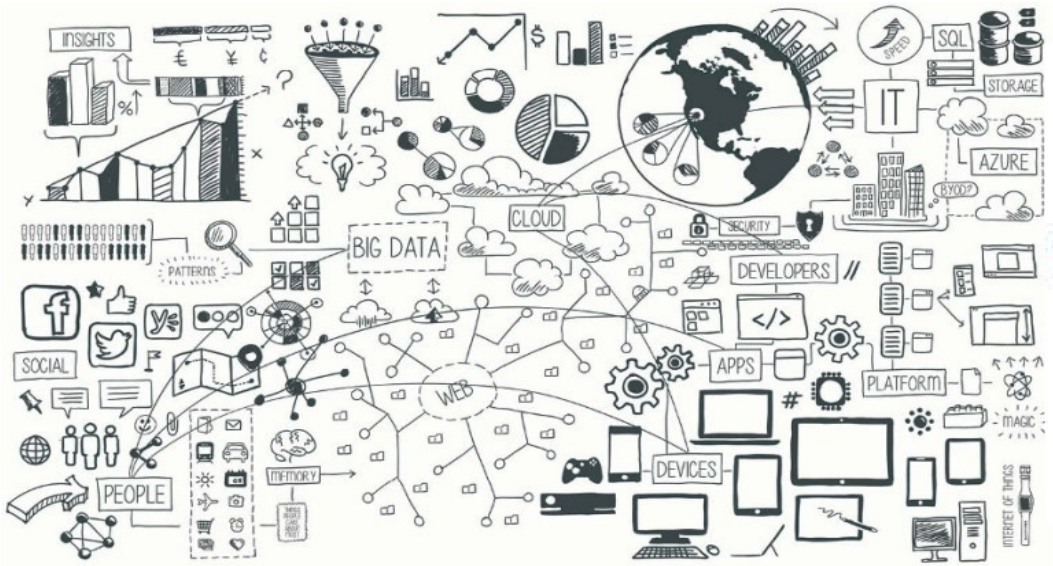
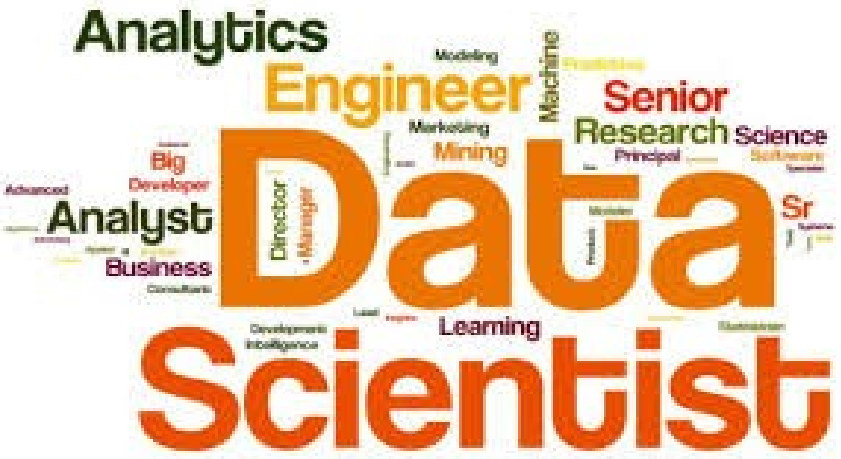
M1970 – Machine Learning II
Redes Probabilísticas Discretas





Sixto Herrera (sixto.herrera@unican.es)
Mike Legasa

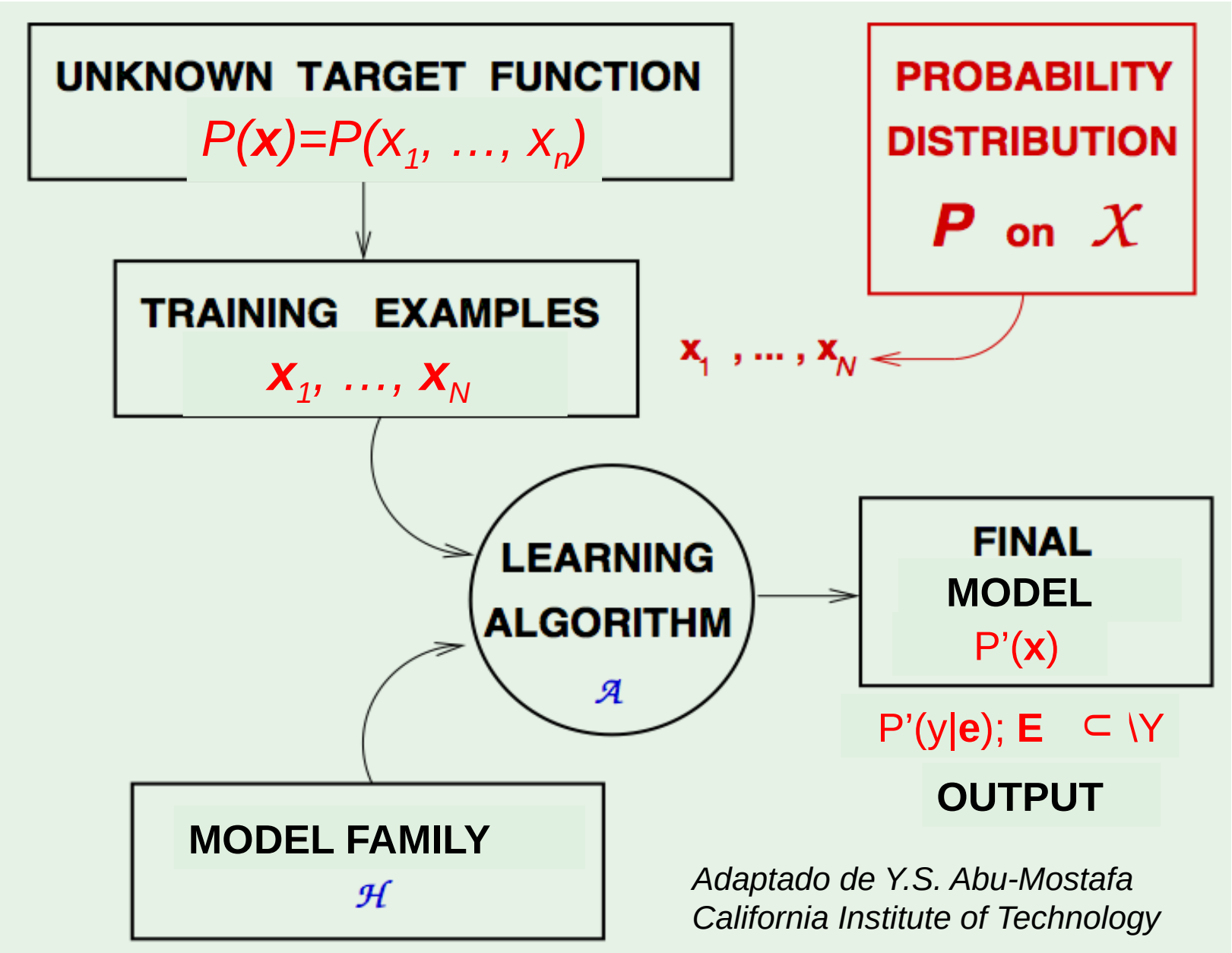
Grupo de Meteorología

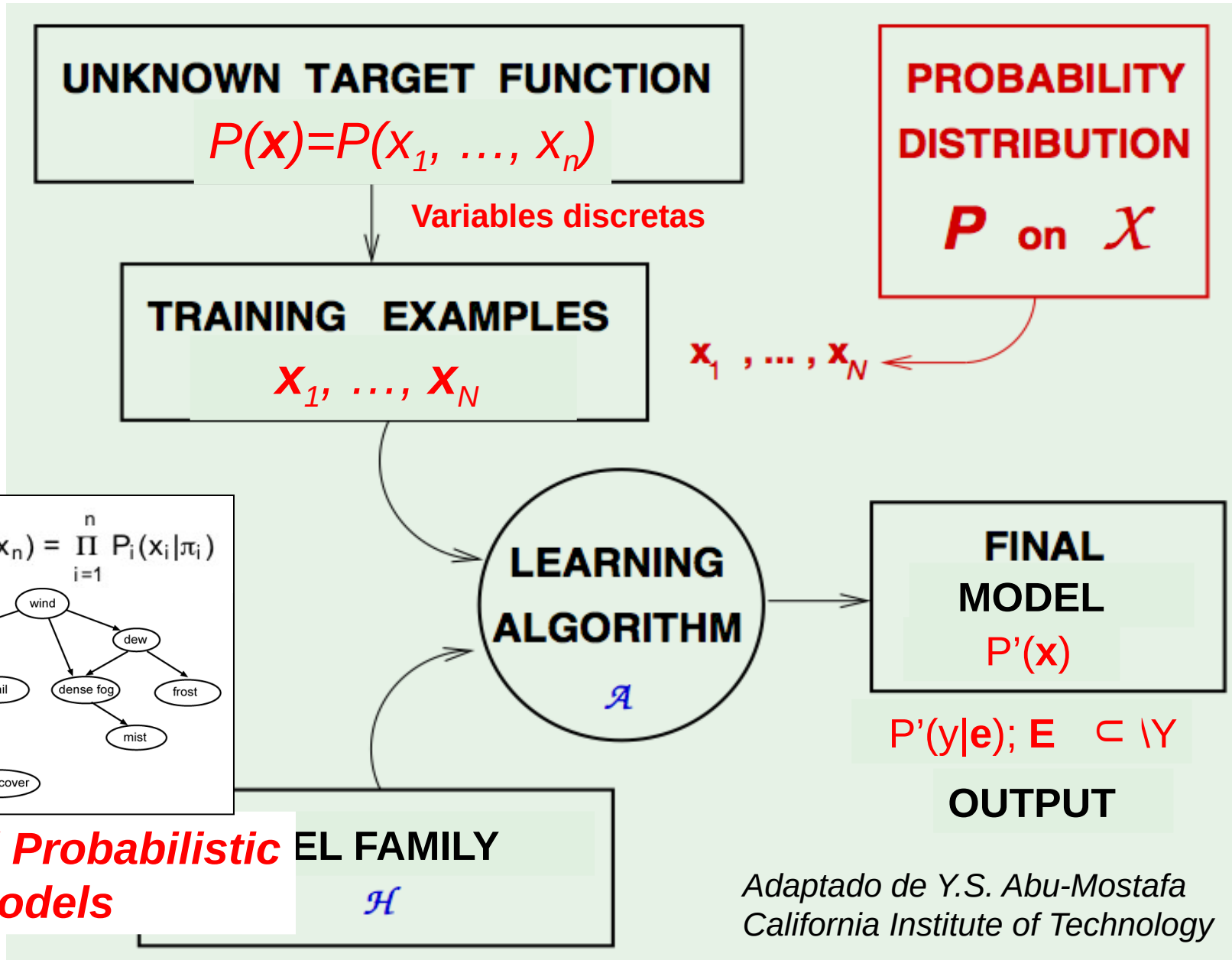
Univ. de Cantabria – CSIC
MACC / IFCA

M1970 – Machine Learning (L 15:30-17:30; X 15:30-17:30)

Feb	**28**	**L**	**Redes Probabilísticas Discretas (2h-T)**
Mar	**2**	**X**	**Redes Bayesianas: Creación e Inferencia (2h-L)**
	7	**L**	**Clasificacidores Bayesianos. Naive Bayes (2h-L)**
	9	**X**	**Redes Bayesianas: Aprendizaje Estructural (2h-T)**
	14	**L**	**Redes Bayesianas: Aprendizaje Paramétrico (2h-LT)**
	16	**X**	**Redes Bayesianas: Aprendizaje (2h-L)**
	21	**L**	**Evaluación (2h)**

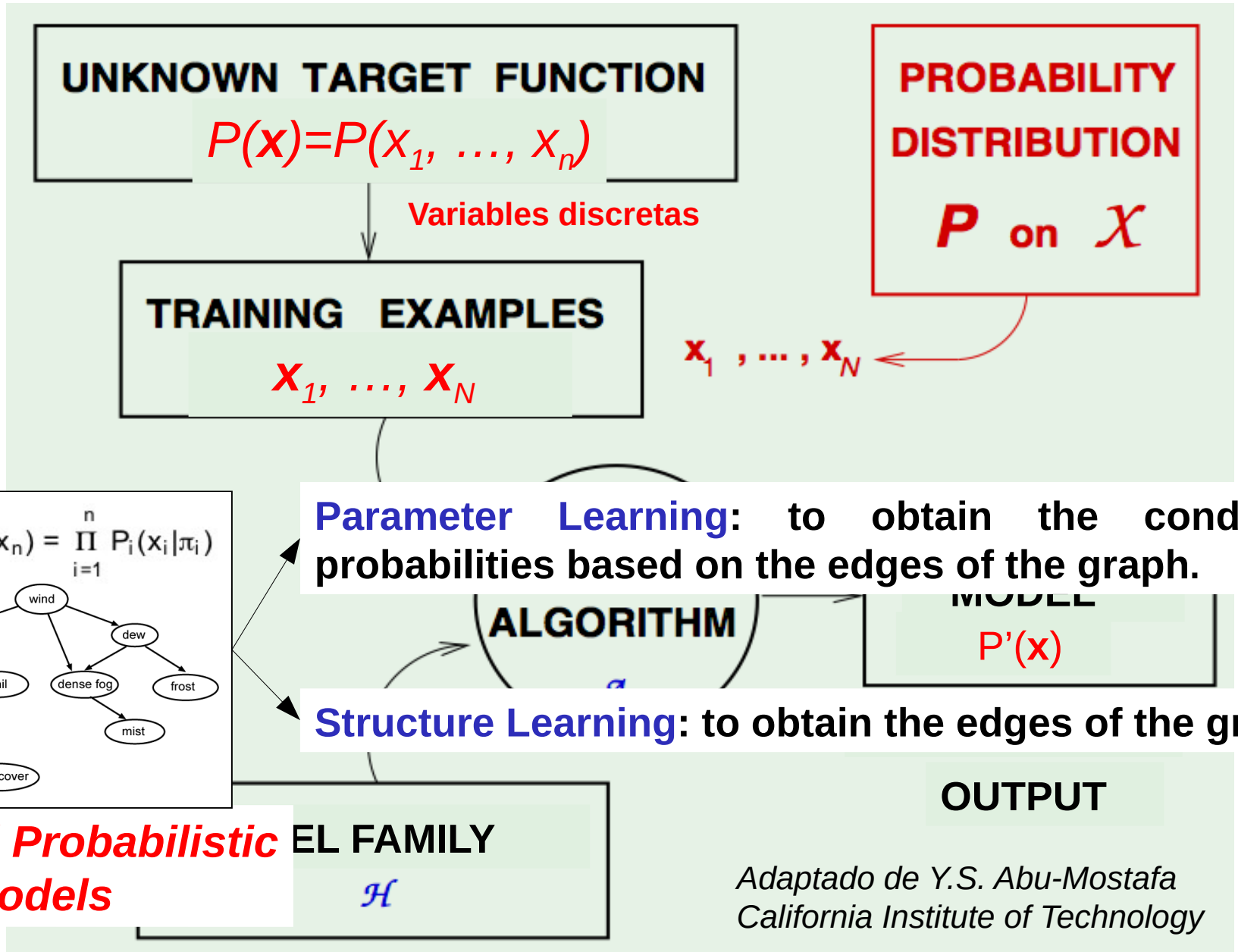
NOTA: Las líneas de código de R en esta presentación se muestran sobre un fondo gris.


UNKNOWN TARGET FUNCTION
$P(x)=P(x_1, \ldots, x_n)$
PROBABILITY DISTRIBUTION
P on X
TRAINING EXAMPLES
$x_1, \ldots, x_N$
$x_1, \ldots, x_N$
LEARNING ALGORITHM
$\mathcal{A}$
FINAL MODEL
$P'(x)$
$P'(y|e)$; $E \subset \backslash Y$
OUTPUT
MODEL FAMILY
$\mathcal{H}$
Adaptado de Y.S. Abu-Mostafa
California Institute of Technology



Graphical Probabilistic Models

Adaptado de Y.S. Abu-Mostafa
California Institute of Technology

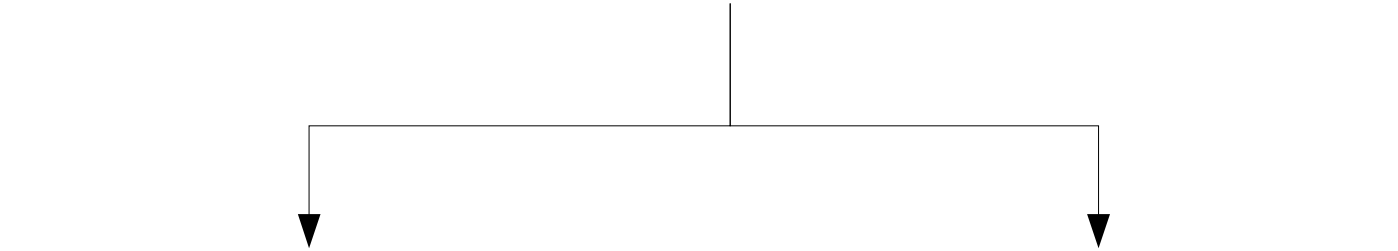


Graphical Probabilistic Models

Adaptado de Y.S. Abu-Mostafa
California Institute of Technology

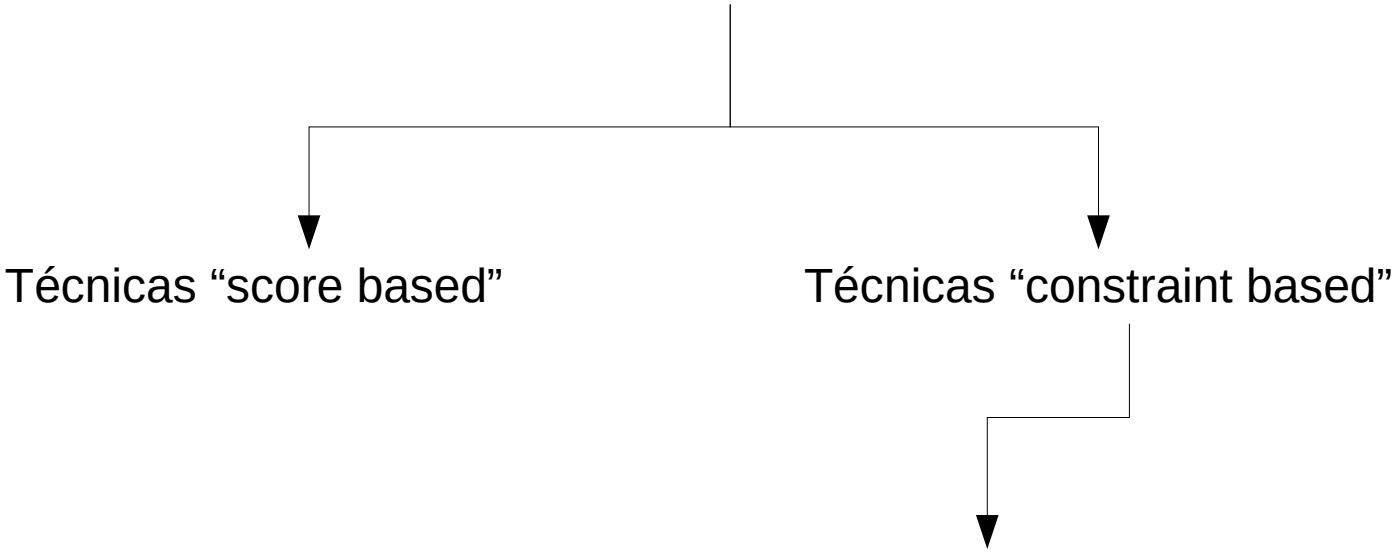
Aprendizaje Estructural:

Aprender la estructura (grafo) a partir del dataset



Aprendizaje Estructural:

Aprender la estructura (grafo) a partir del dataset



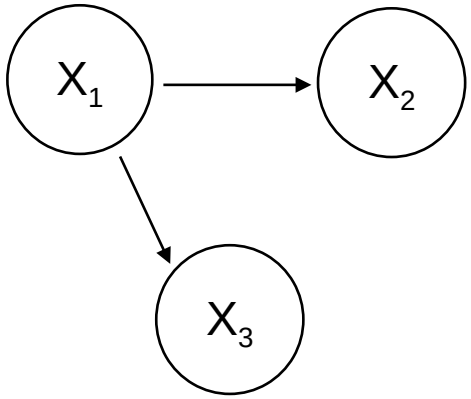
· Determinar las independencias a nivel local.
  · Utilizando tests de independencia.
  · A menudo primero buscando la Markov Blanket

· Algoritmos (Específicos), basados en Inductive Causation:
  PC, Grow-Shrink, Incremental Association Markov Blanket...

D-separation concept for directed graphs enrich the representativity of the model → **Moral graph.**

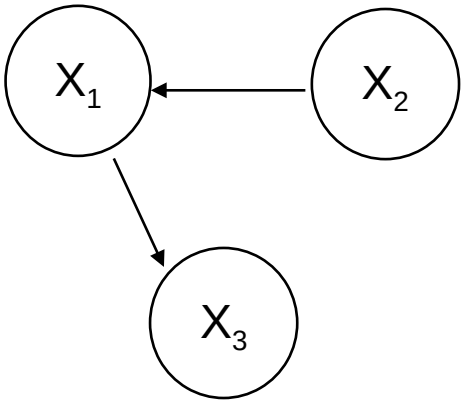
Two directed graph are **equivalents** when they lead to the same probabilistic model:

$$P(X_1, X_2, X_3) = P(X_1)P(X_2|X_1)P(X_3|X_1) = P(X_1, X_2)P(X_3|X_1)$$



Equivalents

$$P(X_1, X_2, X_3) = P(X_2)P(X_1|X_2)P(X_3|X_1) = P(X_1, X_2)P(X_3|X_1)$$



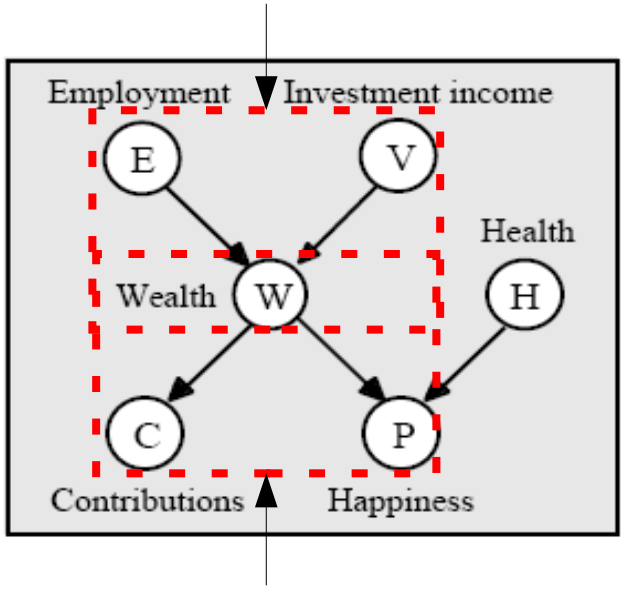
D-separation concept for directed graphs enrich the representativity of the model → **Moral graph.**

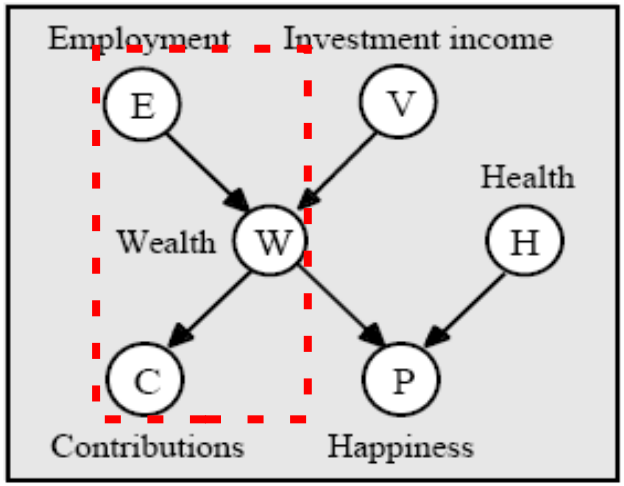
Two directed graph are **equivalents** when they lead to the same probabilistic model.

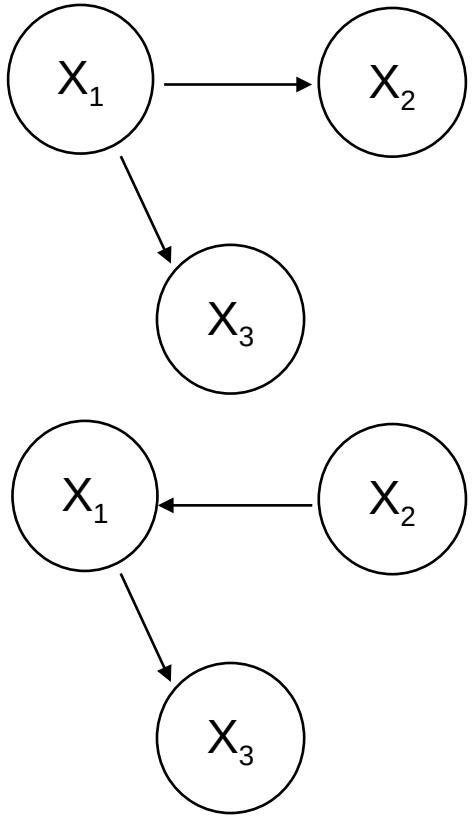
This occurs when the **subyacent non-directed graph** is the same and include the same **V-structures**.

$$P(X_1, X_2, X_3) = P(X_1, X_2)\,P(X_3 | X_1)$$

Common effect







Common cause

Indirect evidential/causal effect

Algoritmo Inductive Causation

1) **(Identificar pares dependientes)**
1.1) Para cada par A, B buscar $S \subset X$ t.q. $A \perp B \mid S$.
1.2) Si $\nexists$ S, colocar un arco no dirigido A — B

Algoritmo Inductive Causation

1) **(Identificar pares dependientes)**
1.1) Para cada par A, B buscar $S \subset X$ t.q. $A \perp B \mid S$.
1.2) Si $\nexists$ S, colocar un arco no dirigido A — B

2) **(Identificar v-estructuras)**
2.1) Para cada par A, B no adyacentes con vecino común C.
2.2) Si $C \notin S$: dirigir A — C, C — B como $A \rightarrow C \leftarrow B$.
2.3) Si $C \in S$: no hacer nada.

Algoritmo Inductive Causation

1) **(Identificar pares dependientes)**
1.1) Para cada par A, B buscar $S \subset X$ minimal t.q. $A \perp B \mid S$.
1.2) Si $\nexists$ S, colocar un arco no dirigido A — B

2) **(Identificar y construir v-estructuras)**
2.1) Para cada par A, B no adyacentes con vecino común C.
2.2) Si $C \notin S$: dirigir A — C, C — B como $A \rightarrow C \leftarrow B$.
2.3) Si $C \in S$: no hacer nada.

3) **(Dirigir arcos)** Repetir para los arcos no dirigidos:
3.1) Si A — B y hay un camino estrictamente dirigido de A a B:
Fijar dirección de A — B a $A \rightarrow B$.
3.2) Si A y B no son adyacentes pero $\exists C$ t.q $A \rightarrow C$ y C — B:
Fijar dirección de C — B a $C \rightarrow B$.

4. **Devolver** el grafo **(Partially Directed Acyclic Graph).**

Algoritmo Inductive Causation

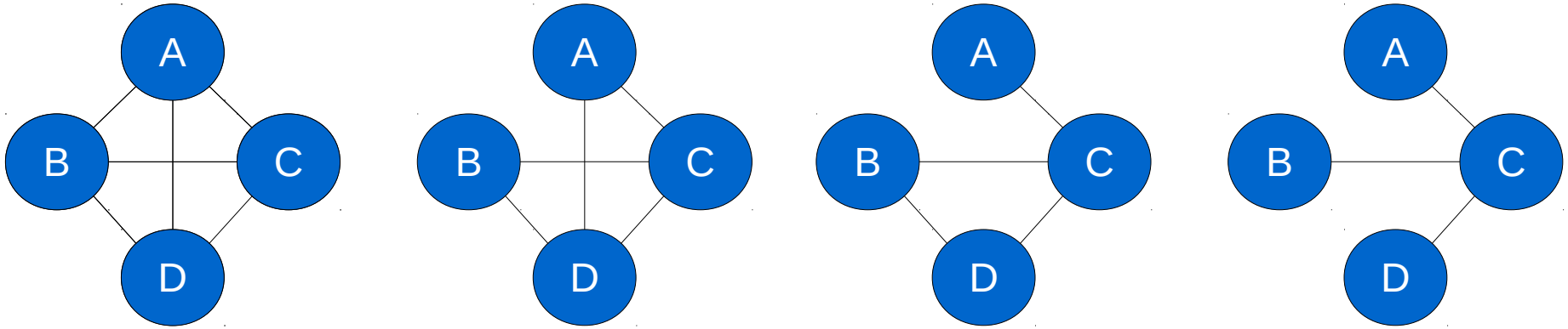
(todos dependientes)

$A \perp B$

$A \perp D \mid C$

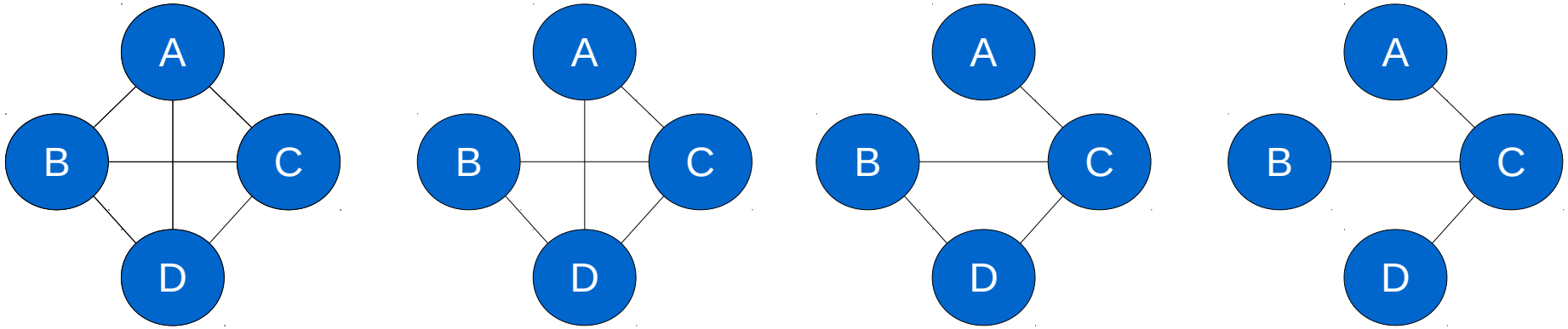
$B \perp D \mid C$

1)

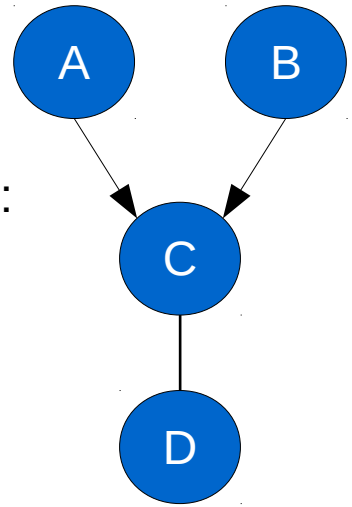


Algoritmo Inductive Causation

1) (todos dependientes) $A \perp B$ $A \perp D \mid C$ $B \perp D \mid C$



2) V-estructura sobre C:



Algoritmo Inductive Causation

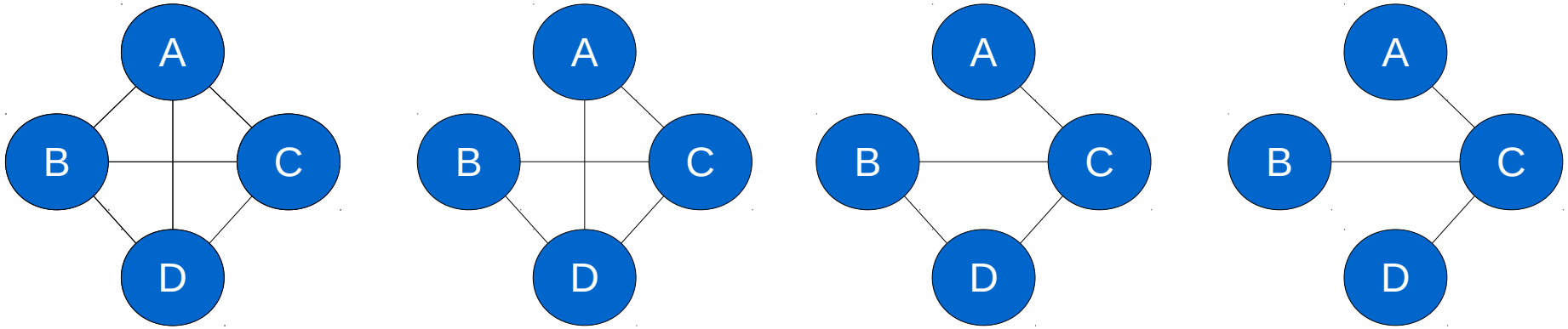
(todos dependientes)

$A \perp B$

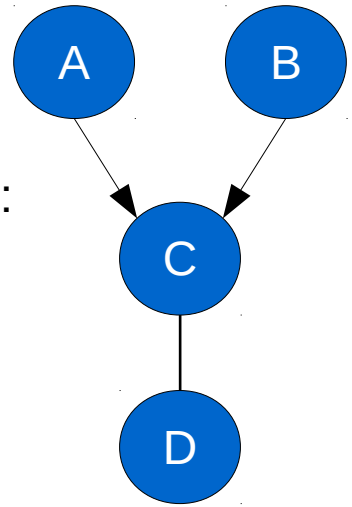
$A \perp D \mid C$

$B \perp D \mid C$

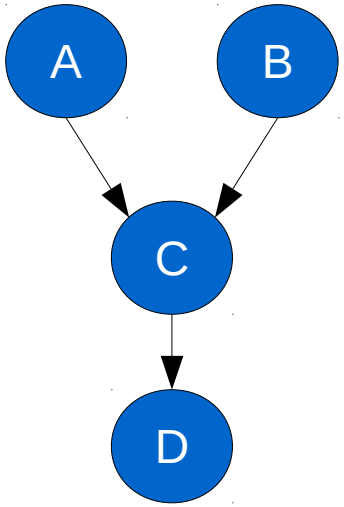
1)



2) V-estructura sobre C:



3.2)



Test de Independencia

Test χ^2 :

$$\mathrm{X}^2(X, Y \mid \mathbf{Z}) = \sum_{i=1}^{R}\sum_{j=1}^{C}\sum_{k=1}^{L}\frac{(n_{ijk} - m_{ijk})^2}{m_{ijk}}, \quad \text{where} \quad m_{ijk} = \frac{n_{i+k}n_{+jk}}{n_{++k}}.$$

Para las L combinaciones de padres **Z** y los R, C estados de X,Y, respectivamente

Otros :

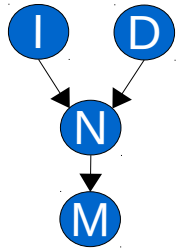
Mutual Information, correlación...

Ejercicios:

1) En el algoritmo Inductive Causation, ¿Por qué es necesario el paso 3.1?

2) En el algoritmo Inductive Causation, ¿Por qué es necesario el paso 3.2?

3) ¿Puede aplicarse Inductive Causation para un problema suficientemente grande?

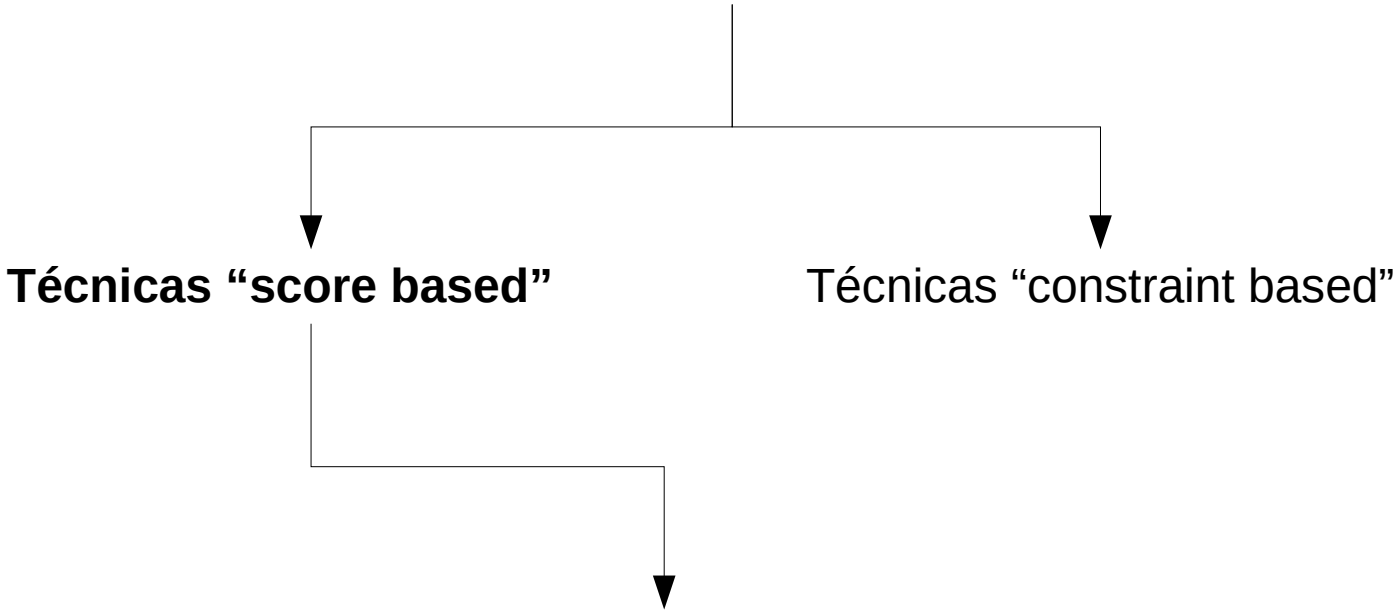
4) Recorre manualmente los pasos del algoritmo IC para construir un DAG con 4 nodos: Inteligencia (I), Dificultad de Examen (D), Nota (N), Nota Media (M). Debe construir el DAG 

.

5) **Bnlearn** implementa el algoritmo PC, función *pc.stable()*, aplícalo al dataset survey utilizando el argumento *debug* = *TRUE*, explora los argumentos *alpha, max.sx.*

Aprendizaje Estructural:

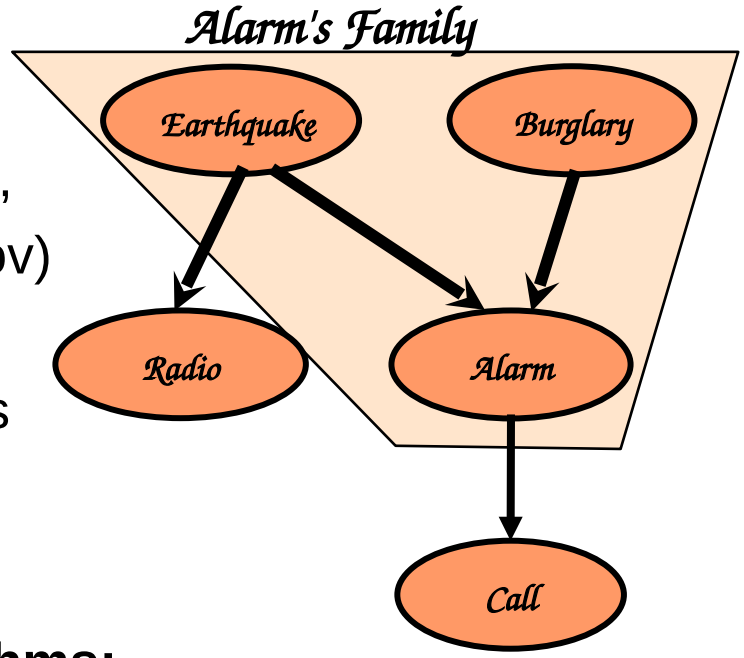
Aprender la estructura (grafo) a partir del dataset



1) Determinar "score", medida de bondad de ajuste.
2) Maximizar el "score".

· Algoritmos: (¡No específicos de RB!)
Cualquiera de optimización, e.g. hill-climbing.

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences

- **Constraint-Based Algorithms:**
  - **IC, PC,** Grow-Shrink **(GS),** Incremental Association **(IAMB),** etc...
- **Score-Based Algorithms:**
  - Hill-Climbing, tabu, **K2, B,** etc...

Find DAG that maximize a **predefined score**

For each step **search** the edge contributing to the score

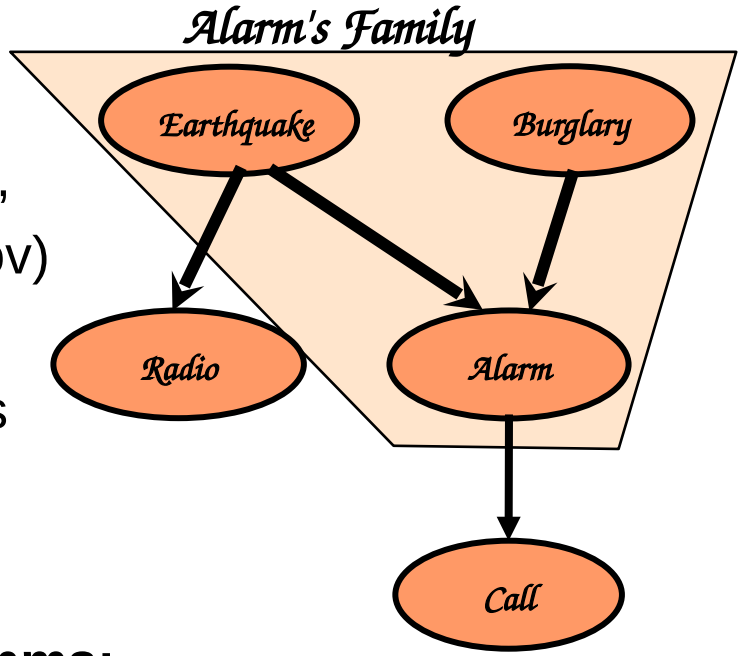
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



- **Constraint-Based Algorithms:**
  - **IC, PC,** Grow-Shrink **(GS),** Incremental Association **(IAMB),** etc...
- **Score-Based Algorithms:**
  - Hill-Climbing, tabu, **K2, B,** etc...

Find DAG that maximize a **predefined score** → **Which score?**

For each step **search** the edge contributing to the score → **How to search?**

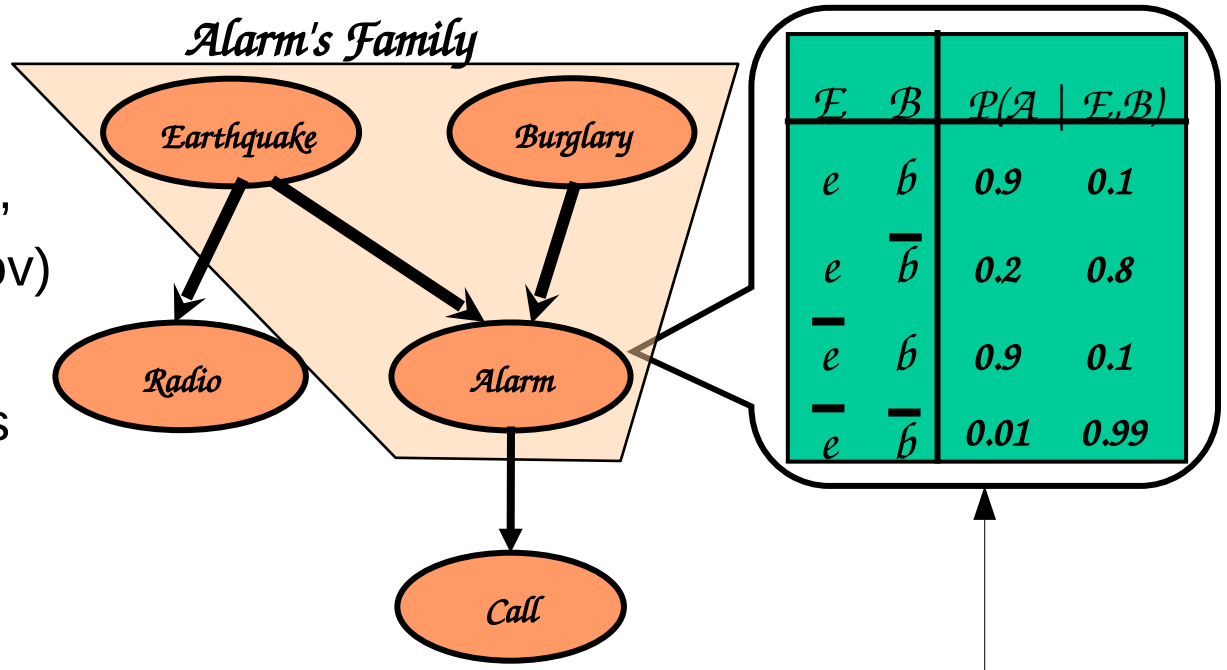
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



Factorization of the joint probability function.

$$P(B,E,A,R,C)=P(E)\,P(B)\,P(R|E)\,P(A|E,B)\,P(C|A)$$

Parameters: Probabilities and tables.

To learn the Bayesian Network consists in the estimation of the dependence structure and the parameters based on the training sample.

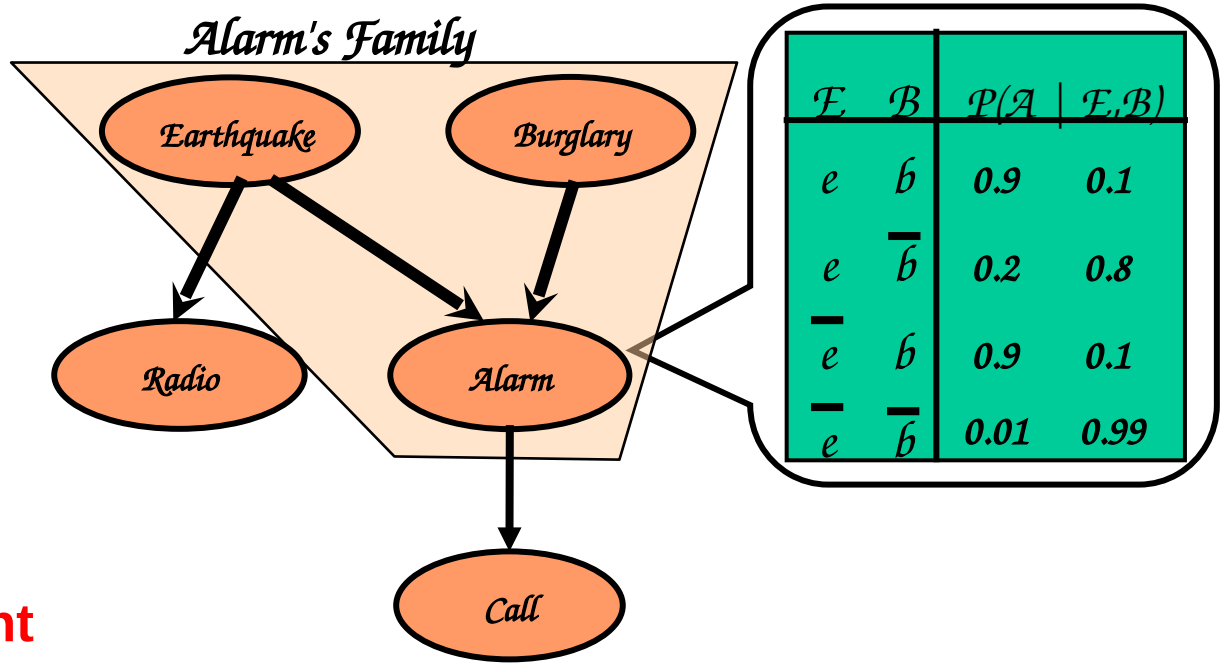
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Score:

P(G) - "A priori" information

P(data|G) – Explain capability

Model complexity



Bayesian Dirichlet Equivalent

$$P(G|data)=\frac{P(data|G)\,P(G)}{P(data)}\Rightarrow BDe=\log(P(data|G))+\log(P(G))$$

Minimum Description Length // Bayesian Information Criterium

$$BIC(G,data)=MDL(G|data)=\sum_{i=1}^{p}\left(\log(Pr(X_i|\Pi_{x_i}))-\frac{|\theta_{x_i}|*\log(N)}{2}\right)$$

$$MDL(G|data)\rightarrow BDe(G|data)$$ ← **Converges asymptotically (N)**

Score de bondad de ajuste

G que maximice $P(G \mid \mathcal{D})$

$$P(G \mid \mathcal{D}) = \frac{P(G, \mathcal{D})}{P(\mathcal{D})} = \frac{P(\mathcal{D} \mid G)P(G)}{P(\mathcal{D})}.$$

Score de bondad de ajuste

G que maximice $P(G \mid \mathcal{D})$

$$P(G \mid \mathcal{D}) = \frac{P(G, \mathcal{D})}{P(\mathcal{D})} = \frac{P(\mathcal{D} \mid G)P(G)}{P(\mathcal{D})}.$$

G que maximice $P(\mathcal{D} \mid G)$

¿Cómo calcularlo?

Utilizando la factorización del propio grafo

G que maximice $P(\mathcal{D} \mid G)$

$$\mathrm{BIC}(G, \mathcal{D}) = \sum_{i=1}^{p} \left[\log \Pr(X_i \mid \Pi_{X_i}) - \frac{|\Theta_{X_i}|}{2} \log n \right]$$

Con:

$|\mathcal{D}| = n.$

$|\Theta_{X_i}|$ el número de parámetros.

Score de bondad de ajuste, estimándolo...

G que maximice $P(G \mid \mathcal{D})$

Ejemplo:

De:

$$\Pr(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T}) = \Pr(\mathtt{A}) \Pr(\mathtt{S}) \Pr(\mathtt{E} \mid \mathtt{A}, \mathtt{S}) \Pr(\mathtt{O} \mid \mathtt{E}) \Pr(\mathtt{R} \mid \mathtt{E}) \Pr(\mathtt{T} \mid \mathtt{O}, \mathtt{R})$$

Obtenemos:

Score de bondad de ajuste, estimándolo...

G que maximice $P(G \mid \mathcal{D})$

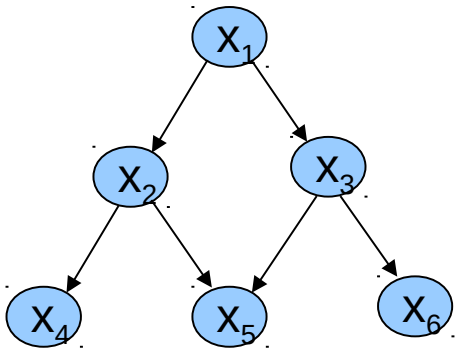
Ejemplo:

De:

$$\Pr(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T}) = \Pr(\mathtt{A}) \Pr(\mathtt{S}) \Pr(\mathtt{E} \mid \mathtt{A}, \mathtt{S}) \Pr(\mathtt{O} \mid \mathtt{E}) \Pr(\mathtt{R} \mid \mathtt{E}) \Pr(\mathtt{T} \mid \mathtt{O}, \mathtt{R})$$

Obtenemos:

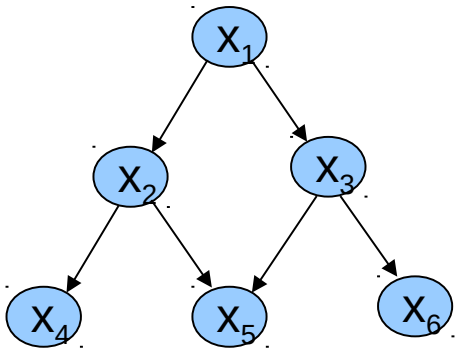
$$\begin{aligned}
\mathrm{BIC} &= \log \widehat{\Pr}(\mathtt{A}, \mathtt{S}, \mathtt{E}, \mathtt{O}, \mathtt{R}, \mathtt{T}) - \frac{d}{2} \log n = \\
&= \left[\log \widehat{\Pr}(\mathtt{A}) - \frac{d_{\mathtt{A}}}{2} \log n\right] + \left[\log \widehat{\Pr}(\mathtt{S}) - \frac{d_{\mathtt{S}}}{2} \log n\right] + \\
&+ \left[\log \widehat{\Pr}(\mathtt{E} \mid \mathtt{A}, \mathtt{S}) - \frac{d_{\mathtt{E}}}{2} \log n\right] + \left[\log \widehat{\Pr}(\mathtt{O} \mid \mathtt{E}) - \frac{d_{\mathtt{O}}}{2} \log n\right] + \\
&+ \left[\log \widehat{\Pr}(\mathtt{R} \mid \mathtt{E}) - \frac{d_{\mathtt{R}}}{2} \log n\right] + \left[\log \widehat{\Pr}(\mathtt{T} \mid \mathtt{O}, \mathtt{R}) - \frac{d_{\mathtt{T}}}{2} \log n\right]
\end{aligned}$$



Score de bondad de ajuste, estimándolo...

x_1	$p(x_1)$
0	0.3
1	0.7

$p(0,1,1,1,0,0) =$
$p(x_1=0)$

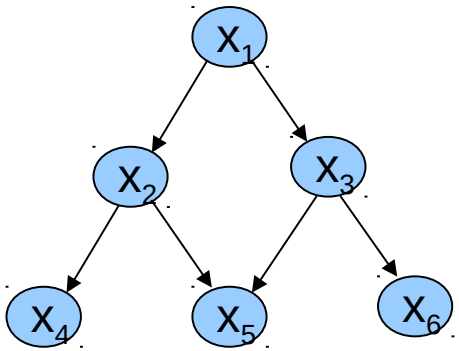


Score de bondad de ajuste, estimándolo...

x_1	$p(x_1)$
0	0.3
1	0.7

x_1	x_2	$p(x_2\|x_1)$
0	0	0.4
0	1	0.6
1	0	0.1
1	1	0.9

$p(0,1,1,1,0,0) =$
$p(x_1=0)p(x_2=1|x_1=0)$



Score de bondad de ajuste, estimándolo...

x_1	$p(x_1)$
0	0.3
1	0.7

x_1	x_2	$p(x_2\|x_1)$
0	0	0.4
0	1	0.6
1	0	0.1
1	1	0.9

x_1	x_3	$p(x_3\|x_1)$
0	0	0.2
0	1	0.8
1	0	0.5
1	1	0.5

x_2	x_4	$p(x_4\|x_2)$
0	0	0.3
0	1	0.7
1	0	0.2
1	1	0.8

x_3	x_6	$p(x_6\|x_3)$
0	0	0.1
0	1	0.9
1	0	0.4
1	1	0.6

x_2	x_3	x_5	$p(x_5\|x_2,x_3)$
0	0	0	0.4
0	0	1	0.6
0	1	0	0.5
0	1	1	0.5
1	0	0	0.7
1	0	1	0.3
1	1	0	0.2
1	1	1	0.8

$p(0,1,1,1,0,0) =$
$p(x_1=0)p(x_2=1|x_1=0)p(x_3=1|x_1=0)p(x_4=1|x_2=1)$
$p(x_6=0|x_3=1)p(x_5=0|x_2=1,x_3=1) =$

$0.3 \times 0.6 \times 0.8 \times 0.8 \times 0.4 \times 0.2 = \mathbf{0.009216}$

Ejercicios:

5) En los algoritmos score-based se utilizan tres operadores para los arcos: adición, borrado e inversión. ¿Por qué es necesario este último?

6) Explica el término final en la expresión del BIC:

¿Qué pasaría si este término no existiera?

7) **Bnlearn** implementa el algoritmo hill-climbing, función *hc()*, aplícalo al dataset survey utilizando el argumento *debug* = *TRUE.* Explora los argumentos *start, maxp* y *k.*

Referencias de ampliación:

D. Koller, N. Friedman: *Probabilistic Graphical Models*

D. Heckerman, D. Geiger, D. M. Chickering: *Learning Bayesian Networks: The combination of Knowledge and Statistical Data*

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

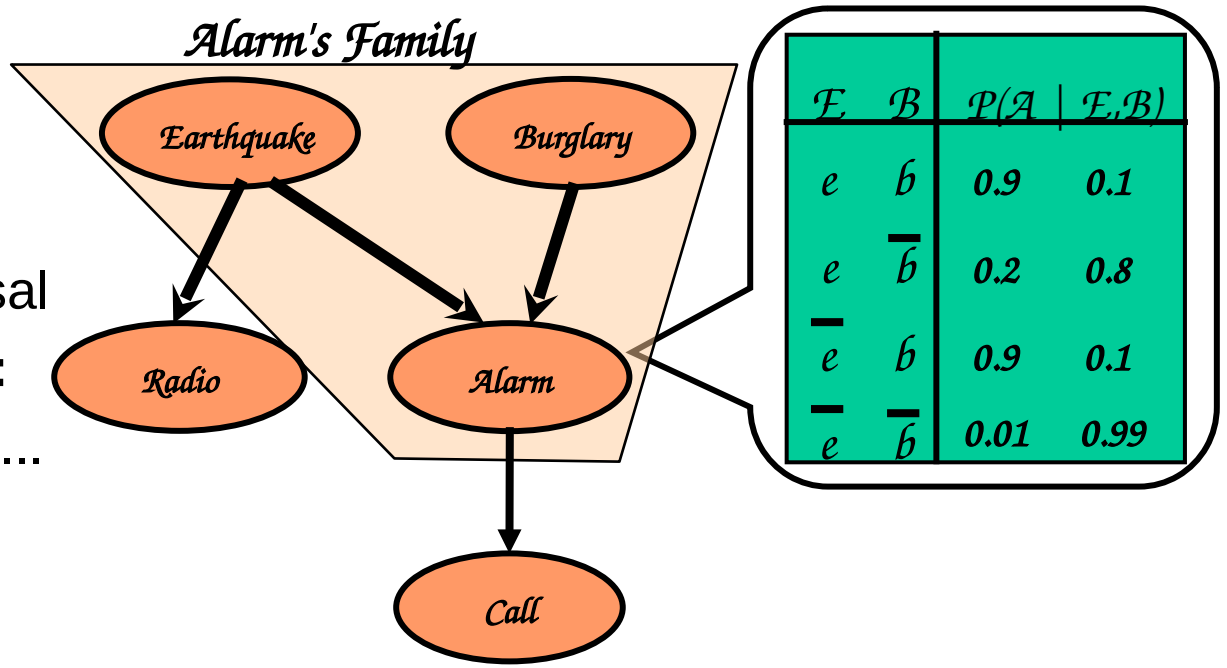
Search:

Operators:

Edge addition, deletion, reversal

Techniques (global or local):

Hill-climbing, tabu, K2, B, etc...



Algorithm:

1) Start with a given network.

2) Obtains the score for all possible changes (**greedy**).

3) Applies change that most improves the score.

4) Repeat **2)** and **3)** until no modification improves the score.

To obtain the score for all possible changes is computationally expensive

Algoritmo Hill-Climbing

1) Inicializar grafo G (e.g. G = Ø)

2) maxscore ← score(G)

Algoritmo Hill-Climbing

1) Inicializar grafo G (e.g. G = Ø)

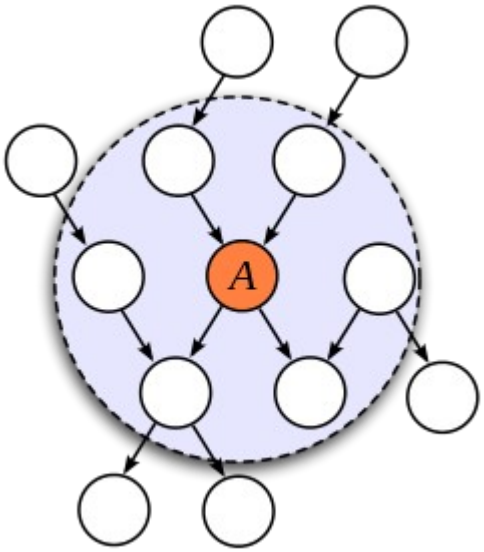
2) maxscore ← score(G)

3) Repetir **mientras** maxscore aumente:
   3.1) **Para cada** operador ω(G):
      3.1.1) G' ← ω(G)
      3.1.2) **Si** score(G') > score(G)
         maxscore ← score(G')
         G ← G'

4) Devolver G **(Directed Acyclic Graph)**

Operadores ω():
- a) añadir arco
- b) borrar arco
- c) invertir arco

D-separation concept for directed graphs enrich the representativity of the model → **Moral graph.**
Two directed graph are **equivalents** when they lead to the same probabilistic model.
This occurs when the **subyacent non-directed graph** is the same and include the same **V-structures**.
The **Skeleton** of the graph is the undirected graph underlying.
The **Markov Blanket** of a node ***A*** is the set of nodes that completely separates ***A*** from the rest of the graph. In particular, it includes the parents and childrens of the node ***A***, and those children's other parents.



The **Markov Blanket** of is the set of nodes that includes all the knowledge needed to do inference on the node ***A***, from estimation to hypothesis testing to prediction.

Source: Image from https://en.wikipedia.org/wiki/Markov_blanket

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

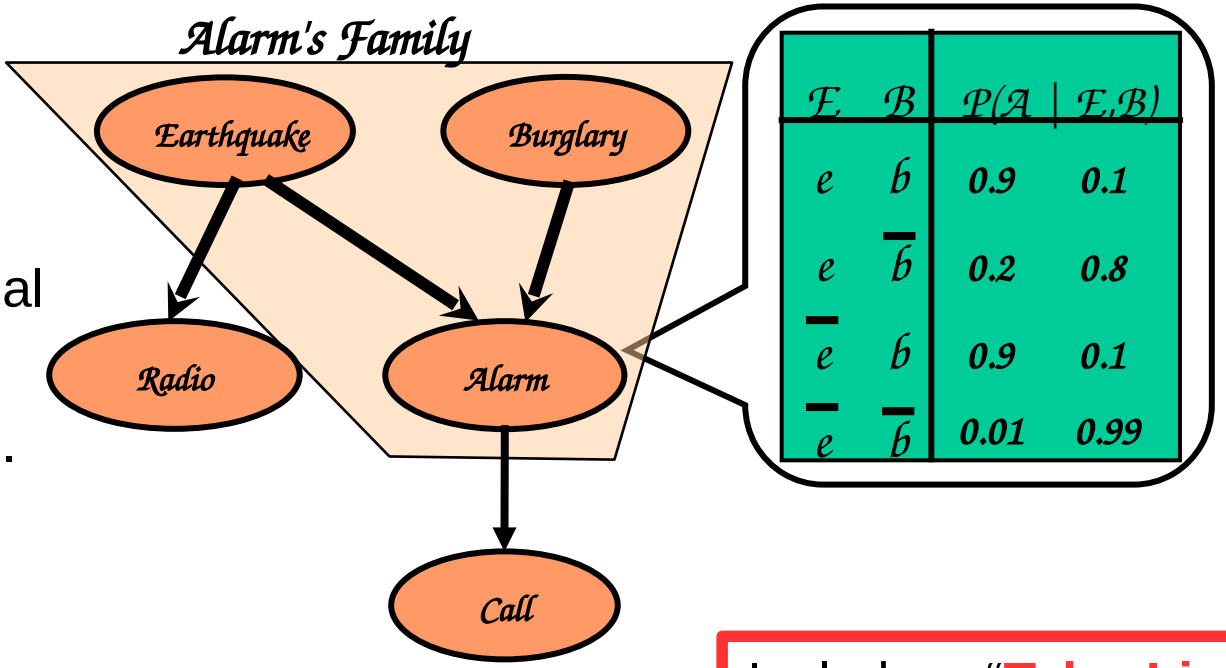
Search:

Operators:

Edge addition, deletion, reversal

Techniques (global or local):

Hill-climbing, **tabu**, K2, B, etc...

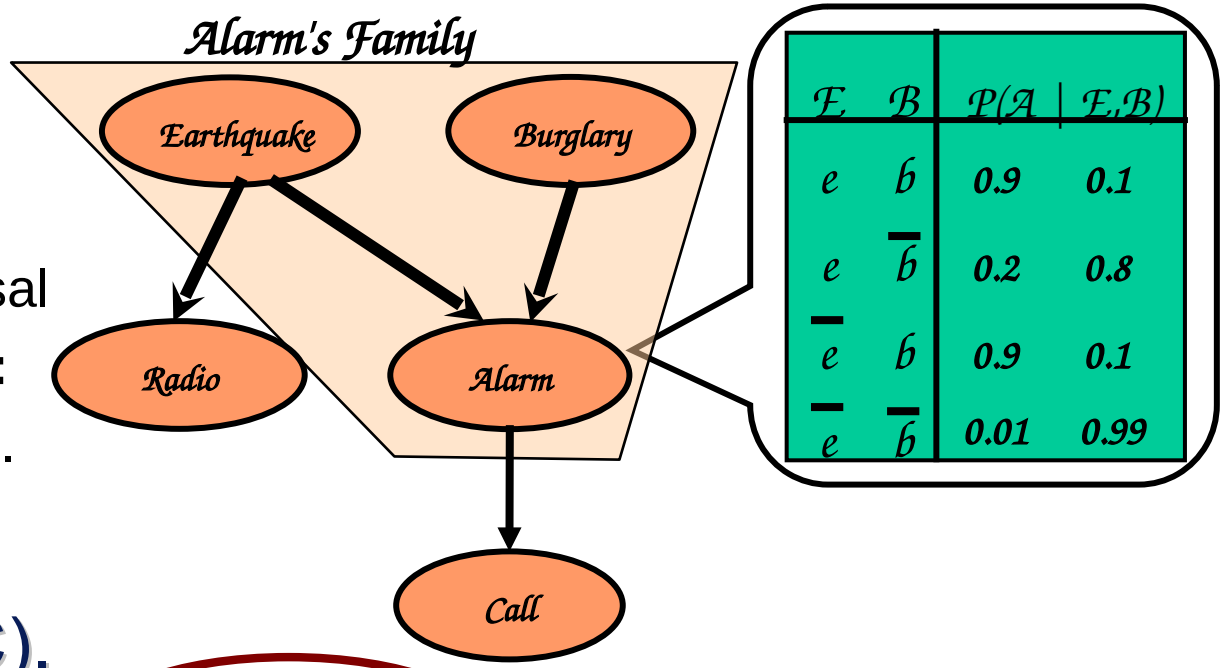


Algorithm:

1) Start with a given network.
2) Obtains the score for all possible changes (**greedy**).
3) Applies change that most improves the score.
4) Repeat **2)** and **3)** until no modification improves the score.

Include a "**Tabu List**": Keep a list of K steps most recently taken which cannot be reversed.

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



Search:

Operators:

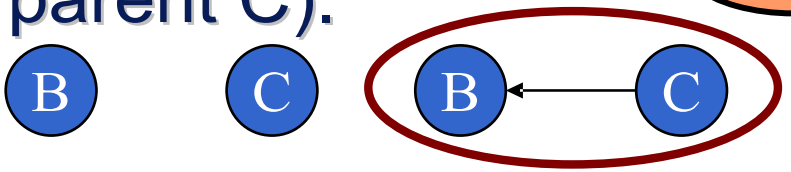
Edge addition, deletion, reversal

Techniques (global or local):

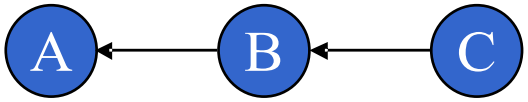
Hill-climbing, tabu, **K2**, B, etc...

A<B<C

B (possible parent C).



The model:



Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

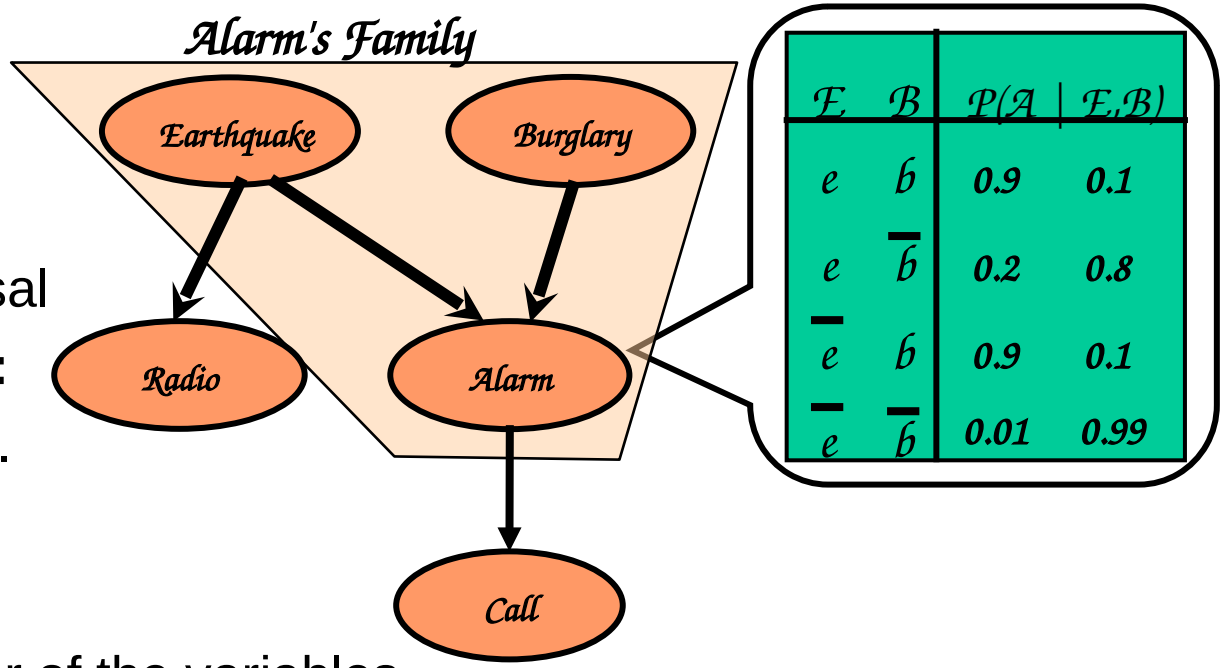
Search:

Operators:

Edge addition, deletion, reversal

Techniques (global or local):

Hill-climbing, tabu, K2, B, etc...



Algorithm:

1) **Randomly chosen** an order of the variables.

2) Obtains the score for **all changes fulfilling some local criteria**.

3) Applies change that most improves the score.

4) Repeat **2)** and **3)** until no modification improves the score.

To obtain the score for all possible changes is computationally expensive

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Search:

Operators:

Edge addition, deletion, reversal

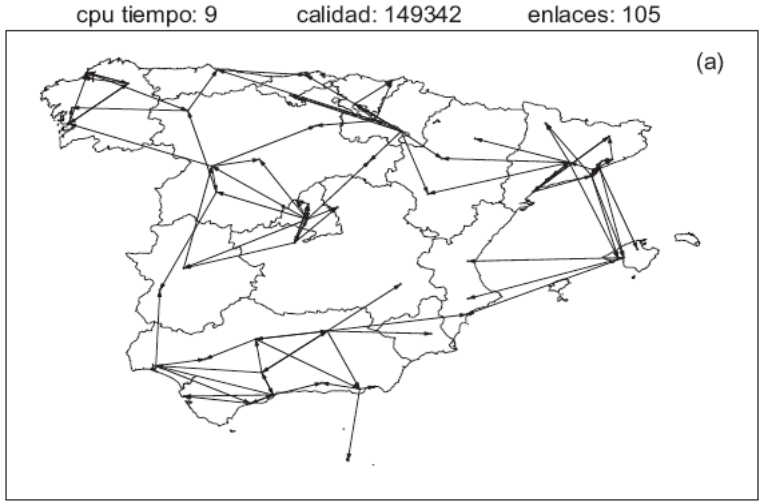
Techniques (global or local):

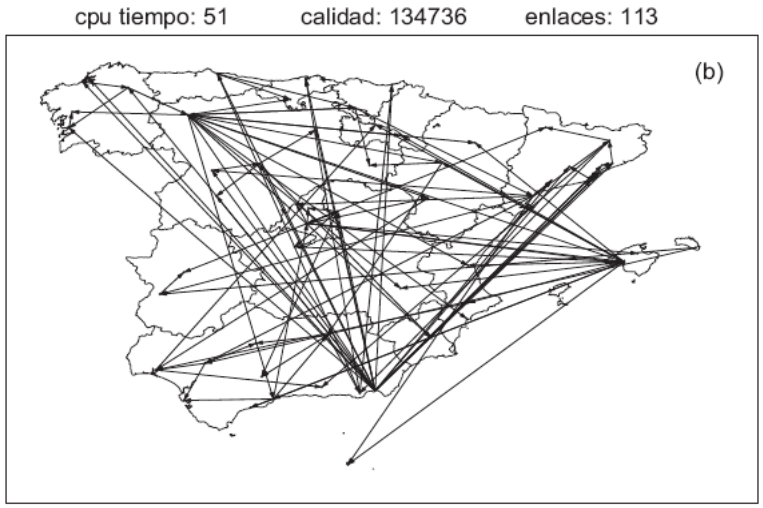
Hill-climbing, tabu, K2, B, etc...

Local search algorithm

1. Maximum number of parents per node
2. Ordering (if K2)
3. For every node
4. Dependency vector (MI or correlation)
5. Selection of potential parents
6. For every potential parent
7. DAG control (if B)
8. Quality measure (MDL or RSA)
9. Stop criterion
10. Take de best link

Sparse Candidate, Friedman (99)





Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

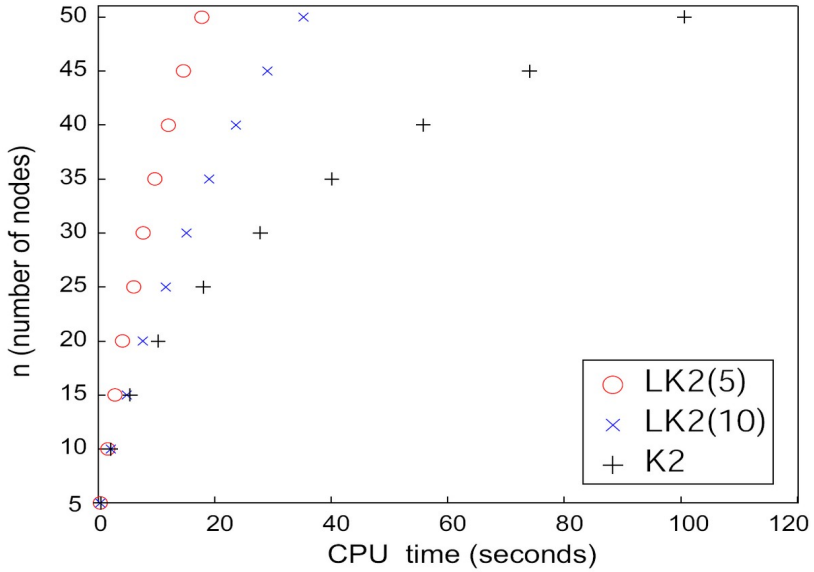
Search:

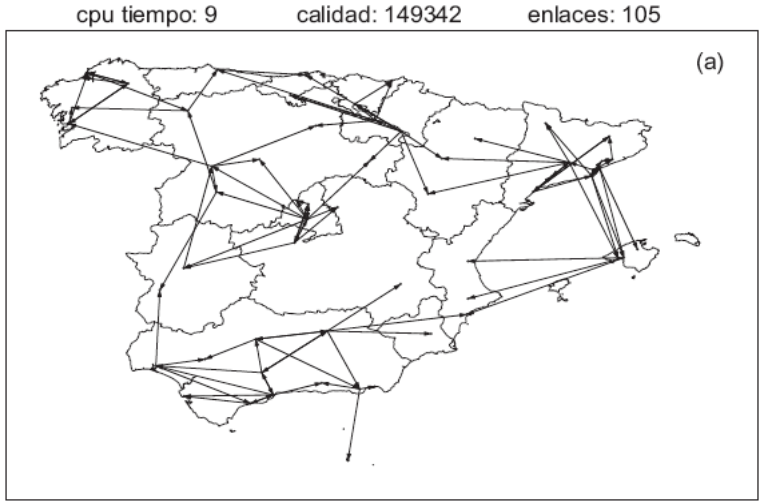
Operators:

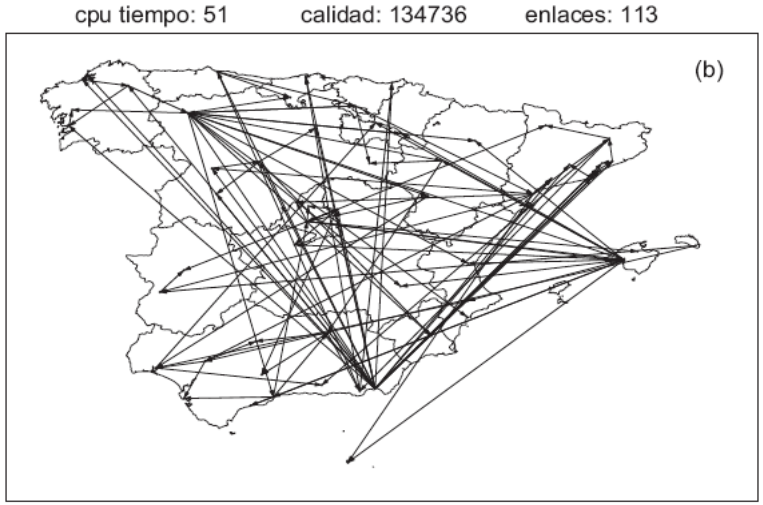
Edge addition, deletion, reversal

Techniques (global or local):

Hill-climbing, tabu, K2, B, etc...







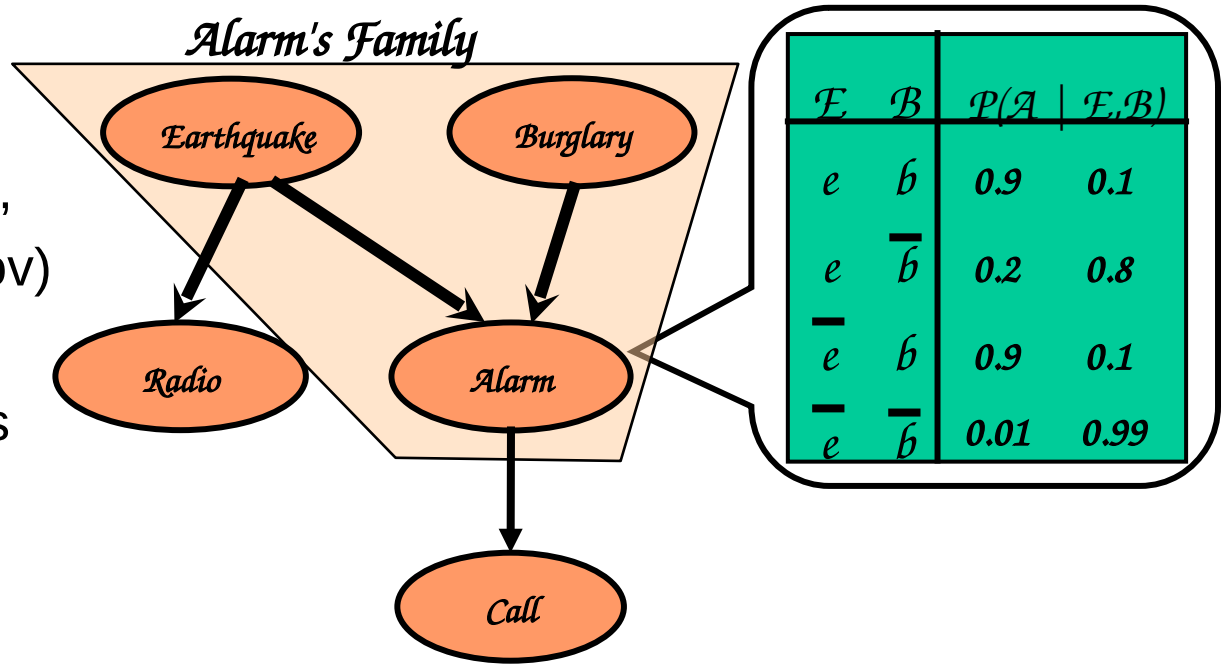
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



Factorization of the joint probability function.

$$P(B,E,A,R,C)=P(E)\,P(B)\,P(R|E)\,P(A|E,B)\,P(C|A)$$

Parameters: Probabilities and tables.

- Maximum likelihood estimation (**MLE**)
- Bayesian Estimation.

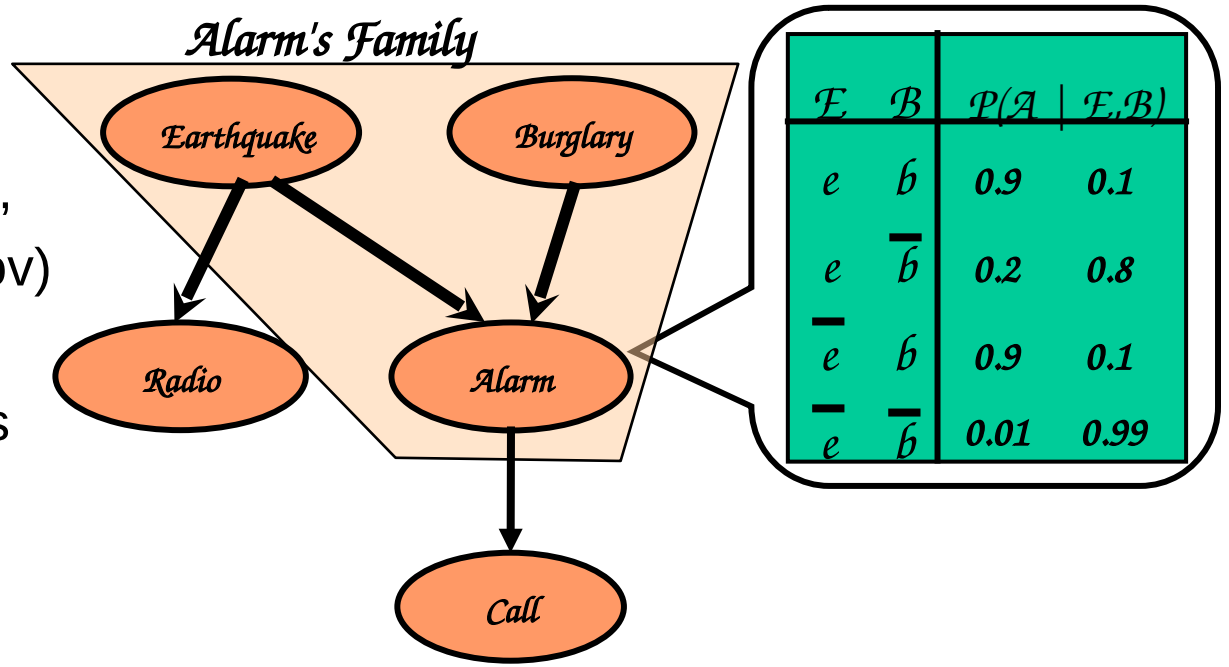
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



Factorization of the joint probability function.

$$P(B,E,A,R,C)=P(E)\,P(B)\,P(R|E)\,P(A|E,B)\,P(C|A)$$

Parameters: Probabilities and tables.

- Maximum likelihood estimation (**MLE**)

$$MLE(\theta:data)=\prod_{i=1}^{N} P(X_i,\pi_i:\theta)=\prod_{i=1}^{N} P(\pi_i:\theta)\,P(X_i|\pi_i:\theta)$$ ← **"Frequentist" approach**

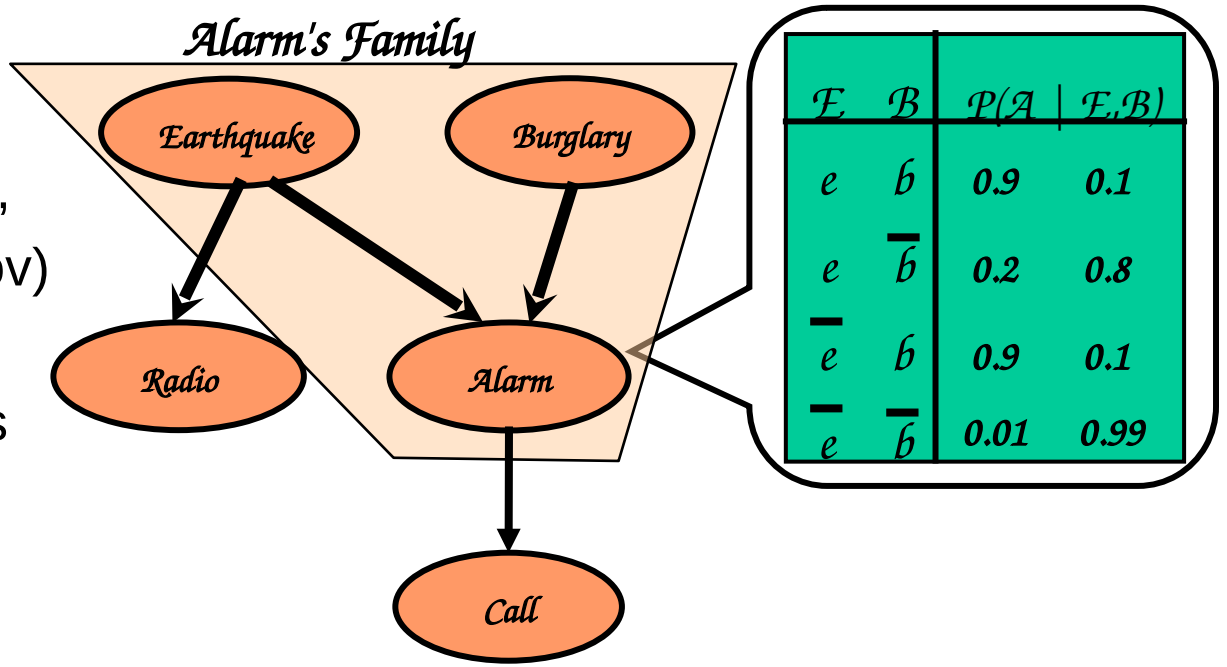
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



Factorization of the joint probability function.

$$P(B,E,A,R,C)=P(E)\,P(B)\,P(R|E)\,P(A|E,B)\,P(C|A)$$

Parameters: Probabilities and tables.

- Bayesian Estimation → Parameters are random variables

"A priori" distribution of the parameters

$$P(X_1,\ldots,X_N,\theta)=P(\theta)\prod_{i=1}^{N}P(X_i|\theta) \qquad P(\theta)=\frac{1}{Z}\prod_{i=1}^{k}\theta_i^{\alpha_i-1} \wedge Z=\prod_{i=1}^{k}\Gamma(\alpha_i)/\Gamma\left(\sum_{i=1}^{k}\alpha_i\right)$$ ← **Dirichlet**

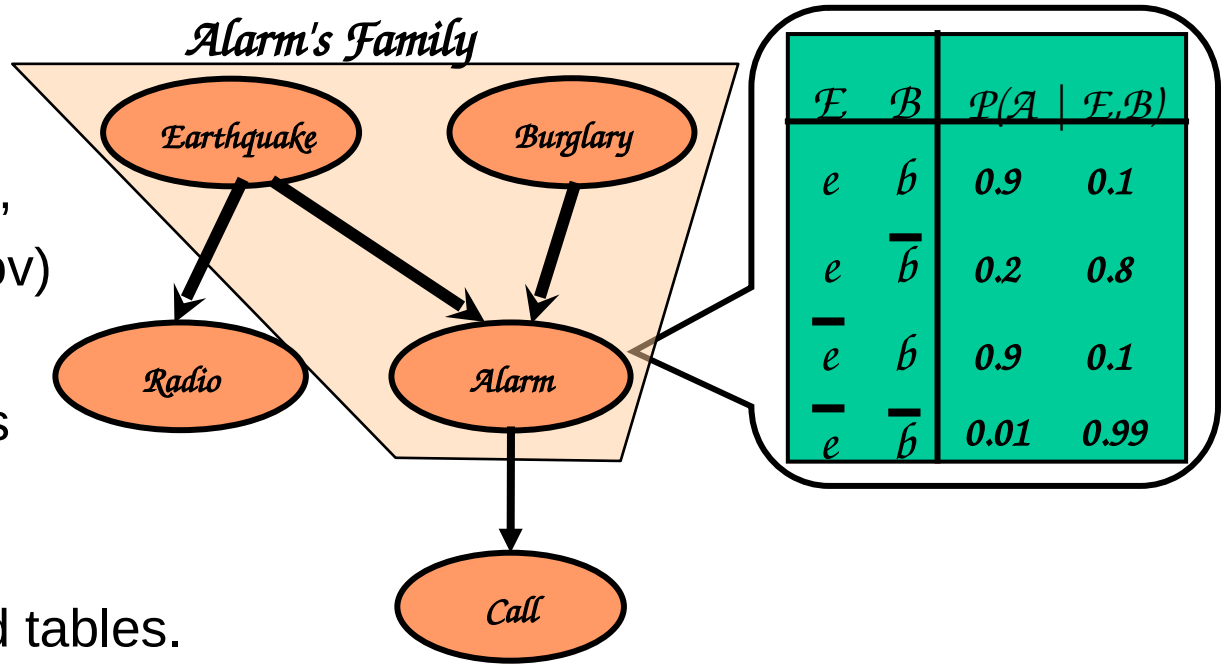
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



Parameters: Probabilities and tables.

- Bayesian Estimation.

"A priori" distribution of the parameters

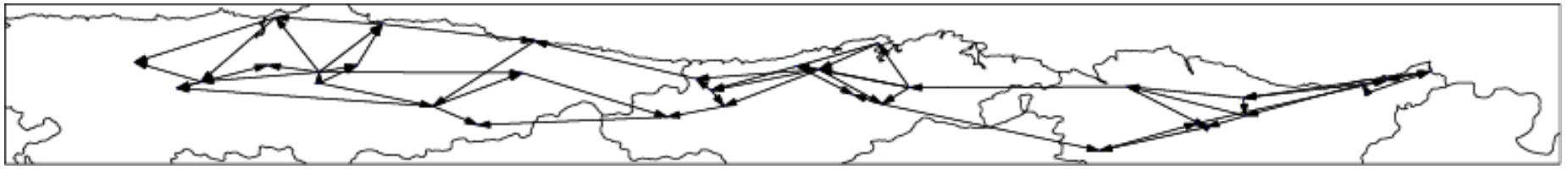
$$P(X_1,\ldots,X_N,\theta)=P(\theta)\prod_{i=1}^{N}P(X_i|\theta) \qquad P(\theta)=\frac{1}{Z}\prod_{i=1}^{k}\theta_i^{\alpha_i-1}\;\backslash\; Z=\prod_{i=1}^{k}\Gamma(\alpha_i)/\Gamma\left(\sum_{i=1}^{k}\alpha_i\right) \leftarrow \textbf{Dirichlet}$$

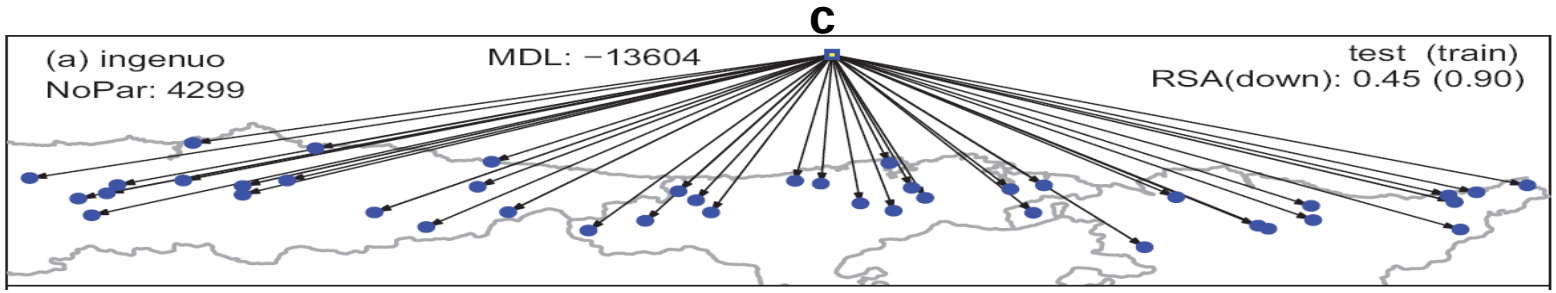
Multinomial Case:

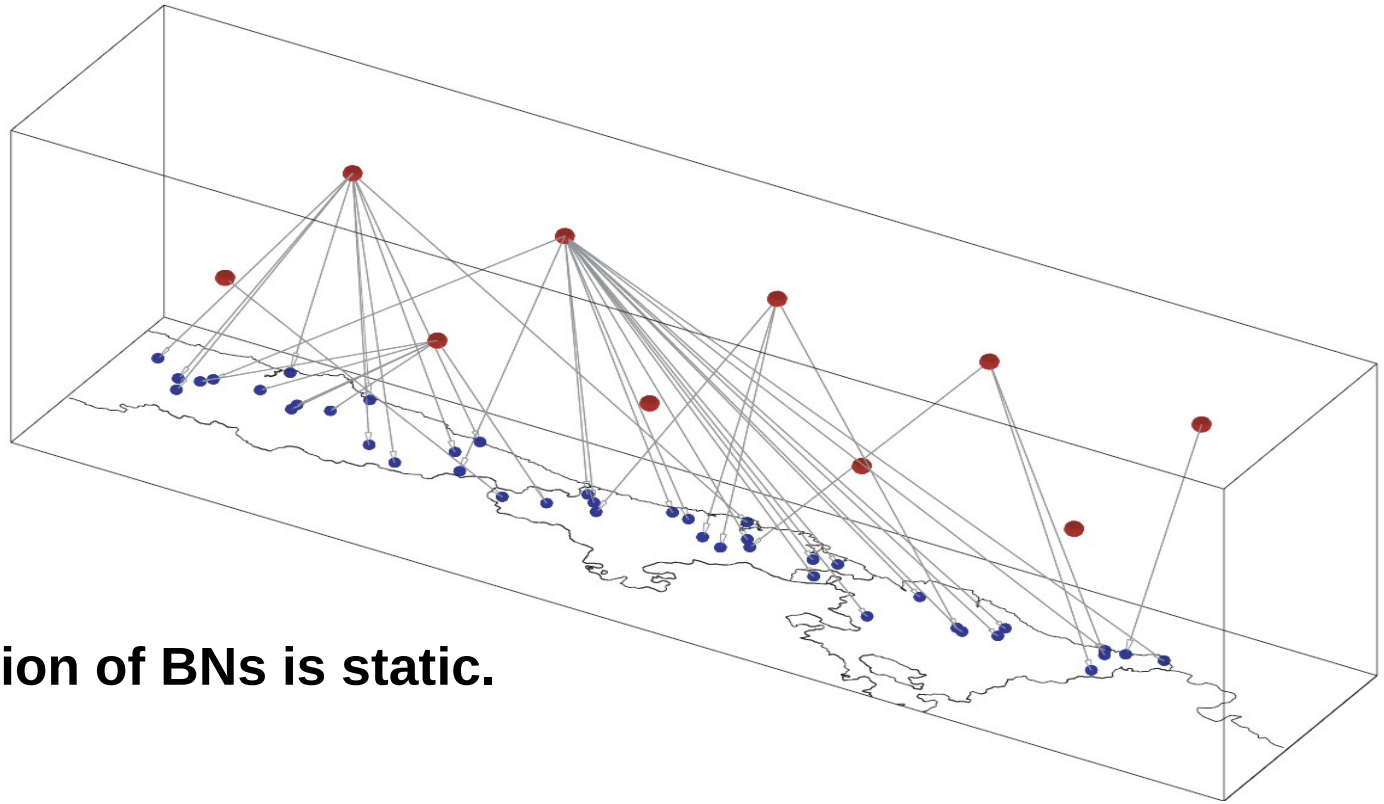
Number of samples

$$P(\theta|data)\propto P(data|\theta)*P(\theta)\Rightarrow P(data|\theta)=\prod_{i=1}^{N}\theta_i^{M_i}$$

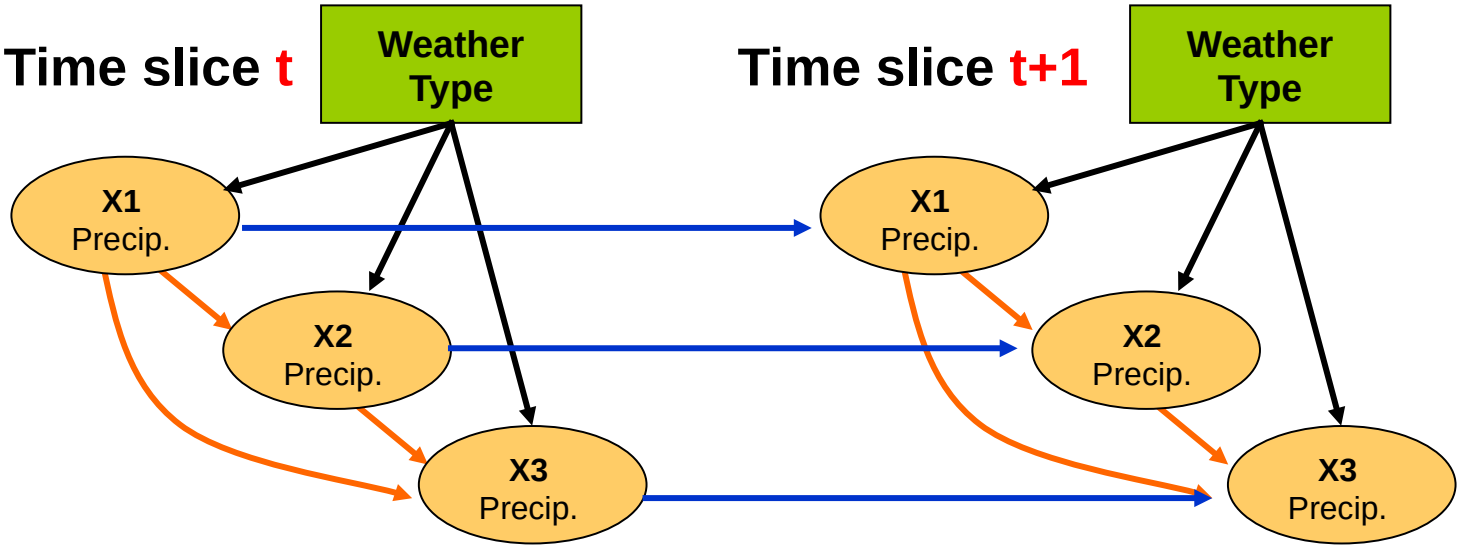
Data counts

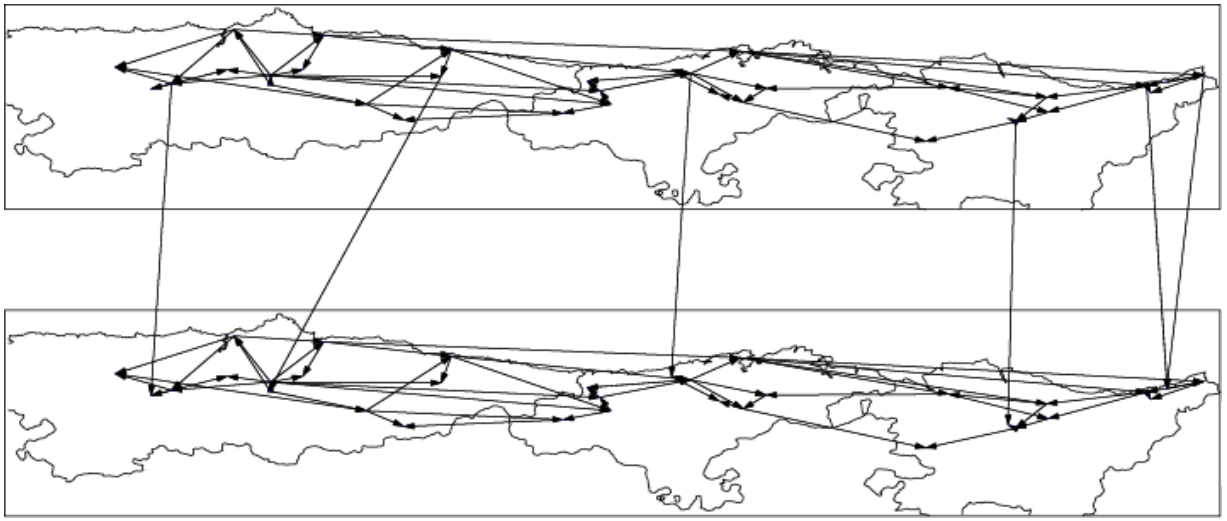






The most used configuration of BNs is static.



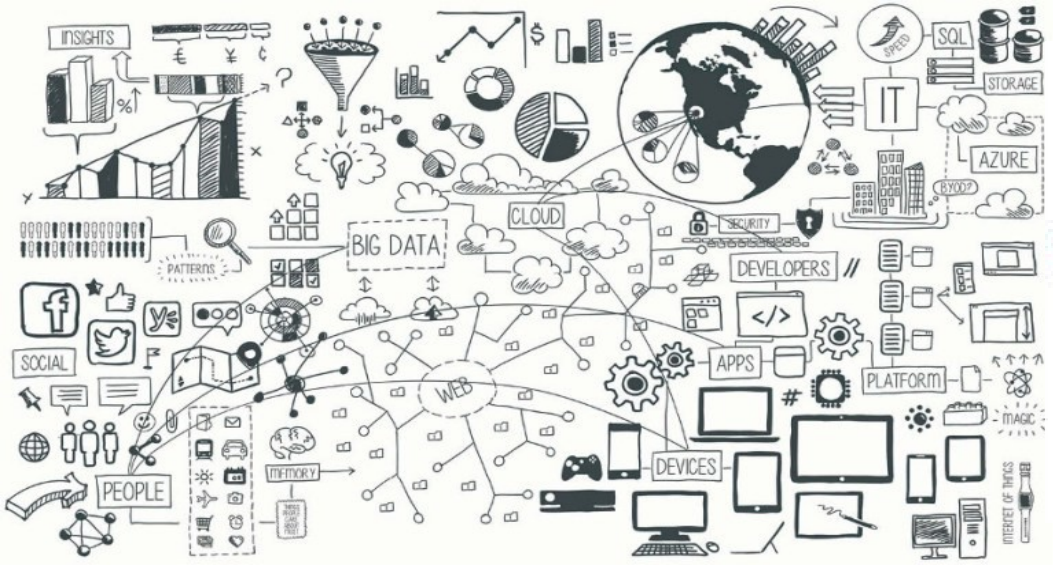


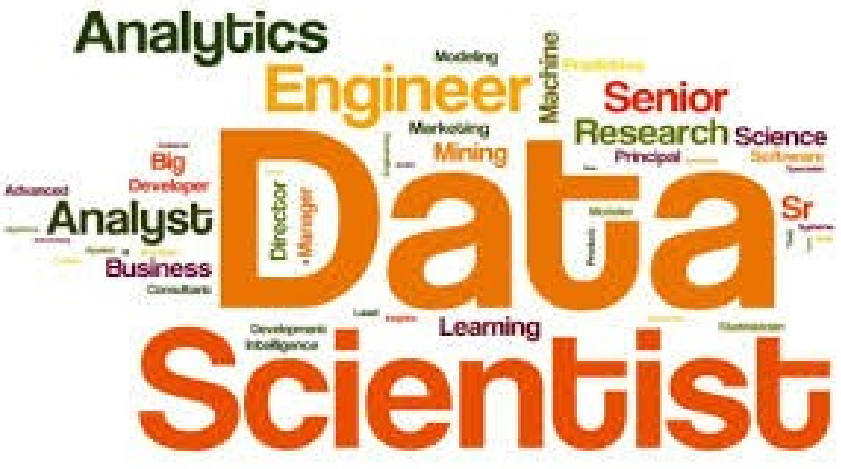
Time slice t

Time slice t+1

The most used configuration of BNs is static. However, there are not theoretical limitations to consider a dynamic configuration.

Machine Learning II
Gaussian and Hybrid Bay. Net.

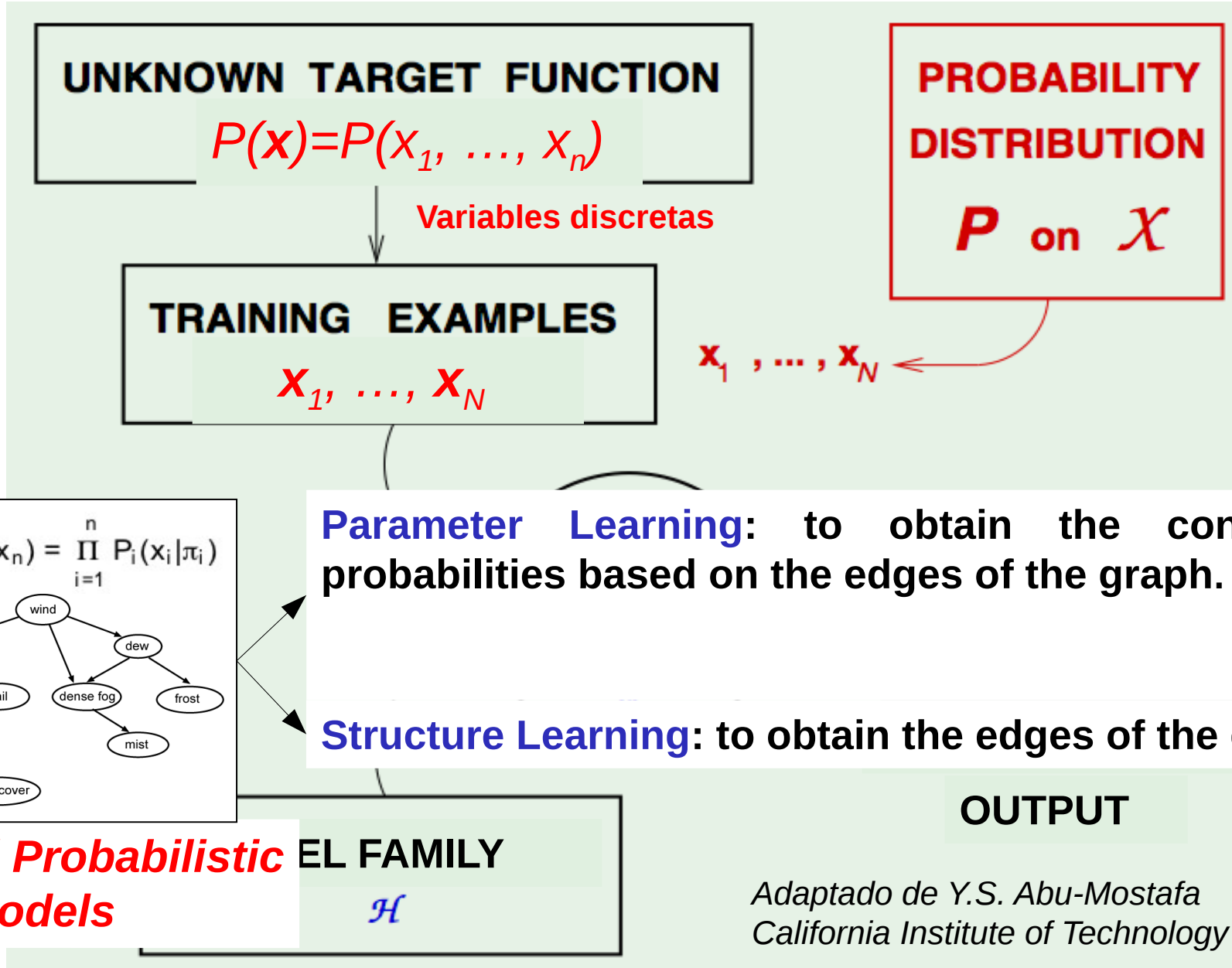




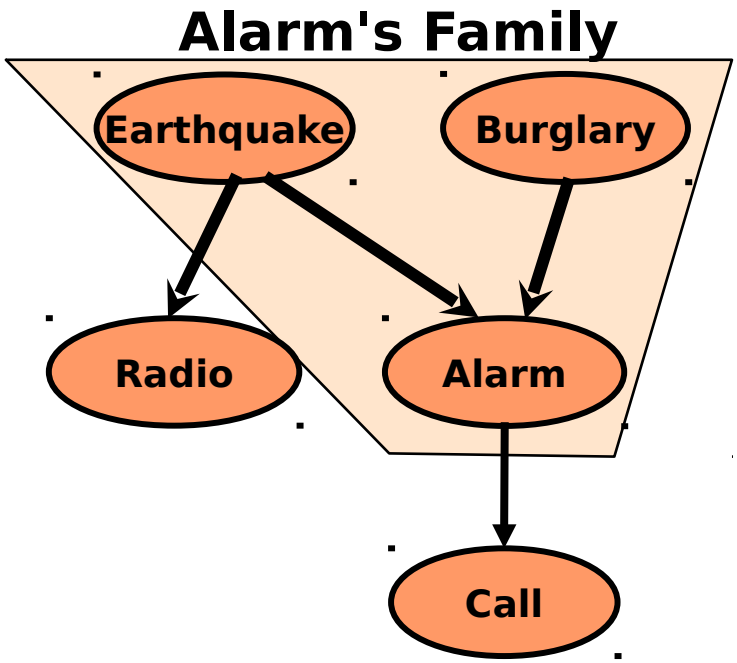
Sixto Herrera (herreras@unican.es)

Grupo de Meteorología

Univ. de Cantabria – CSIC
MACC / IFCA


UNKNOWN TARGET FUNCTION
$P(x)=P(x_1, \ldots, x_n)$
Variables discretas
PROBABILITY DISTRIBUTION
P on X
TRAINING EXAMPLES
$x_1, \ldots, x_N$
$x_1, \ldots, x_N$
$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P_i(x_i|\pi_i)$
wind
rainfall
dew
storm
hail
dense fog
frost
snow
mist
snow cover
Parameter Learning: to obtain the conditional probabilities based on the edges of the graph.
Structure Learning: to obtain the edges of the graph.
EL FAMILY
$\mathcal{H}$
OUTPUT
Graphical Probabilistic Models
Adaptado de Y.S. Abu-Mostafa
California Institute of Technology

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

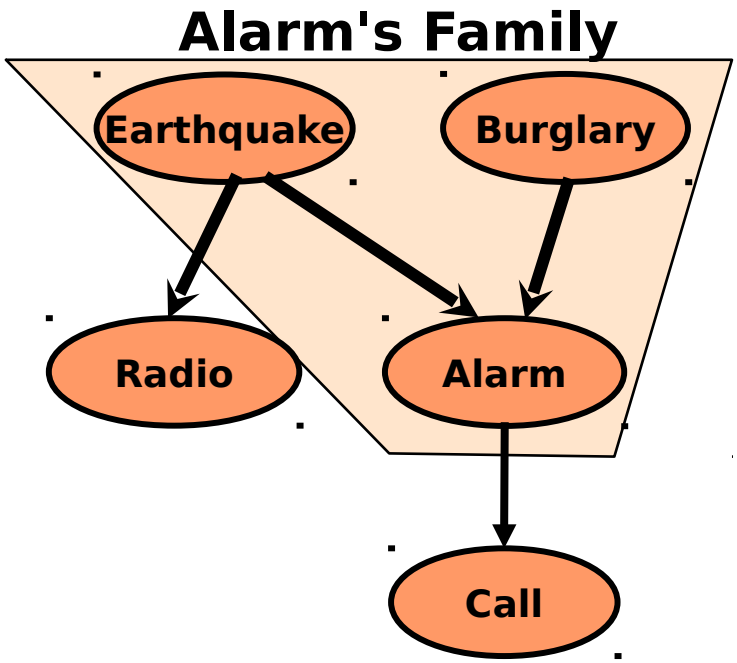


← Multivariate Normal Distribution $N(\mu, \Sigma)$

Mean vector

Covariance Matrix

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



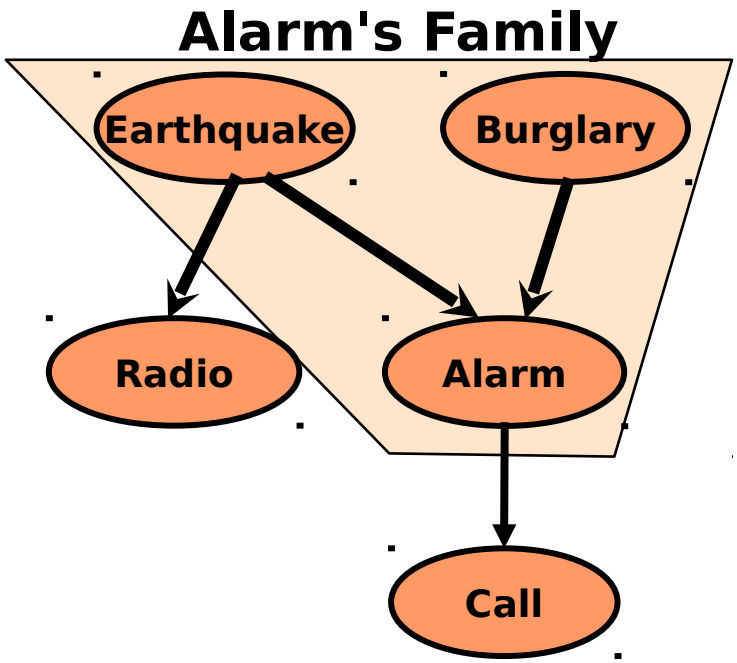
← Multivariate Normal Distribution $N(\mu, \Sigma)$

$$f(x) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{(-1/2)(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

Factorization

$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



← Multivariate Normal Distribution $N(\mu, \Sigma)$

$$f(x) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{(-1/2)(x-\nu)^T \Sigma^{-1} (x-\nu)}$$

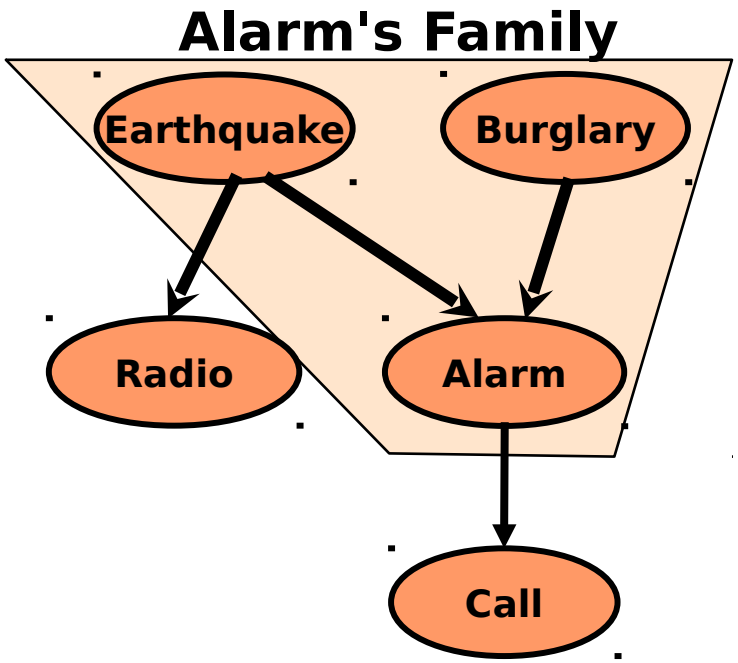
Factorization

$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Regression Coefficients

Conditional Variance

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



← Multivariate Normal Distribution $N(\mu, \Sigma)$

$$f(x) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{(-1/2)(x-\nu)^T \Sigma^{-1} (x-\nu)}$$

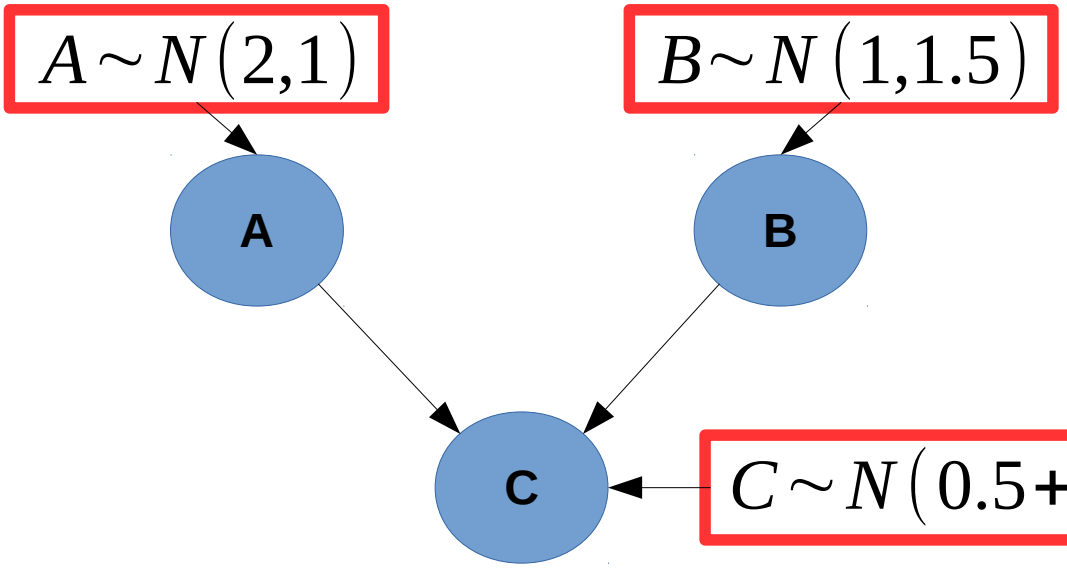
Factorization

$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Conditional (In)dependence ← Regression Coefficients

Conditional Variance

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



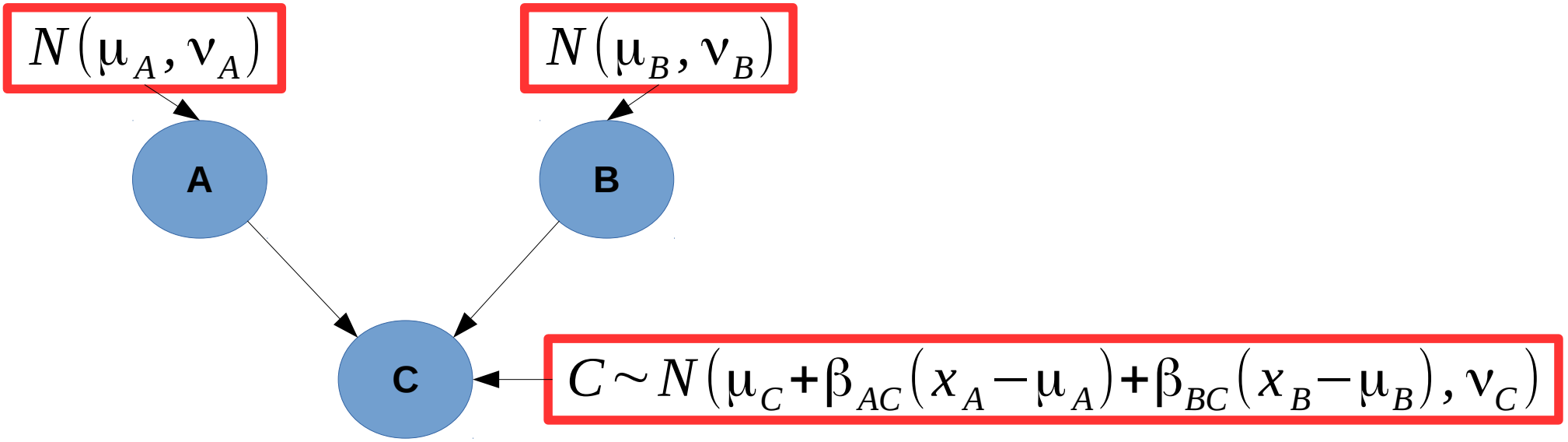
$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Conditional (In)dependence ⟵ **Regression Coefficients**

```
distA = list(coef = c("(Intercept)" = 2), sd = 1)
distB = list(coef = c("(Intercept)" = 1), sd = 1.5)
distC = list(coef = c("(Intercept)" = 0.5, "A" = 0.75, "B" = 1.32), sd = 0.4)
cfit = custom.fit(net, dist = list(A = distA, B = distB, C = distC))
```

← **We could specify the parameters**

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



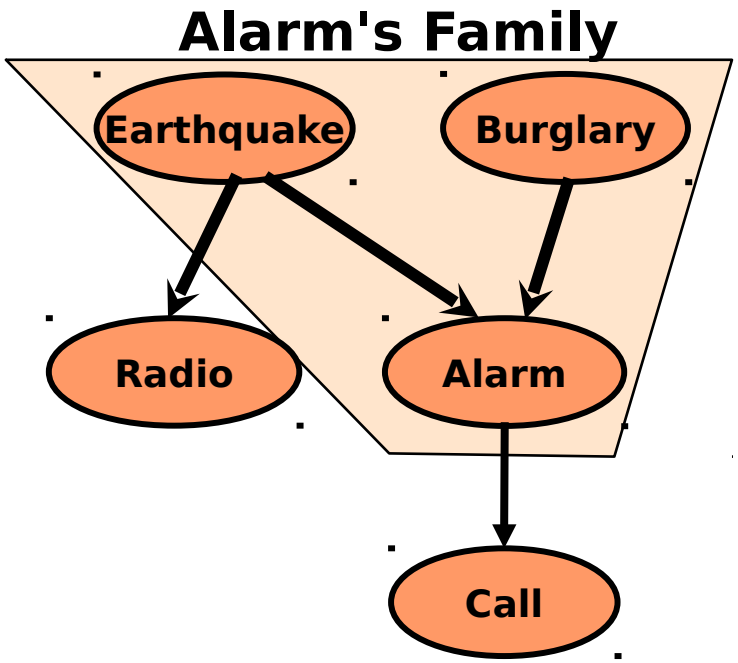
$$f(x|\pi_i) \sim N(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i)$$

Conditional (In)dependence ⟸ **Regression Coefficients**

← We could learn the parameters

```
distA = lm(A ~ 1, data = gaussian.test)
distB = lm(B ~ 1, data = gaussian.test)
distC = lm(C ~ A + B, data = gaussian.test)
cfit = custom.fit(net, dist = list(A = distA, B = distB, C = distC))
```

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



← Multivariate Normal Distribution $N(\mu, \Sigma)$

$$f(x) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{(-1/2)(x-\nu)^T \Sigma^{-1} (x-\nu)}$$

Factorization

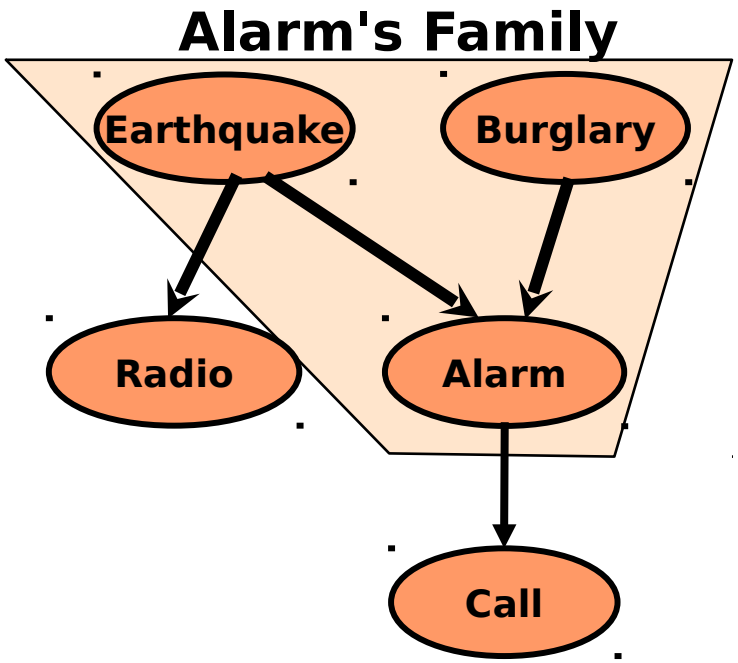
$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Conditional (In)dependence ← Regression Coefficients

```
dag = model2network("[A][B][E][G][C|A:B][D|B][F|A:D:E:G]")
fitted = bn.fit(dag, gaussian.test)
```

← We could specify the DAG

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



← Multivariate Normal Distribution $N(\mu, \Sigma)$

$$f(x) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{(-1/2)(x-\nu)^T \Sigma^{-1} (x-\nu)}$$

Factorization

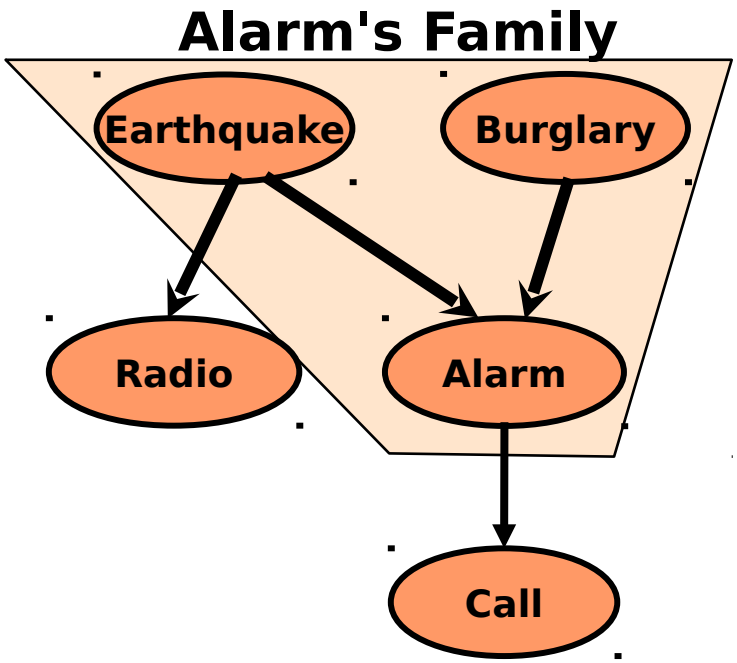
$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j), \nu_i\right)$$

Conditional (In)dependence ← Regression Coefficients

```
learned <- hc(gaussian.test, debug = FALSE) # Structural learning
dag <- model2network(modelstring(learned)) # Define the DAG
bn.gauss <- bn.fit(dag, data = gaussian.test, method = "mle") # Define the BN
```

← We could learn the BN

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

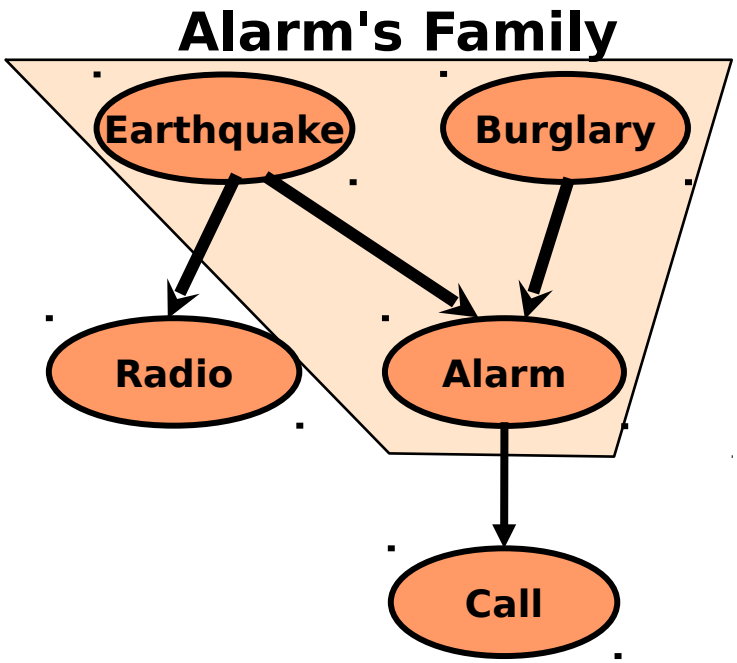
- **Nodes** – discrete and continuous
- **Links** – direct dependences

Joint Distribution is Conditional Gaussian.

$$f\left(x|Y=i\right)\sim N\left(\mu\left(x|i\right),\Sigma\left(x|i\right)\right)$$

For a theoretical description: https://www2.eecs.berkeley.edu/Pubs/TechRpts/1998/CSD-98-990.pdf

Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.



Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- **Nodes** – discrete and continuous
- **Links** – direct dependences

Joint Distribution is Conditional Gaussian.

$$f\left(x|Y=i\right)\sim N\left(\mu\left(x|i\right),\Sigma\left(x|i\right)\right)$$

Discrete: defined by the CPT given its parents (**only discrete**)
Continuous: defined by a Gaussian function given its parents (discrete or continuous)

For a theoretical description: https://www2.eecs.berkeley.edu/Pubs/TechRpts/1998/CSD-98-990.pdf