0.563 0.761 A
0.886 0.5 0.813 0.886 0.848 0.761 I
Weighted Avg. 0.774 0.388 0.763 0.774 0.765 0.761

```

```

=== Confusion Matrix ===

```

```

a b <-- classified as
9 9 | a = A
5 39 | b = I

```

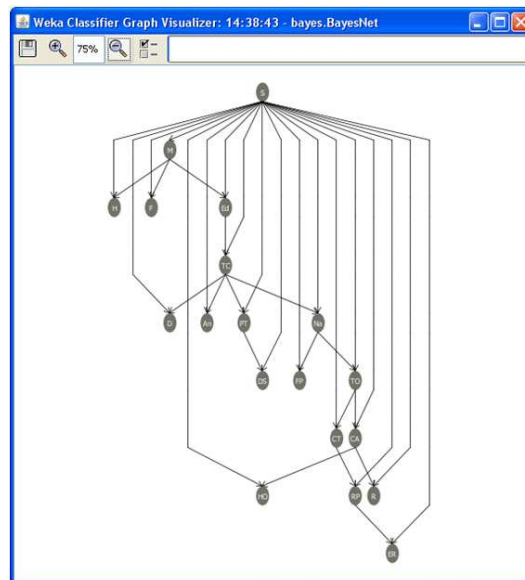


Figura 4.21: Ventana en la que se visualiza la red.

searchAlgorithm: ICS

El proceso es el mismo que el detallado para el algoritmo Naive Bayes (ver Figura (4.22)).

```

=== Run information ===

```

```

Scheme: weka.classifiers.bayes.BayesNet -D -Q
weka.classifiers.bayes.net.search.ci.ICSSearchAlgorithm -- -S
BAYES -cardinality 2 -E weka.classifiers.bayes.net.estimate.SimpleEstimator --

```

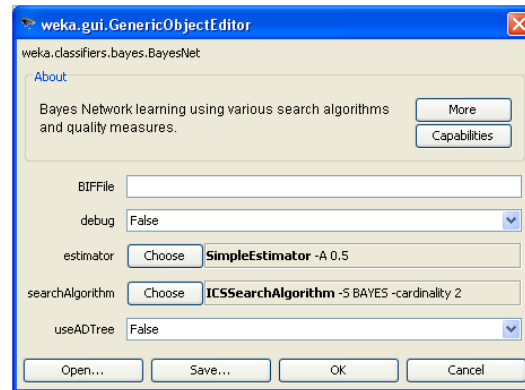


Figura 4.22: Ventana en la que se selecciona el algoritmo ICS.

-A 0.5

Relation: Encuesta

Accidentes-E-weka.filters.unsupervised.attribute.Reorder-
R2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,1-weka.filter
s.unsupervised.attribute.Discretize-0-B10-M-1.0-R4,5,7

Instances: 62

Attributes: 19

Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

Bayes Network Classifier

not using ADTree

#attributes=19 #classindex=18

Network structure (nodes followed by parents)

H(5): D M R

D(6):

M(7): H Ed F

H0(2):

Ed(3): M TC S

Na(4):
An(2): TC PT ER CT
TC(3): Ed An PT FP
TO(3): PT FP CA ER CT
PT(5): An TC TO RP DS
F(3): M CA
RP(2): PT ER CT
FP(2): Na TC TO S
CA(4): HO TO R
R(2):
ER(2): An TO RP
CT(2): An TO RP
DS(3):
S(2): Ed FP CA

LogScore Bayes: -955.1307961360313
LogScore BDeu: -11007.626158670493
LogScore MDL: -6359.781412940882
LogScore ENTROPY: -2509.165031693762
LogScore AIC: -4375.165031693796

Time taken to build model: 1.16 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances 38 61.2903%
Incorrectly Classified Instances 24 38.7097%
Kappa statistic -0.0054
Mean absolute error 0.3857
Root mean squared error 0.5069
Relative absolute error 92.7182%
Root relative squared error 111.4131%
Total Number of Instances 62

```
=== Detailed Accuracy By Class ===
```

```
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
0.222 0.227 0.286 0.222 0.25 0.582 A
0.773 0.778 0.708 0.773 0.739 0.581 I
Weighted Avg. 0.613 0.618 0.586 0.613 0.597 0.582
```

```
=== Confusion Matrix ===
```

```
a b <-- classified as
4 14 | a = A
10 34 | b = I
```

Estos son sólo algunos ejemplos en los que se construye, a partir de unos datos, una red bayesiana y después se utiliza para clasificar.

No podemos extraer conclusiones ya que nos hemos limitado a construir redes bayesianas con distintos algoritmos de búsqueda, para ver qué estructura de red proporcionaban (más o menos compleja) y ver cuál era la precisión de la clasificación.

4.2. Análisis de datos con GeNIe

GeNIe es un entorno desarrollado para la creación de modelos gráficos de decisión. Ha sido y continúa desarrollándose en el Decision Systems Laboratory, en la Universidad de Pittsburgh. GeNIe permite construir modelos de cualquier tamaño y complejidad, limitado únicamente por la capacidad de la memoria de operación del ordenador.

Consideremos el problema siguiente: En el negocio de los créditos los bancos se encuentran interesados en obtener información que les revele si el cliente pagará el crédito o, por el contrario, sea un moroso. El objetivo de una calificación del riesgo crediticio es modelizar o predecir la probabilidad de que un cliente con unas ciertas características se debe considerar como un riesgo potencial.

Consideraremos un conjunto de datos en el que se incluyen 1000 créditos de un banco alemán. Para cada cliente se dispone de la variable respuesta “solvencia”. Además, se consideran otras 20 variables que se supone influyen en la solvencia de la operación (ver Cuadros (4.2) y (4.3)).

Utilizaremos este programa para describir los procesos de aprendizaje estructural y paramétrico de una red bayesiana construida sobre los datos que acabamos de describir.

En primer lugar hacemos una lectura de los datos (Figura (4.23)). Para cargar el archivo de datos escogemos la opción *File*→*Open Data File...* del menú principal. GeNIe utiliza una visualización de los datos en forma de cuadrícula que permite a los usuarios trabajar con los mismos como si se tratase de una hoja de cálculo.

Solvencia	Balance	Duración crédito	Previos	Objetivo	Importe crédito	Valor ahorros/reservas
1	1	18	4	2	1049	1
1	1	9	4	0	2799	1
1	2	12	2	9	841	2
1	1	12	4	0	2122	1
1	1	12	4	0	2171	1
1	1	10	4	0	2241	1
1	1	8	4	0	3388	1
1	1	6	4	0	1361	1
1	4	18	4	3	1098	1
1	2	24	2	3	3758	3
1	1	11	4	0	3905	1
1	1	30	4	1	6187	2
1	1	6	4	3	1957	1
1	2	48	3	10	7582	2
1	1	18	2	3	1936	5
1	1	6	2	3	2647	3
1	1	11	4	0	3539	1
1	2	18	2	3	3213	3
1	2	36	4	3	2337	1
1	4	11	4	0	7228	1
1	1	6	4	0	3676	1
1	2	12	4	0	3124	1
0	2	36	2	5	2384	1
1	2	12	4	4	1424	1
1	1	6	4	0	4716	5
1	2	11	3	3	4771	1
1	1	12	2	2	652	1
1	2	9	4	3	1154	1
1	4	15	2	0	3556	5
1	3	42	4	1	4796	1
1	3	30	4	3	3017	1
1	4	36	4	0	3535	1
1	4	36	4	0	6614	1
1	4	24	2	3	1376	3

Figura 4.23: Ventana de Genie en la que se muestran los datos.

Una vez se ha cargado el fichero se procede al preprocesado de los mismos. Consideraremos el siguiente análisis:

- *Valores faltantes:*

Para seleccionar las filas que contienen valores faltantes se debe seleccionar la opción *Data*→*Missing Values*→*Select* del menú principal. Se ofrecen entonces dos opciones: (1) seleccionar todas las filas que contienen valores faltantes ó (2) seleccionar las filas con valores faltantes sólo en la columna seleccionada (para seleccionar una columna basta con desplazar el cursor a una de sus celdas). Si el archivo de datos no contiene

Variable	Valores
<i>Solvencia</i>	Sí,No
<i>Balance de cuenta corriente (en DM)</i>	No hay balance o débito, $0 \leq \dots \leq 200$, $\dots \geq 200$ o cuenta corriente por al menos un año, ninguna cuenta corriente
<i>Duración del crédito (en meses)</i>	$\leq 6, 6 \leq \dots \leq 12, 12 \leq \dots \leq 18, 18 \leq \dots \leq 24$, $24 \leq \dots \leq 30, 30 \leq \dots \leq 36, 36 \leq \dots \leq 42, 42 \leq \dots \leq 48$, $48 \leq \dots \leq 54, \geq 54$
<i>Pago de créditos anteriores</i>	Ningún crédito anterior/pago de todos los créditos, pagados los créditos a este banco, ningún problema con los créditos anteriores en este banco, pagamiento dudoso de los créditos, problemáticas en la cuenta/otros créditos en otros bancos.
<i>Objetivo del crédito</i>	Coche nuevo, coche de segunda mano, mobiliario, radio/televisión, hipoteca, reparación, educación, vacaciones, reciclaje, negocios, otros.
<i>Cantidad del crédito</i>	$\leq 500, 500 \leq \dots \leq 1000, 1000 \leq \dots \leq 1500$, $1500 \leq \dots \leq 2500, 2500 \leq \dots \leq 5000, 5000 \leq \dots \leq 7500$, $7500 \leq \dots \leq 10000, 10000 \leq \dots \leq 15000$, $15000 \leq \dots \leq 20000, \geq 20000$
<i>Valor de las propiedades</i>	$< 100, -DM, 100, - \leq \dots \leq 500, -DM$, $500, - \leq \dots \leq 1000, -DM, \geq 1000, -DM$, no se sabe/no propiedades.
<i>Duración trabajo actual (en años)</i>	desempleado, $\leq 1, 1 \leq \dots \leq 4, 4 \leq \dots \leq 7, \geq 7$.
<i>Cuota en % de la renta disponible</i>	$\geq 35, 25 \leq \dots \leq 35, 20 \leq \dots \leq 25, < 20$
<i>Estado civil y sexo</i>	hombre:divorciado/viviendo sólo, mujer:divorciada/viviendo sólo/casada, hombre:soltero, hombre:casado/viúdo, mujer:soltera.
<i>Futuros deudores/avalistas</i>	ninguno, co-solicitante, avalista.
<i>Tiempo viviendo en casa actual (en años)</i>	$< 1, 1 \leq \dots < 4, 4 \leq \dots < 7, > 7$

Cuadro 4.2: Cuadro de las variables y valores asociados para el problema de *credit-scoring*
- PARTE I.

<i>Posesiones más valuosas</i>	Propiedad de casa o tierra, seguro de vida, coche, otros, no disponible/ninguno.
<i>Edad</i> (en años)	$0 \leq \dots \leq 25, 26 \leq \dots \leq 39, 49 \leq \dots \leq 59,$ $60 \leq \dots \leq 64, > 64$
<i>Otros créditos</i>	En otros bancos, ninguno.
<i>Vivienda</i>	Piso de alquiler, piso en propiedad, otro.
<i>Número de créditos en este banco</i> <i>-incluyendo el actual-</i>	Uno, dos o tres, cuatro o cinco, seis o más.
<i>Ocupación</i>	Desempleado/no cualificado sin residencia permanente, no cualificado con residencia permanente, trabajador cualificado, funcionario menor, ejecutivo/autónomo/funcionario superior.
<i>Número de personas con derecho a manutención</i>	De cero a dos, tres o más.
<i>Teléfono</i>	Sí, no.
<i>Extranjero</i>	Sí, no.

Cuadro 4.3: Cuadro de las variables y valores asociados para el problema de *credit-scoring* - PARTE II.

ningún valor faltante el propio programa nos informa. En otro caso GeNIe nos informa de el número de filas que se seleccionan (y aparecen resaltadas en la tabla de datos).

Si el fichero de datos contiene algún valor faltante se puede reemplazar seleccionando *Data*→*Missing Values*→*Replace* del menú principal. Se puede reemplazar (1) con un valor específico ó (2) con una media de la columna seleccionada. Los valores reemplazados se distinguirán con un color rojo.

Para eliminar todos los reemplazamientos en alguna de las columnas basta con seleccionar la columna y seleccionar la opción *Data*→*Missing Values*→*Restore* del menú principal. Los valores insertados serán eliminados de la tabla de datos.

En nuestro conjunto de datos no existen valores faltantes, por lo que no tendremos que acudir a esta opción.

- *Discretización:*

GeNIe proporciona una herramienta para discretizar valores. Para invocar la interfaz

de discretización se debe seleccionar la opción *Data*→*Discretize* del menú principal. En la ventana que nos aparece para realizar la discretización se puede seleccionar la forma en que se desea realizar: (1) de forma jerárquica, (2) utilizando anchos uniformes ó (3) estableciendo que todos los valores de la columna se distribuyan de manera uniforme (es decir, el mismo número de valores en cada grupo). Una vez seleccionado el método se selecciona el número de *bins* y se pulsa el botón de discretizar. Una vez discretizada la columna aparece en color azul en la tabla de datos.

Para eliminar la discretización se selecciona la opción *Data*→*Stop Discretization* del menú principal.

En nuestro archivo de datos las variables *Importe del crédito*, *Duración del crédito* y *Edad* son continuas, por lo que deben discretizarse. Las ventanas de discretización asociadas a cada una de las variables se muestran, respectivamente, en las Figuras (4.24), (4.25) y (4.26).

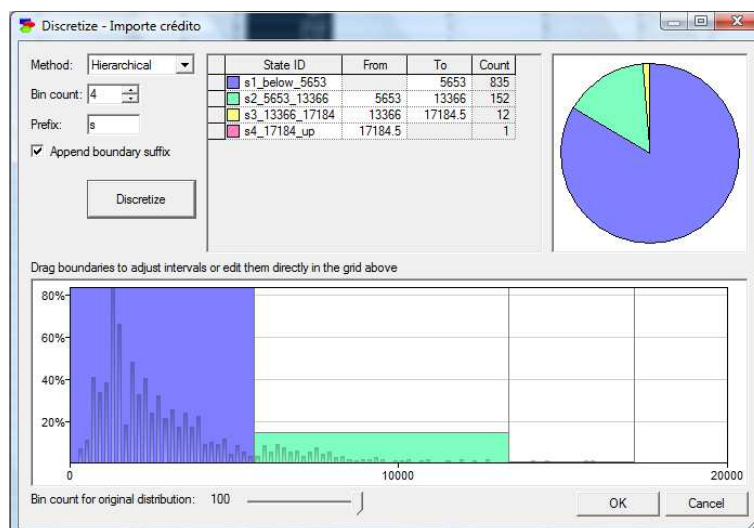


Figura 4.24: Ventana de Genie en la que se discretiza la variable Importe del crédito.

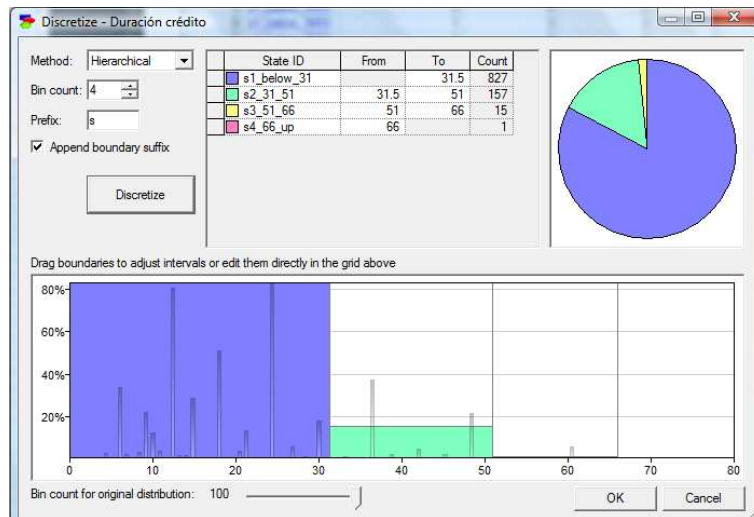


Figura 4.25: Ventana de Genie en la que se discretiza la variable Duración del crédito.

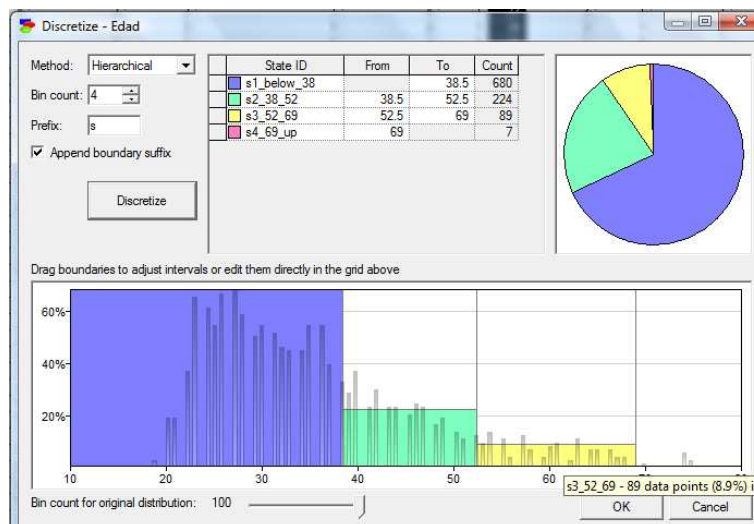
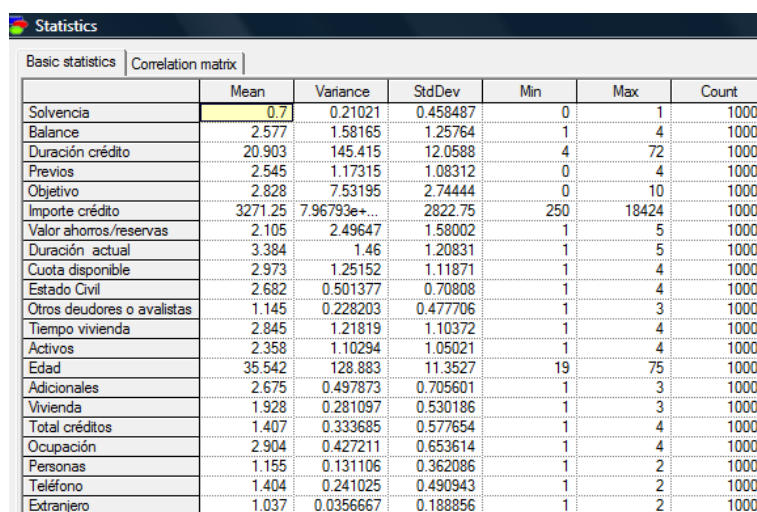


Figura 4.26: Ventana de Genie en la que se discretiza la variable Edad.

■ *Estadísticos:*

GeNIe permite mostrar algunos estadísticos útiles para cada una de las columnas de la tabla de datos. Estos estadísticos son: media, varianza, desviación típica, mínimo, máximo y número de valores la columna. Se puede mostrar también la matriz de correlación. Para realizar esta acción se debe seleccionar la opción *Data->Statistics* del menú principal.

Para nuestros datos, la ventana en la que se muestran los estadísticos aparecen en la Figura (4.27).



	Mean	Variance	StdDev	Min	Max	Count
Solvencia	0.7	0.21021	0.458487	0	1	1000
Balance	2.577	1.58165	1.25764	1	4	1000
Duración crédito	20.903	145.415	12.0588	4	72	1000
Previos	2.545	1.17315	1.08312	0	4	1000
Objetivo	2.828	7.53195	2.74444	0	10	1000
Importe crédito	3271.25	7.96793e+...	2822.75	250	18424	1000
Valor ahorros/reservas	2.105	2.49647	1.58002	1	5	1000
Duración actual	3.384	1.46	1.20831	1	5	1000
Cuota disponible	2.973	1.25152	1.11871	1	4	1000
Estado Civil	2.682	0.501377	0.70808	1	4	1000
Otros deudores o avalistas	1.145	0.228203	0.477706	1	3	1000
Tiempo vivienda	2.845	1.21819	1.10372	1	4	1000
Activos	2.358	1.10294	1.05021	1	4	1000
Edad	35.542	128.883	11.3527	19	75	1000
Adicionales	2.675	0.497873	0.705601	1	3	1000
Vivienda	1.928	0.281097	0.530186	1	3	1000
Total créditos	1.407	0.333685	0.577654	1	4	1000
Ocupación	2.904	0.427211	0.653614	1	4	1000
Personas	1.155	0.131106	0.362086	1	2	1000
Teléfono	1.404	0.241025	0.490943	1	2	1000
Extranjero	1.037	0.0356667	0.188856	1	2	1000

Figura 4.27: Ventana de Genie en la que se muestran los estadísticos de todas las variables.

- *Histogramas:*

Para ver la distribución de los valores en columnas en un histograma podemos seleccionar *Data*→*Histogram* del menú principal. Exponemos aquí los histogramas asociados a las variables continuas que hemos discretizado, realizándose el histograma sobre la variable continua. Los histogramas asociados al *Importe del crédito*, a la *Duración del crédito* y a la *Edad* se muestran en las Figuras (4.28), (4.29) y (4.30), respectivamente.

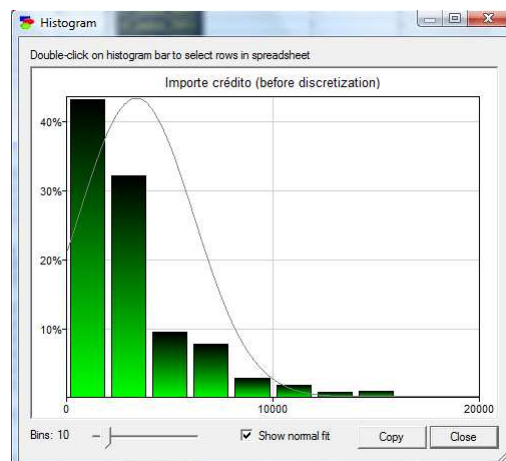


Figura 4.28: Ventana de Genie en la que se muestra el histograma de la variable Importe del crédito.

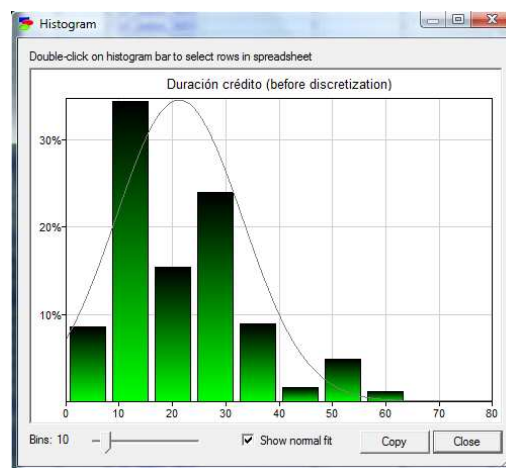


Figura 4.29: Ventana de Genie en la que se muestra el histograma de la variable Duración del crédito.

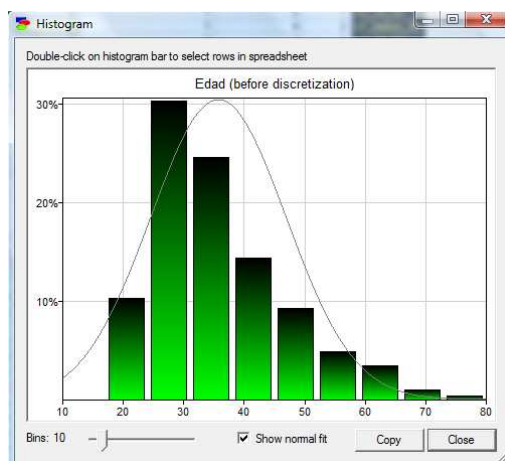


Figura 4.30: Ventana de Genie en la que se muestra el histograma de la variable Edad.

Otras aplicaciones que se pueden realizar es la fusión de estados y la realización de diagramas de dispersión o series de tiempo. El procedimiento es muy similar al descrito en los apartados anteriores y omitiremos los detalles aquí.

Una vez realizado el preproceso se procede al aprendizaje de la estructura. A esta opción se accede mediante *Data->Learn New Network*.

En la ventana que aparece se debe seleccionar las variables (columnas) que se desea que participen en el aprendizaje y el método que se desea utilizar. El método en el que nos centraremos nosotros en el *Naive Bayes*, utilizando como variable clase la *Solvencia*. La ventana en que se realiza la selección mencionada aparece en la Figura (4.31).

La red resultante viene dada en la Figura (4.32).

A continuación y por último procederemos al aprendizaje de parámetros sobre la estructura definida. Para acceder a esta opción seleccionamos *Data->Learn Parameters* del menú principal.

A continuación se procede a crear una especie de trazado de mapas entre las variables definidas en la red (columna de la izquierda) y las variables definidas en el conjunto de datos (columna de la derecha). No haremos ninguna modificación en esta parte, por lo que la ventana relacionada con esta parte queda como viene dada en la Figura (4.33).

Una vez se pulsa Ok se puede definir el nivel de confianza del parámetro (ver Figura (4.34))

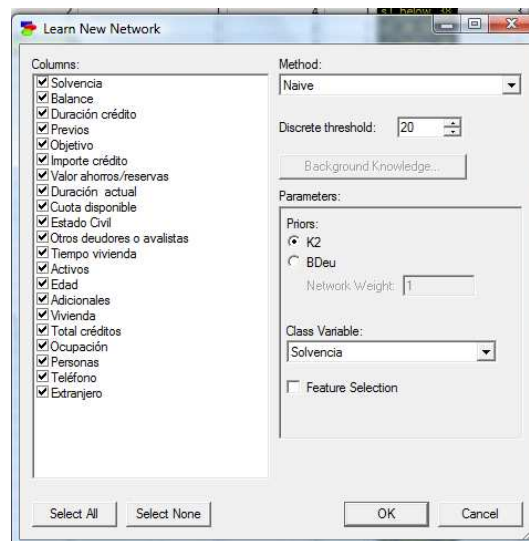


Figura 4.31: Ventana de Genie en la que se realiza la selección de variables que se desean incluir en la red.

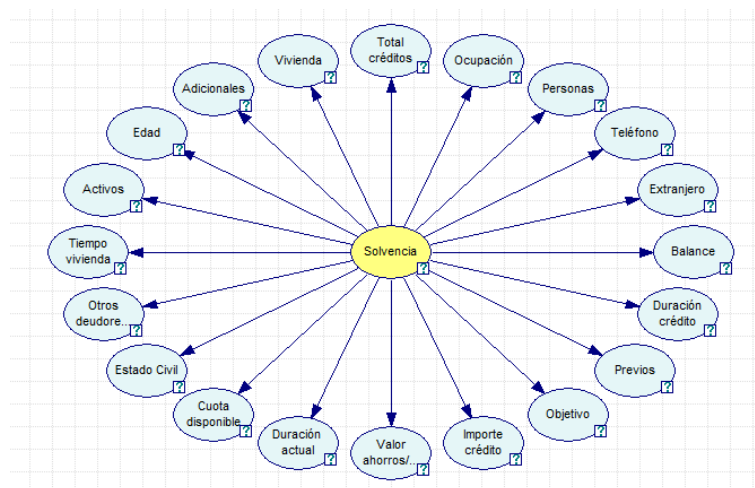


Figura 4.32: Ventana de Genie en la que se muestra la red resultante.

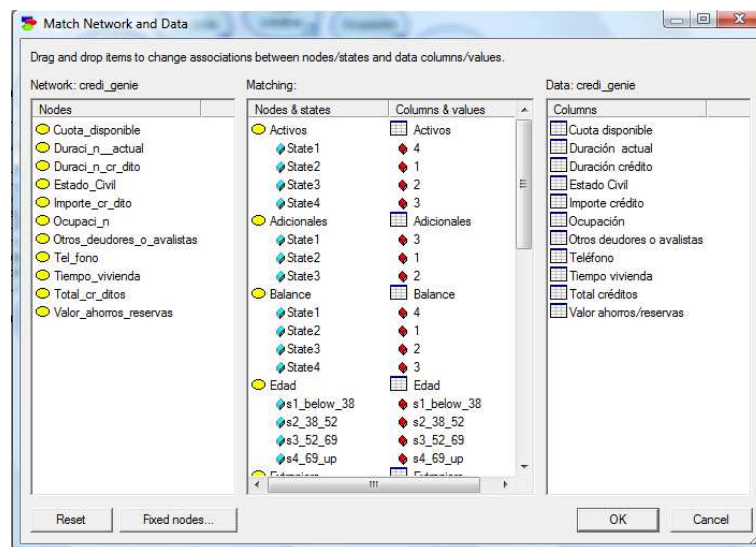


Figura 4.33: Ventana de Genie en la que se realiza el trazado de mapas para el aprendizaje paramétrico.

y finalmente en la Figura (4.35) tenemos el resultado del proceso de aprendizaje.

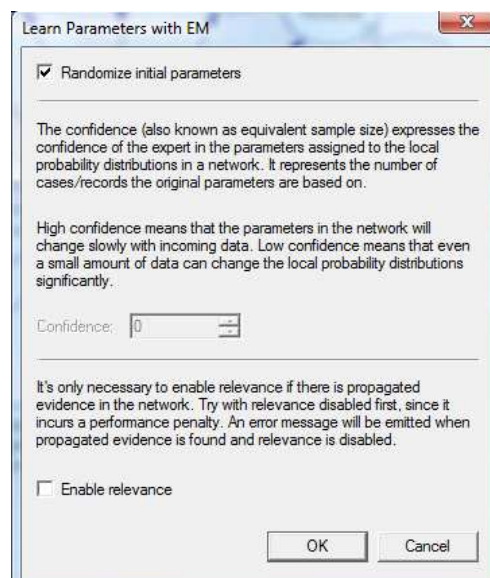


Figura 4.34: Ventana de Genie en la que se selecciona el nivel de confianza del parámetro.

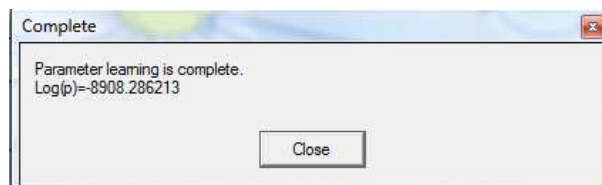


Figura 4.35: Ventana de Genie en la que se muestra el resultado del proceso de aprendizaje.

4.3. Análisis de datos con R

R es un entorno de software libre para el cómputo estadístico y resoluciones gráficas. Se trata de un proyecto GNU y es similar al lenguaje y entorno de S desarrollado en los laboratorios Bell por John Chambers y colegas. Existen diferencias importantes entre R y S, pero muchos de los códigos realizados en S funciona en R inalterados.

En este apartado nombraremos tan sólo los paquetes y/o funciones que se deberían utilizar para realizar un aprendizaje estructural y paramétrico de una red bayesiana, así como la clasificación por el método de Naive Bayes. Supuesta una base de datos, dejaremos el ejercicio de programación abierto a la posible realización del lector.

Supongamos que se desea realizar el análisis de la base de datos *ksl* contenida en el paquete *deal*. Este conjunto de datos proviene de un estudio de medición de características sociales y de la salud sobre muestras representativas de daneses de 70 años de edad recogidas en los años 1967 y en 1984. En total se recogieron 1083 casos y cada uno contiene observaciones de nueve variables diferentes (ver Tabla 4.4).

El interés será determinar qué variables influyen en la presencia o ausencia de *Hyp*.

4.3.1. Aprendizaje de redes bayesianas

Para la realización de esta tarea se pueden utilizar las funciones dadas por los paquetes *deal* y *bnlearn*.

Nodo	Variable	Explicación
1	Fev	<i>Forced ejection volume - lung function</i>
2	Kol	<i>Cholesterol</i>
3	Hyp	<i>Hypertension (no/yes)</i>
4	BMI	<i>Body Mass Index</i>
5	Smok	<i>Smoking (no/yes)</i>
6	Alc	<i>Alcohol consumption (seldom/frequently)</i>
7	Work	<i>Working (yes/no)</i>
8	Sex	<i>Gender (male/female)</i>
9	Year	<i>Survey year (1967/1984)</i>

Cuadro 4.4: Variables que aparecen en la base de datos *ksl*.

4.3.2. Clasificadores

Para realizar la clasificación utilizando el método Naive Bayes se puede utilizar la función *NaiveBayes* del paquete **klaR**.

Bibliografía

- [1] Ben-Gal, I. (2007) *Bayesian Networks*, in Ruggeri F., Faltin F. and Kenett R. Encyclopedia of Statistics in Quality and Reliability, Wiley and Sons.
- [2] Bouckaert, R.R. (2004) *Bayesian Network Classifiers in Weka for Version 3-5-8*. University of Waikato. <http://switch.dl.sourceforge.net/project/weka/documentation/3.5.x/ExplorerGuide>
- [3] Decision Systems Laboratory (2009) *GeNIe, The genie inside! and SMILE, SMILE on board!*. Decision Systems Laboratory, University of Pittsburgh, USA. <http://genie.sis.pitt.edu>.
- [4] Heckerman, David (1995) *Learning Bayesian Networks: The Combination of Knowledge and Statistical Data*. Machine Learning, **20**, 197 – 243.
- [5] Heckerman, David (1997) *Bayesian Networks for Data Mining*. Data Mining and Knowledge Discovery, **1**, 79 – 119.
- [6] Hernández Orallo, J. et al. (2007) *Introducción a la minería de datos*. Pearson-Prentice Hall.
- [7] Jensen, F.V and Nielsen, T.D (2007) *Bayesian Networks and Decision Graphs*. Springer-Information Science and Statistics.
- [8] López de Castilla, C. (2005) *Clasificadores por Redes Bayesianas*. Tesis sometida en cumplimiento parcial de los requisitos para el grado de MAESTRO EN CIENCIAS en MATEMATICA (Estadística), Universidad de Puerto Rico.
- [9] Pardalos, Panos M. *Bayesian Networks*. Universidad de Florida. <http://www.ise.ufl.edu/cao/DMinAgriculture/Lecture3.bayesian.pdf>.
- [10] R Development Core Team (2009) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. <http://www.R-project.org>.

- [11] Sierra Araujo, B. (2006) *Aprendizaje automático: conceptos básicos y avanzados: aspectos prácticos utilizando el software Weka*. Pearson-Prentice Hall.