TRABAJO DE FIN DE GRADO

Estimación paramétrica y validación en redes Bayesianas

Realizado por: Elena Mellado Cabrerizo

Supervisado por: Eduardo Conde Sánchez

FACULTAD DE MATEMÁTICAS
DPTO DE ESTADÍSTICA E INVESTIGACIÓN OPERATIVA

Índice general

Introducción **3**

1. Introducción a las redes Bayesianas **5**
1.1. Elementos de la Teoría de Grafos 5
1.2. Redes Bayesianas: Definiciones y propiedades esenciales . . . 7
1.3. Estructuras equivalentes y envolvente de Markov 9
1.4. Modelado de redes Bayesianas 10
1.5. Naturaleza de los datos 12

2. Estimación paramétrica de una red de Bayes **13**
2.1. Estimación de máxima verosimilitud 13
2.1.1. Descripción 13
2.1.2. Propiedades 14
2.1.3. Estimación de MV de distribuciones discretas 15
2.2. Estimación mediante funciones de riesgo 20
2.2.1. Estimación minimax 22
2.2.2. Estimación Bayesiana 22
2.3. Uso de la librería *bnlearn* del *software R* en la estimación de parámetros 31
2.3.1. Estructura y aprendizaje de una red Bayesiana 31
2.3.2. Estimación paramétrica con *bnlearn* 32

3. Validación de la red de Bayes y aplicación experimental **34**
3.1. Uso del método *bootstrap* con el *software R* 34
3.2. Validación cruzada con el *software R* 36
3.3. Aplicación experimental 39

A. Scripts del *software R* **61**

Resumen

En este Trabajo de Fin de Grado estudiamos con detalle varios métodos de estimación paramétrica (máxima verosimilud, minimax y estimación Bayesiana) en un modelo grafo probabilístico llamado red Bayesiana. También analizamos dos de las técnicas más utilizadas para validar modelos estadísticos (*bootstrap* y validación cruzada). El trabajo concluye con una aplicación experimental en el contexto de diagnóstico médico realizada con el *software R*.

Abstract

In this Final Undergraduate Project we study with detail several estimation methods (maximum likelihood, minimax and Bayesian estimation) in a probabilistic graphical model called Bayesian Network. We also analyse two of the most common techniques used to validate statistical models (bootstrap and cross-validation). The project ends with an experimental application in the context of medical diagnosis carried out with the *software R*.

Introducción

Uno de los objetivos fundamentales del Modelado Estadístico es representar las relaciones estadísticas existentes entre un determinado conjunto de variables aleatorias y explicarlas de forma clara. A través de la distribución conjunta del vector aleatorio bajo estudio, las redes Bayesianas (BNs) nos permiten identificar relaciones estadísticas de dependencia simplemente observando la estructura de la red.

Trabajar con un gran conjunto de variables y sus parámetros no es tarea fácil. A pesar de ello, los avances informáticos de los últimos años han facilitado la tarea de las redes Bayesianas, y esto ha hecho que hayan cobrado una gran importancia en aplicaciones estadísticas en ingeniería, gestión de riesgos, toma de decisiones o diagnóstico médico.

En este trabajo, introducimos los elementos y la estructura de una red Bayesiana. Nos limitamos al caso de variables aleatorias discretas. Esta familia de variables cubren gran parte de las aplicaciones prácticas en los campos previamente comentados.

En aplicaciones reales, las redes Bayesianas pueden ser modeladas por expertos o aprendidas a través de bases de datos. En el primer capítulo, indicamos varias técnicas utilizadas para el aprendizaje topológico de la red. Una vez la red ha sido fijada, se puede proceder a la estimación de los parámetros de la misma.

El segundo capítulo, el principal de este trabajo, está dedicado a explicar los métodos de estimación en el caso particular de redes Bayesianas discretas. Se estudian con detalle los métodos de máxima verosimilitud, *minimax* y estimación Bayesiana. Terminamos este capítulo explicando cómo utilizar estos métodos en el software estadístico *R*.

Una vez construido el modelo estadístico, es importante validar sus elementos para así comprobar la bondad de ajuste del mismo frente a datos reales. En el último capítulo presentamos dos técnicas de validación en redes Bayesianas: el método *bootstrap* y la validación cruzada. Para finalizar, mos-

tramos cómo emplear todos los conceptos estudiados a lo largo del trabajo en una aplicación experimental en el contexto de diagnóstico médico.

Capítulo 1

Introducción a las redes Bayesianas

En este capítulo definiremos el concepto de *red Bayesiana* y explicaremos, aunque no de manera muy detallada, los algoritmos que se llevan a cabo para el aprendizaje de la estructura de una red. Finalmente, destacaremos dos tipos de redes en función de la naturaleza de los datos. La referencia principal para este capítulo es el libro de R. Nagarajan et al. [5, Capítulos 1 y 2].

Previamente debemos tener en cuenta algunas nociones de Teoría de Grafos.

1.1. Elementos de la Teoría de Grafos

Definición 1.1.1 (Grafo)**.** *Un grafo es un par $G = (V, E)$, donde $V \neq \emptyset$ es el conjunto de vértices o nodos y $E \subseteq V \times V$ es el conjunto de arcos.*

Definición 1.1.2. *Si $(u, v) \in E$ y $(v, u) \in E$ se dice que (u,v) es un eje. Dos nodos u,v $\in V$ son adyacentes o vecinos si existe $(u, v) \in E$ ó $(v, u) \in E$*

Definición 1.1.3. *Un grafo $G = (V, E)$ se dice dirigido si todos los elementos de E tienen un sentido definido. En el caso en el que todos los arcos son ejes, se dice que G es un grafo no dirigido. Si contiene arcos dirigidos y ejes, se dice que es un grafo mixto o parcialmente dirigido (véase figura 1.1)*

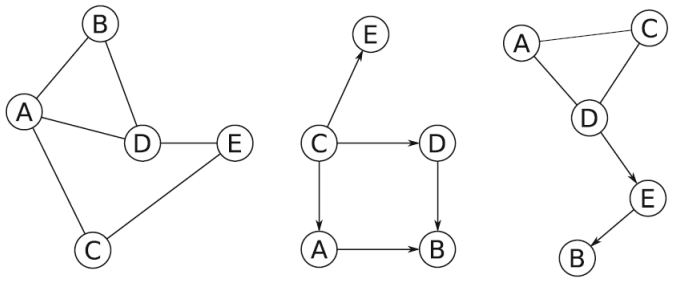


Figura 1.1: Grafo no dirigido, grafo dirigido y grafo parcialmente dirigido, respectivamente.

En lo que sigue, trabajaremos únicamente con grafos dirigidos.

Definición 1.1.4 (Camino)**.** *Se dice que $v_1 v_2 ... v_n$ es un camino entre $v_1 \in V$ y $v_n \in V$ si $(v_i, v_{i+1}) \in E$ para todo $1 \leq i \leq n-1$.*

Definición 1.1.5 (Ciclo)**.** *Un ciclo es un camino cerrado, es decir, aquel en el que $v_1 = v_n$*

Definición 1.1.6 (Grafo acíclico)**.** *Un grafo se dice acíclico si no contiene ciclos.*

Definición 1.1.7 (Relaciones entre nodos)**.** *Dado un grafo $G = (V, E)$ y un nodo $u \in V$, se dice que $v \in V$ es antecesor de u si existe un camino dirigido de v a u. En tal caso, diremos que u es descendiente de v. Se dirá que v es padre de u si son adyacentes y v es antecesor de u. Análogamente, v es hijo de u si son adyacentes y v es descendiente de u. Véase figura 1.2.*

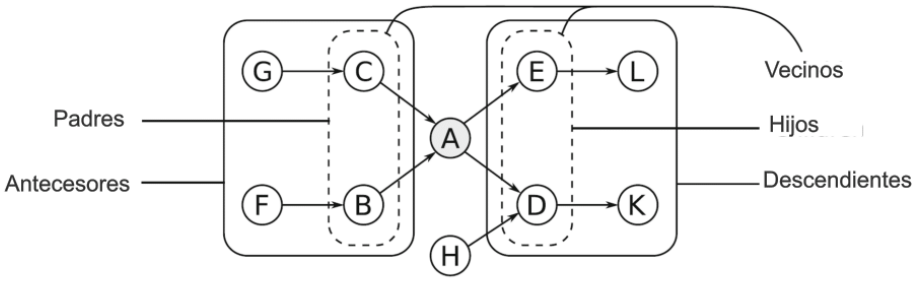


Figura 1.2: Antecesores, descendientes, vecinos, padres e hijos de un nodo.

Definición 1.1.8 (Conexiones elementales)**.** *Dado un camino independientemente de las direcciones de los arcos, se dice que un nodo es de arcos convergentes si solo tiene arcos incidentes. Similarmente, se dice que es un nodo de arcos divergentes si solo tiene arcos salientes. Por último, un nodo se dice de arcos en serie si posee un solo arco incidente y un solo arco saliente. Véase figura 1.3.*

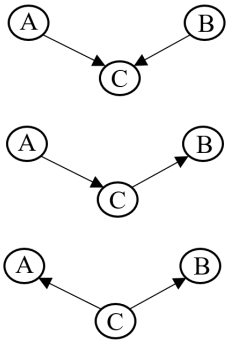


Figura 1.3: Conexión convergente, conexión en serie y conexión divergente.

Definición 1.1.9 (D-Separación)**.** *Sea $G = (V, E)$ un grafo y sean $A, B, C \subseteq V$ subconjuntos de nodos disjuntos. Se dice que C d-separa A de B, denotado $A \perp\!\!\!\perp_G B|C$, si a lo largo de cada secuencia de arcos entre un nodo de A y uno de B, existe un nodo v que satisface una de las siguientes condiciones:*

1. *v tiene arcos convergentes y ni v ni sus descendentes están en C.*
2. *v está en C y no tiene arcos convergentes.*

Ejemplo 1.1.1. *En los casos 2 y 3 de la figura 1.3, C d-separa A de B, mientras que en el primer caso, no es así.*

Definición 1.1.10 (v-estructura)**.** *Se define una v-estructura como el conjunto de conexiones entre tres nodos en las que los dos nodos no adyacentes no son d-separados por el tercero.*

Ejemplo 1.1.2. *La primera red representada en la figura 1.3 se corresponde con una v-estructura.*

1.2. Redes Bayesianas: Definiciones y propiedades esenciales

Una **red Bayesiana** es un tipo particular de grafo que permite explicar de manera concisa las dependencias probabilísticas existentes entre las componentes de un vector aleatorio $\mathbf{X}$. A partir de $\mathbf{X} = (X_1, ..., X_p)$, definimos

un grafo acíclico dirigido (DAG) $G = (V, A)$, donde cada vértice o nodo de V se corresponde con una variable X_i.

Recordemos el concepto de independencia probabilística, el cual, como veremos unas líneas más adelante, está ligado al concepto de d-separación.

Definición 1.2.1. *Dos variables A y B se dicen independientes condicionalmente dado C, denotada $A \perp\!\!\!\perp_P B|C$, si se verifica*

$$P(AB|C) = P(A|C)P(B|C) \tag{1.1}$$

Como comentábamos, existe una relación entre la separación del grafo ($\perp\!\!\!\perp_G$) y la independencia probabilística ($\perp\!\!\!\perp_P$) y es conocida como *mapa de independencia*. Se define como sigue:

Definición 1.2.2 (Mapas)**.** *Dados un grafo $G = (V, E)$ y un vector aleatorio $\boldsymbol{X}$ con función de probabilidad P, se dice que G es un mapa de independencia (I-map) de P si existe una correspondencia biunívoca entre las variables aleatorias X_i y los nodos de V, de modo que para todos los subconjuntos disjuntos $A, B, C \subseteq V$ se tiene*

$$A \perp\!\!\!\perp_P B|C \Longleftarrow A \perp\!\!\!\perp_G B|C$$

Análogamente, G es un mapa de dependencia (D-map) de P si tenemos

$$A \perp\!\!\!\perp_P B|C \Longrightarrow A \perp\!\!\!\perp_G B|C.$$

G se dice que es un mapa perfecto de P si es I-map y D-map, es decir,

$$A \perp\!\!\!\perp_P B|C \Longleftrightarrow A \perp\!\!\!\perp_G B|C$$

y en este caso, P se dice que es isomorfo a G.

La *propiedad de Markov*, la cual establece que en una red Bayesiana, cada nodo X_i es condicionalmente independiente de sus nodos no-descendientes dados sus padres, nos permite escribir la siguiente factorización de la función de probabilidad conjunta:

$$P(\mathbf{X}) = \prod_{i=1}^{p} P_{X_i}(X_i|\Pi_{X_i})$$

donde Π_{X_i} es el conjunto formado por los padres de X_i. Podemos ver esta propiedad como una generalización de la fórmula producto o regla de la multiplicación.

Recordemos que cada nodo de la red representa una variable de nuestro modelo probabilístico. Por otro lado, dados dos nodos del grafo A y B, en

la red aparecerá una arista ($A \to B$) si B está condicionada a A.

Asumiendo que las redes de la figura 1.3 representan $I-maps$, la familia de funciones de probabilidad compatibles tendrían la siguiente estructura:

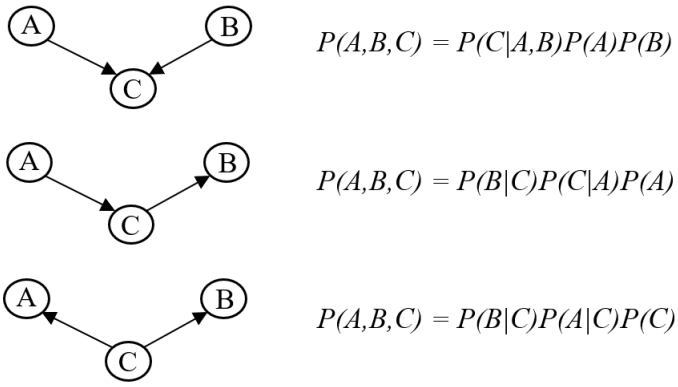


Figura 1.4: Factorización de la función de probabilidad de las tres conexiones fundamentales.

Se dirá que una función de probabilidad factoriza sobre una red Bayesiana si se puede escribir como producto de las probabilidades condicionadas anteriores.

Observación 1.2.1. *Se puede probar que, tanto en la conexión en serie como en la divergente de la figura anterior, se verifica que A y B son independientes condicionalmente dado C.*

1.3. Estructuras equivalentes y envolvente de Markov

No es difícil ver que las factorizaciones de la conexión en serie y divergente (ver figura 1.3) son equivalentes. Cada una de ellas puede ser deducida de la otra aplicando repetidas veces la fórmula de Bayes. Tales estructuras probabilísticamente equivalentes se conocen como *estructuras equivalentes de Markov.*

Dado que la equivalencia es simétrica, reflexiva y transitiva, cada conjunto de estructuras equivalentes forma una clase de equivalencia. En general, se puede probar que los únicos arcos cuya dirección es necesaria para identificar una clase de equivalencia son los que pertenecen, al menos, a una

v-estructura.

Otro concepto importante muy ligado a los mapas de independencia y a la d-separación es la envolvente de Markov. Esencialmente, representa el conjunto de nodos que d-separa completamente un nodo particular del resto.

Definición 1.3.1. *Sea $G = (V, E)$ un grafo. La envolvente de Markov de un nodo $A \in V$ es el menor subconjunto $S \subseteq V$ tal que*

$$\{A\} \perp\!\!\!\perp_G (V - \{A\} - S) | S$$

Observación 1.3.1. *En cualquier red Bayesiana, la envolvente de Markov de un nodo A está formada por los padres de A, los hijos de A y todos los otros nodos que compartan hijos con A.*

Veremos a continuación que la envolvente de Markov cobra gran importancia en la tarea de aprendizaje de una red.

1.4. Modelado de redes Bayesianas

La tarea de construir una red Bayesiana que se ajuste a un modelo probabilístico dado se denomina *aprendizaje* de la red. Se realiza en dos etapas:

1. Aprendizaje estructural. Consiste en identificar, mediante algoritmos, la estructura de grafo de la red Bayesiana. Estos algoritmos se clasifican en *basados en restricciones*, *basados en puntuaciones* y *algoritmos híbridos*.

   Los algoritmos del primer tipo llevan a cabo el aprendizaje estructural mediante contrastes estadísticos de independencia sobre los datos, con el objetivo de identificar las relaciones de dependencia o independencia entre las variables del modelo estadístico. Estos se basan en el algoritmo de Causalidad Inductiva (IC) (ver detalles en R. Nagarajan et al [5]).

   Existen algoritmos mejorados que previamente aprenden la envolvente de Markov de cada nodo de la red. Así, se reduce considerablemente la complejidad del algoritmo. Algunos de estos son:

   - *Grow-Shrink*, basado en el algoritmo de envolvente de Markov de Grow-Shrink (Margaritis [8]).
   - *Incremental Association* (IAMB), basado en el algoritmo global de la asociación incremental de envolventes de Markov (Tsamardinos et al. [4]).

- *Fast Incremental Association*, una variante de IAMB que utiliza la selección especulativa paso a paso para reducir el número de pruebas de independencia condicional (Yaramakala- Margaritis [9]).
- *Interleaved Incremental Association*, otra variante de IAMB que utiliza la selección por pasos hacia adelante (Tsamardinos et al. [4]) para evitar falsos positivos en la fase de detección general de Markov.

Por otro lado, los algoritmos basados en puntuaciones asignan a cada red candidata una puntuación que refleje su bondad de ajuste y, a continuación, se selecciona la red cuya puntuación sea máxima. Destacamos los algoritmos de búsqueda codiciosos (*greedy*), como el algoritmo de escalada. Veamos en qué consiste este último (R. Nagarajan et al. [5]):

Algoritmo 1 Algoritmo de escalada

1: Elija una estructura de red G sobre V, generalmente (pero no necesariamente) vacía.
2: Calcule la puntuación de G, denotada como $score_G$= *score(G).*
3: Establezca $maxscore$=$score_G$.
4: Repita los siguientes pasos siempre y cuando la puntuación máxima ($maxscore$) aumente:
   a) Por cada posible adición, eliminación o inversión de un arco que no dé una red cíclica:
      1) Calcule la puntuación de la red modificada $G*$, $score_{G*}$= $score(G*)$;
      2) Si $\text{score}_{G*} > score_G$, tomar G=G* y score_G=score_{G*}.
   b) Actualice maxscore con el nuevo valor de $score_G$.
5: Devuelva el grafo acíclico dirigido G.

La puntuación de cada estructura se obtiene utilizando la base de datos existente aplicada en contrastes de independencia condicional implicadas por dicha estructura.

Finalmente, los algoritmos híbridos combinan algoritmos basados en restricciones y basados en puntuaciones. Los algoritmos más destacados son el algoritmo del candidato disperso (SC) (Friedman et al. [2]) y el algoritmo de escalada Min-Max (Tsamardinos et al. [10]).

2. Aprendizaje paramétrico. Consiste, como su nombre indica, en estimar los parámetros de la distribución conjunta. En este trabajo nos centraremos en esta tarea, que se puede llevar a cabo, entre otros métodos,

mediante estimación de *máxima verosimilitud* (véase sección 2.1) o mediante *estimación Bayesiana* (véase sección 2.2).

1.5. Naturaleza de los datos

Tanto la función de distribución conjunta como las distribuciones condicionadas dependerán de la naturaleza de los datos. Vamos a destacar los dos casos más frecuentes.

- Variables multinomiales. Las variables toman valores discretos y las distribuciones condicionadas son multinomiales. Estas últimas se representan como tablas condicionadas de probabilidad (CPT). Las redes Bayesianas asociadas a este tipo de variables se conocen como *redes Bayesianas discretas.*
- Variables normales multivariantes. Las variables toman valores continuos. La distribución conjunta es normal multivariante y las distribuciones condicionadas son variables aleatorias normales univariantes. Estas redes Bayesianas se llaman *redes Bayesianas Gaussianas.*

Por último, en el caso de disponer de datos mixtos, podemos optar por la discretización de los datos. Esto consiste en transformar todas las variables continuas en discretas y aplicar las técnicas para redes Bayesianas discretas.

En lo que sigue, nos centraremos en redes Bayesianas discretas.

Capítulo 2

Estimación paramétrica de una red de Bayes

En este trabajo presentamos dos métodos de estimación: la *estimación de máxima verosimilitud* y la estimación mediante funciones de riesgo. En este último tipo distinguiremos la *estimación minimax* y la *estimación de Bayes*.

2.1. Estimación de máxima verosimilitud

2.1.1. Descripción

El principio de *máxima verosimilitud* consiste en, dada una muestra representativa de la población, tomar como estimador aquel valor del parámetro que maximiza la probabilidad conjunta de la muestra. Recordemos que nos limitamos al caso discreto. La referencia principal de esta subsección es el libro de Rohatgi-Saleh [6, Capítulo 8].

Definición 2.1.1 (Función de verosimilitud)**.** *Sea $\boldsymbol{X} = (X_1, ..., X_n)$ un vector aleatorio discreto y sea $\theta \in \Theta \subseteq \mathbb{R}^k$. Se define la función de verosimilitud como*

$$L(\theta, x_1, ..., x_n) = P_\theta(X_1 = x_1, ..., X_n = x_n)$$

donde P_θ denota la función de probabilidad conjunta bajo el parámetro θ.

Observación 2.1.1. *Aunque aparentemente la función de probabilidad conjunta y la función de verosimilitud pueden confundirse, son funciones distintas. Observemos que la función de probabilidad conjunta es una función de $\boldsymbol{x}$ considerando un θ fijo, mientras que la función de verosimilitud es una función de θ en la que la realización muestral $\boldsymbol{x}$ está fija.*

Si $X_1, ..., X_n$ son independientes e idénticamente distribuidas (iid) con función de probabilidad P_θ, se tiene que

$$L(\theta, x_1, ..., x_n) = \prod_{i=1}^{n} P_\theta(X_i = x_i)$$

Definición 2.1.2 (Estimador de máxima verosimilitud)**.** *Sea $\Theta \subseteq \mathbb{R}^k$. Se dirá que $\hat{\theta}_{MV}$ es un estimador de máxima verosimilitud si se verifica*

$$L(\hat{\theta}_{MV}, x_1, ..., x_n) = \sup_{\theta \in \Theta} L(\theta, x_1, ..., x_n)$$

Observación 2.1.2. *Debido a la monotonía de la función logarítmica, la condición anterior es equivalente a*

$$ln(L(\hat{\theta}_{MV}, x_1, ..., x_n)) = \sup_{\theta \in \Theta} ln(L(\theta, x_1, ..., x_n))$$

Nota 2.1.1. *El estimador de máxima verosimilitud no tiene por qué ser único.*

El estimador de máxima verosimilitud guarda relación con la familia de estadísticos suficientes. Recordemos que un estadístico suficiente está caracterizado por el criterio de factorización de Fisher-Neyman.

Teorema 2.1.1 (Criterio de factorización de Fisher-Neyman)**.** *Sea $X_1, ..., X_n$ una muestra aleatoria de X con función de distribución $F \in \{F_\theta : \theta \in \Theta\}$ y sea $S = S(X_1, ..., X_n) = S(\boldsymbol{X})$ un estadístico muestral. Entonces, S es suficiente sí y solo sí podemos escribir*

$$P_\theta(\boldsymbol{x}) = g(\theta, S)h(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \tag{2.1}$$

En este trabajo, suponemos que la estructura de la red Bayesiana es conocida, por lo que la familia $\{F_\theta : \theta \in \Theta\}$ viene dada por las distribuciones que factorizan sobre dicha red.

2.1.2. Propiedades

A continuación, presentamos varias propiedades de este método de estimación.

Teorema 2.1.2. *Sea $\boldsymbol{X} = (X_1, ..., X_n)$ un vector aleatorio, con función de distribución $F \in \{F_\theta : \theta \in \Theta\}$ y sea $S = S(\boldsymbol{X})$ un estadístico suficiente. Si existe un único estimador de máxima verosimilitud para θ, $\hat{\theta}_{MV}$, entonces es una función de S no constante. Si no es único, podemos encontrar uno que sea función de S.*

Definición 2.1.3 (Función de verosimilutud inducida)**.** *Sea h: $\Theta \longrightarrow \Lambda \subseteq \mathbb{R}^k$, una función paramétrica, es decir, que a cada θ asigna $h(\theta)$. Se define la función de verosimilitud inducida por h como*

$$M(\lambda, \boldsymbol{x}) := \sup_{\theta \in \Theta / h(\theta) = \lambda} L(\theta, \boldsymbol{x})$$

Definición 2.1.4 (Estimador de máxima verosimilitud de una función paramétrica)**.** *Sea h=h(θ) una función paramétrica. Se dice que* $\hat{h(\theta)}_{MV}$ *es un estimador de máxima verosimilitud de h(θ) si*

$$\hat{h(\theta)}_{MV} = \sup_{\lambda \in \Lambda} M(\lambda, \boldsymbol{x})$$

Teorema 2.1.3 (Zenha)**.** *Sea h=h(θ) una función paramétrica y sea* $\hat{\theta}_{MV}$ *el estimador de máxima verosimilitud de θ. Se verifica:*

$$\hat{h(\theta)}_{MV} = h(\hat{\theta}_{MV})$$

Teorema 2.1.4. *Sean* $X_1, ..., X_n$ *variables aleatorias iid con función de distribución* $F \in \{F_\theta : \theta \in \Theta\}$ *y sea* $\hat{\theta}_n$ *un estimador de máxima verosimilitud de θ. Bajo ciertas condiciones de regularidad, se tiene que si* $\hat{\theta}_n$ *es único, entonces*

1. $\hat{\theta}_n$ *es fuertemente consistente.*
2. $\hat{\theta}_n$ *es asintóticamente normal, es decir:*

$$\sqrt{(n)}(\hat{\theta}_n - \theta) \xrightarrow{\mathcal{L}} N(0, 1/I(\theta))$$

con $I(\theta)$ *la cantidad de información de Fisher.*

$$I(\theta) = E_\theta \left[\left(\frac{\partial}{\partial \theta} log L(\boldsymbol{X}, \theta) \right)^2 \right]$$

Estas propiedades válidas para variables aleatorias se pueden extender en el caso de los vectores aleatorios representados mediante una red Bayesiana debido a la factorización de las distribuciones de la familia consideradas y a la separabilidad de la función de verosimilitud que debe ser optimizada.

2.1.3. Estimación de MV de distribuciones discretas

En esta subsección seguiremos el libro de Jensen-Nielsen [3, Capítulo 6] para tratar la estimación de máxima verosimilitud en el caso de distribuciones discretas, que usaremos en las redes Bayesianas analizadas en este trabajo.

Sea $X_1, ..., X_n$ una muestra aleatoria de X, variable discreta con función de probabilidad

$$P(X = x_k^0) = \theta_k \ , \ k = 1, ..., r$$

donde r denota el cardinal del soporte de X. Queremos encontrar un estimador de máxima verosimilitud de $\theta = (\theta_1, \ldots, \theta_r)$. Se tiene,

$$L(\mathbf{x}, \theta) = \prod_{k=1}^{r} \theta_k^{n_k} \cdot I(\mathbf{x})_{\{x_1^0, \ldots, x_r^0\}^n} \tag{2.2}$$

donde

$$n_k = \#\{i | x_i = x_k^0, 1 \leq i \leq n\}$$

Supondremos que $n_k \geq 1 \ \forall k = 1, \ldots, r$. Ahora, aplicando la función logarítmica a la expresión 2.2 resulta

$$ln(L(\mathbf{x}, \theta)) = \sum_{k=1}^{r} n_k \cdot ln(\theta_k) \cdot I(\mathbf{x})_{\{x_1^0, \ldots, x_r^0\}^n},$$

que es una función cóncava en θ.

Por otro lado, sabemos que

$$\sup_{\theta \in \Theta} L(\mathbf{x}, \theta) > 0$$

puesto que, tomando $\theta_k = \frac{1}{r} \ \forall k$, tenemos que

$$L(\mathbf{x}, \theta) > 0$$

Además, $L(\mathbf{x}, \theta)$ es continua, luego los siguientes problemas son equivalentes

$$\sup_{\theta \in \Theta} L(\mathbf{x}, \theta) \equiv \sup_{\theta \in S_\delta} L(\mathbf{x}, \theta) > 0$$

con

$$S_\delta = \Theta \cap \{\theta : L(x, \theta) > \delta\}$$

para algún $\delta > 0$. Entonces, sabemos que existe un ε suficientemente pequeño de modo que podemos considerar el siguiente problema

$$\begin{aligned} \sup \quad & ln(L(\mathbf{x}, \theta)) \\ \text{s.t.} \quad & \sum_{k=1}^{r} \theta_k = 1 \\ & \theta_k \geq \varepsilon, k = 1, ..., r \end{aligned} \tag{P}$$

Notemos que el conjunto de restricciones es compacto. Por tanto, como la función objetivo es continua en un conjunto factible compacto, sabemos que se alcanza el máximo por el teorema de Weierstrass. Así, podemos escribir

$$\begin{aligned} \max \quad & ln(L(\mathbf{x},\theta)) \\ \text{s.t.} \quad & \sum_{k=1}^{r}\theta_k = 1 \\ & \theta_k \geq \varepsilon, k = 1,...,r \end{aligned} \tag{P}$$

o equivalentemente,

$$\begin{aligned} \text{-min} \quad & -ln(L(\mathbf{x},\theta)) \\ \text{s.t.} \quad & \sum_{k=1}^{r}\theta_k = 1 \\ & -\theta_k \leq \varepsilon, k = 1,...,r \end{aligned} \tag{P}$$

La función Lagrangiana asociada a (P) es

$$L_a(\theta,\lambda,\mu) = -\sum_{k=1}^{r} n_k \cdot ln(\theta_k) - \lambda\cdot\theta + \mu(\sum_{k=1}^{r}\theta_k - 1)$$

El problema (P) es un problema de optimización convexa diferenciable, por tanto, las condiciones necesarias de optimalidad de **Karush-Tucker-Kuhn (KTK)** son también condiciones suficientes. En este caso, se deben verificar

$$\frac{\partial L_a}{\partial\theta} = 0 \tag{2.3}$$

$$\theta_1 + ... + \theta_r = 1 \tag{2.4}$$

$$\theta_1,...,\theta_r \geqslant \varepsilon \tag{2.5}$$

$$\lambda_k(\varepsilon - \theta_k) = 0 \quad \forall k = 1,...,r \tag{2.6}$$

$$\lambda_k \geq 0 \tag{2.7}$$

Desarrollando la condición 2.3,

$$-\frac{n_k}{\theta_k} - \lambda_k + \mu = 0, \quad \forall k = 1,...,r \tag{2.8}$$

Ahora, como ε es suficientemente pequeño, podemos suponer que

$$\theta_k > \varepsilon \quad \forall k = 1,...,r$$

y por la propiedad de complementariedad, se tiene que

$$\lambda_k = 0 \quad \forall k = 1,...,r$$

Sustituyendo en 2.8,

$$\theta_k = \frac{n_k}{\mu}.$$

Por otro lado, $\theta_1 + ... + \theta_r = 1$. Entonces,

$$1 = \sum_{k=1}^{r} \theta_k = \frac{\sum_{k=1}^{r} n_k}{\mu} = \frac{n}{\mu}$$

Luego, $\mu = n$. Así, $\hat{\theta_k} = \frac{n_k}{n}$ es un estimador de máxima verosimilitud para $\theta_k, \quad \forall k = 1, ..., r$.

Observemos que si $n_k = 0$, el parámetro θ_k no interviene en $L(\mathbf{x}, \theta)$, luego se puede obviar del razonamiento anterior, o lo que es lo mismo, estimarlo como 0. Es decir, el estimador $\hat{\theta_k} = \frac{n_k}{n}$ es válido incluso si no aparecen observaciones en la muestra con valor $X = x_k^0$ $\forall k$.

Ejemplo 2.1.1. *Consideremos una moneda trucada. Asignemos cara* $= 1$*, cruz* $= 0$ *y sean* $x_1^0 = 1$*,* $x_2^0 = 0$*. Supongamos* $P(X = x_1^0) = \theta$*, de modo que* $X \sim Be(\theta)$*. Lanzamos la moneda 100 veces, de las cuales 80 obtenemos cara. Queremos encontrar una estimación de máxima verosimilitud de la probabilidad de obtener cara al lanzar la moneda. Así, tenemos que*

$$L(\boldsymbol{x}, \theta) = \theta^{80} \cdot (1 - \theta)^{20},$$

Resolviendo la ecuación en θ

$$\frac{d}{d\theta} ln(L(\boldsymbol{x}, \theta)) = 0$$

obtenemos que $\theta = 0{,}8$*. Como la función* $L(\theta, \boldsymbol{x})$ *es cóncava en* θ*, tenemos que* $\hat{\theta}_{MV} = \frac{n_1}{n} = 0{,}8$ *es un estimador de máxima verosimilitud de* θ*, donde* n_1 *denota el número de lanzamientos en los que se ha obtenido cara.*

Supongamos ahora que nos encontramos en una red Bayesiana con nodos $\{A, B, C\}$. Queremos encontrar, por ejemplo, un estimador de máxima verosimilitud de $\theta = P(C = c|A = a, B = b)$. Se tiene

$$\hat{\theta}_{MV} = \frac{N(A=a,B=b,C=c)}{N(A=a,B=b)}$$

donde $N(A = a, B = b, C = c)$ denota el número total de casos en los que $A = a, B = b, C = c$ y $N(A = a, B = b)$ es el número de casos en los que $B = b, C = c$.

Esta situación se extiende a cualquier red Bayesiana debido a la factorización de la familia de probabilidades conjuntas y a la separabilidad de la función de verosimilitud respecto a los parámetros involucrados en las

probabilidades condicionadas de un nodo hijo respecto a sus padres.

Veamos un ejemplo.

Ejemplo 2.1.2. *Consideremos las siguientes variables:*

- *Enfermedad X (X)* $\in \{$ *1 (si se padece), 0 (en caso contrario)*$\}$
- *Edad (E)* $\in \{$*Joven (J), Mediana Edad (ME), Avanzada edad (A)*$\}$
- *Sexo (S)* $\in \{$*Hombre (H), Mujer (M)*$\}$
- *Fiebre (F)* $\in \{$ *1 (si se padece), 0 (en caso contrario)*$\}$
- *Tos (T)* $\in \{$ *1 (si se padece), 0 (en caso contrario)*$\}$

Se ha recogido la siguiente muestra del vector aleatorio (X, E, S, F, T)*:*

(1,ME,H,0,1)	*(0,J,H,1,0)*	*(1,A,H,1,1)*
(0,A,M,0,1)	*(1,ME,H,1,0)*	*(0,ME,H,1,1)*
(1,ME,H,1,1)	*(1,J,M,0,1)*	*(1,A,M,0,0)*

Supongamos que queremos estimar la probabilidad de que un hombre de mediana edad tenga tos sabiendo que padece la enfermedad X. Llamemos θ_1 *a dicha probabilidad. De la muestra principal, filtramos aquellos individuos que verifiquen* $E = ME, S = H, X = 1$*. Obtenemos la submuestra*

$$\{(1, ME, H, 0, 0, 1), (1, ME, H, 1, 1, 0), (1, ME, H, 1, 0, 1)\}.$$

Por tanto, tenemos que

$$\hat{\theta}_1 = \frac{N(E{=}ME, S{=}H, X{=}1, T{=}1)}{N(E{=}ME, S{=}H, X{=}1)} = \frac{2}{3}$$

Veamos ahora qué ocurre si queremos estimar la probabilidad de que una mujer de edad avanzada que tiene fiebre y tos padezca la enfermedad X, es decir, queremos estimar $\theta_2 = P(X = 1|E = A, S = M, F = 1, T = 1)$*. Sin embargo, en la base de datos, no se ha encontrado ningún individuo con estas características, por lo que obtendríamos que* $\hat{\theta}_2 = 0$*. Esto es un problema en el modelado de la red, puesto que a través de la estimación de máxima verosimilitud obtendremos un modelo que "niega" la posibilidad de que una mujer con estos síntomas padezca la enfermedad X.*

En la sección 2.3 describiremos la aplicación en R de la estimación de máxima verosimilitud de los parámetros de una red Bayesiana mediante la librería *bnlearn*.

El problema de la inexistencia de casos en la base de datos

Hemos visto anteriormente que el estimador de máxima verosimilitud es fuertemente consistente, sin embargo, si el tamaño muestral no es suficientemente grande o el número de categorías de las variables es excesivo en relación al tamaño de la base de datos, es claro que algunos de los parámetros van a ser estimados como ceros. Esto nos conduce, por ejemplo, a una red Bayesiana que produce errores al identificar causas verosímiles de una enfermedad. Este problema queda resuelto con la *estimación de Bayes*, que además posee buenas propiedades de carácter teórico.

2.2. Estimación mediante funciones de riesgo

En esta sección trataremos la *estimación minimax* y la *estimación de Bayes*. A lo largo de este capítulo seguiremos el capítulo 8 del libro de Rohatgi-Saleh [6].

Sea $X_1, ..., X_n$ una muestra aleatoria de X con función de distribución $F \in \{F_\theta : \theta \in \Theta\}$. Supongamos que se observa la realización muestral $\mathbf{x} = (x_1, ..., x_n)$ y ante ella, debe tomarse una decisión. Sea $\mathcal{D}$ el conjunto de todas las decisiones que se pueden tomar.

Definición 2.2.1 (Función de decisión)**.** *Se define una función de decisión como un estadístico que toma valores en $\mathcal{D}$, es decir, es una función $\delta : \mathbb{R}^n \longrightarrow \mathcal{D}$ medible-Borel.*

Si se observa la realización muestral $\mathbf{X} = \mathbf{x}$, se toma la decisión $\delta(\mathbf{x}) \in \mathcal{D}$.

Ejemplo 2.2.1. *Sea $\mathcal{D} = \{a_1, a_2\}$. En este caso, cualquier función de decisión δ divide el soporte de $(X_1, ..., X_n)$ en un conjunto R y su complementario, R^c. Así, si $\boldsymbol{x} \in R$ entonces tomamos la decisión a_1, y si $\boldsymbol{x} \in R^c$, tomamos a_2.*

Por ejemplo, esto es lo que ocurre en el problema de contrastes de hipótesis, donde R es la región de rechazo de la hipótesis nula y δ es la función test dada por $a_1 = 1$, que corresponde a rechazar la hipótesis nula, y $a_2 = 0$, que corresponde con no rechazarla.

Ejemplo 2.2.2. *Si $\mathcal{D} = \Theta$, estamos ante el problema de estimación paramétrica.*

Otro elemento importante en el modelo de toma de decisiones es la función de pérdidas, que mide el coste de tomar una decisión concreta.

Definición 2.2.2 (Función de pérdidas)**.** *Sea $\mathcal{D}$ un conjunto arbitrario de decisiones. Una función de pérdidas es una función $C : \Theta \times \mathcal{D} \longrightarrow \mathbb{R}$ medible-Borel no negativa.*

El valor $C(\theta, a)$ es la pérdida bajo el parámetro θ de tomar la decisión a. Si consideramos la función de decisión $\delta(\mathbf{X})$, la función de pérdidas C bajo el parámetro θ, la pérdida es la variable aleatoria $C(\theta, \delta(\mathbf{X}))$.

Definición 2.2.3 (Función de riesgo)**.** *Sea Δ un conjunto de funciones de decisión $\delta : \mathbb{R}^n \longrightarrow \mathcal{D}$, y sea C la función de pérdidas en $\Theta \times \Delta$. Se define la función de riesgo como $R : \Theta \times \Delta \longrightarrow \mathbb{R}$ dada por*

$$R(\theta, \delta) = E_\theta[C(\theta, \delta(\boldsymbol{X})]$$

Ejemplo 2.2.3. *Volvamos a los contrastes de hipótesis. Consideremos el constraste*

$$\begin{aligned} H_0 &: \theta = \theta_0 \\ H_1 &: \theta \neq \theta_0 \end{aligned}$$

y tomemos como función de pérdidas

$$C(\theta, a) = aI_{\{\theta_0\}}(\theta)$$

Por tanto, la función de riesgo es

$$E_\theta[C(\theta, \delta(\boldsymbol{X})] = P_{\theta_0}(\delta(\boldsymbol{X}))$$

que coincide con la función potencia. En general, tomando el contraste

$$\begin{aligned} H_0 &: \theta \in \Theta_0 \\ H_1 &: \theta \notin \Theta_0 \end{aligned} \tag{2.9}$$

se tiene que

$$R(\theta, \delta) = P_\theta(\delta(\boldsymbol{X}))I_{\Theta_0}(\theta)$$

que es justamente la función potencia sobre Θ_0 dado el contraste 2.9 .

Ejemplo 2.2.4. *Sea $\mathcal{D} = \Theta \subseteq \mathbb{R}$, $C(\theta, a) = (\theta - a)^2$. Entonces,*

$$R(\theta, \delta) = E_\theta[C(\theta, \delta(\boldsymbol{X})] = E_\theta[(\delta(\boldsymbol{X}) - \theta)^2]$$

que es justamente el Error Cuadrático Medio (ECM). Más aun, si nos limitamos al conjunto de estimadores insesgados, el riesgo no es más que la varianza del estimador, pues recordemos que, si $T = T(\boldsymbol{X})$ es un estimador de θ, entonces

$$ECM_\theta(T) = E_\theta[(T - \theta)^2] = Var_\theta(T) + B_\theta(T)$$

donde $B_\theta(T)$ denota el sesgo del estimador T.

2.2.1. Estimación minimax

El problema principal de la teoría de decisión estadística es el siguiente: dado un espacio de acciones $\mathcal{D}$, y una función de pérdidas $C(\theta, a)$, encontrar una función de decisión $\delta \in \Delta$ tal que el riesgo $R(\theta, \delta)$ sea mínimo para cualquier valor de θ. Puesto que, dada δ, $R(\theta, \delta)$ es una función de θ, necesitamos construir un indicador numérico que sirva para comparar dos funciones de decisión sin depender de θ. Lo usual es considerar el máximo riesgo en θ o el riesgo medio.

Definición 2.2.4. *Se dice que δ^* es una decisión minimax si verifica*

$$\max_{\theta \in \Theta} R(\theta, \delta^*) = \min_{\delta \in \Delta} \max_{\theta \in \Theta} R(\theta, \delta)$$

Es decir, una decisión minimax es aquella que minimiza el máximo riesgo. Si estamos ante un problema de estimación ($\Delta = \Theta$), entonces diremos que δ^* es un estimador minimax de θ.

Ejemplo 2.2.5. *Sea $X \sim b(1, p)$, $p \in \Theta = \{\frac{1}{4}, \frac{1}{2}\}$ y $\mathcal{D} = \{a_1, a_2\}$. Consideremos la siguiente función de pérdidas:*

	a_1	a_2
$p_1 = \frac{1}{4}$	*1*	*4*
$p_2 = \frac{1}{2}$	*3*	*2*

El conjunto de decisiones incluye cuatro funciones: δ_1, δ_2, δ_3, δ_4 dadas por $\delta_1(0) = \delta_1(1) = a_1$; $\delta_2(0) = a_1$, $\delta_2(1) = a_2$; $\delta_3(0) = a_2$, $\delta_3(1) = a_1$; $\delta_4(0) = \delta_4(1) = a_2$. La función de riesgo toma los siguientes valores:

i	$R(p_1, \delta_i)$	$R(p_2, \delta_i)$	$\max_{p_1, p_2} R(p, \delta)$	$\min_i \max_{p_1, p_2} R(p, \delta)$
1	*1*	*3*	*3*	
2	$\frac{7}{4}$	$\frac{5}{2}$	$\frac{5}{2}$	$\frac{5}{2}$
3	$\frac{13}{4}$	$\frac{5}{2}$	$\frac{13}{4}$	
4	*4*	*2*	*4*	

Por tanto, la solución minimax es $\delta_2(x) = a_1$ si x=0 y $\delta_2(x) = a_2$ si x=1.

2.2.2. Estimación Bayesiana

Una perspectiva diferente a la estimación minimax consiste en minimizar el riesgo medio.

Hasta ahora, hemos considerado θ un parámetro fijo desconocido. Sin embargo, en la *estimación Bayesiana*, se considera θ como una variable

aleatoria cuya función de densidad es $\pi(\theta)$ en Θ. La función π se denomina *distribución a priori* de θ.

Sea $f(\mathbf{x}|\theta)$ la función de densidad condicionada del vector aleatorio $\mathbf{X}$, dado un $\theta \in \Theta$ fijo y π, la distribución de θ, se sigue que la función de densidad conjunta viene dada por

$$f(\mathbf{x}, \theta) = \pi(\theta) f(\mathbf{x}|\theta)$$

En esta sección, $R(\theta, \delta)$ es la pérdida promedio condicionada,

$$R(\theta, \delta) = E[C(\theta, \delta(X))|\theta]$$

suponiendo que θ está fijo. Observemos que estamos usando la misma notación para denotar la variable aleatoria θ que para denotar un valor fijo de la misma.

Definición 2.2.5 (Riesgo de Bayes)**.** *Se define el riesgo de Bayes de una función de decisión δ como*

$$R(\pi, \delta) = E_\pi[R(\theta, \delta)]$$

Si θ es una variable aleatoria continua y $\mathbf{X}$ es un vector aleatorio continuo, entonces

$$\begin{aligned} R(\pi, \delta) &= \int R(\theta, \delta)\pi(\theta)d\theta \\ &= \int\int C(\theta, \delta(\mathbf{x}) f(\mathbf{x}|\theta)\pi(\theta)d\mathbf{x}d\theta \\ &= \int\int C(\theta, \delta(\mathbf{x}))f(\mathbf{x}, \theta)d\mathbf{x}d\theta \end{aligned}$$

Si θ es discreta con función de probabilidad π y $\mathbf{X}$ discreto, entonces

$$R(\pi, \delta) = \sum_\theta \sum_{\mathbf{x}} C(\theta, \delta(\mathbf{x}))f(\mathbf{x}, \theta)$$

Expresiones similares resultan para el caso en el que $\mathbf{X}$ sea continuo y π discreta, y en el que $\mathbf{X}$ sea discreto y π continua, aunque en estos casos la distribución conjunta es singular.

Definición 2.2.6 (Decisión de Bayes)**.** *Una función de decisión δ^* se define como una* decisión de Bayes *si "minimiza" el riesgo de Bayes, es decir, si*

$$R(\pi, \delta^*) = \inf_\delta R(\pi, \delta)$$

Definición 2.2.7 (Distribución a posteriori)**.** *La distribución condicionada de la variable aleatoria θ, dada $\boldsymbol{X}$=$\boldsymbol{x}$, se denominada distribución a posteriori de θ.*

En el caso absolutamente continuo podemos escribir la función de densidad conjunta de la forma

$$f(\mathbf{x}, \theta) = g(\mathbf{x})h(\theta|\mathbf{x})$$

donde g denota la densidad marginal de $\mathbf{X}$. La densidad a *priori* $\pi(\theta)$ nos permite conocer la distribución de θ antes de tomar la muestra, mientras que la densidad a *posteriori* $h(\theta|x)$ da la distribución de θ después de tomar la muestra.

Tenemos entonces

$$R(\pi, \delta) = \int g(\mathbf{x}) \left[\int C(\theta, \delta(\mathbf{x}))h(\theta|\mathbf{x})d\theta \right] d\mathbf{x}.$$

En el caso discreto la expresión es similar:

$$R(\pi, \delta) = \sum_x g(\mathbf{x}) \left[\sum_\theta C(\theta, \delta(\mathbf{x}))h(\theta|\mathbf{x})d\theta \right] d\mathbf{x}.$$

Estas expresiones se pueden extender al caso de una distribución conjunta singular.

Teorema 2.2.1. *Consideremos el problema de estimación de un parámetro $\theta \in \Theta \subseteq \mathbf{R}$ y sea la función de pérdidas $C(\theta, \delta) = (\theta - \delta)^2$. Entonces, una decisión de Bayes viene dada por*

$$\delta(\boldsymbol{x}) = E[\theta|\boldsymbol{X} = \boldsymbol{x}] \tag{2.10}$$

La función $\delta(x)$ dada por (2.10) se denomina estimador de Bayes y coincide con la curva de regresión de θ con respecto a $\boldsymbol{X}$.

Demostración. Haremos la demostración para el caso en el que θ y π sean absolutamente continuas. La demostración en el resto de casos es análoga.

Sea π la distribución a priori de θ. Entonces,

$$R(\pi, \delta) = \int g(\mathbf{x}) \left[\int (\theta - \delta(\mathbf{x}))^2 h(\theta|\mathbf{x})d\theta \right] d\mathbf{x}$$

donde g denota la función de densidad marginal de $\mathbf{X}$ y h es la función de densidad condicionada de θ. La *regla de Bayes* es una función δ que minimiza $R(\pi, \delta)$. Minimizar en δ la función $R(\pi, \delta)$ es equivalente a minimizar

$$
\begin{aligned}
G(\delta) :&= \int (\theta - \delta(\mathbf{x}))^2 h(\theta|\mathbf{x}) d\theta = \\
&= E[(\theta - \delta(\mathbf{x}))^2|\mathbf{x}] = E[(\theta - E\left[\theta|\mathbf{x}\right] + E[\theta|\mathbf{x}] - \delta(\mathbf{x}))^2|\mathbf{x}\right] \\
&= E[((\theta - E\left[\theta|\mathbf{x}\right])^2 + (E[\theta|\mathbf{x}] - \delta(\mathbf{x}))^2 + 2(\theta - E\left[\theta|\mathbf{x}\right])(E[\theta|\mathbf{x}] - \delta(\mathbf{x})))|\mathbf{x}\right] \\
&= E\left[(\theta - E\left[\theta|\mathbf{x}\right])^2|\mathbf{x}\right] + (E\left[\theta|\mathbf{x}\right] - \delta(\mathbf{x}))^2 + 2E\left[(\theta - E\left[\theta|\mathbf{x}\right])|\mathbf{x}\right](E\left[\theta|\mathbf{x}\right] - \delta(\mathbf{x}))
\end{aligned}
$$

Ahora bien, por la linealidad de la esperanza

$$E\left[(\theta - E\left[\theta|\mathbf{x}\right])|\mathbf{x}\right] = E\left[\theta|\mathbf{x}\right] - E\left[\theta|\mathbf{x}\right] = 0$$

Por tanto,

$$G(\delta) = E\left[(\theta - E\left[\theta|\mathbf{x}\right])^2|\mathbf{x}\right] + (E\left[\theta|\mathbf{x}\right] - \delta(\mathbf{x}))^2$$

Como el primer sumando no depende de δ, G alcanza el mínimo donde lo alcance $(E\left[\theta|\mathbf{x}\right] - \delta)^2$, es decir, en $\delta = E\left[\theta|\mathbf{x}\right]$. □

Recordemos que, si el vector aleatorio $(\mathbf{X}, \theta)$ es absolutamente continuo, por definición

$$f_{\theta|\mathbf{X}=\mathbf{x}}(\theta) = \frac{f(\mathbf{x}, \theta)}{f_{\mathbf{X}}(\mathbf{x})}$$

Sin embargo, si $(\mathbf{X}, \theta)$ es singular, no tiene sentido hablar de la función de densidad conjunta $f(\mathbf{x}, \theta)$. Es por ello que, para poder trabajar con redes Bayesianas, debemos generalizar la definición anterior. Veamos el siguiente resultado del libro de Bertsekas [1].

Proposición 2.2.1. *Supongamos que $\boldsymbol{X}$ es un vector aleatorio discreto con función de probabilidad P e Y es una variable aleatoria absolutamente continua con función de densidad f_Y. Entonces,*

$$P_\theta(\boldsymbol{X} = \boldsymbol{x})\pi(\theta) = h(\theta|\boldsymbol{x})P(\boldsymbol{X} = \boldsymbol{x})$$

Consideremos entonces un vector aleatorio $\mathbf{X}$ discreto y θ absolutamente continua con función de probabilidad $\pi(\theta)$. Podemos escribir, según lo anterior

$$h(\theta|\mathbf{x}) = \frac{P_\theta(\mathbf{X} = \mathbf{x})\pi(\theta)}{P(\mathbf{X} = \mathbf{x})}$$

Ejemplo 2.2.6. *Sea* $X \sim Bi(n,\theta)$ *y sea* $C(\theta,\delta(x)) = (\theta-\delta(x))^2$. *Sea también la distribución a priori de* θ $\pi(\theta)=1$ *para* $0<\theta<1$. *Entonces*

$$h(\theta|x) = \frac{\binom{n}{x}\theta^x(1-\theta)^{n-x}I_{(0,1)}(x)}{\int_0^1 \binom{n}{x}\theta^x(1-\theta)^{n-x}\,d\theta}$$

Por tanto,

$$E[\theta|x] = \int_0^1 \theta h(\theta|x)d\theta = \frac{x+1}{n+2}$$

Luego, el estimador de Bayes es

$$\delta^*(X) = \frac{X+1}{n+2},$$

el cual nunca va a ser nulo. Como veremos más adelante, esto tiene implicaciones interesantes a la hora de estimar los parámetros de una red Bayesiana de acuerdo a una base de datos en la que no aparecen recogidos toda la casuística modelada por la red, lo que suele ocurrir en muchos casos prácticos, como en el campo de la Medicina.

Calculemos también el riesgo de Bayes

$$\begin{aligned} R(\pi,\delta^*) &= \int \pi(\theta)\sum_{x=0}^{n}(\delta^*-\theta)^2 f(x|\theta)d\theta \\ &= \int_0^1 E\left[\left(\frac{X+1}{n+2}-\theta\right)^2 \mid \theta\right] d\theta \\ &= \frac{1}{(n+2)^2}\int_0^1 [n\theta(1-\theta)+(1-2\theta)^2]d\theta \\ &= \frac{1}{6(n+2)} \end{aligned}$$

Nota 2.2.1. *Aparte de la función de pérdidas*

$$C(\theta,\delta(\boldsymbol{x})) = (\theta-\delta(\boldsymbol{x}))^2$$

se han usado en la literatura otras funciones como

$$|\theta-\delta(X)|, \qquad \frac{|\theta-\delta(X)|^2}{|\theta|}, \qquad |\theta-\delta(X)|^4, \quad y \quad \left(\frac{|\theta-\delta(X)|}{|\theta|+1}\right)^{1/2}$$

Cabe destacar que todas las funciones que contengan valores absolutos conducen a problemas de no diferenciabilidad y, además, muchas de ellas se minimizan en medianas, con las cuales es más difícil trabajar.

La clave para encontrar el estimador de Bayes es la obtención de la distribución a posteriori, $h(\theta|\mathbf{x})$. Presentamos a continuación tres pasos para ello:

1. Encontrar la distribución de $\mathbf{X}|\theta$ y θ dada por $\pi(\theta)f(\mathbf{x}|\theta)$.
2. Encontrar la distribución marginal de $g(\mathbf{x})$ integrando o sumando en $\theta \in \Theta$.
3. Hacer el cociente $\frac{\pi(\theta)f(\mathbf{x}|\theta)}{g(\mathbf{x})}$.

Sin embargo, no siempre es fácil realizar en la práctica estos pasos. En general, es difícil obtener $h(\theta|\mathbf{x})$ explícitamente. Para evitar posibles problemas de integración en los cálculos de las funciones previamente comentadas, se trabaja con *distribuciones conjugadas a priori.*

Definición 2.2.8. *Sea $X \sim f(\boldsymbol{x}|\theta)$ y sea $\pi(\theta)$ la distribución a priori en Θ. Entonces, π se dice familia a priori conjugada si la correspondiente distribución a posteriori $h(\theta|x)$ pertenece a la misma familia que $\pi(\theta)$*

Ejemplo 2.2.7. *Sea $X \sim b(n,p)$ con $0 < \theta < 1$, y sea $\pi(\theta) \sim \beta(a,b)$. Entonces*

$$\begin{aligned} h(\theta|x) &= \frac{\binom{n}{x}\theta^x(1-\theta)^{n-x}\theta^{a-1}(1-\theta)^{b-1}}{\int_0^1 \binom{n}{x}\theta^x(1-\theta)^{n-x}\theta^{a-1}(1-\theta)^{b-1}dp} = \\ &= \frac{\theta^{x+a-1}(1-\theta)^{n-x+b-1}}{\int_0^1 \theta^{x+a-1}(1-\theta)^{n-x+b-1}dp} \\ &= \frac{\theta^{x+a-1}(1-\theta)^{n-x+b-1}}{\beta(x+a,n-x+b)} \end{aligned}$$

que también es la función de densidad de una beta. Por tanto, la familia de las distribuciones beta es una familia a priori conjugada para θ.

Solo faltaría conocer la distribución de la marginal g, pero, una vez conocida $h(\theta|\mathbf{x})$, podemos calcular g como

$$g(\mathbf{x}) = \frac{\pi(\theta)f(\mathbf{x}|\theta)}{h(\theta|\mathbf{x})}$$

En la siguiente tabla se recogen algunos algunos ejemplos de familias conjugadas.

Familias conjugadas		
$f(\mathbf{x}\|\theta)$	Distribución a priori $\pi(\theta)$	Distribución a posteriori $h(\theta\|\mathbf{x})$
$N(\theta,\sigma^2)$	$N(\mu,\tau^2)$	$N(\frac{\sigma^2\mu+x\tau^2}{\sigma^2+\tau^2}, \frac{\sigma^2\tau^2}{\sigma^2+\tau^2})$
$\Gamma(\nu,\beta)$	$\Gamma(\alpha,\beta)$	$\Gamma(\nu+\alpha,\beta+x)$
$b(n,p)$	$\beta(a,b)$	$\beta(a+x,b+n-x)$
$P(\lambda)$	$\Gamma(\alpha,\beta)$	$\Gamma(\alpha+x,\beta+1)$
$\Gamma(\gamma,1/\theta)$	$\Gamma(\alpha,\beta)$	$\Gamma(\gamma+\alpha,\beta+x)$

Veamos ahora dos condiciones suficientes para ser estimador *minimax*.

Teorema 2.2.2. *Sea $\{f_\theta : \theta \in \Theta\}$ una familia de funciones de probabilidad o densidad, y supongamos que δ^* es un estimador de Bayes asociado a la distribución a priori π. Si la función de riesgo $R(\theta, \delta^*)$ es constante en Θ, entonces δ^* es un estimador minimax de θ.*

Demostración. Como δ^* es un estimador de Bayes de θ con función de riesgo constante, r^*, tenemos

$$\begin{aligned} r^* = R(\pi, \delta^*) &= \int_{-\infty}^{\infty} R(\theta, \delta^*)\pi(\theta)d\theta \\ &= \inf_{\delta\in\Delta} \int R(\theta, \delta)\pi(\theta)d\theta \\ &\leq \sup_{\theta\in\Theta} \inf_{\delta\in\Delta} R(\theta, \delta) \\ &\leq \inf_{\delta\in\Delta} \sup_{\theta\in\Theta} R(\theta, \delta) \end{aligned}$$

Además, como $r^* = R(\theta, \delta^*)$ para todo $\theta \in \Theta$, se tiene

$$r^* = \sup_{\theta\in\Theta} R(\theta, \delta^*) \leq \inf_{\delta\in\Delta} \sup_{\theta\in\Theta} R(\theta, \delta)$$

Por tanto,

$$\sup_{\theta\in\Theta} R(\theta, \delta^*) = \inf_{\delta\in\Delta} \sup_{\theta\in\Theta} R(\theta, \delta)$$

y por tanto, δ^* es minimax. □

Teorema 2.2.3. *Sea $\{\pi_k(\theta) : k \geq 1\}$ una sucesión de distribuciones a priori en Θ y sea $\{\delta_k^*\}$ la correspondiente sucesión de estimadores de Bayes con riesgo $R(\pi_k, \delta_k^*)$. Si $\lim\sup_{k\to\infty} R(\pi_k, \delta_k^*) = r^*$ y existe un estimador δ^* tal que*

$$\sup_{\theta\in\Theta} R(\theta, \delta^*) \leq r^* \tag{2.11}$$

entonces δ^* es minimax.

Demostración. Razonamos por reducción al absurdo. Supongamos que δ^* no es minimax. Entonces, existe un estimador $\tilde{\delta}$ tal que

$$\sup_{\theta\in\Theta} R(\theta, \tilde{\delta}) \leq \sup_{\theta\in\Theta} R(\theta, \delta^*)$$

Por otro lado, consideremos los estimadores de Bayes $\{\delta_k^*\}$ asociados a las distribuciones $\{\pi_k(\theta)\}$. Se tiene

$$\begin{aligned} R(\pi_k, \delta_k^*) &= \int R(\theta, \delta_k^*)\pi_k(\theta)d\theta \\ &\leq \int R(\theta, \tilde{\delta})\pi_k(\theta)d\theta \\ &\leq \sup_{\theta\in\Theta} R(\theta, \tilde{\delta}) \end{aligned}$$

lo cual contradice 2.11. Por tanto, δ^* es minimax. □

Estimación Bayesiana en una BN

Veamos a continuación un resultado que nos permite obtener la curva de regresión en un caso muy particular y que nos será de gran utilidad a la hora de obtener estimadores en una red de Bayes. En esta subsección seguiremos el libro de Jensen-Nielsen [3, Capítulo 6].

Teorema 2.2.4. *Sea X una variable binaria. Supongamos que realizamos n+m experimentos independientes y en n de ellos se obtiene X=1. Sea* $\theta = P(X = 1)$. *Entonces, partiendo de la distribución a priori uniforme en (0,1) para* θ, *la distribución a posteriori viene dada por*

$$h(\theta|\boldsymbol{x}) = h(\theta) = \mu\theta^n(1-\theta)^m I_{(0,1)},$$

donde

$$\mu = \frac{(n+m+1)!}{n! \quad m!}$$

Además, el estimador de Bayes para θ *es*

$$\delta = \frac{n+1}{n+m+2}$$

Demostración. Sea $\pi(\theta) = 1$ para $0 \leq \theta \leq 1$. Consideremos también la muestra aleatoria $X_1, ..., X_{n+m}$ de X. Entonces

$$\begin{aligned} h(\theta|\mathbf{x}) &= \frac{f(\mathbf{x}|\theta)\pi(\theta)}{g(\mathbf{x})} \\ &= \mu\theta^n(1-\theta)^m \end{aligned}$$

donde μ verifica

$$1 = \int_0^1 h(\theta|\mathbf{x})d\theta = \mu\int_0^1 \theta^n(1-\theta)^m d\theta = \mu\beta(n+1, m+1)$$

Como $n, m \in \mathbb{N}$, se tiene

$$\mu = \beta(n+1, m+1)^{-1} = \frac{\Gamma(n+m+2)}{\Gamma(n+1)\Gamma(m+1)} = \frac{(n+m+1)!}{n! \quad m!}$$

Por tanto, el estimador de Bayes o la curva de regresión de θ con respecto a $\mathbf{X}$ es

$$\begin{aligned}\delta(\mathbf{X}) &= \int_0^1 \theta h(\theta|x)d\theta = \mu\int_0^1 \theta^{n+1}(1-\theta)^m d\theta = \mu\beta(n+2,m+1)\\ &= \mu\frac{\Gamma(n+2)\Gamma(m+1)}{\Gamma(n+m+3)} = \frac{(n+m+1)!\quad (n+1)!}{n!\quad (n+m+2)!} = \frac{n+1}{n+m+2}\end{aligned}$$

□

Observación 2.2.1. *Observemos que el estimador de Bayes dado por el teorema anterior es siempre distinto de 0, por tanto, hemos eliminado el problema de la estimación nula.*

Análogamente, el teorema 2.2.4 se extiende al caso de variables discretas con soporte finito.

Teorema 2.2.5. *Sea $X_1,\ldots,X_n$ una muestra aleatoria simple de X, variable discreta con función de probabilidad*

$$P(X = x_k^0) = \theta_k \ , \ k = 1,\ldots,r$$

donde r denota el cardinal del soporte de X. Supongamos que realizamos $n = n_1+\cdots+n_r$ experimentos independientes, siendo n_k el número de veces que se ha obtenido $X = x_0^k$. Si $\pi(\theta) = I_{(0,1)^r}$,entonces el estimador de Bayes para θ_k es

$$\delta_k = \frac{n_k+1}{n+r}$$

Demostración. Se tiene

$$h(\theta|\mathbf{x}) = h(\theta) = \mu\theta_1^{n_1}\ldots\theta_r^{n_r}$$

donde μ verifica

$$1 = \int_\Theta \mu\theta_1^{n_1}\ldots\theta_r^{n_r} = \mu\int_\Theta \theta_1^{n_1}\ldots\theta_r^{n_r} \tag{2.12}$$

con $\Theta = \{\theta \in \mathbb{R}^r : \theta_i \geq 0 \quad \forall i = 1,\ldots,r, \quad \sum_{i=1}^r \theta_i = 1\}$. Ahora bien, el segundo factor de la expresión 2.12 se corresponde con la distribución Dirichlet, que es una generalización multivariante de la distribución Beta. Así,

$$1 = \mu\beta(n_1+1,\ldots,n_r+1) = \mu\frac{\Gamma(n_1+1)\ldots\Gamma(n_r+1)}{\Gamma(n+r)} = \mu\frac{n_1!\ldots n_r!}{(n+r-1)!}$$

Por tanto,

$$\mu = \frac{(n+r-1)!}{n_1! \dots n_r!}$$

Calculemos entonces el estimador de Bayes para θ_k:

$$\begin{aligned}\delta_k(\mathbf{X}) &= \mu \int_\Theta \theta_1^{n_1} \dots \theta_k^{n_k+1} \dots \theta_r^{n_r} = \mu\beta(n_1+1, \dots, n_k+2, \dots, n_r+1) \\ &= \mu \frac{\Gamma(n_1+1)\dots\Gamma(n_k+2)\dots\Gamma(n_r+1)}{\Gamma(n+r+1)} \\ &= \frac{(n+r-1)!}{n_1!\dots n_r!} \frac{n_1!\dots(n_k+1)!\dots n_r!}{(n+r)!}\end{aligned}$$

Simplificando los factoriales de la expresión anterior obtenemos que el estimador de Bayes para θ_k es

$$\delta_k = \frac{n_k+1}{n+r}$$

□

2.3. Uso de la librería *bnlearn* del *software R* en la estimación de parámetros

En esta sección, explicaremos brevemente los comandos de la librería *bnlearn* necesarios para la creación de una red Bayesiana dada una base de datos y para la posterior estimación de parámetros mediante los métodos desarrollados en las secciones anteriores.

En la sección 3.3 aplicaremos esos comandos a una base de datos concreta y analizaremos los resultados.

2.3.1. Estructura y aprendizaje de una red Bayesiana

El aprendizaje de la estructura de la red Bayesiana asociada a una base de datos puede ser realizado mediante alguno de los siguientes algoritmos (véase sección 1.4).

- *Grow-Shrink* : `gs(data)`
- *Incremental Association*: `iamb(data)`
- *Fast Incremental Association*: `fast.iamb(data)`
- *Interleaved Incremental Association*: `inter.iamb(data)`
- Algoritmo de escalada: `hc(data)`.

Todos estos algoritmos toman como argumento una estructura de datos o *data frame* `data`, que debe contener todas las variables del modelo.

En ocasiones ocurre que los comandos anteriores devuelven una red no completamente dirigida. Estas funciones no siempre son capaces de detectar la dirección de un arco, entre otras cosas porque recordemos que existen redes diferentes pero que son equivalentes respecto de la familia de probabilidades conjuntas que factorizan sobre ellas. Cuando esto ocurre, existen dos posibles direcciones con la misma puntuación. Para detectar estas puntuaciones usamos los siguientes comandos:

- `score(x, data)`: calcula la puntuación de la estructura *bn* `x`, aprendida a partir de los datos `data`.
- `set.arc(x, from, to)`: fija la dirección de un eje de la estructura *bn* `x`.

Así, si por ejemplo, el arco no dirigido es el que une los nodos A y B, podemos detectar dichas puntuaciones de la siguiente manera:

```
> score(set.arc(metodo(data), from = " A", to = "B"), data)
> score(set.arc(metodo(data), from = "B", to = " A"), data)
```

donde `metodo` es uno de los algoritmos de aprendizaje estructural nombrados anteriormente. Cuando el arco es no dirigido, comprobaremos que las puntuaciones anteriores coinciden.

Como las redes Bayesianas se representan mediante grados acíclicos dirigidos, no podemos tener arcos no orientados, pues, en ese, caso, no sería posible estimar los parámetros de la red (la función $f(\mathbf{x}|\theta)\pi(\theta)$ no estaría completamente determinada). Para solventar este problema, fijamos la dirección de los arcos no orientados. De nuevo, usamos la función `set.arc`.

Continuando con el ejemplo anterior, para fijar la dirección del eje $A-B$ bastaría con introducir en R la siguiente orden:

```
> set.arc(metodo(data), from = "B", to = " A")
```

2.3.2. Estimación paramétrica con *bnlearn*

Una vez tenemos la red completamente dirigida, ya podemos fijar los parámetros de las distribuciones marginales, que adoptan la forma de tablas de probabilidad condicionada, tal y como comentábamos en la sección 1.5.

Para estimar los parámetros de la red, usamos el comando

```
> bn.fit (x,data,method).
```

Este comando toma como argumentos un grafo acíclico de estructura *bn*, una base de datos y un método de estimación.

Si queremos estimar los parámetros mediante máxima verosimilitud debemos introducir `method="mle"`. Por defecto, *R* toma este método de estimación. Por otro lado, si nuestro deseo es obtener los estimadores de Bayes de la red, basta introducir la orden `method="bayes"`.

La estimación de Bayes mediante el comando `bn.fit` se realiza tomando como distribuciones a priori las distribuciones $K2$, que simplemente suman 1 al conteo de cada nodo particular, es decir, no son más que distribuciones uniformes en el intervalo $(0, 1)$. Existe también una distribución a priori más sensible conocida como *BDeu* (*Bayesian Dirichlet equivalent uniform prior*). Para más detalle, véase el capítulo 2 del artículo de *M. Scutari* [7]

En la sección 3.3 veremos con más detalle la aplicación de la función `bn.fit`.

Capítulo 3

Validación de la red de Bayes y aplicación experimental

La validación de una red Bayesiana puede ser llevada a cabo mediante varias técnicas. En esta sección nos centraremos en la técnica *bootstrap* y en la validación cruzada (*cross-validation*). A lo largo de este capítulo seguiremos el libro de R. Nagarajan et al. [5, Capítulo 5, Sección 4].

3.1. Uso del método *bootstrap* con el *software R*

Consideremos una muestra $\mathbf{X}_1, \ldots, \mathbf{X}_n$ del vector aleatorio $\mathbf{X}$, que sigue una distribución F_θ con θ un parámetro desconocido. El método *bootstrap* es una técnica estadística de remuestreo utilizada para aproximar características de la distribución de un estadístico $T(\theta)$.

En las redes Bayesianas, esta técnica se emplea para investigar las propiedades de los parámetros de la red o de la estructura de la misma.

Distinguimos entre *bootstrap* paramétrico, si se supone conocida la familia de funciones de distrubución F_θ que dependen de un parámetro, y *bootstrap* no paramétrico, si la distribución es desconocida. Nos centraremos en este último caso que, además, es el que viene implementado en la librería *bnlearn* de *R*. Veamos en qué consiste.

Supongamos que estamos interesados en estimar la característica f de la estructura de una red. Sea $B \in \mathbb{N}$. Para cada $i = 1, \ldots, B$ tomamos una muestra con reemplazamiento de la base de datos original, generalmente del mismo tamaño que la muestra original. Sea G_i^* la red aprendida a través de la muestra *bootstrap* y sea $f(G_i^*)$ la característica de interés aplicada a la i-ésima red aprendida.

Normalmente, se toma como estimación el promedio o la varianza de los valores $f(G_i^*)$ obtenidos. Veáse la figura 3.1

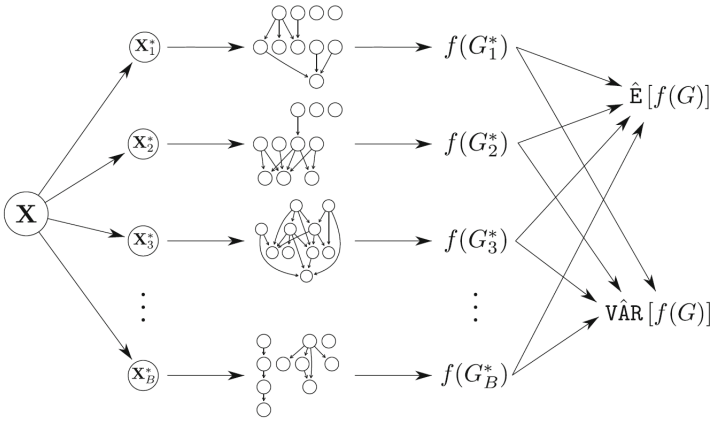


Figura 3.1: Estimación mediante *bootstrap* no paramétrico de la propiedad f de una red Bayesiana.

La función que implementa en el *software* R el método *bootstrap* en el contexto de redes Bayesianas es la siguiente:

```
> bn.boot(data, statistic, R = 200, m = nrow(data), algorithm,
algorithm.args = list(), statistic.args = list(), cluster =
NULL)
```

donde

- `data` es la base de datos, conteniendo las variables del modelo.
- `statistic` es el estadístico o propiedad de la red que se aplicará a cada muestra *bootstrap*.
- `R` es un entero positivo que indica el número de muestras *bootstrap* (en la notación anterior `R` es B).
- `m` es un entero positivo que indica el tamaño de cada muestra *bootstrap*. Por defecto, se toma el tamaño de la muestral original.
- `algorithm` es el algoritmo utilizado para aprender la red.
- `algorithm.args` es una lista de argumentos extra en relación al algoritmo de aprendizaje.
- `statistic.args` es una lista de argumentos extra en relación al estadístico.

- `cluster` es un objeto cluster opcional de la librería *parallel*.

Una aplicación interesante en la que aplicar el método *bootstrap* es la siguiente. Supongamos que queremos estudiar el número de arcos (`narcs`) que contiene la red asociada a nuestra muestra. Esta propiedad de la red nos indica si se trata de una red densa o no. Se dice que una red es poco densa cuando el número de arcos de la misma es mucho menor que el número máximo de arcos posibles, que es $\binom{N}{2}$, donde N denota el número de variables del modelo. Las redes poco densas son particularmente útiles en el análisis de datos reales, es los que el número de variables es considerablemente grande. Mientras más densa sea la red, mayor es la familia de distribuciones que factoriza sobre ella y, por tanto, el método será menos específico e interpretable. Además, las redes poco densas son computacionalmente más manejables.

Otra aplicación interesante de esta técnica es la siguiente. Ya hemos visto que los algoritmos de aprendizaje en el *software R* no siempre nos devuelven redes completamente dirigidas. Es por ello que debemos estimar el sentido de los ejes y posteriormente fijar el arco correspondiente para así poder trabajar con el grafo dirigido. Para ello, usamos la siguiente función de *bnlearn*:

```
> boot.strength(data, R = 200, m = nrow(data), algorithm,
algorithm.args = list())
```

cuyos argumentos son los mismos que los que utiliza la función `bn.boot` previamente comentada.

Esta función estima la relevancia de cada posible arco en la red aprendida aplicando *bootstrap* no paramétrico al conjunto de datos. La función nos devuelve, para cada par de nodos, la probabilidad de que exista un eje entre ambos sin tener en cuenta la dirección y la probabilidad de cada arco. Así, si en la red aprendida aparece un eje, podemos fijar el sentido que tenga una mayor probabilidad estimada.

En la sección 3.3 veremos cómo aplicar estas funciones a nuestro conjunto de datos particular.

3.2. Validación cruzada con el *software R*

La técnica de validación cruzada o *cross-validation* es, probablemente, una de las técnicas más simples y más usadas para validar modelos estadísticos. En nuestro caso, utilizaremos este método de validación para estimar funciones de pérdidas.

Existen distintas técnicas de validación cruzada. Las implementadas en el *software R* son las siguientes:

- Validación cruzada con k pliegues (`k-fold`). Consiste en lo siguiente. Sea un entero k (normalmente k=10). Dividimos aleatoriamente la muestra original en k grupos aproximadamente del mismo tamaño. Así, se originan k submuestras de validación, $X_1^*, \ldots, X_k^*$. Para cada grupo, tomamos este como muestra test, y el resto de grupos como muestra de aprendizaje (construimos la red sobre la muestra de aprendizaje y evaluamos el error o la función de pérdidas sobre la muestra test). Tomamos como estimación del error el promedio de los k errores así obtenidos. Véase la siguiente figura.

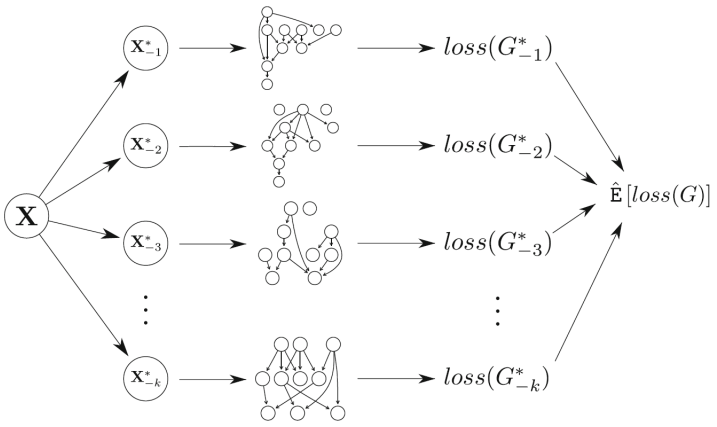


Figura 3.2: Estimación mediante validación cruzada con k pliegues de una función de pérdidas de un algoritmo de aprendizaje de una red Bayesiana.

  Por defecto, la función `bn.cv` de la librería *bnlearn* toma este método como método de validación cruzada.

- Pliegues "a medida"(`custom-folds`). En esta ténica de validación, los datos se dividen manualmente en subconjuntos de tamaño no necesariamente iguales. Estos grupos se utilizan después como los k pliegues en la validación cruzada.

- Hold-out (`hold-out`). Consiste en lo siguiente. Sea n el tamaño de la muestra original, es decir, `nrow(data)`. Dado un entero positivo k, se toman k submuestras sin reemplazamiento de tamaño m a partir de la muestra de datos original. Para cada submuestra, se toma esta como muestra de aprendizaje y las $n - m$ observaciones restantes como muestra test (de nuevo, construimos la red sobre la muestra de aprendizaje y evaluamos la función de pérdidas en la muestra test).

Tomamos como estimación de la función de pérdidas el valor esperado de los k valores obtenidos.

La función que realiza en *R* la validación cruzada en el contexto de redes Bayesianas es la siguiente:

```
bn.cv(data, bn, loss = NULL, ..., algorithm.args = list(),
loss.args = list(), fit, fit.args = list(), method, cluster =
NULL, debug = FALSE)
```

donde

- `data` es la base de datos, conteniendo las variables del modelo.
- `bn` es la etiqueta del algoritmo de aprendizaje aplicada a los datos en cada iteración o un objeto de clase `bn` (una red ya fijada).
- `loss` es el nombre de la función de pérdidas. Más adelante veremos más detalles sobre este argumento.
- `algorithm.args` es una lista de argumentos extra en relación al algoritmo de aprendizaje.
- `loss.args` es una lista de argumentos extra en relación a la función de pérdidas.
- `fit` es el método utilizado para estimar los parámetros de la red (véase subsección 2.3.2).
- `fit.args` es una lista de argumentos extra en relación al método de estimación.
- `method` es uno de los tres métodos de validación cruzada comentados previamente.
- `cluster` es un objeto cluster opcional de la librería *parallel*.

También podemos incluir los siguientes argumentos:

- `k`, un entero positivo que indica el número de grupos en los que se divide la muestra en el método de los k-pliegues o el número de submuestras tomadas en el método de *hold-out*. Como comentábamos, por defecto `k=10`.
- `m`, un entero positivo que indica el tamaño de cada muestra test en el método de *hold-out*.
- `runs`, un entero positivo que indica el número de veces que se ejecuta el método de los k-pliegues o el de *hold-out*.

- `folds`, una lista en la que cada elemento corresponde con un pliegue y contiene las observaciones incluidas en ese pliegue, usado en el método de pliegues "a medida".

Podemos usar, por ejemplo, la siguiente función para crear bucles aleatorios:

```
> random.folds = function(data, k) split(sample(nrow(data)),
seq(k))
```

Algunas de las funciones de pérdidas implementadas en R son las siguientes:

- Error de clasificación (`pred`): es el error de predicción de un nodo en una red discreta. Frecuentemente se utilizan métodos de predicción para predecir el valor de un nodo en particular a partir de sus padres.
- Pérdida de log-verosimilitud (`logl`)
- Pérdida de log-verosimilitud Gaussiana(`logl-g`):
- Posterior error de clasificación (`pred-lw` and `pred-cg`)
- Error cuadrático medio (`mse`): es el error cuadrático medio entre los datos observados y las predicciones de un nodo particular de una red Gaussiana.

3.3. Aplicación experimental

En esta última sección, simularemos una red usando la librería *bnlearn* del *software* R y compararemos los estimadores de máxima verosimilitud y los estimadores de Bayes obtenidos analíticamente en los capítulos anteriores. También usaremos las funciones de validación previamente comentadas.

En nuestro estudio, tomaremos una muestra de tamaño $n = 5000$ con 9 variables, que han sido tomadas del *Informe nº 22. Situación de COVID-19 en España a 13 de abril de 2020. Equipo COVID- 19* elaborado por la Red Nacional de Vigilancia Epidemiológica. Son las siguientes.

- *Edad* (Edad), clasificada por intervalos:
  - <2
  - 2-4
  - 5-14
  - 15-29
  - 30-39

    - 40-49
    - 50-59
    - 60-69
    - > 80
- *Sexo* (Sex)
    - Mujer
    - Hombre

Las variables siguientes están relacionadas con los síntomas del paciente y tomarán el valor 1 si el paciente padece tal síntoma y 0 en caso contrario.

- *Fiebre* (Fie)
- *Dolor de garganta* (DG)
- *Síndrome de distrés respiratorio agudo* (SDRA)

Las siguientes variables están relacionadas con enfermedades y factores de riesgo y tomarán el valor 1 si el paciente padece dicha enfermedad y 0 si no la padece.

- *Neumonía* (Neum)
- *Enfermedad respiratoria* (EnRes)

Las últimas variables corresponden a la situación clínica del paciente:

- *Hospitalización* (Hosp), que valdrá 1 si se requiere de hospitalización y 0 en caso contrario.
- *Defunción* (Def), que valdrá 1 si el paciente fallece y 0 en caso contrario.

Comencemos el estudio en el *software R*.

En primer lugar, cargamos las librerías necesarias e introducimos en *R* la base de datos. La librería *Rgraphviz* nos permitirá representar redes.

```
#install.packages("bnlearn")
#install.packages("BiocManager")
#BiocManager::install("Rgraphviz")
library(bnlearn)
library(ggplot2)
library(Rgraphviz)

datos=read.csv("datoscovid.csv")
head(datos)
```

```
##   X Def  Edad    Sex DG Hosp Neum Fieb SDRA EnRes
## 1 1  No 60-69 Hombre No   Si   No   Si   No    No
## 2 2  No 70-79  Mujer No   No   Si   No   No    No
## 3 3  No 40-49 Hombre Si   Si   Si   Si   No    No
## 4 4  No 70-79 Hombre No   Si   Si   Si   No    No
## 5 5  No 40-49  Mujer No   Si   No   Si   No    No
## 6 6  No 50-59 Hombre No   Si   Si   Si   No    No
```

```
dim(datos)
```

```
## [1] 5002   10
```

```
str(datos)
```

```
## 'data.frame':    5002 obs. of  10 variables:
##  $ X    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Def  : chr  "No" "No" "No" "No" ...
##  $ Edad : chr  "60-69" "70-79" "40-49" "70-79" ...
##  $ Sex  : chr  "Hombre" "Mujer" "Hombre" "Hombre" ...
##  $ DG   : chr  "No" "No" "Si" "No" ...
##  $ Hosp : chr  "Si" "No" "Si" "Si" ...
##  $ Neum : chr  "No" "Si" "Si" "Si" ...
##  $ Fieb : chr  "Si" "No" "Si" "Si" ...
##  $ SDRA : chr  "No" "No" "No" "No" ...
##  $ EnRes: chr  "No" "No" "No" "No" ...
```

Eliminamos la primera columna y transformamos las variables a variables de tipo *factor*.

```r
datos=datos[,-1]

for (i in (1:ncol(datos))){
datos[,i]=factor(datos[,i])}

str(datos)
```

```
## 'data.frame':    5002 obs. of  9 variables:
##  $ Def  : Factor w/ 2 levels "No","Si": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Edad : Factor w/ 10 levels "<2",">80","15-29",..: 9 10 6 10 6 8 8 8 3 10 ...
##  $ Sex  : Factor w/ 2 levels "Hombre","Mujer": 1 2 1 1 2 1 1 1 2 1 ...
##  $ DG   : Factor w/ 2 levels "No","Si": 1 1 2 1 1 1 1 1 1 2 ...
##  $ Hosp : Factor w/ 2 levels "No","Si": 2 1 2 2 2 2 2 1 1 1 ...
##  $ Neum : Factor w/ 2 levels "No","Si": 1 2 2 2 1 2 2 1 1 1 ...
##  $ Fieb : Factor w/ 2 levels "No","Si": 2 1 2 2 2 2 2 2 2 2 ...
##  $ SDRA : Factor w/ 2 levels "No","Si": 1 1 1 1 1 1 1 1 1 1 ...
##  $ EnRes: Factor w/ 2 levels "No","Si": 1 1 1 1 1 1 1 1 1 1 ...
```

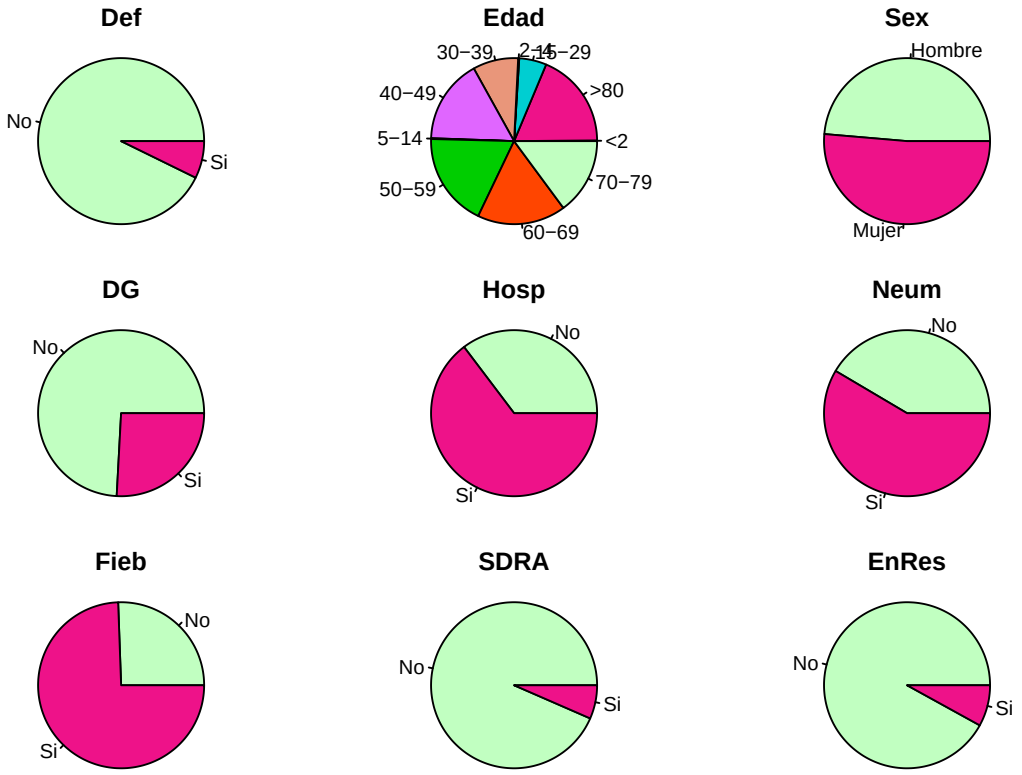
Podemos obtener también un resumen de los datos y una representación gráfica, que nos permitirá tener una idea general de los perfiles de nuestra base de datos.

```r
summary(datos)
```

```
##  Def            Edad         Sex         DG       Hosp       Neum       Fieb
##  No:4639   >80    :928   Hombre:2433   No:3709   No:1766   No:2076   No:1279
##  Si: 363   50-59  :915   Mujer :2569   Si:1293   Si:3236   Si:2926   Si:3723
##            60-69  :862
##            40-49  :819
##            70-79  :744
##            30-39  :443
##            (Other):291
##  SDRA      EnRes
##  No:4675   No:4601
##  Si: 327   Si: 401
##
##
##
##
##
```

```r
colores=c("darkseagreen1","deeppink2",  "darkturquoise",
"darkorchid1","darksalmon", "mediumorchid1","mistyrose",
"green3","orangered")
par(mar=c(1,1,1,1))
```

```
par(mfrow=c(3,3))
for (i in 1:(ncol(datos)))
     {
  pie(table(datos[,i]), main=colnames(datos)[i], col=colores)
}
```



Veamos si los valores que toma la variable *Edad* están ordenados.

```
levels(datos$Edad)
```

```
##  [1] "<2"    ">80"   "15-29" "2-4"   "30-39" "40-49" "5-14"  "50-59" "60-69"
## [10] "70-79"
```

Como no lo están, procedemos a reordenarlos.

```
datos$Edad=factor(datos$Edad, labels = levels(datos$Edad)
[ c( 1,4,7,3,5,6,8,9,10,2)])
```

```
levels(datos$Edad)
```

```
##  [1] "<2"    "2-4"   "5-14"  "15-29" "30-39" "40-49" "50-59" "60-69" "70-79"
## [10] ">80"
```

Ahora que ya tenemos nuestra base de datos correctamente definida, podemos empezar a aprender la red correspondiente a la misma. Vamos a realizar el aprendizaje estructural mediante varios métodos y compararemos los resultados.

En primer lugar, vamos a aprender la red mediante el algoritmo *Incremental Association*.

```
red_iamb=iamb(datos)
red_iamb
```

```
##
##   Bayesian network learned via Constraint-based methods
##
##   model:
##     [partially directed graph]
##   nodes:                                 9
##   arcs:                                  8
##     undirected arcs:                     1
##     directed arcs:                       7
##   average markov blanket size:           2.00
##   average neighbourhood size:            1.78
##   average branching factor:              0.78
##
##   learning algorithm:                    IAMB
##   conditional independence test:         Mutual Information (disc.)
##   alpha threshold:                       0.05
##   tests used in the learning procedure:  284
```

La red aprendida mediante este algoritmo tiene 8 arcos, de los cuales 1 es no dirigido. A continuación, aprendemos la red mediante el algoritmo de escalada y posteriormente compararemos gráficamente ambas redes.

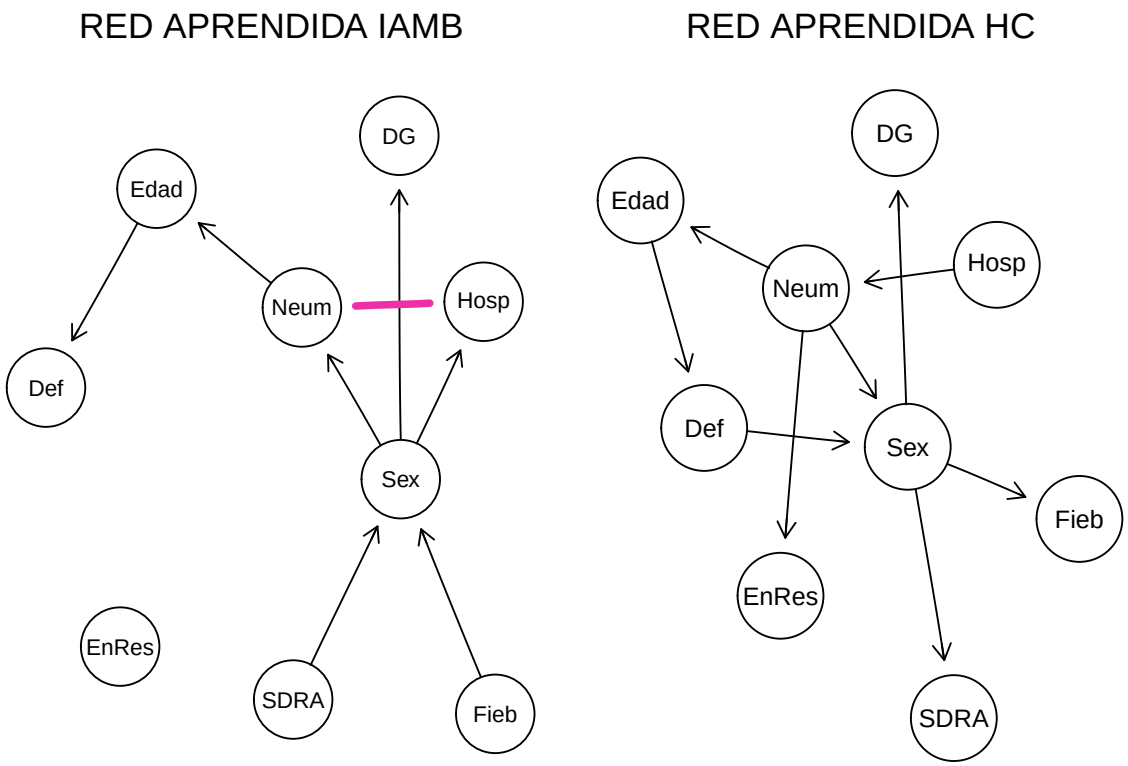
```
red_hc=hc(datos)
red_hc
```

```
##
##   Bayesian network learned via Score-based methods
##
##   model:
##    [Hosp][Neum|Hosp][Edad|Neum][EnRes|Neum][Def|Edad][Sex|Def:Neum][DG|Sex]
```

```
##    [Fieb|Sex][SDRA|Sex]
##   nodes:                                 9
##   arcs:                                  9
##     undirected arcs:                     0
##     directed arcs:                       9
##   average markov blanket size:           2.22
##   average neighbourhood size:            2.00
##   average branching factor:              1.00
##
##   learning algorithm:                    Hill-Climbing
##   score:                                 BIC (disc.)
##   penalization coefficient:              4.258797
##   tests used in the learning procedure:  124
##   optimized:                             TRUE
```

Esta red, en cambio, tiene 9 arcos pero todos están dirigidos.

```
arcos=c("Hosp","Neum")
par(mar=c(1,1,1,1))
par(mfrow=c(1,2))
graphviz.plot(red_iamb, highlight = list( arcs =arcos,
col="maroon2",lwd = 4), layout = "fdp",
main = "RED APRENDIDA IAMB")
#graphviz.plot(red_iamb, layout = "fdp", main = "RED APRENDIDA IAMB")
graphviz.plot(red_hc,layout = "fdp", main = "RED APRENDIDA HC")
```



Obsevamos que, la principal diferencia entre ambas redes aprendidas, es la orientación de los arcos. Vamos a comenzar trabajando con la red aprendida mediante el método de escalada, `red_hc` y, posteriormente, aplicaremos técnicas de validación para orientar el arco no dirigido de `red_iamb` . Recordemos que para poder estimar parámetros de una red, esta debe estar completamente dirigida.

A continuación, estimamos los parámetros de `red_hc` utilizando el método de estimación de Máxima Verosimilitud.

```
bn.fit(red_hc,datos,method="mle")
```

```
##
##   Bayesian network parameters
##
##   Parameters of node Def (multinomial distribution)
##
## Conditional probability table:
##
##     Edad
```

```
## Def           <2         2-4        5-14       15-29       30-39       40-49
##   No 1.000000000 0.760775862 0.996240602 1.000000000 0.995485327 0.993894994
##   Si 0.000000000 0.239224138 0.003759398 0.000000000 0.004514673 0.006105006
##     Edad
## Def        50-59       60-69       70-79         >80
##   No 1.000000000 0.986885246 0.974477958 0.866935484
##   Si 0.000000000 0.013114754 0.025522042 0.133064516
##
##   Parameters of node Edad (multinomial distribution)
##
## Conditional probability table:
##
##        Neum
## Edad              No           Si
##   <2    0.0024084778 0.0006835270
##   2-4   0.1372832370 0.2197539303
##   5-14  0.1006743738 0.0194805195
##   15-29 0.0043352601 0.0003417635
##   30-39 0.1421001927 0.0505809979
##   40-49 0.2138728324 0.1281613124
##   50-59 0.0028901734 0.0006835270
##   60-69 0.1888246628 0.1787423103
##   70-79 0.1329479769 0.2002734108
##   >80   0.0746628131 0.2012987013
##
##   Parameters of node Sex (multinomial distribution)
##
## Conditional probability table:
##
## , , Neum = No
##
##         Def
## Sex             No        Si
##   Hombre 0.3743668 0.5392157
##   Mujer  0.6256332 0.4607843
##
## , , Neum = Si
##
##         Def
## Sex             No        Si
##   Hombre 0.5519700 0.6436782
##   Mujer  0.4480300 0.3563218
##
##
##   Parameters of node DG (multinomial distribution)
##
```

```
## Conditional probability table:
##
##     Sex
## DG      Hombre     Mujer
##   No 0.7866831 0.6987155
##   Si 0.2133169 0.3012845
##
##   Parameters of node Hosp (multinomial distribution)
##
## Conditional probability table:
##         No        Si
## 0.3530588 0.6469412
##
##   Parameters of node Neum (multinomial distribution)
##
## Conditional probability table:
##
##     Hosp
## Neum        No        Si
##   No 0.8640997 0.1699629
##   Si 0.1359003 0.8300371
##
##   Parameters of node Fieb (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## Fieb    Hombre     Mujer
##   No 0.2009864 0.3075127
##   Si 0.7990136 0.6924873
##
##   Parameters of node SDRA (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## SDRA     Hombre      Mujer
##   No 0.91491985 0.95328922
##   Si 0.08508015 0.04671078
##
##   Parameters of node EnRes (multinomial distribution)
##
## Conditional probability table:
##
##      Neum
## EnRes         No         Si
```

```
##    No 0.94412331 0.90259740
##    Si 0.05587669 0.09740260
```

Veamos las estimaciones que nos proporciona el método de Bayes.

```
bn.fit(red_hc,datos,method="bayes")
```

```
##
##   Bayesian network parameters
##
##   Parameters of node Def (multinomial distribution)
##
## Conditional probability table:
##
##     Edad
## Def           <2         2-4        5-14       15-29       30-39       40-49
##   No 0.992957746 0.760747764 0.996054115 0.995049505 0.995373505 0.993834697
##   Si 0.007042254 0.239252236 0.003945885 0.004950495 0.004626495 0.006165303
##     Edad
## Def        50-59       60-69       70-79         >80
##   No 0.993827160 0.986832040 0.974422921 0.866886171
##   Si 0.006172840 0.013167960 0.025577079 0.133113829
##
##   Parameters of node Edad (multinomial distribution)
##
## Conditional probability table:
##
##        Neum
## Edad              No           Si
##   <2    0.0024319769 0.0007004955
##   2-4   0.1372742596 0.2197334700
##   5-14  0.1006742114 0.0194942764
##   15-29 0.0043582952 0.0003587904
##   30-39 0.1420900554 0.0505894413
##   40-49 0.2138454130 0.1281565009
##   50-59 0.0029135565 0.0007004955
##   60-69 0.1888032747 0.1787288570
##   70-79 0.1329400433 0.2002562788
##   >80   0.0746689140 0.2012813942
##
##   Parameters of node Sex (multinomial distribution)
##
## Conditional probability table:
##
## , , Neum = No
```

```
##
##         Def
## Sex              No        Si
##   Hombre 0.3743827 0.5391198
##   Mujer  0.6256173 0.4608802
##
## , , Neum = Si
##
##         Def
## Sex              No        Si
##   Hombre 0.5519651 0.6435407
##   Mujer  0.4480349 0.3564593
##
##
##   Parameters of node DG (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## DG      Hombre     Mujer
##   No 0.7866242 0.6986768
##   Si 0.2133758 0.3013232
##
##   Parameters of node Hosp (multinomial distribution)
##
## Conditional probability table:
##         No        Si
## 0.3530881 0.6469119
##
##   Parameters of node Neum (multinomial distribution)
##
## Conditional probability table:
##
##     Hosp
## Neum        No        Si
##   No 0.8639966 0.1700139
##   Si 0.1360034 0.8299861
##
##   Parameters of node Fieb (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## Fieb    Hombre     Mujer
##   No 0.2010479 0.3075501
##   Si 0.7989521 0.6924499
```

```
##
##   Parameters of node SDRA (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## SDRA     Hombre      Mujer
##   No 0.91483460 0.95320101
##   Si 0.08516540 0.04679899
##
##   Parameters of node EnRes (multinomial distribution)
##
## Conditional probability table:
##
##      Neum
## EnRes         No         Si
##    No 0.94401637 0.90252862
##    Si 0.05598363 0.09747138
```

Vamos a estimar mediante los dos métodos conocidos la probabilidad de que una persona con neumonía que ha fallecido sea una mujer. Denotemos por θ dicha probabilidad. La estimación mediante Máxima Verosimilitud obtenida mediante la función `bn.fit` de θ es 0.3563218. En la sección 2.1.3 vimos que el estimador de Máxima Verosimilitud es

$$\hat{\theta} = \frac{M}{N},$$

donde N es el número de individuos de la base de datos en los que $Def =$ "Si", $Neum =$ "Si" y M es el número de casos en los que $Sex =$ "$Mujer$", $Def =$ "Si", $Neum =$ "Si". Calculemos dicho estimador.

```
M=nrow(subset(datos,datos$Sex=="Mujer" & datos$Neum=="Si" &
datos$Def=="Si"))
N=nrow(subset(datos,datos$Def=="Si" & datos$Neum=="Si"))

thetaMV=M/N
thetaMV
```

```
## [1] 0.3563218
```

Calculemos el error cometido en la estimación de Máxima Verosimilitud.

```
errorMV=abs(thetaMV-0.3563218)
errorMV
```

```
## [1] 3.908046e-08
```

Ya vemos que el error cometido es prácticamente nulo. Realicemos la misma comparación

aplicando el método de estimación Bayesiana. El estimador obtenido para θ mediante el método de Bayes es 0.3564593. Consideremos la submuestra de la base de datos original dada por

$$S = \{Def = "Si", Neum = "Si"\}.$$

Entonces, por el teorema 2.2.4, el estimador de Bayes para la probabilidad buscada viene dadod por

$$\hat{\theta} = \frac{\#(S \cap \{Sex = "Mujer"\}) + 1}{\#S + 2}.$$

Es decir, usando la notación empleada en las estimaciones de Máxima Verosimilitud, los estimadores de Bayes no son más que

$$\hat{\theta} = \frac{M + 1}{N + 2}$$

```
thetaB=(M+1)/(N+2)
thetaB
```

```
## [1] 0.3574144
```

Calculemos el error cometido en las estimaciones.

```
errorB=abs(thetaB-0.3564593)
errorB
```

```
## [1] 0.0009551487
```

En este caso, el error es ligeramente mayor, pero de nuevo, muy cercano a 0.

Volvamos ahora a la red aprendida mediante el algoritmo $IAMB$. Recordemos que el arco no orientado es el que une los nodos $Neum$ y $Hosp$. En efecto,

```
score(set.arc(red_iamb, from = "Neum", to = "Hosp"), datos)
```

```
## [1] -27498.81
```

```
score(set.arc(red_iamb, from = "Hosp", to = "Neum"), datos)
```

```
## [1] -27498.81
```

Vamos a utilizar la técnica *bootstrap* para estimar la dirección del eje $Neum - Hosp$. Recordemos que para ello utilizamos la función `boot.strength()`, que estima la relevancia o importancia de cada posible arco dado un conjunto de nodos.

```r
fuerza=boot.strength(datos, R = 100, algorithm="iamb")
subset(fuerza,fuerza$from=="Neum" & fuerza$to=="Hosp")
```

```
##    from   to strength direction
## 45 Neum Hosp        1     0.245
```

```r
subset(fuerza,fuerza$from=="Hosp" & fuerza$to=="Neum")
```

```
##    from   to strength direction
## 37 Hosp Neum        1     0.755
```

La columna *strength* nos indica la probabilidad de que exista un arco entre ambos nodos, sin importar la dirección, y la columna *direction* nos indica la probabilidad de la dirección. La probabilidad de que el arco sea $Neum \rightarrow Hosp$ es 0.245, por lo que la probabilidad de que la dirección sea $Hosp \rightarrow Neum$ es 1-0.245=0.755. Por tanto, concluimos que es más verosímil tomar como dirección $Hosp \rightarrow Neum$.

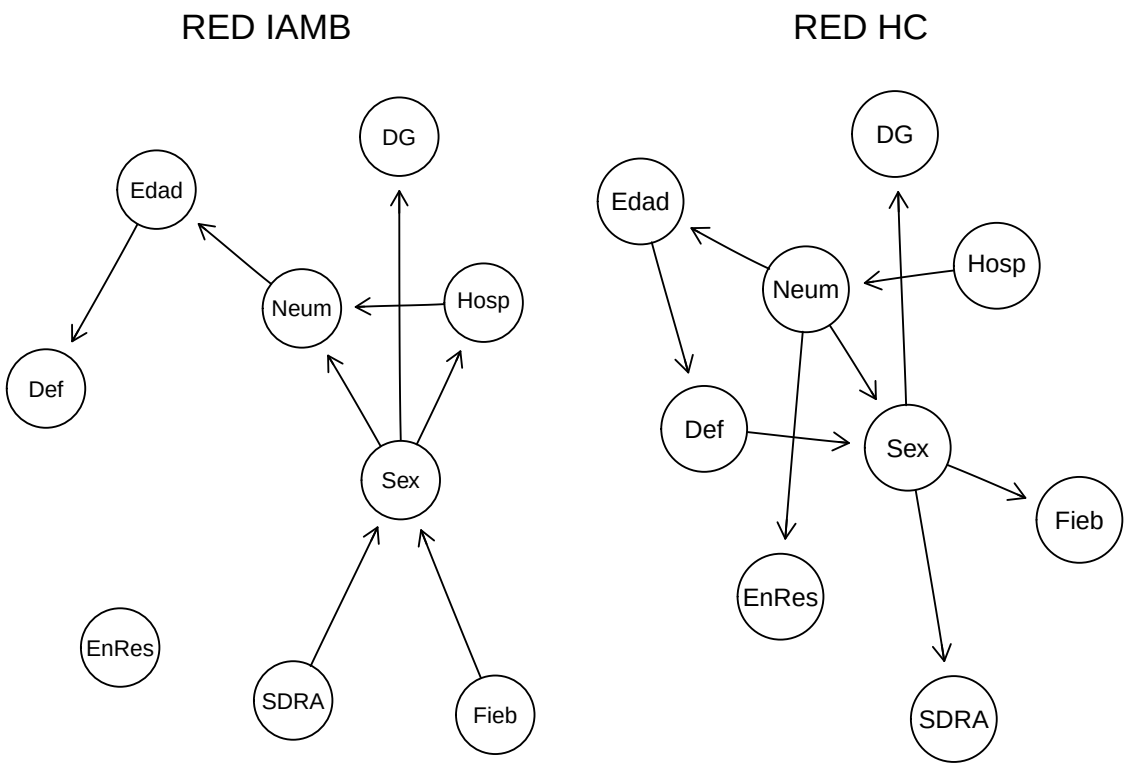
```r
red_iamb=set.arc(red_iamb, from = "Hosp", to = "Neum")
red_iamb
```

```
##
##   Bayesian network learned via Constraint-based methods
##
##   model:
##    [Fieb][SDRA][EnRes][Sex|Fieb:SDRA][DG|Sex][Hosp|Sex][Neum|Sex:Hosp]
##    [Edad|Neum][Def|Edad]
##   nodes:                                 9
##   arcs:                                  8
##     undirected arcs:                     0
##     directed arcs:                       8
##   average markov blanket size:           2.00
##   average neighbourhood size:            1.78
##   average branching factor:              0.89
##
##   learning algorithm:                    IAMB
##   conditional independence test:         Mutual Information (disc.)
##   alpha threshold:                       0.05
##   tests used in the learning procedure:  284
```

Observemos que las redes no se corresponden con el mismo grafo.

```r
par(mar=c(1,1,1,1))
par(mfrow=c(1,2))
```

```
graphviz.plot(red_iamb, layout = "fdp",
main = "RED IAMB ")
graphviz.plot(red_hc,layout = "fdp", main = "RED HC")
```



Una vez tenemos nuestra `red_iamb` completamente dirigida, podemos estimar los parámetros de la misma mediante los dos métodos estudiados en este trabajo.

```
bn.fit(red_iamb,datos,method="mle")
```

```
##
##   Bayesian network parameters
##
##   Parameters of node Def (multinomial distribution)
##
## Conditional probability table:
##
##     Edad
## Def           <2         2-4        5-14       15-29       30-39       40-49
##   No 1.000000000 0.760775862 0.996240602 1.000000000 0.995485327 0.993894994
##   Si 0.000000000 0.239224138 0.003759398 0.000000000 0.004514673 0.006105006
```

```
##     Edad
## Def        50-59       60-69       70-79         >80
##   No 1.000000000 0.986885246 0.974477958 0.866935484
##   Si 0.000000000 0.013114754 0.025522042 0.133064516
##
##   Parameters of node Edad (multinomial distribution)
##
## Conditional probability table:
##
##        Neum
## Edad              No           Si
##   <2    0.0024084778 0.0006835270
##   2-4   0.1372832370 0.2197539303
##   5-14  0.1006743738 0.0194805195
##   15-29 0.0043352601 0.0003417635
##   30-39 0.1421001927 0.0505809979
##   40-49 0.2138728324 0.1281613124
##   50-59 0.0028901734 0.0006835270
##   60-69 0.1888246628 0.1787423103
##   70-79 0.1329479769 0.2002734108
##   >80   0.0746628131 0.2012987013
##
##   Parameters of node Sex (multinomial distribution)
##
## Conditional probability table:
##
## , , SDRA = No
##
##         Fieb
## Sex             No        Si
##   Hombre 0.3821815 0.5086356
##   Mujer  0.6178185 0.4913644
##
## , , SDRA = Si
##
##         Fieb
## Sex             No        Si
##   Hombre 0.3846154 0.7108434
##   Mujer  0.6153846 0.2891566
##
##
##   Parameters of node DG (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
```

```
## DG      Hombre     Mujer
##   No 0.7866831 0.6987155
##   Si 0.2133169 0.3012845
##
##   Parameters of node Hosp (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## Hosp    Hombre     Mujer
##   No 0.2766132 0.4254574
##   Si 0.7233868 0.5745426
##
##   Parameters of node Neum (multinomial distribution)
##
## Conditional probability table:
##
## , , Hosp = No
##
##     Sex
## Neum    Hombre     Mujer
##   No 0.8365527 0.8810613
##   Si 0.1634473 0.1189387
##
## , , Hosp = Si
##
##     Sex
## Neum    Hombre     Mujer
##   No 0.1312500 0.2161247
##   Si 0.8687500 0.7838753
##
##
##   Parameters of node Fieb (multinomial distribution)
##
## Conditional probability table:
##         No        Si
## 0.2556977 0.7443023
##
##   Parameters of node SDRA (multinomial distribution)
##
## Conditional probability table:
##          No         Si
## 0.93462615 0.06537385
##
##   Parameters of node EnRes (multinomial distribution)
##
```

```
## Conditional probability table:
##          No         Si
## 0.91983207 0.08016793
```

```
bn.fit(red_iamb,datos,method="bayes")
```

```
##
##   Bayesian network parameters
##
##   Parameters of node Def (multinomial distribution)
##
## Conditional probability table:
##
##     Edad
## Def            <2         2-4        5-14       15-29       30-39       40-49
##   No 0.992957746 0.760747764 0.996054115 0.995049505 0.995373505 0.993834697
##   Si 0.007042254 0.239252236 0.003945885 0.004950495 0.004626495 0.006165303
##     Edad
## Def         50-59       60-69       70-79         >80
##   No 0.993827160 0.986832040 0.974422921 0.866886171
##   Si 0.006172840 0.013167960 0.025577079 0.133113829
##
##   Parameters of node Edad (multinomial distribution)
##
## Conditional probability table:
##
##        Neum
## Edad              No           Si
##   <2    0.0024319769 0.0007004955
##   2-4   0.1372742596 0.2197334700
##   5-14  0.1006742114 0.0194942764
##   15-29 0.0043582952 0.0003587904
##   30-39 0.1420900554 0.0505894413
##   40-49 0.2138454130 0.1281565009
##   50-59 0.0029135565 0.0007004955
##   60-69 0.1888032747 0.1787288570
##   70-79 0.1329400433 0.2002562788
##   >80   0.0746689140 0.2012813942
##
##   Parameters of node Sex (multinomial distribution)
##
## Conditional probability table:
##
## , , SDRA = No
##
```

```
##         Fieb
## Sex             No        Si
##   Hombre 0.3822060 0.5086350
##   Mujer  0.6177940 0.4913650
##
## , , SDRA = Si
##
##         Fieb
## Sex             No        Si
##   Hombre 0.3849840 0.7106319
##   Mujer  0.6150160 0.2893681
##
##
##   Parameters of node DG (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## DG      Hombre     Mujer
##   No 0.7866242 0.6986768
##   Si 0.2133758 0.3013232
##
##   Parameters of node Hosp (multinomial distribution)
##
## Conditional probability table:
##
##     Sex
## Hosp    Hombre     Mujer
##   No 0.2766591 0.4254719
##   Si 0.7233409 0.5745281
##
##   Parameters of node Neum (multinomial distribution)
##
## Conditional probability table:
##
## , , Hosp = No
##
##     Sex
## Neum    Hombre     Mujer
##   No 0.8364278 0.8809742
##   Si 0.1635722 0.1190258
##
## , , Hosp = Si
##
##     Sex
## Neum    Hombre     Mujer
```

```
##   No 0.1313024 0.2161727
##   Si 0.8686976 0.7838273
##
##
##   Parameters of node Fieb (multinomial distribution)
##
## Conditional probability table:
##         No        Si
## 0.2557466 0.7442534
##
##   Parameters of node SDRA (multinomial distribution)
##
## Conditional probability table:
##          No         Si
## 0.93453928 0.06546072
##
##   Parameters of node EnRes (multinomial distribution)
##
## Conditional probability table:
##          No         Si
## 0.91974815 0.08025185
```

Así por ejemplo, la estimación de Máxima Verosimilitud de la probabilidad de que un hombre que está hospitalizado padezca neumonía es 0.8687500, mientras que la estimación Bayesiana de dicha probabilidad es 0.8686976.

Por último, vamos a utilizar validación cruzada con 10 pliegues para estimar la pérdida de log-verosimilitud de cada una de las redes aprendidas.

```
bn.cv(datos,red_iamb)
```

```
##
##   k-fold cross-validation for Bayesian networks
##
##   target network structure:
##    [Fieb][SDRA][EnRes][Sex|Fieb:SDRA][DG|Sex][Hosp|Sex][Neum|Sex:Hosp]
##    [Edad|Neum][Def|Edad]
##   number of folds:                         10
##   loss function:                           Log-Likelihood Loss (disc.)
##   expected loss:                           5.462146
```

```
bn.cv(datos,red_hc)
```

```
##
##   k-fold cross-validation for Bayesian networks
```

```
##
##   target network structure:
##    [Hosp][Neum|Hosp][Edad|Neum][EnRes|Neum][Def|Edad][Sex|Def:Neum][DG|Sex]
##    [Fieb|Sex][SDRA|Sex]
##   number of folds:                        10
##   loss function:                          Log-Likelihood Loss (disc.)
##   expected loss:                          5.464837
```

La pérdida de log-verosimilitud asociada a la red aprendida mediante el algoritmo $IAMB$ es 5.462146 y la asociada a la red aprendida mediante el método de escalada es 5.464837. Cuanto menor es la pérdida, más se ajusta la red a nuestro modelo. Sin embargo, en este caso, la diferencia de pérdidas es muy pequeña, por lo que podríamos concluir que es indiferente el método utilizado para aprender la red.

Apéndice A

Scripts del *software R*

```
# PAQUETES Y LIBRERÍAS # install.packages("bnlearn")
# install.packages("BiocManager")
library(bnlearn)
library(ggplot2)
library(Rgraphviz)

# 1) LECTURA Y ANÁLISIS DE LOS DATOS
datos=read.csv("datoscovid.csv")
head(datos)
dim(datos)
str(datos)
datos=datos[,-1]

# Transformamos las variables a factor
for (i in (1:ncol(datos)))
  datos[,i]=factor(datos[,i])

# Resumen
str(datos)
summary(datos)

# Representación de las variables
colores=c("darkseagreen1","deeppink2", "darkturquoise",
  "darkorchid1","darksalmon", "mediumorchid1","mistyrose",
  "green3"," orangered")
par(mar=c(1,1,1,1)
par(mfrow=c(3,3))
for (i in 1:(ncol(datos)))
  { pie(table(datos[,i]), main=colnames(datos)[i],col=colores)
}
```

```
# Ordenamos los valores de la variable Edad
levels(datos$Edad)
datos$Edad=factor(datos$Edad, labels = levels(datos$Edad) [ c(
1,4,7,3,5,6,8,9,10,2)])
levels(datos$Edad)

# 2) APRENDIZAJE DE LA RED
# Método IAMB
red_iamb=iamb(datos)
red_iamb

# Método de escalada
red_hc=hc(datos)
red_hc

# Gráficos de las redes
arcos=c("Hosp","Neum")
par(mar=c(1,1,1,1))
par(mfrow=c(1,2))
graphviz.plot(red_iamb, highlight = list( arcs =arcos,
col="maroon2",lwd = 4),
layout = "fdp", main = " RED APRENDIDA IAMB ")
# graphviz.plot(red_iamb, layout = "fdp", main = " RED
APRENDIDA IAMB ")
graphviz.plot(red_hc,layout = "fdp", main = " RED APRENDIDA HC
")

# 3) ESTIMACIÓN DE PARÁMETROS RED_HC
bn.fit(red_hc,datos,method="mle")
bn.fit(red_hc,datos,method="bayes")

# Estimación P(Sex=Mujer|Neumonia=Sí, Def=Sí) MV
M=nrow(subset(datos,datos$Sex=="Mujer"& datos$Neum=="Si"&
   datos$Def=="Si"))
N=nrow(subset(datos,datos$Def=="Si"& datos$Neum=="Si"))
thetaMV=M/N
thetaMV
errorMV=abs(thetaMV-0.3563218)
errorMV

# Estimación P(Sex=Mujer|Neumonia=Sí, Def=Sí) Bayes
thetaB=(M+1)/(N+2)
thetaB
```

```
errorB=abs(thetaB-0.3564593)
errorB

# 4) ESTIMACIÓN DE LOS EJES DE RED_IAMB
score(set.arc(red_iamb, from = "Neum", to = "Hosp"), datos)
score(set.arc(red_iamb, from = "Hosp", to = "Neum"), datos)
fuerza=boot.strength(datos, R = 100, algorithm=" iamb")
subset(fuerza,fuerza$from=="Neum"& fuerza$to=="Hosp")
subset(fuerza,fuerza$from=="Hosp"& fuerza$to=="Neum")
red_iamb=set.arc(red_iamb, from = "Hosp", to = "Neum")
red_iamb

# Gráfico de las redes
par(mar=c(1,1,1,1))
par(mfrow=c(1,2))
graphviz.plot(red_iamb, layout = "fdp", main = " RED IAMB ")
graphviz.plot(red_hc,layout = "fdp", main = " RED HC ")

# 5) ESTIMACIÓN DE PARÁMETROS RED_IAMB
bn.fit(red_iamb,datos,method="mle")
bn.fit(red_iamb,datos,method="bayes")

# 6) VALIDACIÓN CRUZADA bn.cv(datos,red_iamb)
bn.cv(datos,red_iamb)
```

Bibliografía

[1] Bertsekas, D.P., Tsitsiklis, J.N. (2002) *Introduction to probability*, Athena Scientific, Belmont

[2] Friedman, J., Peer, D., Nachman, I. (1999), *Learning Bayesian network structure from massive datset: the "Sparce Candidate" algorithm*, In: Proceedings of 15th conference on uncertainty in artificial intelligence (UAI), Morgan Kaufmann, 206-215

[3] Jensen, F.V., Nielsen, T.D. (2007) *Bayesian Networks and Decision Graphs*, Springer, Berlin

[4] Margaritis, D. (2003), *Learning Bayesian network model structure from data*, PhD thesis, School of Computer Science, CArnegie-Mellon University, Pittsburgh, PA, available as Technical Report CMU-CS-03-153

[5] Nagarajan, R., Scurari, M., Lèbre, S. (2013), *Bayesian Networks in R with Application in Systems Biology*, Springer

[6] Rohatgi, V.K., Saleh, A.K.M.E., (2015) *An Introduction to Probability and Statistics: Third Edition*, John Wiley & Sons Inc Hoboken, New Jersey

[7] Scutari, M. (2017), *Dirichlet Bayesian Network Scores and the Maximum Relative Entropy Principle*. Journal of Machine Learning Research (73, Proceedings Track, AMBN 2017), 8-20 (arXiv:1708.00689)

[8] Spirtes, P., Glymour, C., Scheines, R. (2001), *Causation, prediction, and search, 2nd edn*, MIT Press, Cambridge

[9] Tsamardinos, I., Aliferis, CF., Statnikov A. (2003), *Algorithms for large scale MArkov Blanket discovery*, In: PToceedings of the 16th international Florida artificial intelligence research society conference, AAAI Press, 376-381

[10] Tsamardinos, I., Brown, L.E., Aliferis, C.F. (2006), *The max-min hill-cliimbing Bayesian network structure learning algorithm*, Machine Learn 65(1):31-78