Algoritmos de Estimación de Distribuciones = Computación Evolutiva + Modelos Gráficos Probabilísticos

Pedro Larrañaga

Departamento de Ciencias de la Computación e Inteligencia Artificial

Universidad del País Vasco

http://www.sc.ehu.es/isg/

Dpto. Lenguajes y Ciencias de la Computación · Universidad de Málaga · 30 Septiembre 2002

Esquema del tutorial

- 1. Introducción
- 2. Modelos Gráficos Probabilísticos
- 3. Algoritmos de Estimación de Distribuciones
- 4. Aplicaciones de los Algoritmos de Estimación de Distribuciones
- 5. Conclusiones
- Referencias

1 Introducción i

Motivación

- Algoritmos Evolutivos
  - varios parámetros a determinar
  - dificultad en la predicción de las poblaciones a través de las generaciones
  - *building blocks*
  - relación entre las variables (*linkage learning*)
  - problemas engañosos

1 Introducción ii

Nueva aproximación a la Computación Evolutiva

- Basada en poblaciones
- Sin operadores de cruce ni mutación
- En cada generación se estima de los individuos seleccionados, la distribución de probabilidad subyacente a los mismos
- Muestreando esta distribución se obtiene la siguiente población
- Se repiten los dos pasos anteriores hasta el criterio de terminación

EDA (Estimation of Distribution Algorithms) Mühlenbein y Paaβ, 1996

2. Modelos Gráficos Probabilísticos

- 2.1 Introducción
- 2.2 Redes Bayesianas
  - 2.2.1 Introducción
  - 2.2.2 Independencia condicional
  - 2.2.3 Factorización de la probabilidad conjunta
  - 2.2.4 Propagación de la evidencia
  - 2.2.5 Simulación
  - 2.2.6 Aprendizaje estructural
- 2.3 Redes Gausssianas
  - 2.3.1 Normal multidimensional y redes Gaussianas
  - 2.3.2 Simulación
  - 2.3.3 Aprendizaje estructural

2.1 Introducción i

- Limitación de los sistemas basados en reglas
- Intuición humana, cierto paralelismo con la probabilidad
- A finales de la década de los 80 se superaron las dificultades computacionales que limitaban los primitivos sistemas probabilísticos
- Superación de las asunciones simplificadas con las que se construian los sistemas expertos probabilísticos en los origenes de la IA
- Concepto de independencia condicional
- Los modelos gráficos probabilísticos permiten construir sistemas con características de razonamiento –humano versus animal (visión, voz, movimiento, ...)–

2.1 Introducción ii

- Redes Bayesianas (Pearl, 1988)
- Redes Gaussianas (Shachter y Kenley, 1989)
- Redes de Markov
- Modelos ocultos de Markov
- Modelos log-lineales
- Grafos cadena
- ...

2.1 Introducción iii

Información en Internet

- http://bayes.stat.washington.edu/almond/belief.html
- http://http.cs.berkeley.edu/~murphyk/Bayes/bayes.html
- http://www.afit.af.mil/Schools/EN/AI
- http://www.auai.org/
- http://www.cs.auc.dk/research/DSS/
- http://www.maths.nott.ac.uk/hsss/
- http://www.research.microsoft.com/research/dtg/

2.1 Introducción iv

Software libre en Internet

- http://www.ia.uned.es/~elvira
- http://hss.cmu.edu/html/departments/philosophy/TETRAD
- http://http.cs.Berkeley.edu/~murphyk/Bayes/bnsoft.html
- http://kmi.open.ac.uk/projects/bkd
- http://www.city.ac.uk/~rgc
- http://www.cs.cmu.edu/~javabayes/Home/
- http://www.mrc-bsu.cam.ac.uk/bugs/Welcome.html
- http://www2.sis.pitt.edu/~genie

2.1 Introducción v

Software comercial en Internet

- HUGIN http://www.hugin.dk/
- DXPRESS http://www.kic.com/
- NETICA http://www.norsys.com/netica.html

2.2.1 Introducción i

- Variables discretas, multinomiales: redes Bayesianas
- Variables continuas, normales: redes Gaussianas
- Representación utilizada para codificar incertidumbre en sistemas expertos
- Dependencias entre las variables
- En un lugar central entre los siguientes dos casos extremos
  - $p(\boldsymbol{x}) = p(x_1, x_2, \ldots, x_n)$ necesita $2^n - 1$ parámetros
  - $p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i)$ necesita n parámetros

2.2.1 Introducción ii

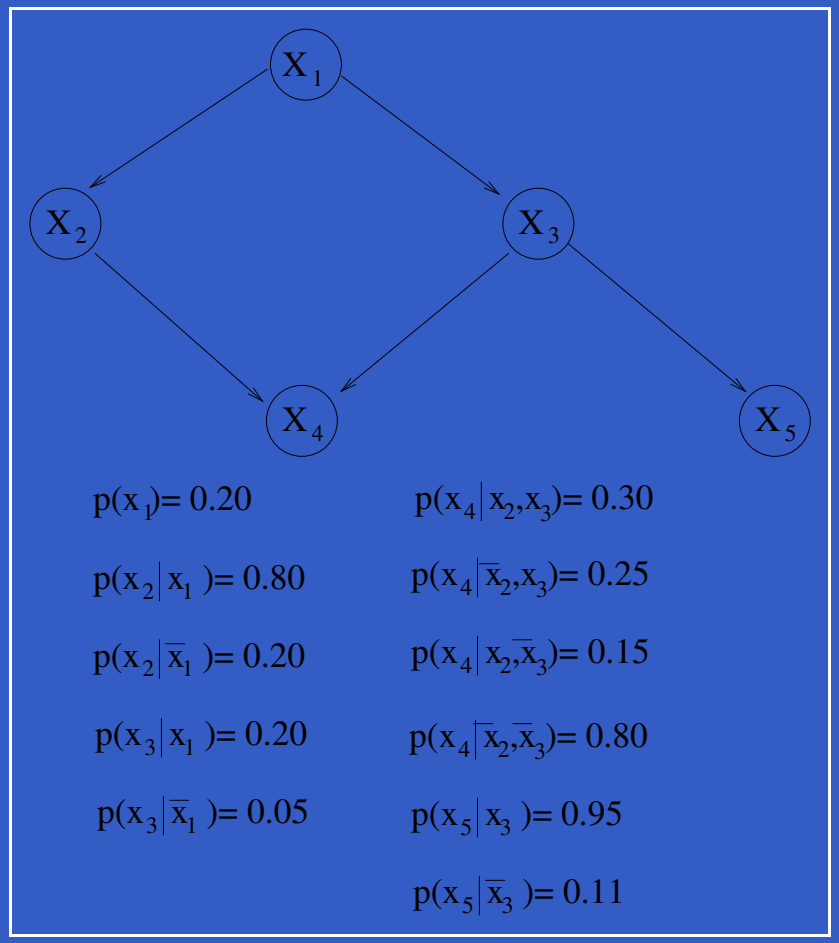
- Grafos acíclicos dirigidos (DAGs)
- Nodos: variables
- Arcos: dependencias condicionales
- Factorización de la distribución de probabilidad $(X_1, ..., X_n)$:

$$\rho(\boldsymbol{x}) = \rho(x_1, ..., x_n) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}(x_i))$$

x_i representa el valor de la variable aleatoria X_i

$\boldsymbol{pa}(x_i)$ representa el valor de las variables aleatorias padres –variables de las cuales parte un arco– de X_i

2.2.1 Introducción iii



$$p(x_1, x_2, x_3, x_4, x_5) = p(x_1) \cdot p(x_2 \mid x_1) \cdot p(x_3 \mid x_1) \cdot p(x_4 \mid x_2, x_3) \cdot p(x_5 \mid x_3)$$

2.2.1 Introducción iv

$\boldsymbol{X} = (X_1, \ldots, X_n)$ un conjunto de variables aleatorias

- $\boldsymbol{X} = (X_1, \ldots, X_n)$ un conjunto de variables aleatorias
- x_i valor de X_i (la i-ésima componente de $\boldsymbol{X}$)
- $\boldsymbol{y} = (x_i)_{X_i \in \boldsymbol{Y}}$ valor de $\boldsymbol{Y} \subseteq \boldsymbol{X}$
- El modelo gráfico probabilístico para $\boldsymbol{X}$ proporciona una factorización de la distribución de probabilidad conjunta $\rho(\boldsymbol{x})$
  - $p(\boldsymbol{x})$ caso discreto
  - $f(\boldsymbol{x})$ caso continuo
- Dos componentes
  - estructura S (DAG)
  - un conjunto de densidades locales
- S representa un conjunto de independencias condicionales sobre tripletas de variables de $\boldsymbol{X}$

2.2.2 Independencia condicional i

Distribución de probabilidad conjunta de (X, Y, Z) : $p(x, y, z)$

x	y	z	$p(x, y, z)$
0	0	0	0.12
0	0	1	0.18
0	1	0	0.04
0	1	1	0.16
1	0	0	0.09
1	0	1	0.21
1	1	0	0.02
1	1	1	0.18

2.2.2 Independencia condicional ii

Distribuciones marginales: $p(x)$, $p(y)$, $p(z)$

$$p(x) = \sum_{y,z} p(x, y, z)$$
$$p(y) = \sum_{x,z} p(x, y, z)$$
$$p(z) = \sum_{x,y} p(x, y, z)$$

x	$p(x)$
0	0.5
1	0.5

y	$p(y)$
0	0.6
1	0.4

z	$p(z)$
0	0.27
1	0.73

2.2.3 Independencia condicional iii

Marginales de orden 2: $p(x,y)$, $p(x,z)$, $p(y,z)$

$$p(x,y) = \sum_z p(x,y,z)$$
$$p(x,z) = \sum_y p(x,y,z)$$
$$p(y,z) = \sum_x p(x,y,z)$$

x	y	$p(x,y)$
0	0	0.3
0	1	0.2
1	0	0.3
1	1	0.2

x	z	$p(x,z)$
0	0	0.16
0	1	0.34
1	0	0.11
1	1	0.39

y	z	$p(y,z)$
0	0	0.21
0	1	0.39
1	0	0.06
1	1	0.34

2.2.2 Independencia condicional iv

Probabilidades condicionadas:

$$p(x|y) = \frac{p(x,y)}{p(y)} \quad p(x|z) = \frac{p(x,z)}{p(z)} \quad p(y|z) = \frac{p(y,z)}{p(z)}$$

y	x	$p(x\|y)$
0	0	0.5
0	1	0.5
1	0	0.5
1	1	0.5

z	x	$p(x\|z)$
0	0	16/27
0	1	11/27
1	0	34/73
1	1	39/73

z	y	$p(y\|z)$
0	0	21/27
0	1	6/27
1	0	39/73
1	1	34/73

2.2.2 Independencia condicional v

- X e Y independientes sii:
  - $p(x|y) = p(x) \ \ \forall\, x, y$ ó
  - $p(x, y) = p(x) \cdot p(y) \ \ \forall\, x, y$

  En el ejemplo X e Y son independientes pero X y Z son dependientes
- X condicionalmente independiente de Y dado Z, $I(X, Y|Z)$, sii:
  - $p(x|y, z) = p(x|z) \ \ \forall\, x, y, z$ ó
  - $p(x, y|z) = p(x|z) \cdot p(y|z) \ \ \forall\, x, y, z$

  En el ejemplo, existen x, y, z tal que $p(x|y, z) \neq p(x|z)$, por tanto X no es condicionalmente independiente de Y dado Z

2.2.2 Independencia condicional vi

$$p(x|y,z) = \frac{p(x,y,z)}{p(y,z)}$$

y	z	x	$p(x\|y,z)$
0	0	0	12/21
0	0	1	9/21
0	1	0	18/39
0	1	1	21/39
1	0	0	4/6
1	0	1	2/6
1	1	0	16/34
1	1	1	18/34

2.2.2 Independencia condicional vii

$$p(x|z) = \frac{p(x,z)}{p(z)}$$

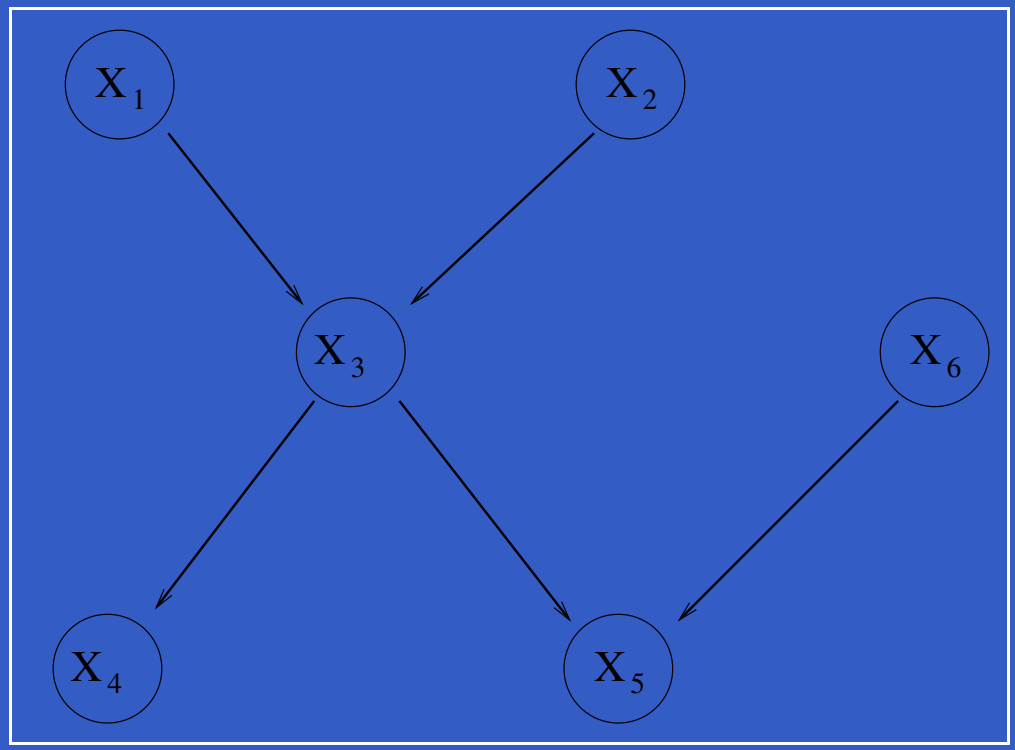
z	x	$p(x\|z)$
0	0	16/27
0	1	11/27
1	0	34/73
1	1	39/73

2.2.2 Independencia condicional viii

¿$I(X, Y|Z)$?

- Basado en el criterio de separación
  - (i) obtener el menor grafo conteniendo X, Y y Z y sus ancestros
  - (ii) moralizar el subgrafo obtenido:
    - añadir una arista entre padres con hijos comunes
    - transformar los arcos en aristas
  - (iii) en el grafo no dirigido obtenido $I(X, Y|Z)$ sii Z bloquea todo camino entre X e Y

2.2.2 Independencia condicional ix



¿$I(X_1, X_5|X_3)$?; ¿$I(X_1, X_6|X_5)$?;

¿$I(X_4, X_5|\{X_1, X_3\})$?; ¿$I(X_4, \{X_5, X_6\}|X_1)$?

2.2.3 Factorización de la probabilidad conjunta i

Factorización de la función de densidad generalizada:

- $\boldsymbol{Pa}_i$ conjunto de padres de X_i en S

$$\rho(\boldsymbol{x}) = \rho(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \rho(x_i | \boldsymbol{pa}_i^S)$$

- Suponiendo que la distribución de probabilidad depende de un conjunto finito de parámetros

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

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

- Modelo gráfico probabilístico $M = (S, \boldsymbol{\theta}_S)$

2.2.3 Factorización de la probabilidad conjunta ii

i) Árbol $$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|x_{j(i)}, \boldsymbol{\theta}_i)$$

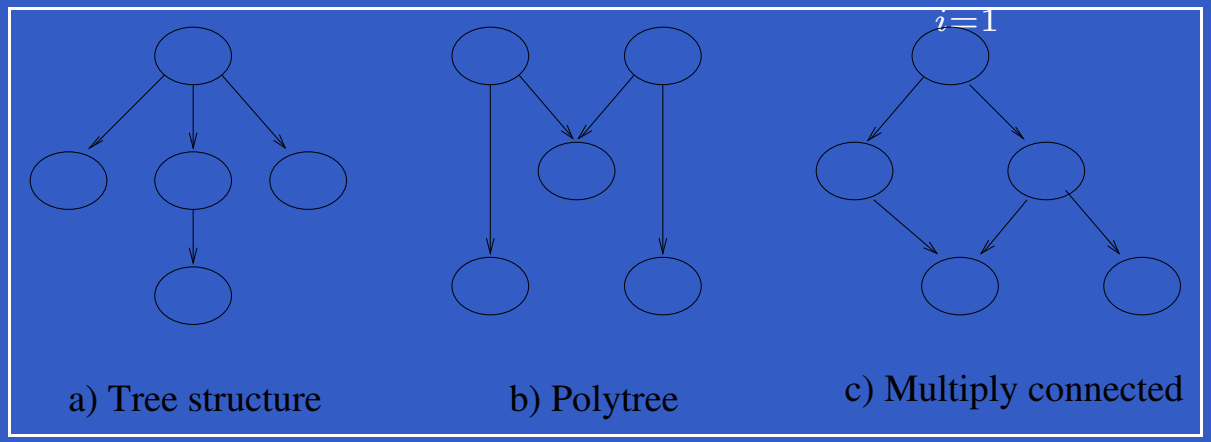
$X_{j(i)}$ es el (posiblemente vacío) padre de X_i

ii) Poliárbol $$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|x_{j1(i)}, \ldots, x_{jr(i)}, \boldsymbol{\theta}_i)$$

$\{X_{j1(i)}, \ldots, X_{jr(i)}\}$ es el (posiblemente vacío) conjunto de padres de X_i, los cuales son mútuamente independientes:

$$\rho(x_{j1(i)}, x_{j2(i)}, \ldots, x_{jr(i)}) = \prod_{k=1}^{r} \rho(x_{jk(i)}) \quad i = 1, \ldots, n$$

iii) Múltiplemente conectados $$\rho(\boldsymbol{x}|\boldsymbol{\theta}_S) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}_i^S, \boldsymbol{\theta}_i)$$



2.2.3 Factorización de la probabilidad conjunta iii

Notación

- Para todo i, X_i es discreta
- $x_i^1, \ldots, x_i^{r_i}$ denotan los r_i posibles valores de X_i
- $p(x_i^k | \boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i) = \theta_{ijk}$ es la probabilidad condicional de que X_i esté en su k-ésimo valor, dado que el conjunto de sus padres está en su j-ésimo valor
- $q_i = \prod_{X_g \in \boldsymbol{Pa}_i} r_g$ denota el número de posibles instanciaciones distintas de $\boldsymbol{Pa}_i$
- Parámetros locales $\boldsymbol{\theta}_i = (((\theta_{ijk})_{k=1}^{r_i})_{j=1}^{q_i})$

2.2.3 Factorización de la probabilidad conjunta iv

Componentes

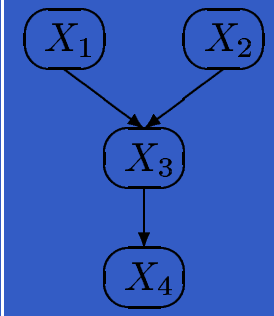
a) La estructura S (DAG) refleja el conjunto de independencias condicionales entre las variables

b) Distribuciones a priori θ_{i-k} para todos los nodos raíces $p(x_i^k|\emptyset, \boldsymbol{\theta}_i)$

c) Probabilidades condicionadas, θ_{ijk}, para todos los nodos dadas todas las posibles combinaciones de los padres $p(x_i^k|\boldsymbol{pa}_i^{j,S}, \boldsymbol{\theta}_i)$

2.2.3 Factorización de la probabilidad conjunta v

Structure



Local probabilities

$$\boldsymbol{\theta_1} = (\theta_{1-1}, \theta_{1-2}) \qquad p(x_1^1), p(x_1^2)$$

$$\boldsymbol{\theta_2} = (\theta_{2-1}, \theta_{2-2}, \theta_{2-3}) \qquad p(x_2^1), p(x_2^2), p(x_2^3)$$

$$\begin{aligned}\boldsymbol{\theta_3} = (&\theta_{311}, \theta_{321}, \theta_{331}, \\ &\theta_{341}, \theta_{351}, \theta_{361}, \\ &\theta_{312}, \theta_{322}, \theta_{332}, \\ &\theta_{342}, \theta_{352}, \theta_{362})\end{aligned}$$

$$\begin{aligned}&p(x_3^1|x_1^1, x_2^1), p(x_3^1|x_1^1, x_2^2), p(x_3^1|x_1^1, x_2^3), \\ &p(x_3^1|x_1^2, x_2^1), p(x_3^1|x_1^2, x_2^2), p(x_3^1|x_1^2, x_2^3), \\ &p(x_3^2|x_1^1, x_2^1), p(x_3^2|x_1^1, x_2^2), p(x_3^2|x_1^1, x_2^3), \\ &p(x_3^2|x_1^2, x_2^1), p(x_3^2|x_1^2, x_2^2), p(x_3^2|x_1^2, x_2^3)\end{aligned}$$

$$\boldsymbol{\theta_4} = (\theta_{411}, \theta_{421}, \theta_{412}, \theta_{422}) \qquad p(x_4^1|x_3^1), p(x_4^1|x_3^2), p(x_4^2|x_3^1), p(x_4^2|x_3^2)$$

Factorization of the joint mass-probability

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2)p(x_3|x_1, x_2)p(x_4|x_3)$$

Estructura, probabilidades locales y factorización resultante de una red Bayesiana con 4 variables (X_1, X_3 y X_4 con dos posibles valores, y X_2 con tres posibles valores)

2.2.4 Propagación de la evidencia i

- Métodos exactos (Lauritzen, Spiegelhalter 1988)
- Métodos aproximados
  - Muestreo lógico probabilístico (Henrion 1988)

2.2.4 Propagación de la evidencia ii

Métodos exactos (Lauritzen, Spiegelhalter 1988)

- Moralización del DAG
- Triangulación del grafo moral
- Creación del grafo de juntura
- Creación del árbol de juntura
- Modificación del algoritmo de propagación de la evidencia para árboles al árbol de juntura

El único paso problemático en el proceso de DAG a árbol de juntura es la triangulación

2.2.5 Simulación i

Simulación Probabilistic Logic Sampling (PLS) Henrion, 1988

Dado un orden ancestral, π, en los nodos

Para $j = 1, 2, \ldots, N$

Para $i = 1, 2, \ldots, n$

$x_{\pi(i)} \leftarrow$ generar un valor de $p(x_{\pi(i)}|\boldsymbol{pa}_{\pi(i)})$

- Un orden ancestral es aquel en el que las variables $\boldsymbol{Pa}_i$ son anteriores a la variable X_i
- Los casos son generados variable a variable
- Una variable es muestreada una vez que lo han sido sus variables padres

2.2.6 Aprendizaje estructural i

- Aprendizaje estructural $\equiv$ inducción del modelo $\equiv$ búsqueda del modelo
- Dos aproximaciones
  - detección de independencias condicionales
  - score + búsqueda

2.2.5 Aprendizaje estructural ii

Detección de independencias condicionales:

- A partir de tests de independencia condicional entre tripletas de variables obtener una lista de independencias y dependencias condicionales
- Obtener la estructura de red Bayesiana que mejor refleje las independencias y dependencias condicionales anteriores

2.2.5 Aprendizaje estructural iii

score:

- Máxima verosimilitud penalizada: $\log p(D \mid S, \boldsymbol{\theta}) - pe(N)dim(S)$

$$\sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - pe(N)dim(S)$$

  - N_{ijk} denota el número de casos en D en los cuales X_i toma el valor x_i^k y $\boldsymbol{Pa}_i$ toma su j-ésimo valor; $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$
  - $dim(S) = \sum_{i=1}^{n} q_i(r_i - 1)$
  - $pe(N) = \begin{cases} 1 & \text{AIC, Akaike, 1974} \\ \frac{1}{2}\log N & \text{BIC, Schwarz, 1978} \end{cases}$

- Verosimilitud marginal: $p(D|S, \boldsymbol{\theta})$

$$\prod_{i=1}^{n}\prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}$$

2.2.5 Aprendizaje estructural iv

búsqueda:

- Búsqueda voraz
- Heurísticos: Enfriamiento Estadístico, Algoritmos Genéticos, Búsqueda Tabu, Colonias de Hormigas, ...

2.2.5 Aprendizaje estructural v

Algoritmo K2. Cooper y Herskovits (1992)

- Las variables de la base de datos son discretas
- Dado un modelo de Red Bayesiana los casos ocurren independientemente
- No hay casos que tengan variables con datos perdidos
- $f(B_P|B_S)$ tiene una distribucion uniforme

2.2.5 Aprendizaje estructural vi

Algoritmo K2. Cooper y Herskovits (1992)

- Un conjunto de n variables discretas X_i $(i = 1, \ldots, n)$, con r_i posibles valores
- D la base de datos con N casos
- B_S una estructura de Red Bayesiana
- $\boldsymbol{Pa}_i$ a los padres de la variable X_i
- q_i el número de instanciaciones diferentes de $\boldsymbol{Pa}_i$
- N_{ijk} el número de casos en D para los cuales X_i toma su k-ésimo valor y $\boldsymbol{Pa}_i$ está en su j-ésima instanciación
- $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$

$$P(B_S, D) = P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} (N_{ijk})! = P(B_S) \prod_{i=1}^{n} g(i, \boldsymbol{Pa}_i).$$

2.2.5 Aprendizaje estructural vii

INPUT: Un conjunto de n nodos, un orden entre los nodos, una cota superior u
del numero de padres que puede tener un nodo, y una base de datos D conteniendo m casos
OUTPUT: Para cada nodo, el conjunto de sus nodos padres
BEGIN *K2*
FOR $i := 1$ TO n DO
BEGIN
$\boldsymbol{Pa}_i := \emptyset$;
$P_{\text{old}} := g(i, \boldsymbol{Pa}_i)$:
OKtoProceed := TRUE
WHILE OKtoProceed AND $|\boldsymbol{Pa}_i| < u$ DO
BEGIN
Sea Z el nodo en Pred$(X_i) - \boldsymbol{Pa}_i$ que maximiza $g(i, \boldsymbol{Pa}_i \cup \{Z\})$;
$P_{\text{new}} := g(i, \boldsymbol{Pa}_i \cup \{Z\})$;
IF $P_{\text{new}} > P_{\text{old}}$ THEN
BEGIN
$P_{\text{old}} := P_{\text{new}}$;
$\boldsymbol{Pa}_i := \boldsymbol{Pa}_i \cup \{Z\}$
END
ELSE OKtoProceed := FALSE;
END;
WRITE('Nodo:', X_i, 'Padres de este nodo:', $\boldsymbol{Pa}_i$)
END;
END *K2*

2.2.5 Aprendizaje estructural viii

- Se necesita un *orden* entre las variables
- Hace falta una *cota superior* del número de padres para cada variables
- Todas las estructuras son igualmente probables al inicio
- Para cada nodo K2 busca el conjunto de padres que maximiza:

$$g(i, \boldsymbol{Pa}_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!$$

- K2 comienza asumiendo que un nodo no tiene padres
- En cada paso K2 añade de manera incremental aquel nodo padre cuya inclusión mas incrementa $g(i, \boldsymbol{Pa}_i)$
- K2 para cuando la inclusión de un padre simple no incrementa $g(i, \boldsymbol{Pa}_i)$
- K2 es un algoritmo *greedy*

2.3.1 Normal multidimensional y redes Gaussianas i

- Es un modelo gráfico probabilístico donde cada variable $X_i \in \boldsymbol{X}$ es continua
- Las funciones de densidad locales coinciden con el modelo de regresión lineal:

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

- $\mathcal{N}(x; \mu, \sigma^2)$ distribución normal univariada con media μ y varianza σ^2
- b_{ji} intensidad de la relación entre X_j y X_i ($b_{ji} = 0$ sii no existe el arco de X_j a X_i)
- v_i es la varianza de X_i condicionada a $\boldsymbol{Pa}_i$
- $\boldsymbol{\theta}_i = (m_i, \boldsymbol{b}_i, v_i)$ parámetros locales, $\boldsymbol{b}_i = (b_{1i}, \ldots, b_{i-1i})^t$

2.3.1 Normal multidimensional y redes Gaussianas ii

De distribuciones normales multivariantes a redes Gaussianas

$$f(\boldsymbol{x}) \equiv \boldsymbol{\mathcal{N}}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \equiv (2\pi)^{-\frac{n}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

- $\boldsymbol{\mu}$ vector de esperanzas matemáticas
- $\boldsymbol{\Sigma}$ matriz de covarianzas $n \times n$
- $\boldsymbol{W} = \boldsymbol{\Sigma}^{-1}$ matriz de precisión
- $f(\boldsymbol{x}) = \prod_{i=1}^{n} f(x_i|x_1, \ldots, x_{i-1}) = \prod_{i=1}^{n} \mathcal{N}(x_i; \mu_i + \sum_{x_j \in \boldsymbol{pa}_i} a_{ji}(x_j - \mu_j), {\sigma_i}^2)$
  interpretación de distribución normal multivariante como una red Gaussiana:
  $m_i \equiv \mu_i, b_{ji} \equiv a_{ji}, v_i \equiv {\sigma_i}^2$

2.3.1 Normal multidimensional y redes Gaussianas iii

De una red Gaussiana a una distribución normal multivariante

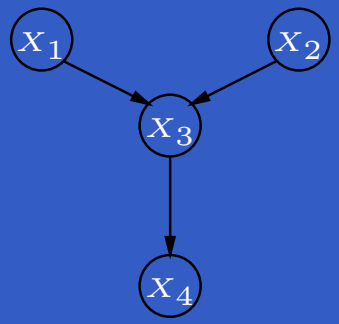
- $\mu_i = m_i$
- $\boldsymbol{W}$ de v_i y $\boldsymbol{b}_i$ por medio de la siguiente fórmula recursiva:

$$W(1) = \frac{1}{v_1}$$

$$\boldsymbol{W}(i+1) = \begin{pmatrix} \boldsymbol{W}(i) + \frac{\boldsymbol{b}_{i+1}\boldsymbol{b}_{i+1}^t}{v_{i+1}}, & -\frac{\boldsymbol{b}_{i+1}}{v_{i+1}} \\ -\frac{\boldsymbol{b}_{i+1}^t}{v_{i+1}}, & \frac{1}{v_{i+1}} \end{pmatrix} \quad i = 1, \ldots, n-1$$

2.3.1 Normal multidimensional y redes Gaussianas iv

Structure



Local densities

$$\boldsymbol{\theta_1} = (m_1, -, v_1) \qquad f_{X_1} \hookrightarrow \mathcal{N}(x_1; m_1, v_1)$$

$$\boldsymbol{\theta_2} = (m_2, -, v_2) \qquad f_{X_2} \hookrightarrow \mathcal{N}(x_2; m_2, v_2)$$

$$\boldsymbol{\theta_3} = (m_3, \boldsymbol{b_3}, v_3) \qquad f_{X_3|X_1=x_1, X_2=x_2} \hookrightarrow$$
$$\boldsymbol{b_3} = (b_{13}, b_{23})' \qquad \mathcal{N}(x_3; m_3 + b_{13}(x_1 - m_1) + b_{23}(x_2 - m_2), v_3)$$

$$\boldsymbol{\theta_4} = (m_4, \boldsymbol{b_4}, v_4) \qquad f_{X_4|X_3=x_3} \hookrightarrow \mathcal{N}(x_4; m_4 + b_{34}(x_3 - m_3), v_4)$$
$$\boldsymbol{b_4} = (b_{34})'$$

Factorization of the joint density function

$$f(x_1, x_2, x_3, x_4) = f(x_1)f(x_2)f(x_3|x_1, x_2)f(x_4|x_3) =$$
$$\frac{1}{\sqrt{2\pi v_1}} e^{-\frac{1}{2v_1}(x_1 - m_1)^2} \cdot \frac{1}{\sqrt{2\pi v_2}} e^{-\frac{1}{2v_2}(x_2 - m_2)^2} \cdot$$
$$\frac{1}{\sqrt{2\pi v_3}} e^{-\frac{1}{2v_3}(x_3 - (m_3 + b_{13}(x_1 - m_1) + b_{23}(x_2 - m_2)))^2} \cdot$$
$$\frac{1}{\sqrt{2\pi v_4}} e^{-\frac{1}{2v_4}(x_4 - (m_4 + b_{34}(x_3 - m_3)))^2}$$

Estructura, densidades locales y factorización resultante de una red Gaussiana con cuatro variables

2.3.2 Simulación

Dado un orden ancestral, π, de los nodos

Para $j = 1, 2, \ldots, N$

Para $i = 1, 2, \ldots, n$

$x_{\pi(i)} \leftarrow$ generar un valor de $f(x_{\pi(i)}|\boldsymbol{pa}_{\pi(i)})$

- Un orden ancestral es aquel en el que las variables $\boldsymbol{Pa}_i$ son anteriores a la variable X_i
- Los casos son generados variable a variable
- Una variable es muestreada una vez que lo han sido sus variables padres
- Para muestrear una distribución normal univariada:
  - suma de 12 variables uniformes independientes, Box y Muller, 1958
  - ratio de uniformes, Ripley, 1983

2.3.3 Aprendizaje estructural i

score:

- Máxima verosimilitud penalizada: $\log f(D \mid S, \boldsymbol{\theta}) - pe(N)dim(S)$

$$\sum_{r=1}^{N}\sum_{i=1}^{n}\left[-\log(\sqrt{2\pi v_i}) - \frac{1}{2v_i}\left(x_{ir} - m_i - \sum_{x_j \in \boldsymbol{pa}_i} b_{ji}(x_{jr} - m_j)\right)^2\right] +$$

$$-pe(N)\cdot(2n + \sum_{i=1}^{n}|\boldsymbol{Pa}_i|)$$

  - x_{ir} es la i-ésima componente del r-ésimo elemento de la muestra
  - $pe(N) = \begin{cases} 1 & \text{AIC, Akaike, 1974} \\ \frac{1}{2}\log N & \text{BIC, Schwarz, 1978} \end{cases}$

2.3.3 Aprendizaje estructural ii

- Verosimilitud marginal: $f(D|S,\boldsymbol{\theta})$

$$f(D \mid S) = \prod_{i=1}^{n} \frac{f(D^{X_i \boldsymbol{Pa}_i} \mid S_c)}{f(D^{\boldsymbol{Pa}_i} \mid S_c)}$$

donde

$$f(D \mid S_c) = (2\pi)^{-\frac{nN}{2}} \left(\frac{\nu}{\nu+N}\right)^{\frac{n}{2}} \frac{c(n,\alpha)}{c(n,\alpha+N)} |T_0|^{\frac{\alpha}{2}} |T_N|^{-\frac{\alpha+N}{2}}$$

búsqueda:

- Búsqueda voraz
- Heurísticos: Enfriamiento Estadístico, Algoritmo Genético, ...

3. Algoritmos de Estimación de Distribuciones

- 3.1 Introducción
- 3.2 Optimización combinatorial por medio del aprendizaje y la simulación de redes Bayesianas
  - 3.2.1 Sin dependencias
  - 3.2.2 Dependencias a pares
  - 3.2.3 Dependencias múltiples
- 3.3 Optimización en dominios continuos por medio del aprendizaje y la simulación de redes Gausssianas
  - 3.3.1 Sin dependencias
  - 3.3.2 Dependencias a pares
  - 3.3.3 Dependencias múltiples

3.1 Introducción i

Motivación

- Algoritmos Evolutivos
  - varios parámetros a determinar
  - dificultad en la predicción de las poblaciones a través de las generaciones
  - *building blocks*
  - relación entre las variables (*linkage learning*)
  - problemas engañosos

3.1 Introducción ii

Nueva aproximación a la Computación Evolutiva

- Basada en poblaciones
- Sin operadores de cruce ni mutación
- En cada generación se estima de los individuos seleccionados, la distribución de probabilidad subyacente a los mismos
- Muestreando esta distribución se obtiene la siguiente población
- Se repiten los dos pasos anteriores hasta el criterio de terminación

EDA (Estimation of Distribution Algorithms) Mühlenbein y Paaβ, 1996

3.1 Introducción iii

$$\text{máx } h(\boldsymbol{x}) = \sum_{i=1}^{6} x_i \text{ con } x_i = 0, 1$$

(a) D_0 $\qquad\qquad$ $p_0(X_i = 1) = 0{,}5$ para $i = 1, \ldots, 6$

3.1 Introducción iv

	X_1	X_2	X_3	X_4	X_5	X_6	$h(\boldsymbol{x})$
1	1	0	1	0	1	0	3
2	0	1	0	0	1	0	2
3	0	0	0	1	0	0	1
4	1	1	1	0	0	1	4
5	0	0	0	0	0	1	1
6	1	1	0	0	1	1	4
7	0	1	1	1	1	1	5
8	0	0	0	1	0	0	1
9	1	1	0	1	0	0	3
10	1	0	1	0	0	0	2
11	1	0	0	1	1	1	4
12	1	1	0	0	0	1	3
13	1	0	1	0	0	0	2
14	0	0	0	0	1	1	2
15	0	1	1	1	1	1	5
16	0	0	0	1	0	0	1
17	1	1	1	1	1	0	5
18	0	1	0	1	1	0	3
19	1	0	1	1	1	1	5
20	1	0	1	1	0	0	3

3.1 Introducción v

(b) $|D_0^{Se}| = 10$ truncación

	X_1	X_2	X_3	X_4	X_5	X_6
1	1	0	1	0	1	0
4	1	1	1	0	0	1
6	1	1	0	0	1	1
7	0	1	1	1	1	1
11	1	0	0	1	1	1
12	1	1	0	0	0	1
15	0	1	1	1	1	1
17	1	1	1	1	1	0
18	0	1	0	1	1	0
19	1	0	1	1	1	1

3.1 Introducción vi

(c)

$$p_1(\boldsymbol{x}) = p_1(x_1, \ldots, x_6) = \prod_{i=1}^{6} p(x_i | D_0^{Se})$$

modelo a aprender

$$\hat{p}(X_1 = 1 | D_0^{Se}) = 0{,}7$$

$$\hat{p}(X_2 = 1 | D_0^{Se}) = 0{,}7$$

$$\hat{p}(X_3 = 1 | D_0^{Se}) = 0{,}6$$

$$\hat{p}(X_4 = 1 | D_0^{Se}) = 0{,}6$$

$$\hat{p}(X_5 = 1 | D_0^{Se}) = 0{,}8$$

$$\hat{p}(X_6 = 1 | D_0^{Se}) = 0{,}7$$

3.1 Introducción vii

(d) muestreando $p_1(\boldsymbol{x})$

D_1

	X_1	X_2	X_3	X_4	X_5	X_6	$h(\boldsymbol{x})$
1	1	1	1	1	1	1	6
2	1	0	1	0	1	1	4
3	1	1	1	1	1	0	5
4	0	1	0	1	1	1	4
5	1	1	1	1	0	1	5
6	1	0	0	1	1	1	4
7	0	1	0	1	1	0	3
8	1	1	1	0	1	0	4
9	1	1	1	0	0	1	4
10	1	0	0	1	1	1	4
11	1	1	0	0	1	1	4
12	1	0	1	1	1	0	4
13	0	1	1	0	1	1	4
14	0	1	1	1	1	0	4
15	1	1	1	1	1	1	6
16	0	1	1	0	1	1	4
17	1	1	1	1	1	0	5
18	0	1	0	0	1	0	2
19	0	0	1	1	0	1	3
20	1	1	0	1	1	1	5

3.1 Introducción viii

(e) $|D_1^{Se}| = 10$ truncación

	X_1	X_2	X_3	X_4	X_5	X_6
1	1	1	1	1	1	1
2	1	0	1	0	1	1
3	1	1	1	1	1	0
5	1	1	1	1	0	1
6	1	0	0	1	1	1
8	1	1	1	0	1	0
9	1	1	1	0	0	1
15	1	1	1	1	1	1
17	1	1	1	1	1	0
20	1	1	0	1	1	1

3.1 Introducción ix

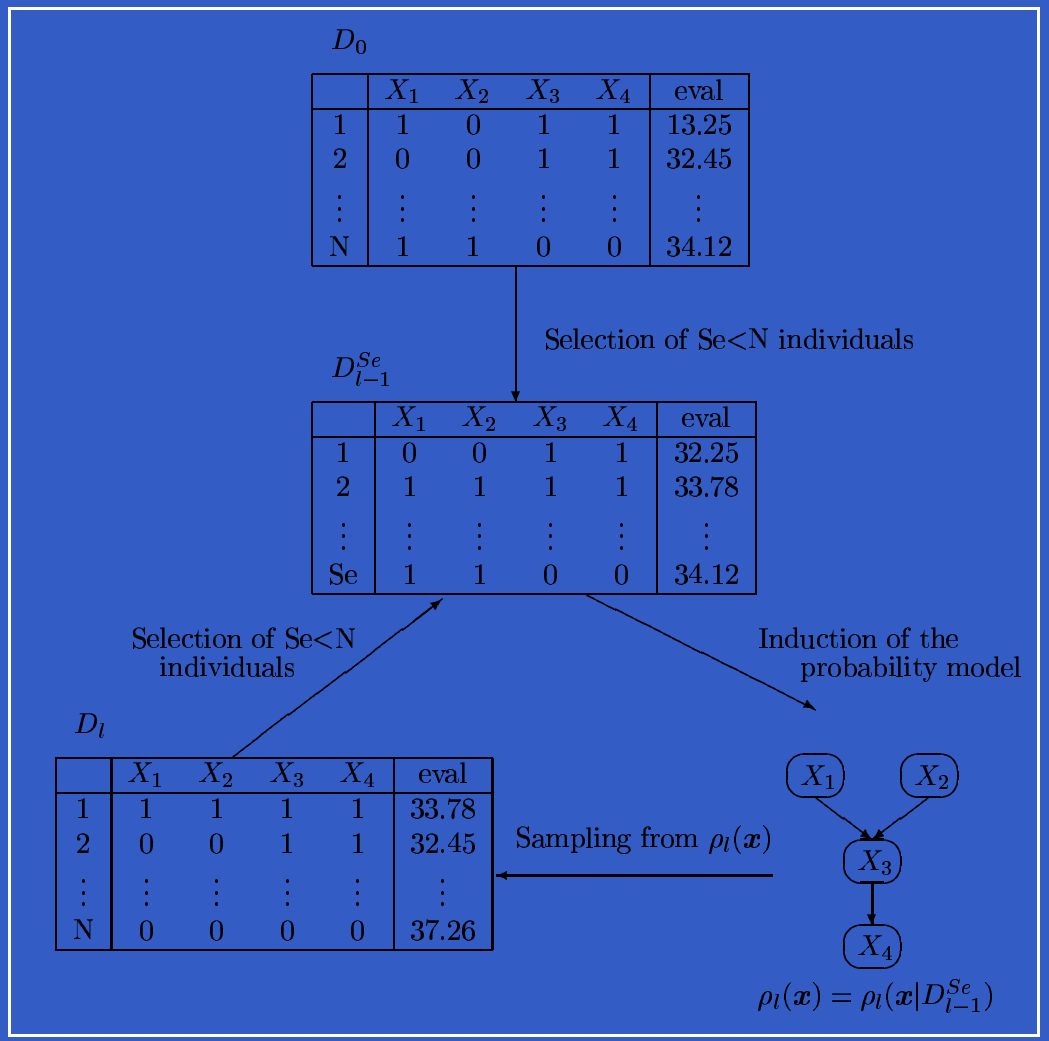
(f) repetir

- Seleccionar Se individuos de D_l obteniendo D_l^{Se}
- Aprender la distribución de probabilidad de los seleccionados

$$p_l(\boldsymbol{x}) = \prod_{i=1}^{6} p(x_i | D_{l-1}^{Se})$$

- Muestrear $p_l(\boldsymbol{x})$ obteniendo D_l

3.1 Introducción x



	X_1	X_2	X_3	X_4	eval
1	1	0	1	1	13.25
2	0	0	1	1	32.45
⋮	⋮	⋮	⋮	⋮	⋮
N	1	1	0	0	34.12

	X_1	X_2	X_3	X_4	eval
1	0	0	1	1	32.25
2	1	1	1	1	33.78
⋮	⋮	⋮	⋮	⋮	⋮
Se	1	1	0	0	34.12

	X_1	X_2	X_3	X_4	eval
1	1	1	1	1	33.78
2	0	0	1	1	32.45
⋮	⋮	⋮	⋮	⋮	⋮
N	0	0	0	0	37.26

3.1 Introducción xi

EDA

$D_0 \leftarrow$ Generar N individuos (la población inicial) al azar

Repetir para $l = 1, 2, \ldots$ hasta la condición de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $Se \leq N$ individuos de D_{l-1} siguiendo un método de selección

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) \leftarrow$ Estimar la distribución de probabilidad de los individuos seleccionados

$D_l \leftarrow$ Muestrear N individuos (la nueva población) de $p_l(\boldsymbol{x})$

3.2.1 Optimización combinatorial por medio del aprendizaje y la simulación de redes Bayesianas i

- Sin dependencias
- Dependencias bivariadas
- Redes Bayesianas
  - Ideas básicas
  - Aprendizaje
  - Simulación
- Dependencias múltiples

3.2 Optimización combinatorial por medio del aprendizaje y la simulación de redes Bayesianas ii

- Sin dependencias (UMDA, BSC, PBIL, cGA)
- Dependencias bivariadas (MIMIC, COMIT, BMDA)
- Dependencias múltiples
  - FDA, EcGA
  - .....Redes Bayesianas
  - EBNA, BOA, LFDA

3.2.1 Sin dependencias i

UMDA (Univariate Marginal Distribution Algorithm) Mühlenbein, 1998

UMDA

$D_0 \leftarrow$ Generar M individuos (la poblacion inicial) al azar

Repeat for $l = 1, 2, \ldots$ hasta que se verifique el criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} de acorde con un metodo de seleccion

$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D_{l-1}^{Se}) = \prod_{i=1}^{n} p_l(x_i) = \prod_{i=1}^{n} \frac{\sum_{j=1}^{N} \delta_j(X_i = x_i|D_{l-1}^{Se})}{N} \leftarrow$ Estimar la distribucion de probabilidad conjunta

$D_l \leftarrow$ Muestrear M individuos (la nueva poblacion) de $p_l(\boldsymbol{x})$

3.2.1 Sin dependencias ii

PBIL (Population Based Incremental Learning) Baluja, 1994

Obtener un vector de probabilidad inicial $p_0(\boldsymbol{x})$

while no convergencia **do**
begin

Usando $p_l(\boldsymbol{x})$ obtener M individuos: $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$

Evaluar y ordenar $\boldsymbol{x}_1^l, \ldots, \boldsymbol{x}_k^l, \ldots, \boldsymbol{x}_M^l$

Seleccionar los N ($N \leq M$) mejores individuos: $\boldsymbol{x}_{1:M}^l, \ldots, \boldsymbol{x}_{k:M}^l, \ldots, \boldsymbol{x}_{N:M}^l$

Actualizar el vector de probabilidades $p_{l+1}(\boldsymbol{x}) = (p_{l+1}(x_1), \ldots, p_{l+1}(x_n))$

for $i = 1, \ldots n$ **do**
$p_{l+1}(x_i) = (1-\alpha)p_l(x_i) + \alpha \frac{1}{N} \sum_{k=1}^{N} x_{i,k:M}^l$

end

3.2.1 Sin dependencias iii

cGA (compact Genetic Algorithm) Harik et al., 1998

Paso 1. Inicializar el vector de probabilidades $p_0(\boldsymbol{x})$
$p_0(\boldsymbol{x}) = p_0(x_1, \ldots, x_i, \ldots, x_n) = (p_0(x_1), \ldots, p_0(x_i), \ldots, p_0(x_n)) =$
$(0{,}5, \ldots, 0{,}5, \ldots, 0{,}5)$

Paso 2. $l = l + 1$. Muestrear $p_l(\boldsymbol{x})$ with $l = 0, 1, 2, \ldots$ obteniendo dos individuos: $\boldsymbol{x}_1^l, \boldsymbol{x}_2^l$

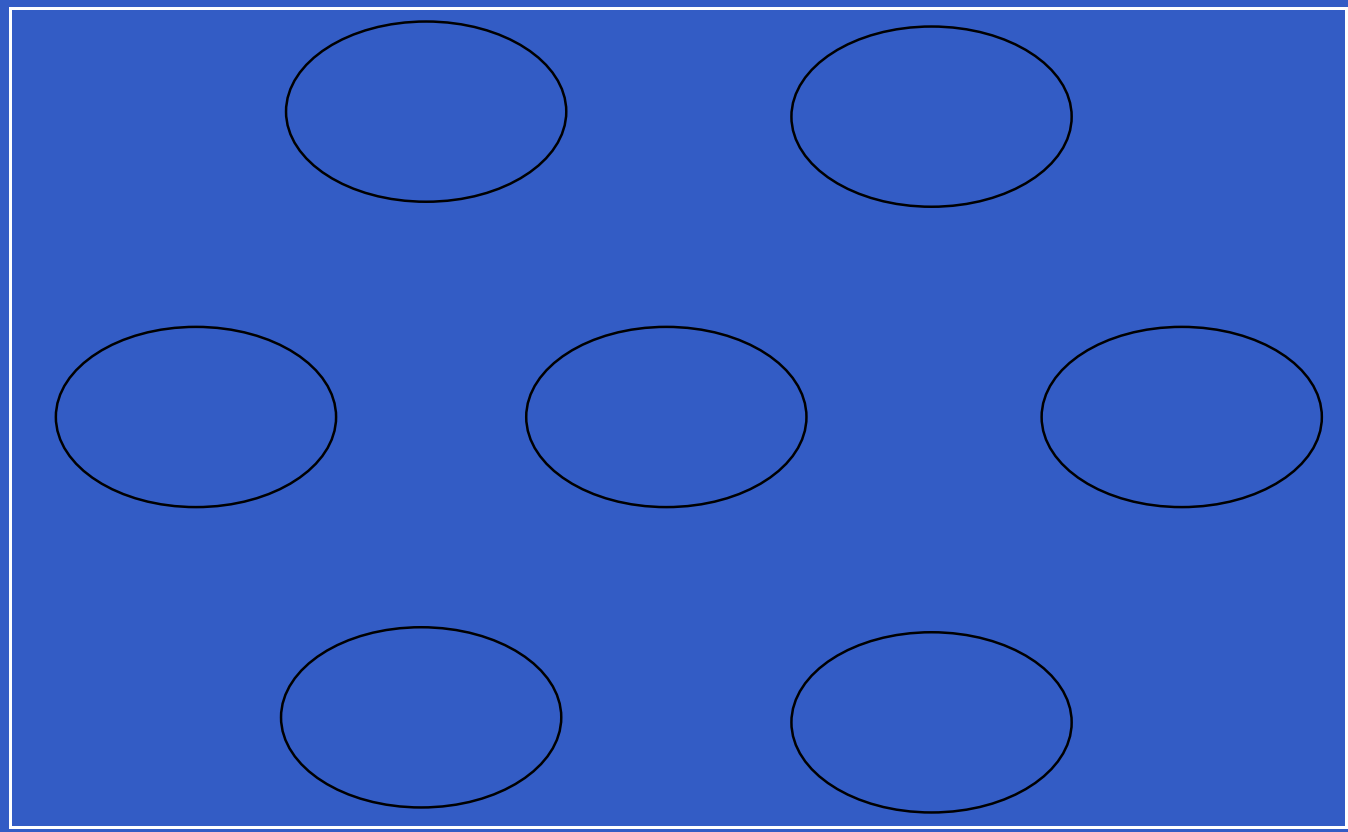
Paso 3. Evaluar y ordenar $\boldsymbol{x}_1^l$ and $\boldsymbol{x}_2^l$ obteniendo:
$\boldsymbol{x}_{1:2}^l$ (el mejor) y $\boldsymbol{x}_{2:2}^l$ (el peor)

Paso 4. Actualizar el vector de probabilidades $p_l(\boldsymbol{x})$ hacia $\boldsymbol{x}_{1:2}^l$
for $i = 1$ to n
 if $x_{i,1:2}^l \neq x_{i,2:2}^l$ entonces
  if $x_{i,1:2}^l = 1$ entonces $p_l(x_i) = p_{l-1}(x_i) + \frac{1}{K}$
  if $x_{i,1:2}^l = 0$ entonces $p_l(x_i) = p_{l-1}(x_i) - \frac{1}{K}$

Paso 5. Chequear si el vector de probabilidades $p_l(\boldsymbol{x})$ ha convergido
for $i = 1$ to n do
 if $p_l(x_i) > 0$ y $p_l(x_i) < 1$ entonces volver al Paso 2

Paso 6. $p_l(\boldsymbol{x})$ representa la solucion final

3.2.1 Sin dependencias iv



3.2.2 Dependencias bivariadas i

MIMIC (Mutual Information Maximization for Input Clustering) De Bonet y col., 1997

$$p_l^{\pi}(\boldsymbol{x}) = p_l(x_{i_1}|x_{i_2}) \cdot p_l(x_{i_2}|x_{i_3}) \cdot \ldots \cdot p_l(x_{i_{n-1}}|x_{i_n}) \cdot p_l(x_{i_n})$$

donde $\pi = (i_1, i_2, \ldots, i_n)$ es una permutación de los índices $1, \ldots, n$

- en cada generación tratar de encontrar la permutación π que minimiza la distancia de Kullback-Leibler entre $p_l^{\pi}(\boldsymbol{x})$ y $\hat{p}(\boldsymbol{x}|D_{l-1}^{Se})$
- algoritmo voraz basado en la teoría de la información
  - buscar la variable X_{i_n} con menor entropía
  - en cada paso seleccionar –del conjunto de variables no elegidas hasta el momento– la variable cuya entropía condicional media con respecto a la variable seleccionada en el paso anterior es mínima

3.2.2 Dependencias bivariadas ii

- COMIT (Combining Optimizers with Mutual Information Trees) Baluja y Davies, 1997

$$p_l(\boldsymbol{x}) = \prod_{i=1}^{n} p_l(x_i|x_{j(i)})$$

  - en cada generación se utiliza una adaptación del algoritmo MWST de Chow y Liu, 1968 para buscar, para cada variable X_i, la variable $X_{j(i)}$:

3.2.2 Dependencias bivariadas iii

- para cada par de variables calcular:

$$I(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)}$$

- asignar las dos ramas más largas al árbol que se está construyendo
- examinar la siguiente rama más larga y añadirla al árbol a menos que forme un ciclo, en cuyo caso descartar la rama actual y considerar la siguiente más larga
- repetir los pasos anteriores hasta que se hayan seleccionado $n - 1$ ramas
- $p_l(\boldsymbol{x})$ puede ser calculado seleccionando al azar un nodo raíz y siguiendo la estructura de árbol obtenida

3.2.2 Dependencias bivariadas iv

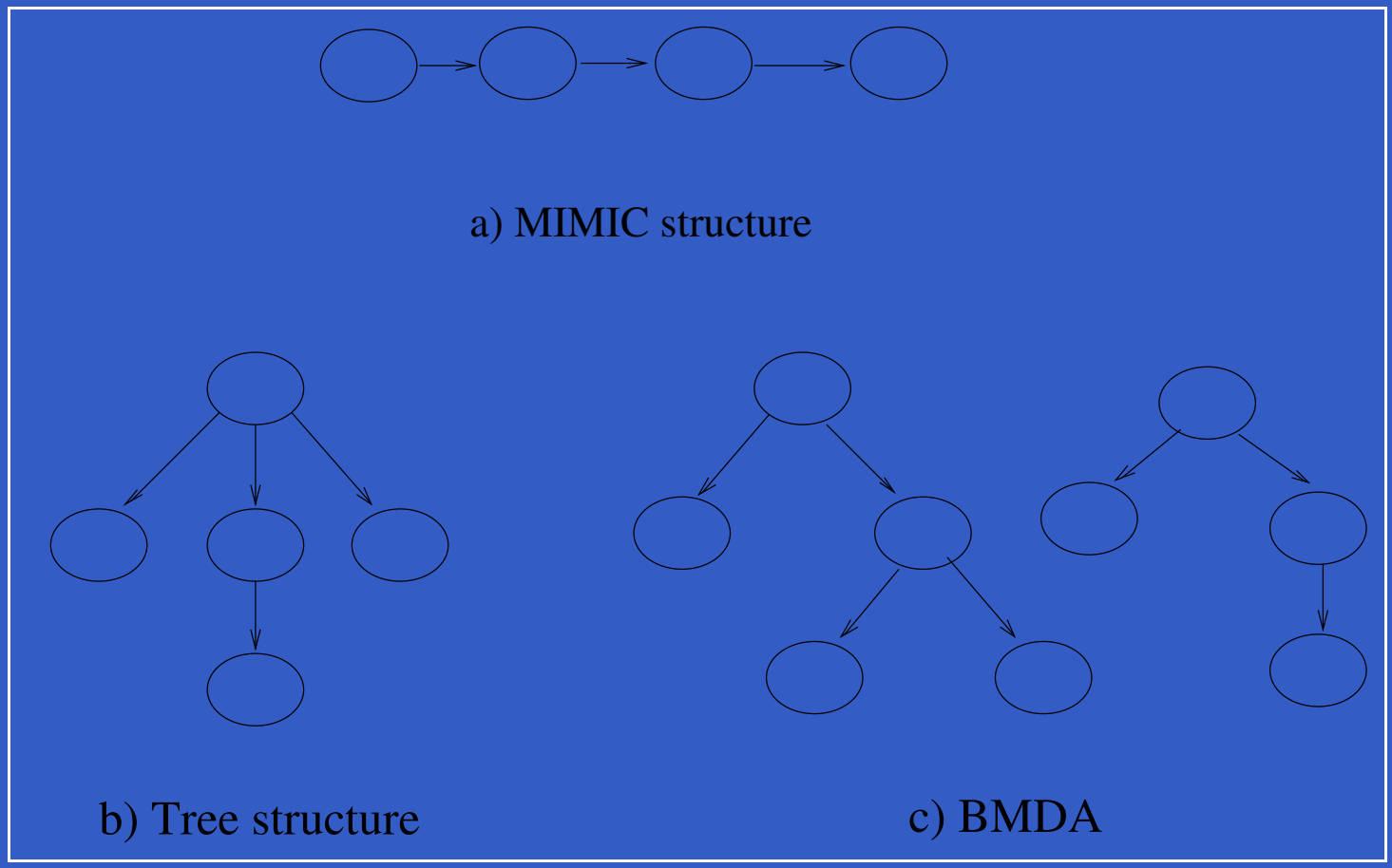
BMDA (Bivariate Marginal Distribution Algorithm) Pelikan y Mühlenbein, 1998

- Basado en la construcción de un grafo de dependencias: conjunto de árboles no necesariamente conexos

3.2.2 Dependencias bivariadas v

- Escoger al azar una variable como nodo raiz
- Seleccionar de entre las variables no incorporadas al árbol la que tenga mayor dependencia (test chi-cuadrado) con cualquiera de las previamente incorporadas
- Repetir el paso anterior hasta que ninguna de las variables que quedan fuera pase el test. En ese momento seleccionar al azar una variable de entre las que faltan por incorporar como nodo raiz de un nuevo árbol

3.2.2 Dependencias bivariadas vi



3.2.3 Dependencias múltiples i

- EBNA (Estimation of Bayesian Network Algorithm) Etxeberria y Larrañaga, 1999
  - uso de redes Bayesianas para aprender y simular $p_l(\boldsymbol{x}|D_{l-1}^{Se})$
  - M_0 es un DAG sin ningún arco $p(X_i = x_i) = \frac{1}{r_i}, i = 1, \ldots, n$
  - la búsqueda voraz comienza con el modelo obtenido en la generación previa

3.2.3 Dependencias múltiples ii

EBNA$_{BIC}$

$M_0 \leftarrow (S_0, \boldsymbol{\theta}_0)$

$D_0 \leftarrow$ Muestreando M_0, obtener N individuos

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

$D_{l-1}^{Se} \leftarrow$ Seleccionar Se individuos de D_{l-1}

$S_l^* \leftarrow$ Buscar la estructura que maximice $BIC(S_l, D_{l-1}^{Se})$

$\boldsymbol{\theta}^l \leftarrow$ Calcular $\{\theta_{ijk}^l = \frac{N_{ijk}^l + 1}{N_{ij}^l + r_i}\}$ usando D_{l-1}^{Se} como conjunto de datos

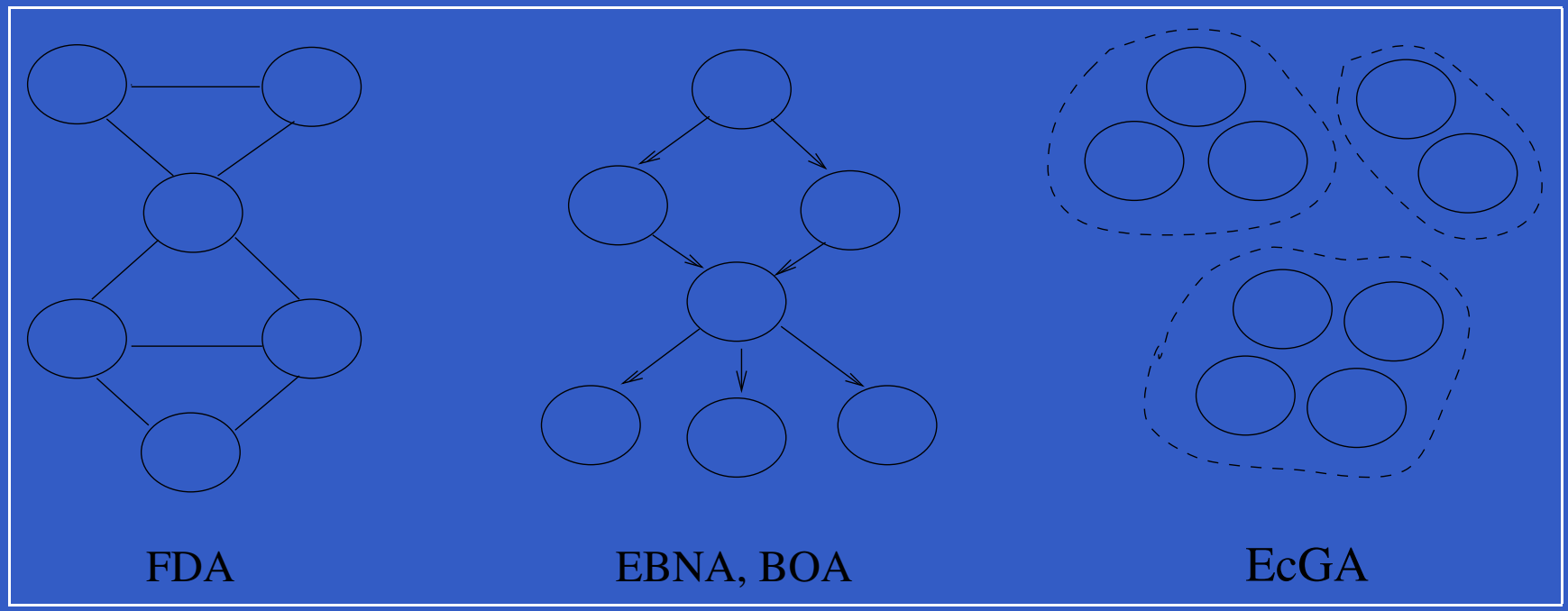
$M_l \leftarrow (S_l^*, \boldsymbol{\theta}^l)$

$D_l \leftarrow$ Utilizando PLS muestrear M_l para obtener N individuos

3.2.3 Dependencias múltiples iii

- BOA (Bayesian Optimization Algorithm) Pelikan y col., 1999
  - score: verosimilitud marginal
  - búsqueda: búsqueda voraz partiendo de una estructura vacía en cada generación
- LFDA (Learning Factorized Distribution Algorithm) Mühlenbein y Mahnig, 1999a
  - score: BIC
  - búsqueda: búsqueda voraz partiendo de una estructura vacía en cada generación
- EcGA (Extended compact Genetic Algorithm) Harik, 1999
  - Factorización: la distribución de probabilidad conjunta se factoriza como un producto de distribuciones marginales $p_l(\boldsymbol{x}) = \prod_{c=1}^{C} p_l(\boldsymbol{x}_c)$

3.2.3 Dependencias múltiples iv



3.3 Optimización en dominios continuos via aprendizaje y simulación de redes Gaussianas

- Sin dependencias (UMDA$_c$, SHCLVND, PBIL$_c$)
- Dependencias bivariadas (MIMIC$_c$)
- Dependencias múltiples
  - EMNA$_a$
  - EMNA$_{ee}$
  - EGNA$_{BIC}$,
  - EGNA$_{BGe}$

3.3.1 Sin dependencias i

- UMDA$_c$ (Univariate Marginal Distribution Algorithm continuous) Larrañaga y col., 2000

$$f_l(\boldsymbol{x};\boldsymbol{\theta}^l) = \prod_{i=1}^{n} f_l(x_i,\boldsymbol{\theta}_i^l)$$

  - en cada generación cada variable puede seguir una distribución de probabilidad diferente
  - estimación máximo verosímil de los parámetros
  - UMDA$_c^G$ cada variable sigue una normal univariante $\boldsymbol{\theta}_i^l = (\mu_i^l, \sigma_i^l)$:

$$\widehat{\mu_i^l} = \overline{X_i}^l = \frac{1}{Se}\sum_{r=1}^{Se} x_{i,r}^l; \quad \widehat{\sigma_i^l} = \sqrt{\frac{1}{Se}\sum_{r=1}^{Se}(x_{i,r}^l - \overline{X_i}^l)^2}$$

3.3.1 Sin dependencias ii

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

- SHCLVND (Stochastic Hill Climbing with Learning by Vectors of Normal Distributions) Rudlof y Köppen, 1996
  - el vector de medias $\boldsymbol{\mu} = (\mu_1, ..., \mu_i, ..., \mu_n)$ se adapta por medio de una regla Hebbiana:

$$\boldsymbol{\mu}^{l+1} = \boldsymbol{\mu}^{l} + \alpha \cdot (\boldsymbol{b}^{l} - \boldsymbol{\mu}^{l})$$

    con $0 < \alpha < 1$ y $\boldsymbol{b}^l$ el baricentro de los k mejores individuos
  - adaptación del vector de varianzas $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma}^{l+1} = \boldsymbol{\sigma}^{l} \cdot \beta \quad (\beta < 1)$$

3.3.1 Sin dependencias iii

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

- PBIL$_c$ (Population Based Incremental Learning. Continuous) Sebag y Ducoulombier, 1998
  - dados $\boldsymbol{x}^{1,l}$, $\boldsymbol{x}^{2,l}$ y $\boldsymbol{x}^{Se,l}$ el mejor, segundo mejor y el peor individuo en la l-ésima generación:

$$\boldsymbol{\mu}^{l+1} = (1-\alpha)\cdot\boldsymbol{\mu}^{l} + \alpha\cdot(\boldsymbol{x}^{1,l} + \boldsymbol{x}^{2,l} - \boldsymbol{x}^{Se,l})$$

  - cuatro heurísticas de adaptación de $\boldsymbol{\sigma}^{l+1}$

3.3.2 Dependencias a pares i

MIMIC$_c^G$ (Mutual Information Maximization for Input Clustering. Continuous. Gaussian) Larrañaga y col. 2000

- La distribución marginal para cada par de variables se supone que es una normal bidimensional
- Dada una permutación $\pi = (i_1, i_2, \ldots, i_n)$ la factorización es:

$$f_\pi(\boldsymbol{x}) = f(x_{i_1} \mid x_{i_2}) \cdot f(x_{i_2} \mid x_{i_3}) \cdot \ldots \cdot f(x_{i_{n-1}} \mid x_{i_n}) \cdot f(x_{i_n})$$

- En cada generación buscar la permutación que minimiza la distancia de Kullback-Leibler entre la función de densidad verdadera $f(\boldsymbol{x})$ y las densidades $f_\pi(\boldsymbol{x})$
- Esto es equivalente a minimizar:

$$J_\pi(\boldsymbol{x}) = h(X_{i_1} \mid X_{i_2}) + h(X_{i_2} \mid X_{i_3}) + \ldots + h(X_{i_{n-1}} \mid X_{i_n}) + h(X_{i_n})$$

donde $h(X)$ es la entropía de Shannon de la variable X y $h(X|Y) = h(X,Y) - h(Y)$

3.3.2 Dependencias a pares ii

- En el caso Gaussiano:

$$h(X) = \frac{1}{2}(1 + \log 2\pi) + \log \sigma_X$$

$$h(X \mid Y) = \frac{1}{2}\left[(1 + \log 2\pi) + \log(\frac{\sigma_X^2 \sigma_Y^2 - \sigma_{XY}^2}{\sigma_Y^2})\right]$$

donde σ_{XY}^2 es la covarianza entre las variables X e Y

MIMIC$_c^G$

Paso 1. Escoger $i_n = \arg\min_j \hat{\sigma}^2_{X_{i_j}}$

Paso 2. Escoger $i_k = \arg\min_j \hat{\sigma}^2_{X_{i_j}} - \frac{\hat{\sigma}^2_{X_{i_j} X_{i_{k+1}}}}{\hat{\sigma}^2_{X_{i_{k+1}}}}$ $j \neq i_{k+1}, \ldots, i_n; k = n-1, n-2, \ldots, 1$

3.3.3 Dependencias múltiples i

EMNA$_{global}$(Estimation of Multivariate Normal Algorithm. Global) Larrañaga y Lozano 2001

EMNA$_{global}$

$D_0 \leftarrow$ Generar M individuos (la poblacion inicial) al azar

Repetir for $l = 1, 2, \ldots$ hasta que se verifique el criterio de parada

$D_{l-1}^{Se} \leftarrow$ Seleccionar $N \leq M$ individuos de D_{l-1} de acorde a un metodo de seleccion

$f_l(\boldsymbol{x}) = f(\boldsymbol{x}|D_{l-1}^{Se}) = \boldsymbol{\mathcal{N}}(\boldsymbol{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \leftarrow$ Estimar la funcion de densidad normal mutidimensional a partir de los individuos seleccionados

$D_l \leftarrow$ Muestrear M individuos (la nueva poblacion) de $f_l(\boldsymbol{x})$

3.3.3 Dependencias múltiples ii

EMNA$_{global}$

- $\boldsymbol{\mu}_l = (\mu_{1,l}, \ldots, \mu_{n,l})$
- $\boldsymbol{\Sigma}_l$ con elementos: $\sigma^2_{ij,l}$ with $i, j = 1, \ldots, n$
- En cada generación hace falta estimar $2n + \begin{pmatrix} n-1 \\ 2 \end{pmatrix}$ parámetros: n esperanzas matemáticas, n varianzas y $\begin{pmatrix} n-1 \\ 2 \end{pmatrix}$ covarianzas

$$\hat{\mu}_{i,l} = \overline{X}^l_i = \frac{1}{N} \sum_{r=1}^{N} x^l_{i,r} \quad i = 1, \ldots, n$$

$$\hat{\sigma}^2_{i,l} = \frac{1}{N} \sum_{r=1}^{N} (x^l_{i,r} - \overline{X}^l_i)^2 \quad i = 1, \ldots, n$$

3.3.3 Dependencias múltiples iii

EMNA$_a$ (Estimation of Multivariate Normal Algorithm. Adaptive) Larrañaga y Lozano, 2001

EMNA$_a$

$D_0 \leftarrow$ Generar al azar N individuos (la población inicial)

Obtener la primera densidad normal multivariante $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$

Repetir para $l = 1, 2, \ldots$ hasta cumplir la condición de parada

Generar un individuo $\boldsymbol{x}_{ge}^{l}$ de $\mathcal{N}(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$

si $\boldsymbol{x}_{ge}^{l}$ es mejor que el peor individuo, $\boldsymbol{x}^{l,N}$, **entonces**

añadir $\boldsymbol{x}_{ge}^{l}$ a la población y borrar $\boldsymbol{x}^{l,N}$ de esta

obtener $\mathcal{N}(\boldsymbol{\mu}_{l+1}, \boldsymbol{\Sigma}_{l+1})$

3.3.3 Dependencias múltiples iv

EMNA$_i$ Fórmulas de adaptación

Modificar la función de densidad multivariante:

$$\boldsymbol{\mu}_{l+1} = \boldsymbol{\mu}_l + \frac{1}{N}(\boldsymbol{x}^l_{ge} - \boldsymbol{x}^{l,N})$$

$$\sigma^2_{ij,l+1} = \sigma^2_{ij,l} - \frac{1}{N^2}(x^l_{ge,i} - x^{l,N}_i) \cdot \sum_{r=1}^{N}(x^{l,r}_j - \mu^l_j) +$$

$$-\frac{1}{N^2}(x^l_{ge,j} - x^{l,N}_j) \cdot \sum_{r=1}^{N}(x^{l,r}_i - \mu^l_i) + \frac{1}{N^2}(x^l_{ge,i} - x^{l,N}_i)(x^l_{ge,j} - x^{l,N}_j) +$$

$$-\frac{1}{N}(x^{l,N}_i - \mu^{l+1}_i)(x^{l,N}_j - \mu^{l+1}_j) + \frac{1}{N}(x^l_{ge,i} - \mu^{l+1}_i)(x^l_{ge,j} - \mu^{l+1}_j)$$

3.3.3 Dependencias múltiples v

EMNA$_i$ (Estimation of Multivariate Normal Algorithm incremental) Larrañaga y Lozano, 2001

EMNA$_i$

$D_0 \leftarrow$ Generar M individuos (la poblacion inicial) al azar
Seleccionar $N \leq M$ individuos de D_0 de acorde con un metodo de seleccion
Obtener la primera densidad normal multivariante $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$
Repetir for $l = 1, 2, \ldots$ hasta que se verifique un criterio de parada
  Generar un individuo $\boldsymbol{x}_{ge}^l$ de $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$
  Añadir $\boldsymbol{x}_{ge}^l$ a la poblacion
  Obtain $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{l+1}, \boldsymbol{\Sigma}_{l+1})$

3.3.3 Dependencias múltiples vi

EMNA$_i$ Fórmulas de adaptación:

(1) $$\boldsymbol{\mu}_{l+1} = \frac{N_l}{N_l+1}\boldsymbol{\mu}_l + \frac{1}{N_l+1}\boldsymbol{x}^l_{ge}$$

(2) $$\sigma^2_{ij,l+1} = \frac{N_l}{N_l+1}\sigma^2_{ij,l} + \frac{1}{N_l+1}(x^l_{ge,i} - \mu^l_i)(x^l_{ge,j} - \mu^l_j).$$

3.3.3 Dependencias múltiples vii

EMNA$_{ee}$ (Estimation of Multivariate Normal Algorithm. Edge Exclusion) Larrañaga y col., 2000

- Basado en la detección de independencias entre pares de variables
- El aprendizaje se realiza por medio de $\begin{pmatrix} n \\ 2 \end{pmatrix}$ test de exclusión de arcos
  - X_i y X_j son independientes sii se acepta el siguiente test de hipótesis (Smith y Whittaker, 1998)

$$\begin{cases} H_0 : w_{ij} = 0 & \text{hipótesis nula} \\ H_A : w_{ij} \neq 0 & \text{hipótesis alternativa} \end{cases}$$

  - test de la razón de verosimilitud:

$$T_{lik} = -n \log(1 - r^2_{ij|rest}) \text{ con } r_{ij|rest} = -\hat{w}_{ij}(\hat{w}_{ii}\hat{w}_{jj})^{-1/2}$$

3.3.3 Dependencias múltiples viii

- EGNA$_{BIC}$ (Estimation of Gaussian Network Algorithm. BIC) Larrañaga y Lozano, 2001
  - score: máxima verosimilitud penalizada (BIC)
  - búsqueda: búsqueda voraz
    - primera generación: se comienza con una estructura sin arcos
    - el resto de las generaciones: comenzar con el modelo obtenido en la generación previa
- EGNA$_{BGe}$ (Estimation of Gaussian Network Algorithm. BGe) Larrañaga y col., 2000
  - score: verosimilitud marginal (BGe métrica Bayesiana para Gaussianas verificando la propiedad de *score equivalence*)
  - búsqueda: búsqueda voraz
    - primera generación: se comienza con una estructura sin arcos
    - el resto de las generaciones: comenzar con el modelo obtenido en la generación previa

4 Aplicaciones de los Algoritmos de Estimación de Distribuciones

- 4.1 Introducción
- 4.2 Algoritmos de estimación de distribuciones en optimización
  - 4.2.1 Introducción
  - 4.2.2 Problema de la mochila
  - 4.2.3 Problema del viajante del comercio
- 4.3 Algoritmos de estimación de distribuciones en aprendizaje automático
  - 4.3.1 Introducción
  - 4.3.2 Selección de variables
  - 4.3.3 Pesado de variables en K-NN
  - 4.3.4 Inducción de reglas
  - 4.3.5 Clustering particional
  - 4.3.6 Ajuste de pesos en redes neuronales
  - 4.3.7 Inferencia abductiva en redes Bayesianas

4.1 Introducción i

Optimización versus aprendizaje automático con EDAs

- Similitudes:
  - Ambos se pueden ver como un problema de búsqueda
  - Problemática común de generar soluciones factibles
- Diferencias en la evaluación:
  - En optimización directamente el valor de la función objetivo
  - En aprendizaje automático se estima (*holdout, k-fold croos-validation, bootstrapping, ...*)

4.2.1 Introducción i

- Dificultades al aplicar los EDAs derivadas de:
  - generación de soluciones factibles
  - utilización de codificaciones basadas en permutaciones
- Soluciones:
  - modificar el PLS
  - obtener permutaciones a partir de simulaciones de redes Gaussianas

4.2.2 Problema de la mochila i

- $\max \sum_{i=1}^{n} b_i x_i$ sujeto a
  - $\sum_{i=1}^{n} w_i x_i \leq C$
  - $x_i = 0, 1$
- b_i beneficio; w_i peso; C capacidad
- Problema NP-completo

4.2.2 Problema de la mochila ii

- EDAs utilizados: discretos y continuos
- Codificación: binaria y permutación
- Gestión de las restricciones:
  - penalización
  - algoritmo *first fit*
- Inicialización:
  - uniforme
  - probabilidad proporcional al ratio (beneficio/peso)
  - siguiendo la siguiente distribución de probabilidad:

$$p_0(x_i) = \begin{cases} \alpha & \text{si el item } x_i \text{ es seleccionado por el } \textit{first fit} \\ 1-\alpha & \text{si el item } x_i \text{ no es seleccionado por el } \textit{first fit} \end{cases}$$

4.2.2 Problema de la mochila iii

Resultados experimentales: 50, 200, 1000

Representación binaria (200):

	penalización			*first fit*		
	unif.	*prop.*	*d. prob.*	*unif.*	*prop.*	*d. prob.*
UMDA	7964.0	7977.3	8011.6	8018.0	8018.0	8018.2
MIMIC	7977.2	7990.2	8013.0	8017.2	8018.0	8018.6
UMDA$_c$	7935.4	8003.8	8010.4	8016.8	8016.8	8014.5
MIMIC$_c$	7950.2	7985.0	8010.0	8017.8	8016.8	8014.1

Representación basada en permutaciones (200):

	uniforme	*proporcional*	*dis. prob.*
UMDA$_c$	8012.0	8012.1	8016.5
MIMIC$_c$	8005.8	8014.4	8016.2

4.2.2 Problema de la mochila iv

Conclusiones:

- *first fit* funciona mejor que la penalización
- $n = 50$, inicialización uniforme, representación binaria
- $n = 200$, evaluación *first fit*, representación binaria, UMDA
- $n = 1000$, evaluación *first fit*, representación basada en permutaciones y distribución de probabilidad

4.2.3 Problema del viajante de comercio i

- Problema: dado un conjunto de ciudades, y una matriz de distancias entre pares de ciudades, encontrar una ruta de costo mínimo que pasa por cada ciudad una sola vez y vuelve a la ciudad de origen
- Problema NP-completo
- Algoritmos de búsqueda local: 2-opt, 3-opt, etc.

4.2.3 Problema del viajante de comercio ii

- EDAs discretos: UMDA, MIMIC, TREE, EBNA
- EDAs continuos: UMDA$_c$, MIMIC$_c$, EGNA, EMNA
- Codificación: basada en permutaciones
- Hibridación: 2-opt
- Experimentos: Gröstel24, Gröstel48, Gröstel120

4.2.3 Problema del viajante de comercio iii

Gröstel24

	500-sin		*500-con*	
	Mejor	*Media*	*Mejor*	*Media*
Local-opt.			1272	1285
GA-ER*	1272	1272		
GA-OX2*	1300	1367		
UMDA	1339	1495	1272	1272
MIMIC	1391	1486	1272	1272
TREE	1413	1486	1272	1272
EBNA	1431	1528	1272	1272
UMDA_c	1289	1289	1272	1272
MIMIC_c	1289	1289	1272	1272
EGNA	1289	1306	1272	1272
EMNA	1289	1289	1272	1272

4.2.3 Problema del viajante de comercio iv

Gröstel48

	500-sin		*500-con*	
	Mejor	*Media*	*Mejor*	*Media*
Local-opt.			5200	5290
GA-ER*	5074	5138		
GA-OX2*	5251	5715		
UMDA	6715	7432	5079	5149
MIMIC	6679	7083	5046	5053
TREE	-	-	5046	5071
EBNA	7044	7476	5165	5193
UMDA_c	5142	5248	5046	5048
MIMIC_c	5122	5176	5046	5046
EGNA	5122	5249	5046	5046
EMNA	5336	5532	5046	5048

4.2.3 Problema del viajante de comercio v

Gröstel120

	500-sin		*500-con*	
	Mejor	*Media*	*Mejor*	*Media*
UMDA	14550	15530	7171	7257
MIMIC	13644	14432	7050	7092
UMDA_c	7546	7667	7077	7113
MIMIC_c	7658	7767	7055	7078

4.2.3 Problema del viajante de comercio vi

Conclusiones:

- EDAs continuos obtienen mejores resultados que discretos
- con poblaciones pequeñas mejores resultados que con grandes
- mejores resultados con optimización local
- sugerencia: combinación de EDA discretos y continuos

4.3.1 Introducción i

- Evaluación imprecisa: validación del modelo
  - entrenamiento-testeo
  - K*-fold CV*
  - *bootstraping*
- Búsqueda en el espacio de modelos

4.3.2 Selección de variables i

- Objetivo: reducir el número de características de cara a mejorar el rendimiento de los algoritmos de aprendizaje
- Variables irrelevantes, variables redundantes
- Aproximación indirecta (*filter*), aproximación directa (*wrapper*)
- Clasificador Naive-Bayes
- Codificación: array binario
- Validación: 5×2 CV

4.3.2 Selección de variables ii

Dimensionalidad pequeña y mediana

Dominio	*Num. de ejemplos*	*Num. de variables*
(1)Ionosphere	351	34
(2)Horse-colic	368	22
(3)Soybean-large	683	35
(4)Anneal	898	38
(5)Image	2,310	19
(6)Sick-euthyroid	3,163	25

4.3.2 Selección de variables iii

Dom.	*W. FSS*	*SFS*	*SBE*	*FSS-GA-o*	*FSS-GA-u*	*FSS-EBNA*
(1)	84,84	90,25	91,39	91,17	90,97	92,40
(2)	78,97	83,31	82,12	83,43	83,51	83,93
(3)	81,96	86,38	87,78	85,64	86,09	88,64
(4)	93,01	86,72	92,49	92,95	93,13	94,10
(5)	79,95	88,65	88,82	88,67	89,12	88,98
(6)	84,77	90,73	95,57	95,97	95,90	96,14

- Sin selección de variables (W. FSS)
- Selección secuencial hacia adelante (SFS)
- Eliminación secuencial hacia atrás (SBE)
- Algoritmo genético con cruce basado en un punto (FSS-GA-o)
- Algoritmo genético con cruce uniforme (FSS-GA-u)
- FSS-EBNA

4.3.2 Selección de variables iv

Alta dimensionalidad

Dominio	*Num. de ejemplos*	*Num. de variables*
Audiology	226	69
Arrhythmia	452	279
Cloud	1,834	204
DNA	3,186	180
Internet advertisements	3,279	1558
Spambase	4,601	57

4.3.2 Selección de variables v

Dominio	*Sin FSS*	*FSS-GA-o*	*FSS-GA-u*
Audiology	52,39	68,29	68,44
Arrhythmia	39,91	63,23	64,73
Cloud	68,18	74,49	75,17
DNA	93,93	94,00	95,01
Internet adv.	95,23	96,10	96,38
Spambase	81,71	88,92	88,77

Dominio	*FSS-PBIL*	*FSS-BSC*	*FSS-MIMIC*	*FSS-TREE*
Audiology	70,22	68,29	68,88	70,09
Arrhythmia	64,62	65,01	64,33	64,51
Cloud	75,18	76,24	76,31	75,84
DNA	94,86	95,40	95,53	95,40
Internet adv.	96,49	96,37	96,46	96,69
Spambase	88,63	89,52	89,80	89,60

4.3.3 Pesado de variables en K-NN i

- K-NN
  - almacena todos los conjuntos de entrenamiento
  - al presentarle un ejemplo de testeo, lo compara con el ejemplo del conjunto de entrenamiento que se encuentre mas cercano
  - usa dicho ejemplo mas cercano para predecir la clase del ejemplo de testeo

$$\textit{distancia}(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n} w_i \times \textit{diferencia}(x_i,y_i)^2}$$

  - $\mathbf{x} = (x_1, ..., x_i, ..., x_n)$, $\mathbf{y} = (y_1, ..., y_i, ..., y_n)$ son ejemplos de entrenamiento
  - w_i denota el peso asignado a la variable X_i
- Objetivo: usar EDAs para buscar los pesos óptimos

4.3.3 Pesado de variables en K-NN ii

- Codificación: $\mathbf{w} = (w_1, ..., w_i, ..., w_n)$
  - Discreta: $\{0{,}0, 0{,}5, 1{,}0\}$
  - Continua: $w_i \in \mathbb{R}$
- Aproximación *wrapper*: clasificador1-NN
- Validación: 5$\times$2 CV

4.3.3 Pesado de variables en K-NN iii

Dominio	*Número de ejemplos*	*Número de variables*
(1) *LED24*	600	24
(2) *Waveform-21*	600	21
(3) *3-Weights*	600	12
(4) *C-Weights*	600	10
(5) *Glass*	214	9
(6) *CRX*	690	15
(7) *Vehicle*	846	18
(8) *Contraceptive*	1,473	9

4.3.3 Pesado de variables en K-NN iv

Dominio	*no-FW*	*DIET-10*	*GA-o*	*EBNA*	*IB4*	*EGNA*
(1)	47,37	63,84	68,64	69,03	66,70	61,55
(2)	76,20	76,66	76,71	76,87	77,96	76,90
(3)	77,19	81,91	82,88	85,99	80,32	82,00
(4)	81,01	83,55	83,93	83,55	81,98	84,33
(5)	64,85	71,34	71,32	71,12	61,13	70,09
(6)	81,56	82,12	83,14	83,74	85,48	82,17
(7)	67,33	68,71	69,86	69,43	64,65	69,58
(8)	43,61	47,54	48,10	48,32	45,66	44,95

4.3.4 Inducción de reglas i

- Reglas SI-ENTONCES
  - simplicidad, transparencia y comprensión
- Notación:
  - $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ variables predictoras
  - C etiqueta de clase
  - $|X_i| = r_i$
  - $|C|$
- Codificación: $n + 1$ variables
  - las primeras n posiciones representan antecedentes de las reglas
  - la última posición representa el consecuente

4.3.4 Inducción de reglas ii

$X_1, \ldots, X_5$ con $r_1 = r_2 = r_3 = 3$ y $r_4 = r_5 = 4$, $|C| = 2$

$$(1, 2, 2, 3, 4, 1) \equiv$$

SI ($X_1 = 1$ y $X_2 = 2$ y $X_3 = 2$ y $X_4 = 3$ y $X_5 = 4$) ENTONCES

$$C = 1$$

- Aproximación x_i, cardinalidad del espacio de búsqueda:
  $| C | \prod_{i=1}^{n} r_i$
  $(1, 2, 2, 3, 4, 1) \equiv$ SI ($X_1 = 1$ y $X_2 = 2$ y $X_3 = 2$ y $X_4 = 3$ y $X_5 = 4$) ENTONCES $C = 1$
- Aproximación $x_i, \#$, cardinalidad del espacio de búsqueda:
  $| C | \prod_{i=1}^{n} (r_i + 1)$
  $(1, 2, 4, 3, 5, 2) \equiv$ SI ($X_1 = 1$ y $X_2 = 2$ y $X_4 = 3$) ENTONCES $C = 2$

4.3.4 Inducción de reglas iii

$X_1, \ldots, X_5$ con $r_1 = r_2 = r_3 = 3$ y $r_4 = r_5 = 4$, $|C| = 2$

$$(1, 2, 2, 3, 4, 1) \equiv$$

SI ($X_1 = 1$ y $X_2 = 2$ y $X_3 = 2$ y $X_4 = 3$ y $X_5 = 4$) ENTONCES $C = 1$

- Aproximación $x_i, \#, \neq x_i$, cardinalidad del espacio de búsqueda:
  $| C | \prod_{i=1}^{n} (2r_i + 1)$
  $(5, 2, 7, 5, 8, 2) \equiv$ SI ($X_1 \neq 1$ y $X_2 = 2$ y $X_3 \neq 3$ y $X_5 \neq 3$) ENTONCES $C = 2$
- Regla de complejidad aumentada, cardinalidad del espacio de búsqueda:
  $| C | \prod_{i=1}^{n} (2r_i + 1)^k$
  $(2, 2, 2, 1, 3, 3, 2, 1, 4, 1, 2) \equiv$ SI ($X_1 = 2$ y $X_2 = 2$ y $X_3 = 2$ y $X_4 = 1$ y $X_5 = 3$)
  or ($X_1 = 3$ y $X_2 = 2$ y $X_3 = 1$ y $X_4 = 4$ y $X_5 = 1$) ENTONCES $C = 2$

4.3.4 Inducción de reglas iv

Dominio	*Número de ejemplos*	*Número of variables*
Heart	270	13
Cleveland	303	13
2-Attractors	2,000	12

Dominio	*CN2*	*RIPPER*
Heart	$78{,}17 \pm 2{,}61$	$77{,}04 \pm 1{,}44$
Cleveland	$79{,}84 \pm 3{,}18$	$78{,}88 \pm 3{,}30$
2-Attractors	$87{,}05 \pm 0{,}85$	$81{,}15 \pm 0{,}88$

4.3.4 Inducción de reglas v

Disyunción de dos reglas simples

Dominio	*UMDA*	*TREE*	*EBNA*
Heart	$77{,}33 \pm 0{,}42$	$77{,}11 \pm 1{,}46$	$79{,}70 \pm 1{,}91$
Cleveland	$77{,}35 \pm 0{,}82$	$78{,}41 \pm 0{,}89$	$77{,}22 \pm 0{,}96$
2-Attractors	$77{,}34 \pm 0{,}25$	$75{,}10 \pm 0{,}62$	$76{,}96 \pm 0{,}40$

Disyunción de 4 reglas simples

Dominio	*UMDA*	*TREE*	*EBNA*
Heart	$80{,}22 \pm 2{,}08$	$77{,}92 \pm 2{,}36$	$80{,}07 \pm 1{,}23$
Cleveland	$78{,}01 \pm 1{,}70$	$78{,}48 \pm 1{,}58$	$78{,}08 \pm 1{,}50$
2-Attractors	$80{,}85 \pm 0{,}37$	$78{,}27 \pm 0{,}62$	$80{,}15 \pm 0{,}33$

4.3.5 Clustering i

- Problema: distribuir n objetos en k clusters
- Cardinalidad del espacio de búsqueda:

$$S(n,k) = \frac{1}{k!}\sum_{j=1}^{k}(-1)^{k-1}\left(\begin{array}{c} k \\ j \end{array}\right)j^n$$

- k-Medias:

---

k-Medias (k clusters, i iteraciones) {
  Tomar los k primeros objetos como clusters iniciales
  Repetir para cada objeto
    Asignar el objeto al cluster más cercano
    Calcular los nuevos centroides de los clusters
  Hasta ningún objeto cambia de grupo **o** número de iteraciones $> i$
}

- EDAs como método alternativo a k-Medias

4.3.5 Clustering ii

- Codificación: 6 objectos en 3 clusters, $\{A, B, C, D, E, F\}$ particionado como sigue: $\{A\}$, $\{B, E\}$, $\{C, D, F\}$, la codificación es: $(1, 2, 3, 3, 2, 3)$
- Una partición es legal si aparecen los k valores $\Longrightarrow$ adaptar PLS
- Minimizar la función:

$$F(W, C) = \sum_{i=1}^{k} \sum_{j=1}^{n} u_{ij} D(v_j, c_i)$$

4.3.5 Clustering iii

	No. Clusters	*No. Objetos*	*No. Variables*
Cleveland	4	303	14
Wine	3	178	13
Iris	3	150	5
Soybean	4	47	36
Voting	2	345	17

4.3.5 Clustering iv

Resultados for Cleveland (k-Medias: 10048.9)

Tamaño	*Heur.*	*GA*	*BSC*	*MIMIC*	*TREE*	*EBNA*$_{BIC}$
200	*Ninguno*	14514.2	9994.2	9755.8	10442.2	9709.2
200	*Híbrido*	14502.6	9841.0	9739.9	10523.8	9705.0
500	*Ninguno*	14636.6	9763.3	9728.7	9956.1	9704.7
500	*Híbrido*	14907.8	9717.9	9702.9	9836.9	9703.7
1000	*Ninguno*	14668.8	9734.7	9719.0	9797.0	9738.7
1000	*Híbrido*	15021.2	9718.4	9698.5	9692.3	9711.0

- EDAs mejoran los resultados obtenidos por k-Medias y algoritmos genéticos
- Mejores resultados con la aproximación híbrida

4.3.6 Ajuste de pesos en redes neuronales i

- Motivación: BP riesgo de quedarse atrapado en mínimos locales
- Objetivo: comparar BP, GAs, ESs, EDAs
- Codificación: cada peso se asocia con cada variable
- Minimizar RMSE (*rooted mean square error*)
- Experimentos: base de datos ECOLI, predicción de la localización de proteinas en células eucariotidas, 336 ejemplos, 8 variables descriptivas, 8 clases, arquitectura de la red neuronal 8-4-2-8

4.3.6 Ajuste de pesos en redes neuronales ii

Algoritmo	*error–entrenamiento*	*error–testeo–5CV*	*porcen–test–5CV*
BP	0.2584±0.005	0.1289±0.001	8.3333±6.390
GA	0.1968±0.016	0.1001±0.003	47.8308±7.894
ES	0.1667±0.008	0.0891±0.002	65.8929±2.530
$\mathrm{UMDA_c}$	0.1830±0.006	0.0808±0.002	58.5970±5.928
$\mathrm{MIMIC_c}$	0.1778±0.013	0.0802±0.001	58.5075±4.815
GA+BP	0.3004±0.012	0.1522±0.004	8.0398±5.992
ES+BP	0.1925±0.020	0.0939±0.001	53.5417±4.251
$\mathrm{UMDA_c}$+BP	0.2569±0.006	0.1593±0.001	9.8209±6.943
$\mathrm{MIMIC_c}$+BP	0.2587±0.006	0.1585±0.001	10.4179±7.328

- Ordenación de los algoritmos respecto al error-testeo-5CV: MIMIC_c, UMDA_c, ES, ES+BP, GA, BP, resto de híbridos
- La aproximación híbrida funciona peor que la no híbrida

4.3.7 Inferencia abductiva en redes Bayesianas i

Tipos de razonamiento en una red Bayesiana sobre $\mathbf{X} = \{X_1, \ldots, X_n\}$

- Propagación de la evidencia: $p(x_i | \boldsymbol{X_O} = \boldsymbol{x_O})$
- Inferencia abductiva total $\equiv$ explicación mas probable (MPE):

$$\boldsymbol{x}_{\boldsymbol{U}}^* = \arg \max_{\boldsymbol{x_U}} p(\boldsymbol{x_U} | \boldsymbol{X_O} = \boldsymbol{x_O})$$

  $\boldsymbol{X_U} = \boldsymbol{X} \setminus \boldsymbol{X_O}$ variables no observadas
- Inferencia abductiva parcial $\equiv$ problema de búsqueda del máximo a posteriori (MAP):

$$\boldsymbol{x}_{\boldsymbol{E}}^* = \arg \max_{\boldsymbol{x_E}} p(\boldsymbol{x_E} | \boldsymbol{X_O} = \boldsymbol{x_O}) = \arg \max_{\boldsymbol{x_E}} \sum_{\boldsymbol{x_R}} p(\boldsymbol{x_E}, \boldsymbol{x_R} | \boldsymbol{x_O})$$

  $\boldsymbol{X_E} \subset \boldsymbol{X_U}$ conjunto de explicación, $\boldsymbol{X_R} = \boldsymbol{X_U} \setminus \boldsymbol{X_E}$
  En general $\boldsymbol{x}_{\boldsymbol{E}}^*$ no coincide con la configuración que se obtiene al quitar de $\boldsymbol{x}_{\boldsymbol{U}}^*$ la parte que no está en $\boldsymbol{X_E}$

4.3.7 Inferencia abductiva en redes Bayesianas ii

- Objetivo: usar EDAs para buscar x_E^*
- Codificación: un individuo es un posible valor del conjunto de explicación
- Función de evaluación: basada en el *junction tree*
- Interés en encontrar los K mejores MAPs

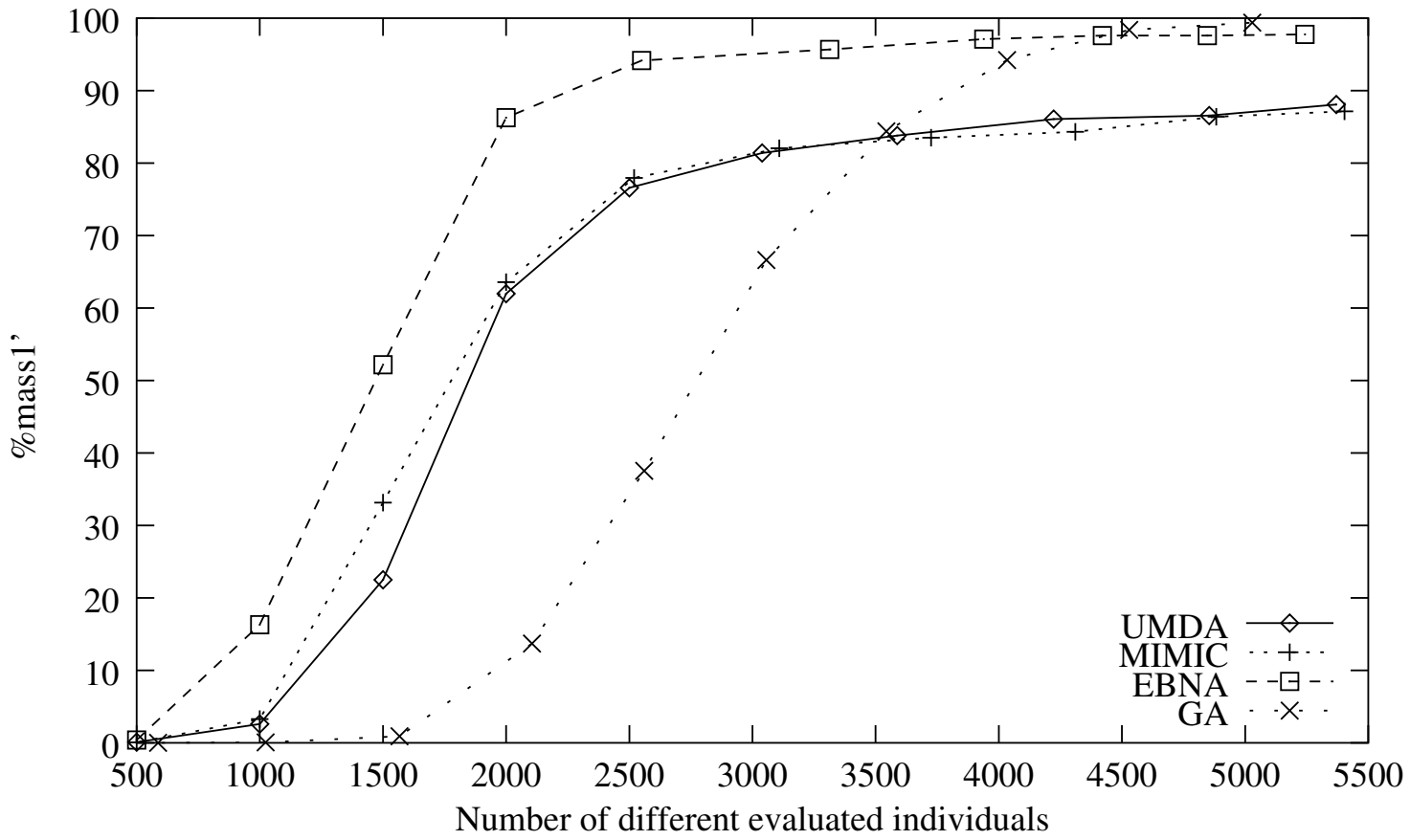
4.3.7 Inferencia abductiva en redes Bayesianas iii

#exp.	$\lvert \boldsymbol{X_E} \rvert$	*red*	$\boldsymbol{X_E}$	$\lvert \Omega_{\boldsymbol{X_E}} \rvert$
1	18	*Alarm*	pseudo-random	143,327,232
2	19	*Alarm*	pseudo-random	214,990,848
3	20	*Alarm*	pseudo-random	382,205,952
4	30	*random100*	pseudo-random	1,073,741,824
5	30	*random100e*	pseudo-random	1,073,741,824

4.3.7 Inferencia abductiva en redes Bayesianas iv

	%mass1′	*%mass10′*	*%mass25′*	*%mass50′*
UMDA	80.77 ± 10	75.16 ± 12	71.07 ± 12	68.25 ± 12
MIMIC	85.21 ± 12	80.43 ± 15	76.66 ± 17	74.19 ± 17
EBNA	95.56 ± 09	93.62 ± 12	92.44 ± 14	91.75 ± 16
GA	97.04 ± 00	89.63 ± 01	85.16 ± 01	83.19 ± 02

4.3.7 Inferencia abductiva en redes Bayesianas iv



5. Conclusiones

- EDAs como una nueva herramienta en computación evolutiva
- Basados en el aprendizaje de una distribución de probabilidad conjunta (la de los mejores individuos en cada generación) y su posterior simulación
- Las relaciones entre las variables utilizadas para codificar los puntos del espacio de búsqueda se explicitan
- Buen comportamiento en problemas con funciones objetivo de computación costosa
- En problemas de búsqueda de la mejor permutación, EDAs continuos mejores resultados que EDAs discretos
- Muchas posibilidades de investigar en este área de la computación evolutiva

Referencias

- P. Larrañaga, J. A. Lozano (2001) (editores) *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers.
- P. Larrañaga, J. A. Lozano (2002) (editores invitados) Número especial titulado *Synergies between Probabilistic Graphical Models and Evolutionary Computation* en *International Journal of Approximate Reasoning*.
- Workshop titulado *Optimization by Building and Using Probabilistic Models*. *OBUPM'2000* dentro del congreso GECCO'2000. Las Vegas 2000.
- Workshop titulado *Optimization by Building and Using Probabilistic Models*. *OBUPM'2001* dentro del congreso GECCO'2001. San Francisco 2001.
- Workshop titulado *Optimization by Building and Using Probabilistic Models*. *OBUPM'2002* dentro del congreso GECCO'2002. Nueva York 2002.
- *International Symposium on Adaptive Systems. Evolutionary Computation and Probabilistic Graphical Models. ISAS1999* La Habana 1999.
- *International Symposium on Adaptive Systems. Evolutionary Computation and Probabilistic Graphical Models. ISAS2001* La Habana 2001.