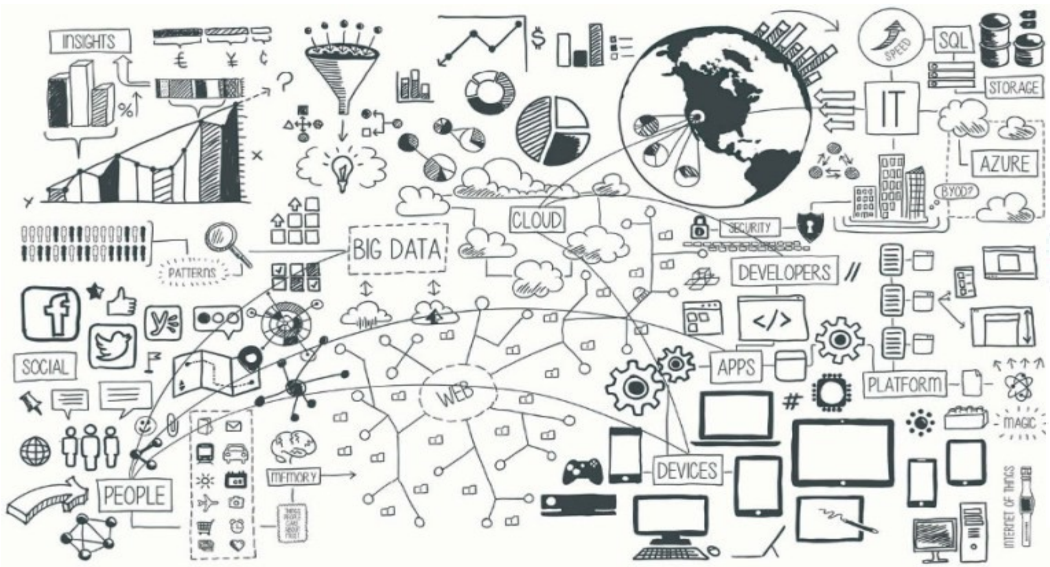
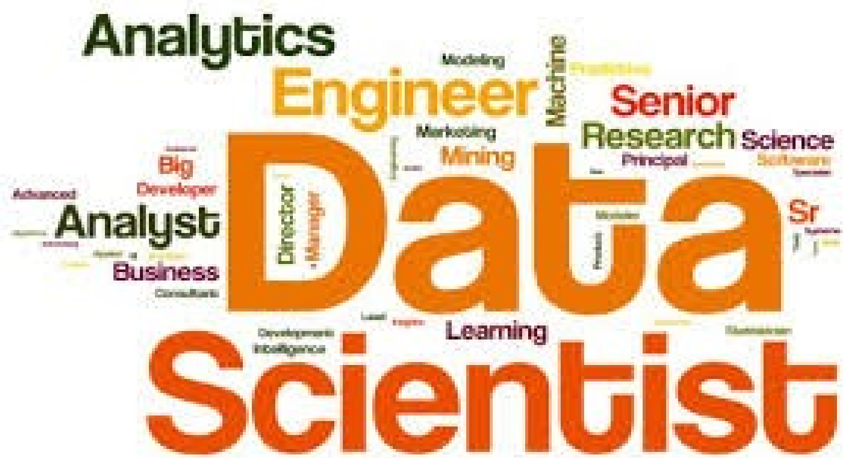
M1970 – Machine Learning II
Redes Probabilísticas Discretas





Sixto Herrera (sixto.herrera@unican.es)
José M. Gutiérrez, Mikel Legasa

Grupo de Meteorología

Univ. de Cantabria – CSIC
MACC / IFCA

M1970 – Machine Learning (L 16:00-18:00; X 16:00-18:00)

Mar	**2**	**L**	**Redes Probabilísticas Discretas (2h-T)**
	4	**X**	**Redes Bayesianas: Creación e Inferencia (2h-L)**
	9	**L**	**Clasificacidores Bayesianos. Naive Bayes (2h-L)**
	11	**X**	**Redes Bayesianas: Aprendizaje (2h-T)**
	16	**L**	**Redes Bayesianas: Aprendizaje (2h-L)**
	18	**X**	**Redes Bayesianas: Aprendizaje (2h-L)**
	23	**L**	**Evaluación (2h)**

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

CATEGÓRICO

Overview Data Kernels Discussion Activity Download (30 KB) New Kernel

http://archive.ics.uci.edu/ml/datasets/Mushroom

Data Set Characteristics:	Multivariate	**Number of Instances:**	8124	**Area:**	Life
Attribute Characteristics:	Categorical	**Number of Attributes:**	22	**Date Donated**	1987-04-27
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	298439

Attribute Information: (classes: edible=e, poisonous=p)

cap-shape: bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
cap-surface: fibrous=f,grooves=g,scaly=y,smooth=s
cap-color: brown=n,buff=b,cinnamon=c,gray=g,green=r,pink=p,purple=u,...
bruises: bruises=t,no=f
odor: almond=a,anise=l,creosote=c,fishy=y,foul=f,musty=m,none=n,...
...

```
mush <- read.csv("Data_mining/datasets/mushrooms.csv")
str(mush)
```

```
'data.frame':   8124 obs. of  23 variables:
 $ class                   : Factor w/ 2 levels "e","p": 2 1 1 2 1 1 1 1 2 1 ...
 $ cap.shape               : Factor w/ 6 levels "b","c","f","k",..: 6 6 1 6 6 6 1 1 6 1 ...
 $ cap.surface             : Factor w/ 4 levels "f","g","s","y": 3 3 3 4 3 4 3 4 4 3 ...
 $ cap.color               : Factor w/ 10 levels "b","c","e","g",..: 5 10 9 9 4 10 9 9 9 10 ...
 $ bruises                 : Factor w/ 2 levels "f","t": 2 2 2 2 1 2 2 2 2 2 ...
 $ odor                    : Factor w/ 9 levels "a","c","f","l",..: 7 1 4 7 6 1 1 4 7 1 ...
 ...
```

CATEGÓRICO

Overview Data Kernels Discussion Activity Download (30 KB) New Kernel

http://archive.ics.uci.edu/ml/datasets/Mushroom

Data Set Characteristics:	Multivariate	**Number of Instances:**	8124	**Area:**	Life
Attribute Characteristics:	Categorical	**Number of Attributes:**	22	**Date Donated**	1987-04-27
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	298439

Attribute Information: (classes: edible=e, poisonous=p)

cap-shape: bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
cap-surface: fibrous=f,grooves=g,scaly=y,smooth=s
cap-color: brown=n,buff=b,cinnamon=c,gray=g,green=r,pink=p,purple=u,...
bruises: bruises=t,no=f
odor: almond=a,anise=l,creosote=c,fishy=y,foul=f,musty=m,none=n,...
...

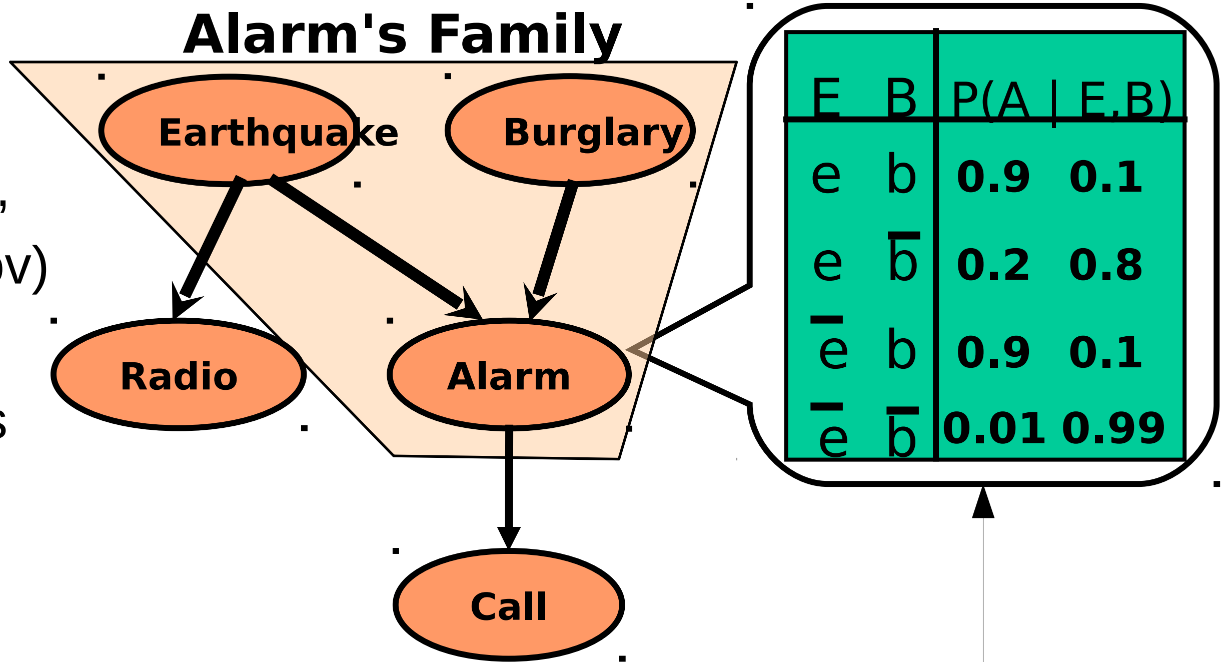
Consider the Mushroom dataset

- How much parameters would be needed?

- How much parameters are obtained for the Naive Bayes?

- Train the model and evaluate the Bayesian Classifier obtained

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)

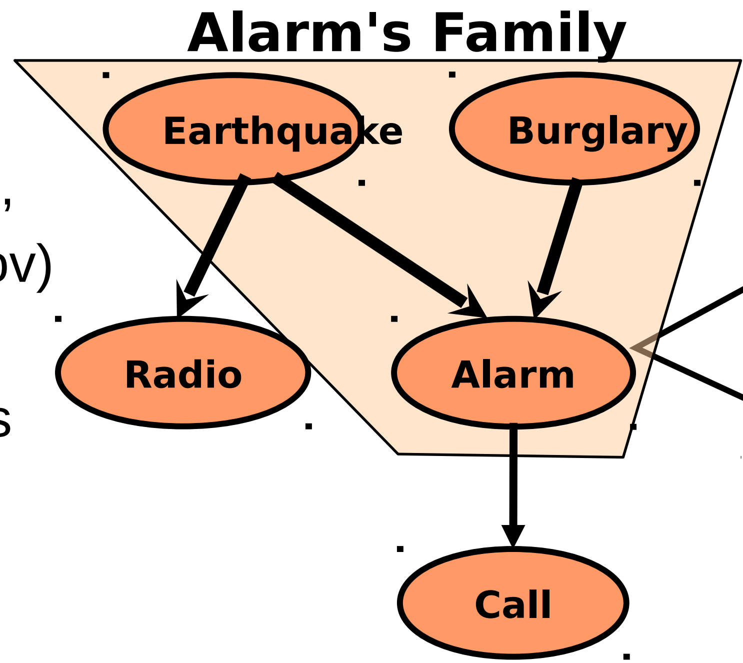
- Nodes – variables
- Links – direct dependences

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



E	B	P(A | E,B)	
e	b	**0.9**	**0.1**
e	$\bar{b}$	**0.2**	**0.8**
$\bar{e}$	b	**0.9**	**0.1**
$\bar{e}$	$\bar{b}$	**0.01**	**0.99**

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.

UNKNOWN TARGET FUNCTION

$P(\mathbf{x})=P(x_1, \ldots, x_n)$

Variables discretas

PROBABILITY DISTRIBUTION

$\boldsymbol{P}$ **on** $\mathcal{X}$

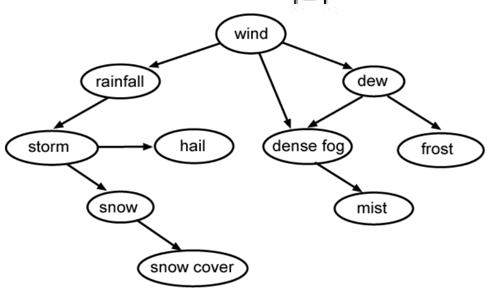
TRAINING EXAMPLES

$\mathbf{x}_1, \ldots, \mathbf{x}_N$

$\mathbf{x}_1, \ldots, \mathbf{x}_N$

Graphical Probabilistic Models

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P_i(x_i|\pi_i)$$



LEARNING ALGORITHM

$\mathcal{A}$

FINAL MODEL

$P'(\mathbf{x})$

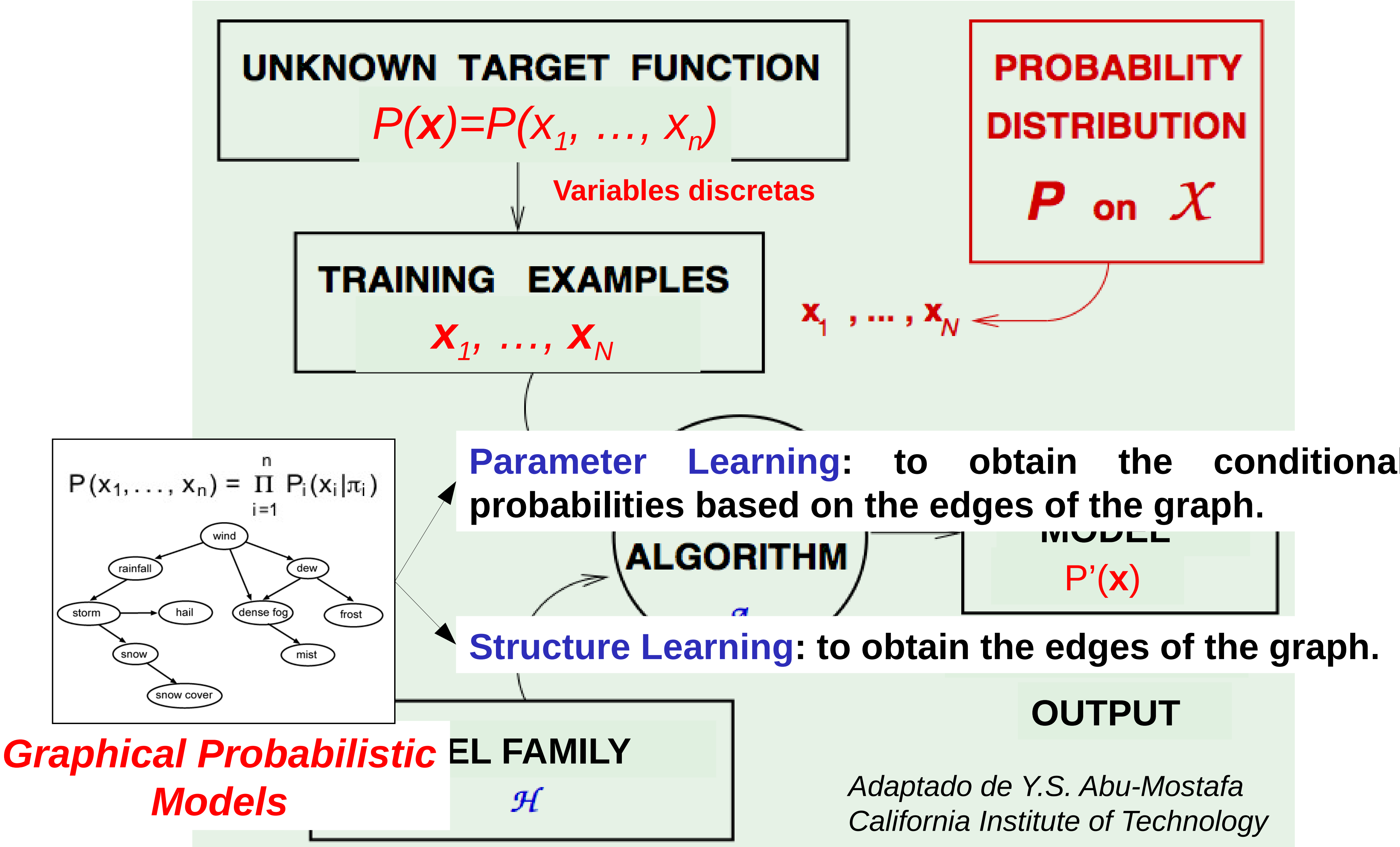
$P'(y|\mathbf{e})$; **E** $\subset$ \Y

OUTPUT

MODEL FAMILY

$\mathcal{H}$

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



UNKNOWN TARGET FUNCTION
$P(\mathbf{x})=P(x_1, \ldots, x_n)$
Variables discretas
PROBABILITY DISTRIBUTION
P on $\mathcal{X}$
TRAINING EXAMPLES
$\mathbf{x}_1, \ldots, \mathbf{x}_N$
$\mathbf{x}_1, \ldots, \mathbf{x}_N$
$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P_i(x_i|\pi_i)$
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.
ALGORITHM
$P'(\mathbf{x})$
Structure Learning: to obtain the edges of the graph.
OUTPUT
Graphical Probabilistic Models
EL FAMILY
$\mathcal{H}$
Adaptado de Y.S. Abu-Mostafa
California Institute of Technology

Aprendizaje Estructural:

Aprender la estructura (grafo) a partir del dataset

Técnicas “score based”

Técnicas “constraint based”

· 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...

Aprendizaje Estructural:

Aprender la estructura (grafo) a partir del dataset

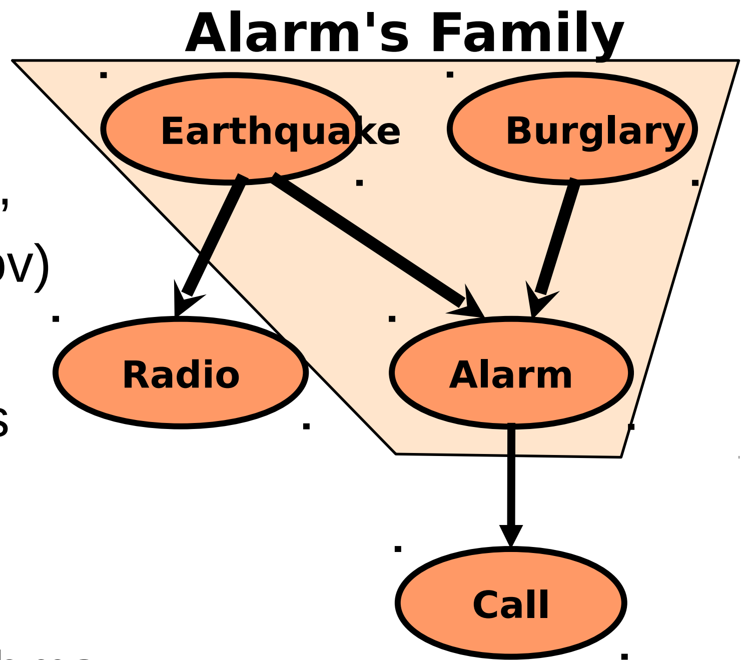
Técnicas “score based”

Técnicas “constraint based”

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)

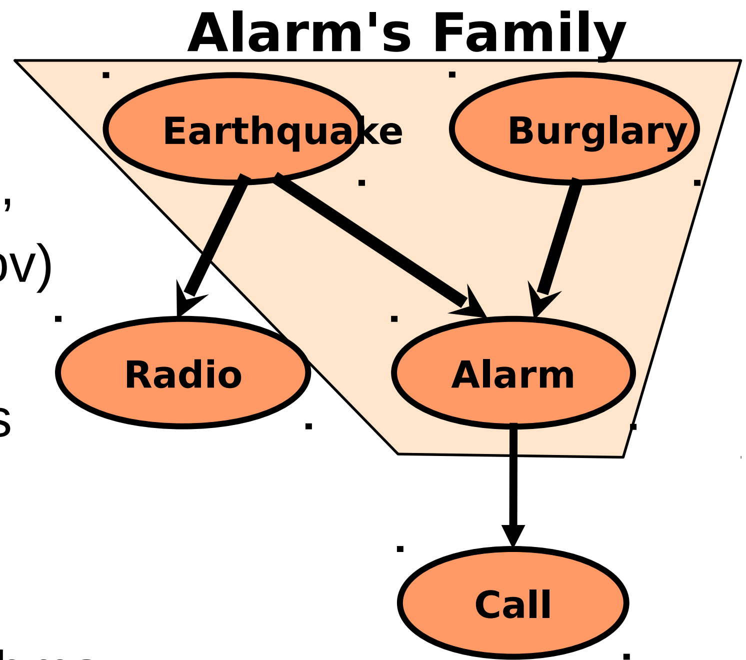
- Nodes – variables
- Links – direct dependences

- **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)

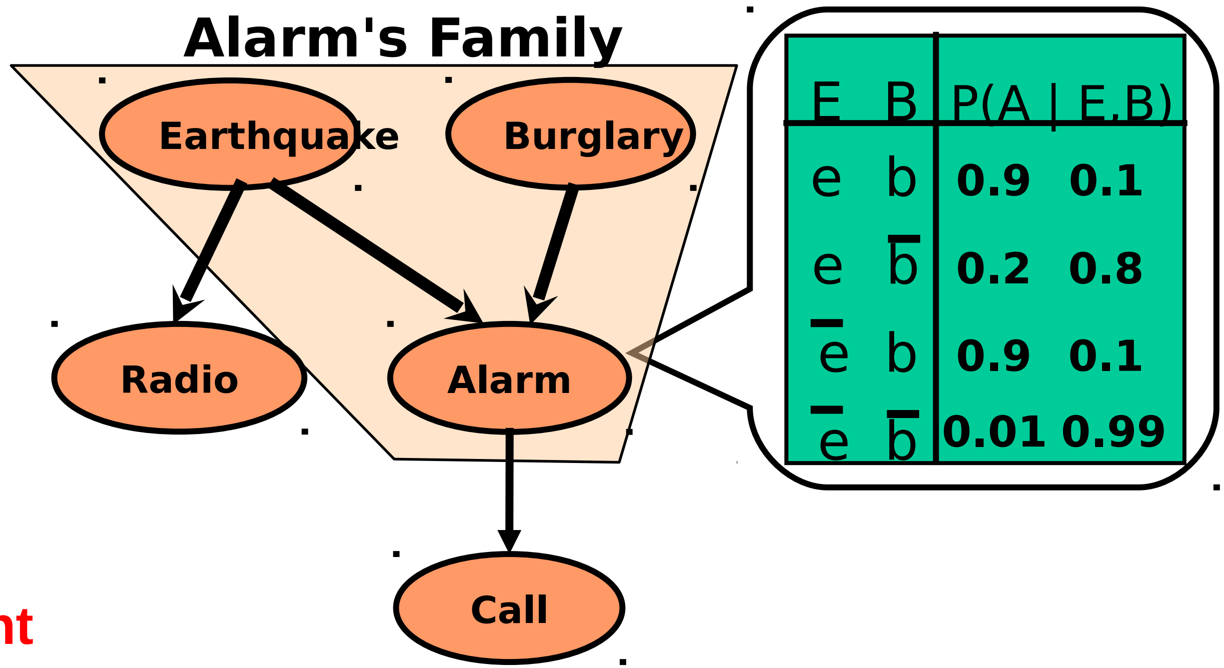
- Nodes – variables
- Links – direct dependences

- **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.



Score:

P(M) - “A priori” information

P(data|M) – Explain capability

Model complexity

Bayesian Dirichlet Equivalent

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

Minimum Description Length // Bayesian Information Criterium

$$MDL(M|data)=\frac{r*\log(N)}{2}-\sum_{i=1}^{N}\log(P_{X_i}(X_i|\pi_i))$$

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

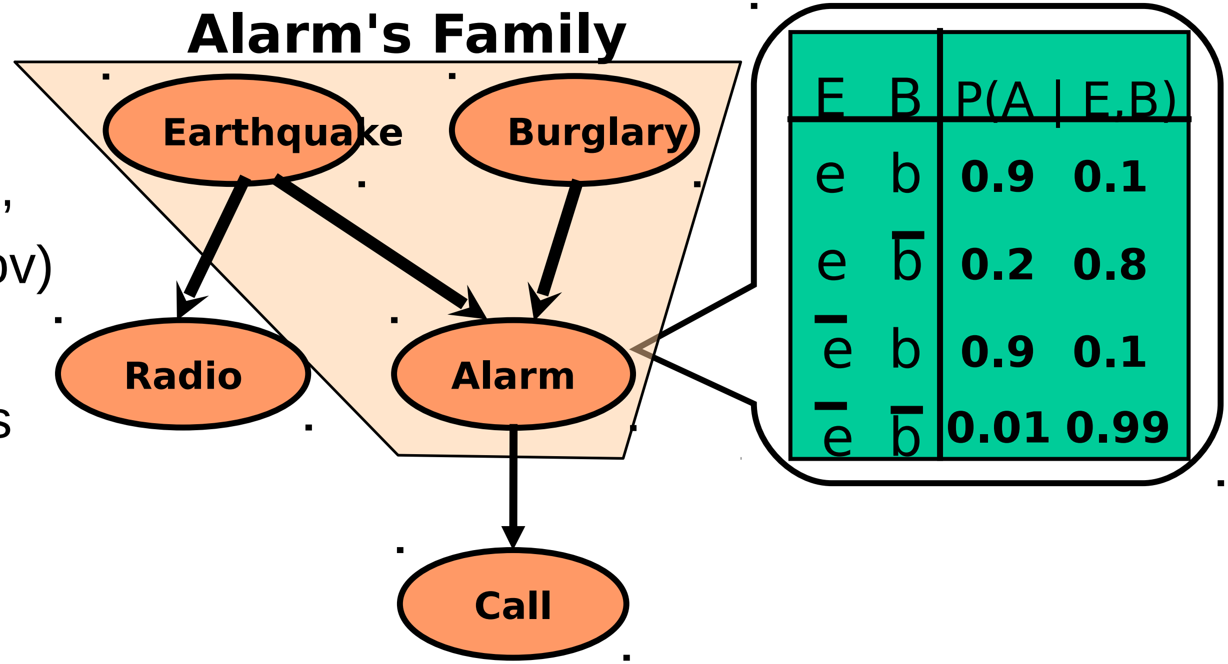
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

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.

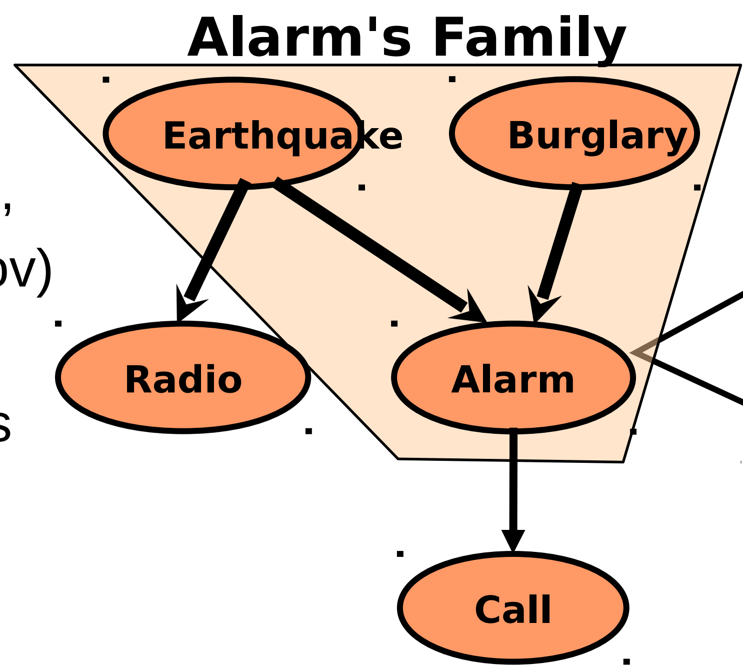
- Maximum likelihood estimation (**MLE**)
- Bayesian Estimation.

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



E	B	P(A | E,B)	
e	b	**0.9**	**0.1**
e	$\bar{b}$	**0.2**	**0.8**
$\bar{e}$	b	**0.9**	**0.1**
$\bar{e}$	$\bar{b}$	**0.01**	**0.99**

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**

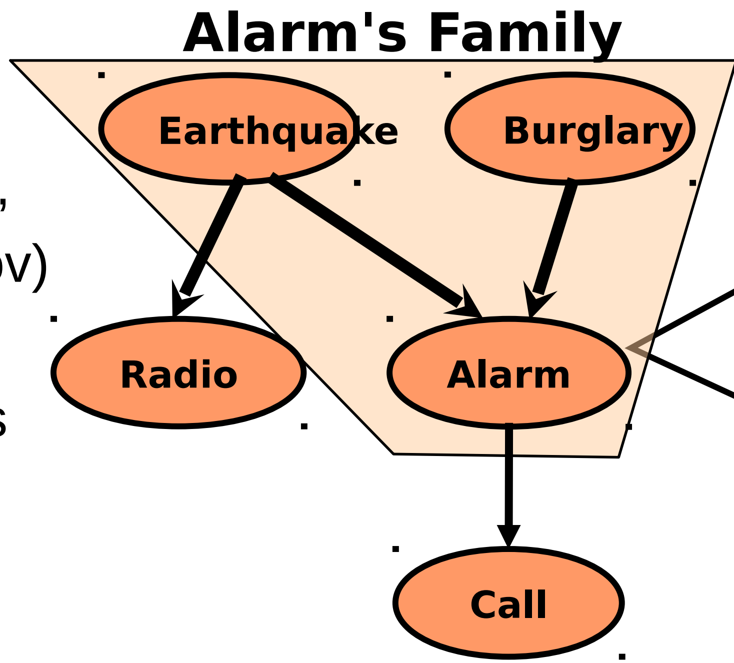
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



E	B	P(A | E,B)	
e	b	0.9	0.1
e	$\bar{b}$	0.2	0.8
$\bar{e}$	b	0.9	0.1
$\bar{e}$	$\bar{b}$	0.01	0.99

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

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

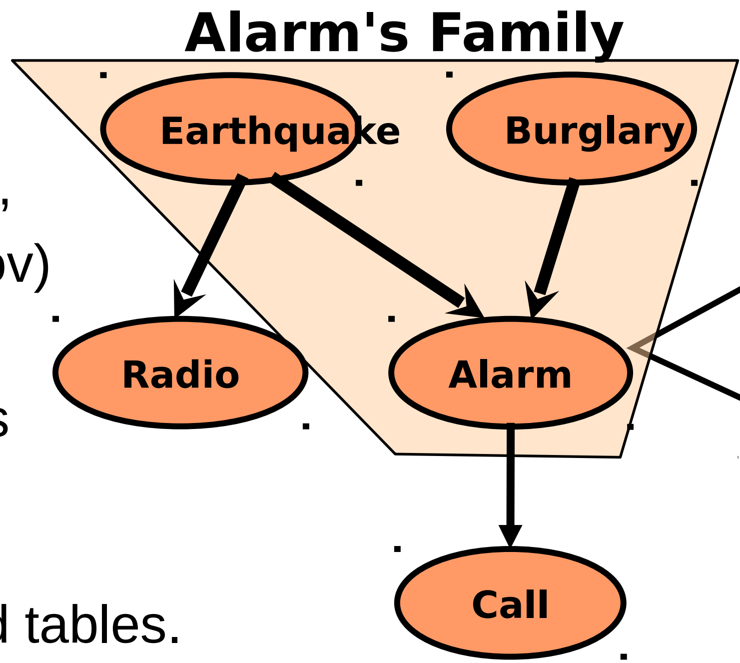
Bayesian Networks obtain a compact representation of the joint probability function through the conditional independences.

Structure:

Acyclic Directed Graph (DAG),

or non-directed graphs (Markov)

- Nodes – variables
- Links – direct dependences



E	B	P(A | E,B)	
e	b	0.9	0.1
e	$\bar{b}$	0.2	0.8
$\bar{e}$	b	0.9	0.1
$\bar{e}$	$\bar{b}$	0.01	0.99

Parameters: Probabilities and tables.

- Bayesian Estimation.

"A priori" distribution of the parameters

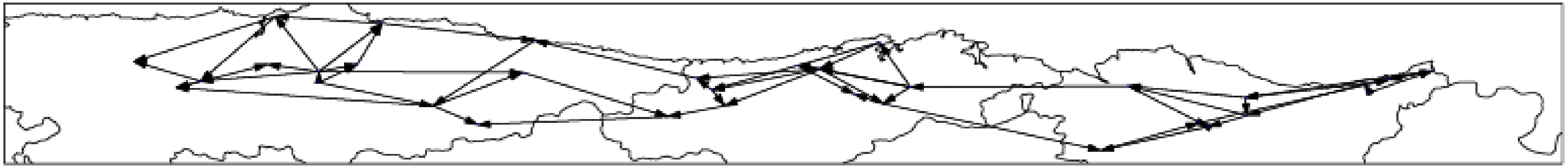
$$P(X_1,\ldots,X_N,\theta)=\boxed{P(\theta)}\prod_{i=1}^{N}P(X_i|\theta) \qquad P(\theta)=\frac{1}{Z}\prod_{i=1}^{k}\theta_i^{\boxed{\alpha_i-1}}, Z=\prod_{i=1}^{k}\Gamma(\alpha_i)/\Gamma\left(\sum_{i=1}^{k}\alpha_i\right) \leftarrow \text{Dirichlet}$$

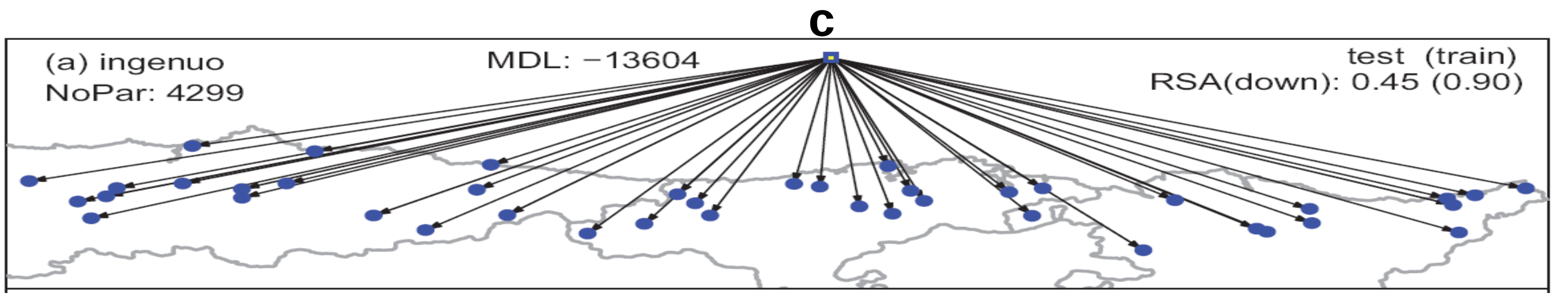
Number of samples

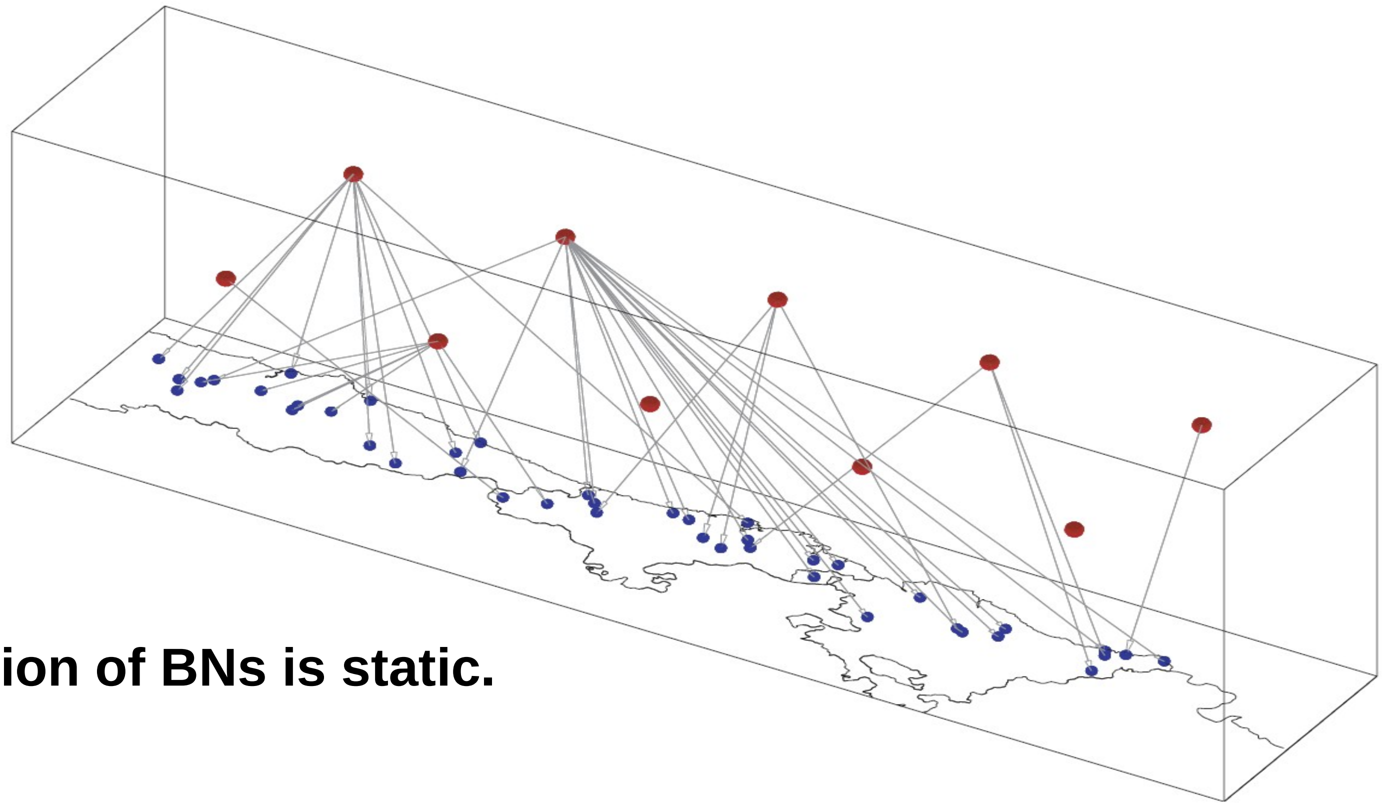
Multinomial Case:

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

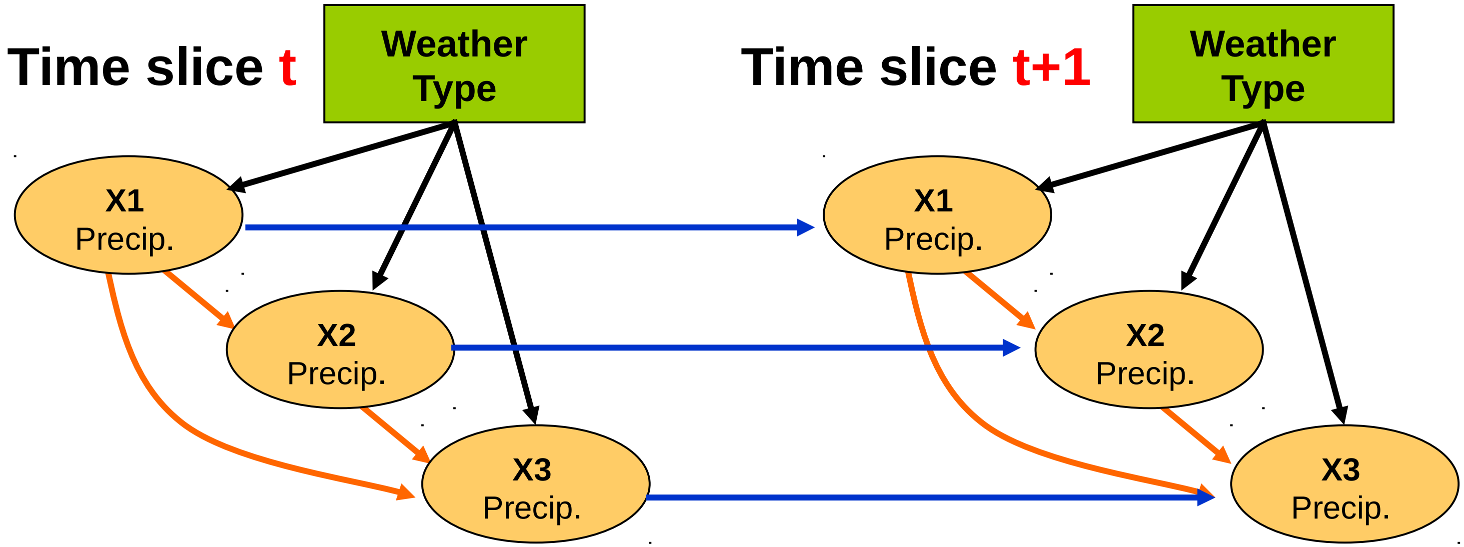
Data counts

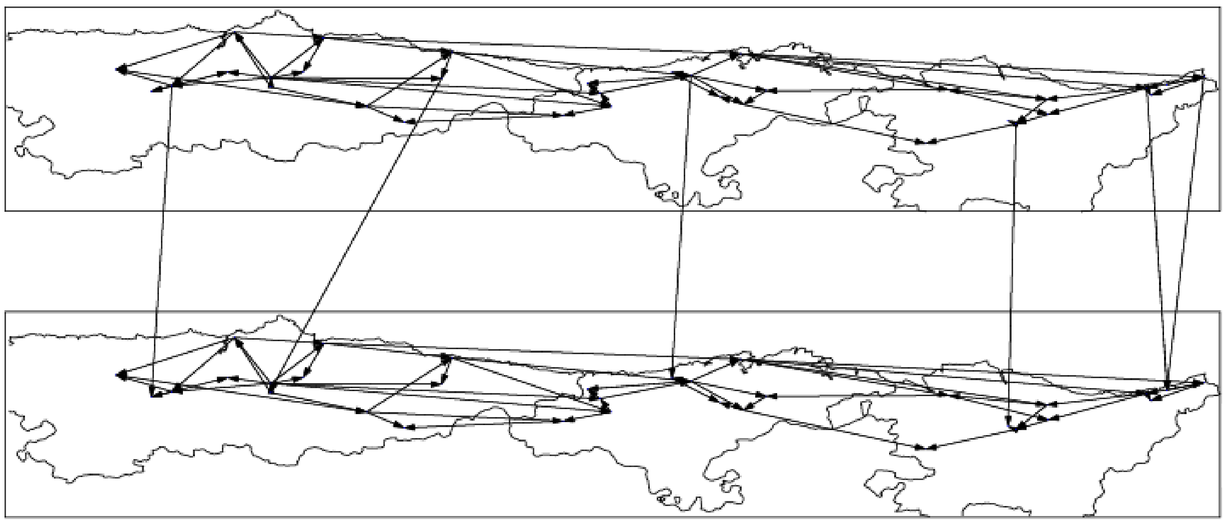






The most used configuration of BNs is static.





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.

Survey is a atificial dataset including the answer of the users of several transports to a fictitious survey. For each user, six variables have been collected:

- Age (**A**) of the user, grouped in **three** states: *Young*, ***Adult*** and ***Old***
- Sex (**S**) of the user: Male (**M**) and Female (**F**)
- Education (**E**) of the user: highest education level reached, ***high school*** and ***university***.
- Ocupation (**O**): ***employee*** and ***self-employed***.
- Residence (**R**): population of the town, ***big*** or ***small***, in which the user lives.
- Transport (**T**): most used transport: ***car, train*** and ***other***.

.

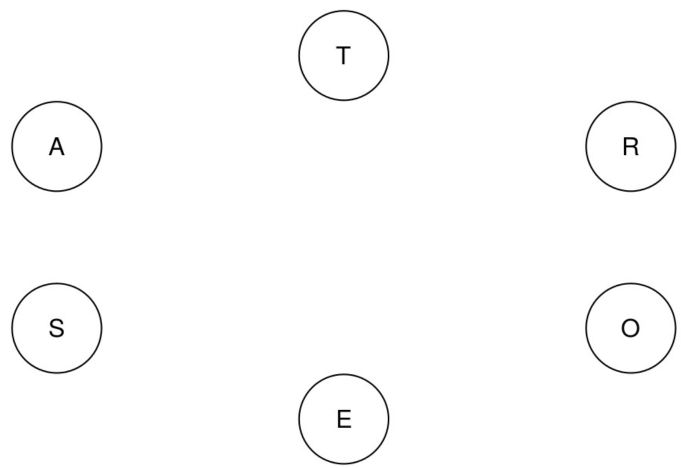
Number of Parameters: 3*2*2*2*2*3-1 = **143**

Survey is a atificial dataset including the answer of the users of several transports to a fictitious survey. For each user, six variables have been collected:

- Age (**A**) of the user, grouped in **three** states: *Young*, ***Adult*** and ***Old***
- Sex (**S**) of the user: Male (**M**) and Female (**F**)
- Education (**E**) of the user: highest education level reached, ***high school*** and ***university***.
- Ocupation (**O**): ***employee*** and ***self-employed***.
- Residence (**R**): population of the town, ***big*** or ***small***, in which the user lives.
- Transport (**T**): most used transport: ***car, train*** and ***other***.

Creating an empty graph:

```
library(bnlearn)
dag<-empty.graph(nodes=c("A","S","E","O","R","T"))
class(dag)
print(dag)
plot(dag)
```



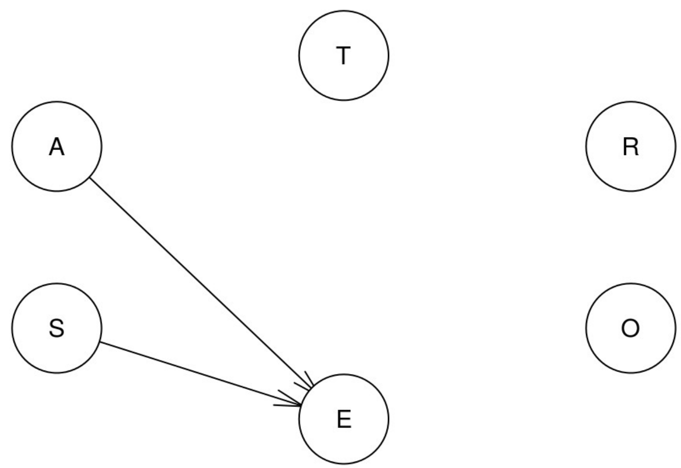
Survey is a atificial dataset including the answer of the users of several transports to a fictitious survey. For each user, six variables have been collected:

- Age (**A**) of the user, grouped in **three** states: *Young*, ***Adult*** and ***Old***
- Sex (**S**) of the user: Male (**M**) and Female (**F**)
- Education (**E**) of the user: highest education level reached, ***high school*** and ***university***.
- Ocupation (**O**): ***employee*** and ***self-employed***.
- Residence (**R**): population of the town, ***big*** or ***small***, in which the user lives.
- Transport (**T**): most used transport: ***car, train*** and ***other***.

Creating an empty graph:

```
library(bnlearn)
dag<-empty.graph(nodes=c("A","S","E","O","R","T"))
# Including edges
dag<-set.arc(dag,from="A",to="E")
dag<-set.arc(dag,from="S",to="E")
print(dag)
plot(dag)
```

Based on our expertise¡¡¡

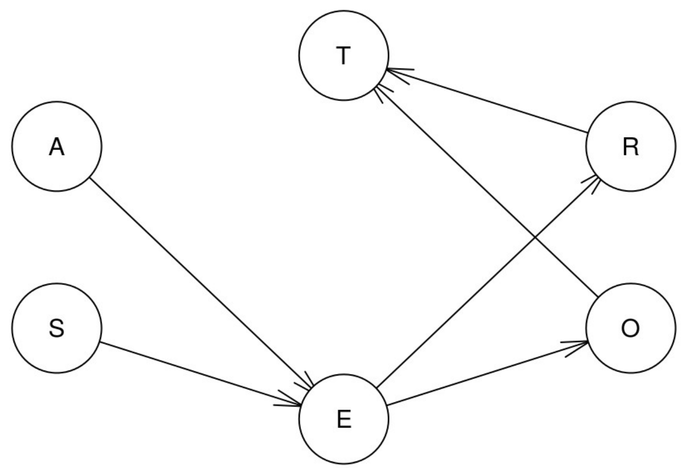


Survey is a atificial dataset including the answer of the users of several transports to a fictitious survey. For each user, six variables have been collected:

- Age (**A**) of the user, grouped in **three** states: *Young*, ***Adult*** and ***Old***
- Sex (**S**) of the user: Male (**M**) and Female (**F**)
- Education (**E**) of the user: highest education level reached, ***high school*** and ***university***.
- Ocupation (**O**): ***employee*** and ***self-employed***.
- Residence (**R**): population of the town, ***big*** or ***small***, in which the user lives.
- Transport (**T**): most used transport: ***car, train*** and ***other***.

Impose the rest of edges:

```
library(bnlearn)
dag<-empty.graph(nodes=c("A","S","E","O","R","T"))
# Including edges
dag<-set.arc(dag,from="A",to="E")
dag<-set.arc(dag,from="S",to="E")
dag<-set.arc(dag,from="E",to="O")
dag<-set.arc(dag,from="E",to="R")
dag<-set.arc(dag,from="R",to="T")
dag<-set.arc(dag,from="O",to="T")
print(dag);plot(dag)
```



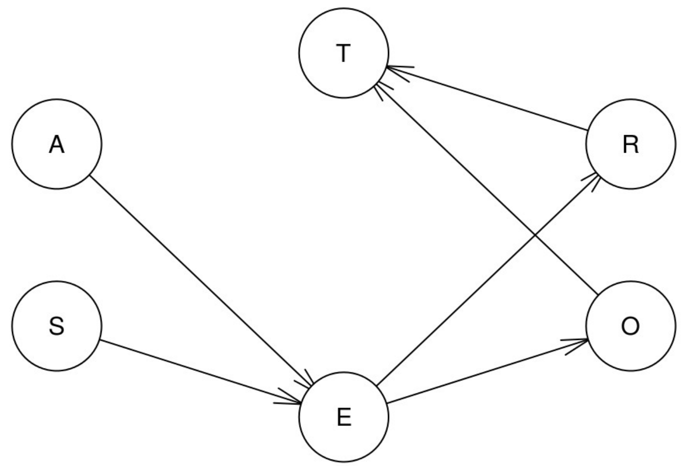
Expert approach

Survey is a atificial dataset including the answer of the users of several transports to a fictitious survey. For each user, six variables have been collected:

- Age (**A**) of the user, grouped in **three** states: *Young*, ***Adult*** and ***Old***
- Sex (**S**) of the user: Male (**M**) and Female (**F**)
- Education (**E**) of the user: highest education level reached, ***high school*** and ***university***.
- Ocupation (**O**): ***employee*** and ***self-employed***.
- Residence (**R**): population of the town, ***big*** or ***small***, in which the user lives.
- Transport (**T**): most used transport: ***car, train*** and ***other***.

Impose the rest of edges:

```
# Factorization
modelstring(dag)
# Properties
nodes(dag)
arcs(dag)
print(dag)
```

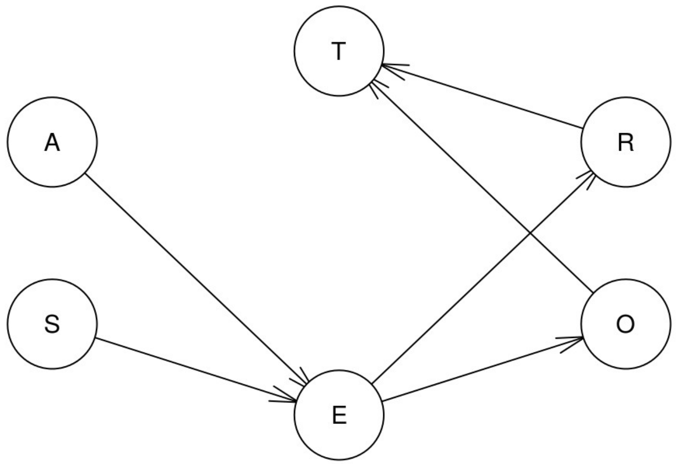


Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
# States of the variables:
estados.A <- c("young", "adult", "old")
estados.S <- c("M", "F")
estados.E <- c("high", "uni")
estados.O <- c("emp","self")
estados.R <- c("small","big")
estados.T <- c("car","train","other")
```

143 Parameters >> 6 Tables

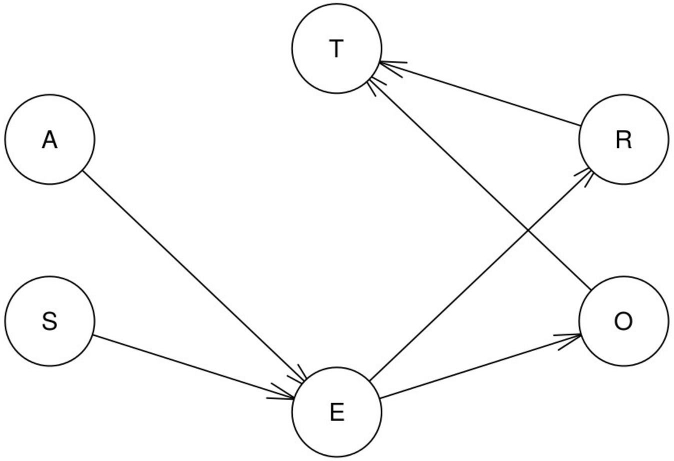


Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
# States of the variables:
estados.A <- c("young", "adult", "old")
estados.S <- c("M", "F")
estados.E <- c("high", "uni")
estados.O <- c("emp","self")
estados.R <- c("small","big")
estados.T <- c("car","train","other")
# Independent variables: A and S
A.prob <- array(c(.3, .5, .2), dim = 3, dimnames = list(A = estados.A))
S.prob <- array(c(.6, .4), dim = 2, dimnames = list(S = estados.S))
A.prob
```

143 Parameters >> 6 Tables



Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

143 Parameters >> 6 Tables

```
# States of the variables:
estados.A <- c("young", "adult", "old")
estados.S <- c("M", "F")
estados.E <- c("high", "uni")
estados.O <- c("emp","self")
estados.R <- c("small","big")
estados.T <- c("car","train","other")
# Independent variables: A and S
A.prob <- array(c(.3, .5, .2), dim = 3, dimnames = list(A = estados.A))
S.prob <- array(c(.6, .4), dim = 2, dimnames = list(S = estados.S))
A.prob
# Conditional probabilities
O.prob <- array(c(.96,.04,.92,.08), dim = c(2,2),
                 dimnames = list(O = estados.O, E = estados.E))
O.prob
R.prob <- array(c(.25,.75,.2,.8), dim = c(2,2),
                  dimnames = list(R = estados.R, E = estados.E))
E.prob <- array(c(.75, .25, .72, .28, .88, .12, .64, .36, .70, .30, .90, .10),
                  dim = c(2, 3, 2), dimnames = list(E = estados.E, A = estados.A,
                        S = estados.S))
T.prob <- array(c(.48, .42, .1, .56, .36, .08, .58, .24, .18, .7, .21, .09),
                  dim = c(3,2,2), dimnames = list(T = estados.T,
                        O = estados.O, R = estados.R))
```

Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

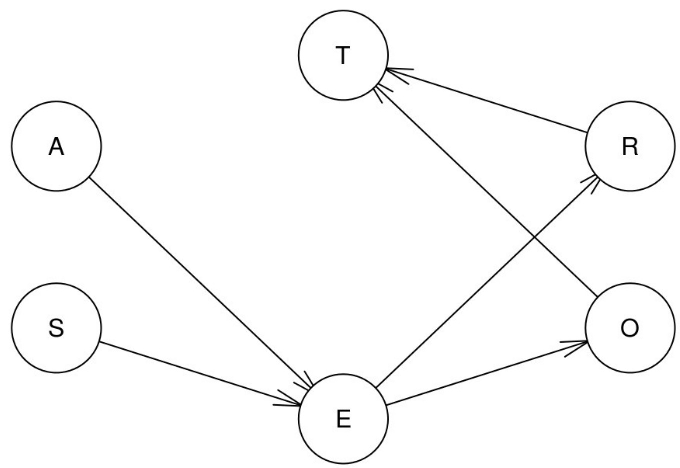
```
# Conditional Probability Tables:
cpt <- list(A = A.prob, S = S.prob, E = E.prob, O = O.prob, R = R.prob, T = T.prob)
str(cpt)
# Bayesian Network: DAG + CPT
bn <- custom.fit(dag, cpt)
# Properties
nparams(bn)  ← 143 Parameters
arcs(bn)
nodes(bn)
parents(bn,"E")
children(bn,"A")
```

Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
# S and R d-separated?
dsep(dag, x = "S", y = "R")
# Is there a path between S and R?
path(dag, from = "S", to = "R")
```

← **DAG-Inference**



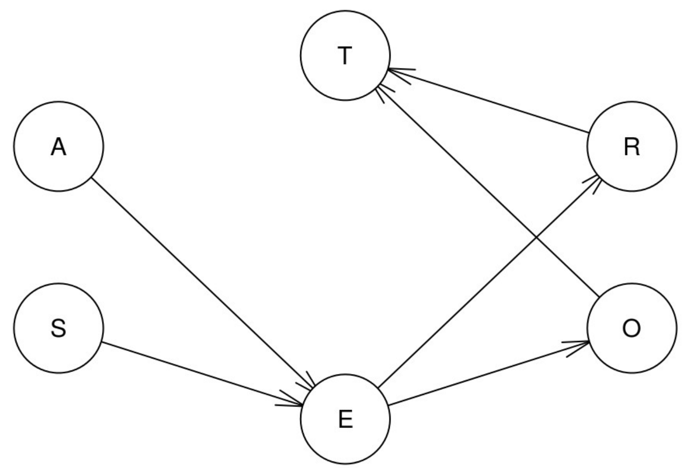
Once the DAG is defined and the corresponding factorization of the joint probability distribution is obtained, the needed conditional probabilities should be obtained for each factor:

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
# S and R d-separated?
dsep(dag, x = "S", y = "R")
# Is there a path between S and R?
path(dag, from = "S", to = "R")
# Given E, are S and R d-separated?
dsep(dag, x = "S", y = "R", z = "E")
```

← **DAG-Inference**

As we have seen, once ***E*** is known, the path between ***S*** and ***R*** is ***blocked*** and, as a result, both variables are ***d-separated given E.***



Exact inference is based on the ***junction tree*** that can be obtained directly ***"moralizing"*** the DAG of the Bayesian Network. **gRain** (**gRa**phical **in**ference) has implemented the exact inference

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
library(gRain) # gRain → Graphical Inference
# Create the Junction Tree and obtain the table of probabilities:
junction <- compile(as.grain(bn))
# Consulting the probabilities: Marginal Probability of the Transport.
querygrain(junction, nodes = "T")$T
# Is the transport dependent on the sex?
jsex <- setEvidence(junction, nodes = "S", states = "F") # Sex - Female
querygrain(jsex, nodes = "T")$T
```

← **Without significant changes**

Exact inference is based on the ***junction tree*** that can be obtained directly ***"moralizing"*** the DAG of the Bayesian Network. **gRain** (**gRa**phical **in**ference) has implemented the exact inference

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

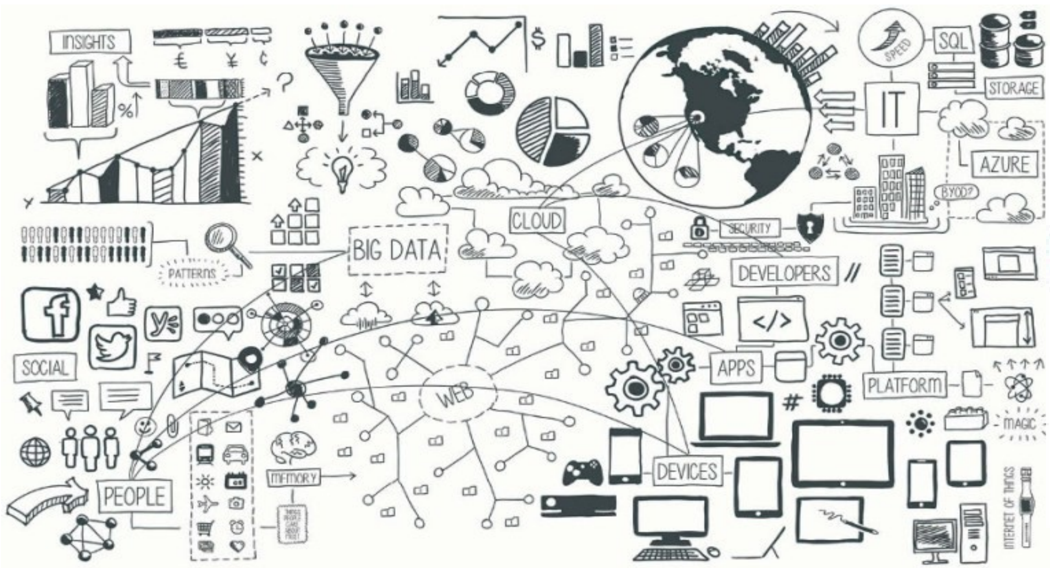
```
library(gRain) # gRain → Graphical Inference
# Create the Junction Tree and obtain the table of probabilities:
junction <- compile(as.grain(bn))
# Consulting the probabilities: Marginal Probability of the Transport.
querygrain(junction, nodes = "T")$T
# Is the transport dependent on the sex?
jsex <- setEvidence(junction, nodes = "S", states = "F") # Sex - Female
querygrain(jsex, nodes = "T")$T    ← Without significant changes
# Is the transport dependent on the town's size?
jres <- setEvidence(junction, nodes = "R", states = "small") # Residence - small
querygrain(jres, nodes = "T")$T    ← Significant changes
```

Exact inference is based on the ***junction tree*** that can be obtained directly ***"moralizing"*** the DAG of the Bayesian Network. **gRain** (**gRa**phical **in**ference) has implemented the exact inference

$$P(A,S,E,O,R,T)=P(A)P(S)P(E|A,S)P(O|E)P(R|E)P(T|O,R)$$

```
library(gRain) # gRain → Graphical Inference
# Create the Junction Tree and obtain the table of probabilities:
junction <- compile(as.grain(bn))
# Consulting the probabilities: Marginal Probability of the Transport.
querygrain(junction, nodes = "T")$T
# Is the transport dependent on the sex?
jsex <- setEvidence(junction, nodes = "S", states = "F") # Sex - Female
querygrain(jsex, nodes = "T")$T    ← Without significant changes
# Is the transport dependent on the town's size?
jres <- setEvidence(junction, nodes = "R", states = "small") # Residence - small
querygrain(jres, nodes = "T")$T    ← Significant changes

# Inexact simulation (see http://www.bnlearn.com/documentation/man/cpquery.html)
cpquery(bn,(T=="car"),(S=="F"))
cpquery(bn,(T=="car"),(R=="small"))
```

Machine Learning II
Discrete Bayesian Networks (Learning)

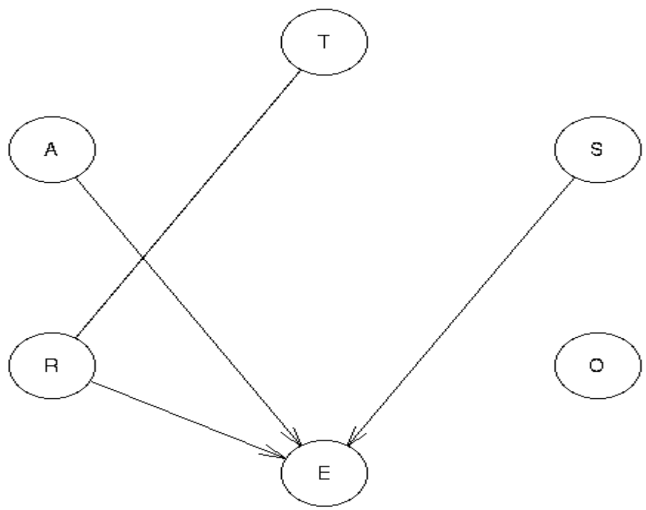


Sixto Herrera (herreras@unican.es)

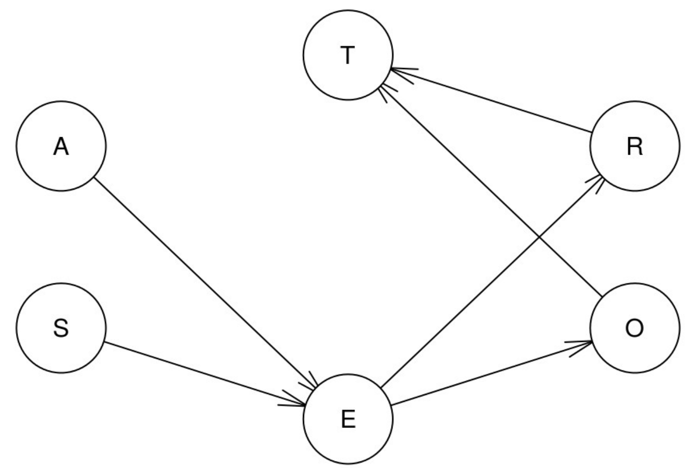
Grupo de Meteorología

Univ. de Cantabria – CSIC
MACC / IFCA

```
bnlearn implements the following constraint-based structure learning algorithms:
    PC (pc.stable)
    Grow-Shrink (gs);
    Incremental Association Markov Blanket (iamb);
    Fast Incremental Association (fast.iamb);
    Interleaved Incremental Association (inter.iamb);
    Max-Min Parents & Children (mmpc);
    Semi-Interleaved Hiton-PC (si.hiton.pc);
 (more details in http://www.bnlearn.com/examples/whitelist/)
# Loading the dataset:
survey <- read.table("survey.txt", header = TRUE)
dag.PC <-pc.stable(survey)
plot(dag)
plot(dag.PC)
```



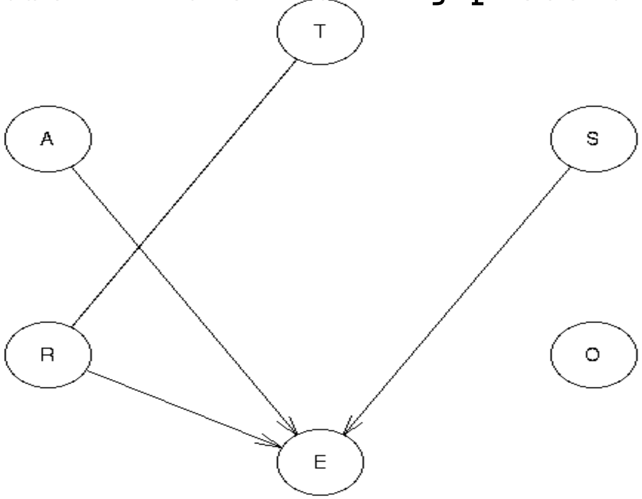
PC-Algorithm



Expert approach

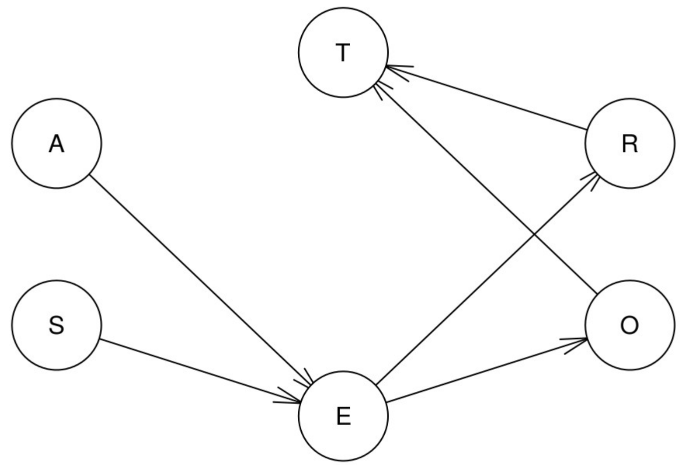
```
print(dag);print(dag.PC)
```

```
B-N learned via Constraint-based methods

model: [partially directed graph]
nodes:                                  6
arcs:                                   4
  undirected arcs:                      1
  directed arcs:                        3
average markov blanket size:            2.33
average neighbourhood size:             1.33
average branching factor:               0.50

learning algorithm:               PC (Stable)
conditional independence test: M.I(disc.)
alpha threshold: 0.05; optimized: FALSE
tests used in the learning procedure:  58
```



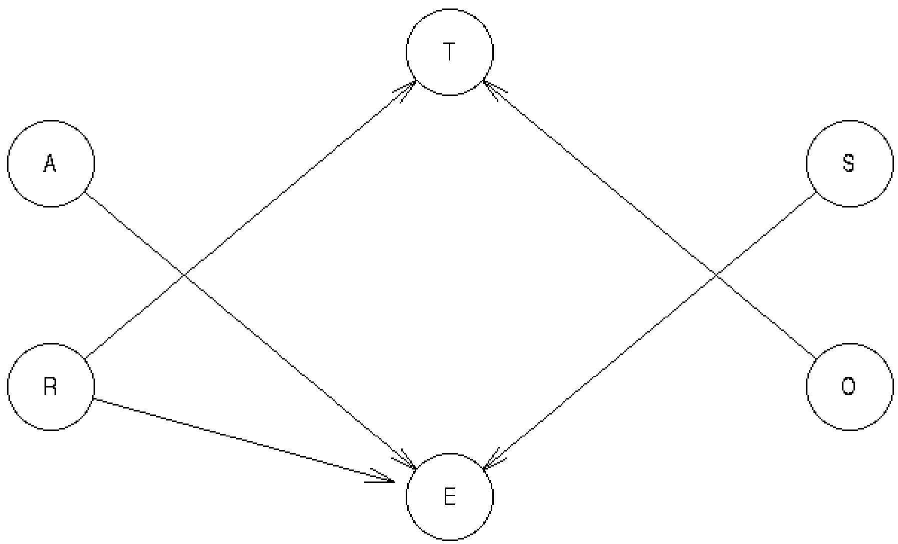
PC-Algorithm

```
Random/Generated Bayesian network

Model: [A][S][E|A:S][O|E][R|E][T|O:R]
nodes:                                      6
arcs:                                       6
  undirected arcs:                          0
  directed arcs:                            6
average markov blanket size:                2.67
average neighbourhood size:                 2.00
average branching factor:                   1.00

generation algorithm:                       Empty
```



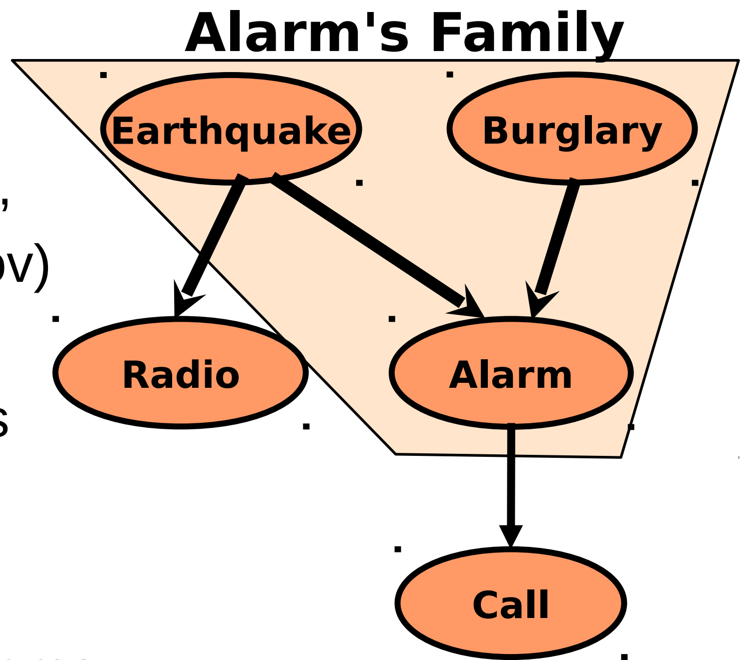
Expert approach

```
dag.PC1 <-pc.stable(survey, whitelist = c("O","T"))
plot(dag.PC1)
print(dag.PC1)
 B-N learned via Constraint-based methods

 model:  [A][R][O][S][E|A:R:S][T|R:O]
 nodes:                                 6
 arcs:                                  5
   undirected arcs:                     0
   directed arcs:                       5
 average markov blanket size:           3.00
 average neighbourhood size:            1.67
 average branching factor:              0.83

 learning algorithm:             PC (Stable)
 conditional independence test: M.I(disc.)
 alpha threshold: 0.05; optimized: FALSE
 tests used in the learning procedure:  58
```



PC-Algorithm + Expert Knowledge

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



Structure:

Acyclic Directed Graph (DAG),

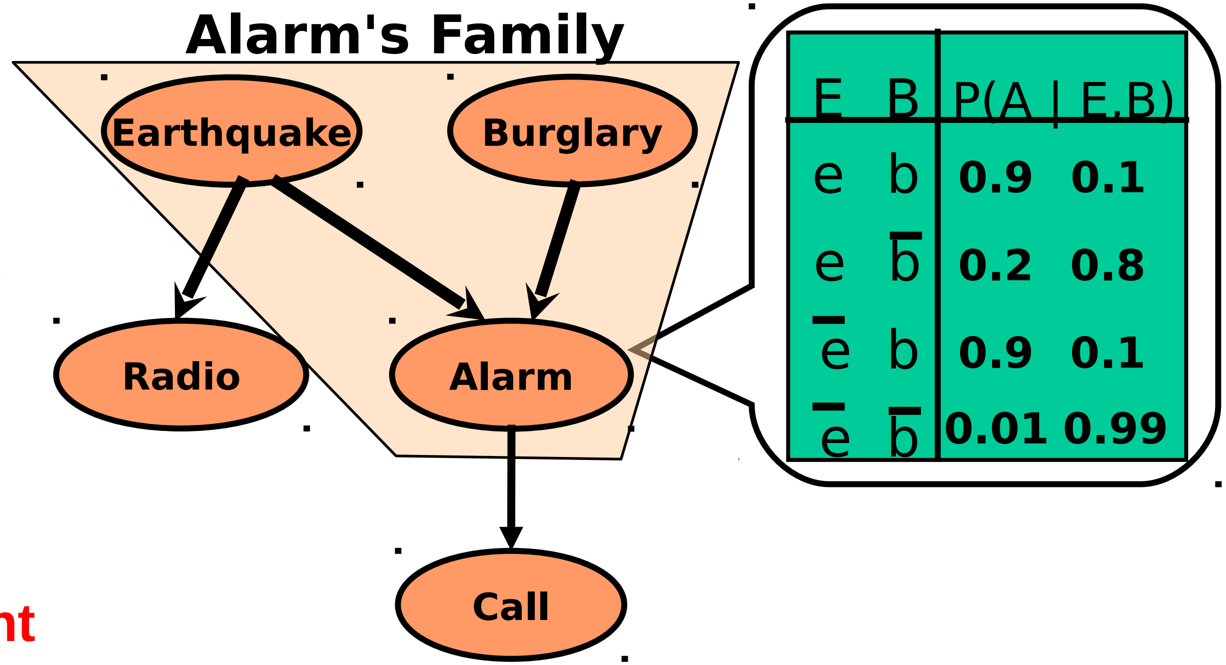
or non-directed graphs (Markov)

- 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...

```
score(dag, data = survey, type = "bde") # Expert approach
score(dag.PC1, data = survey, type = "bde") # Expert approach + PC-Algorithm
score(dag.PC, data = survey, type = "bde") # Error: DAG should be directed
```

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



Score:

P(M) - "A priori" information

P(data|M) – Explain capability

Model complexity

Bayesian Dirichlet Equivalent

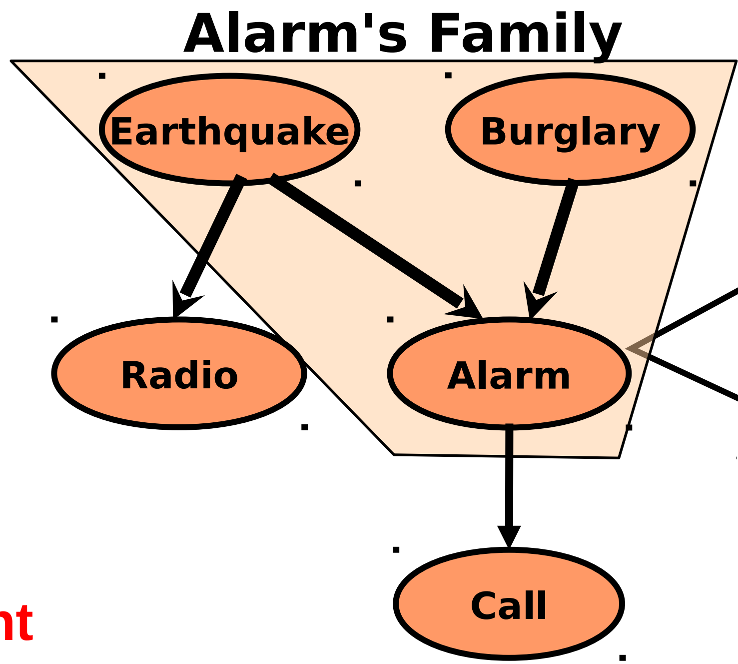
$$P(M|data)=\frac{P(data|M)P(M)}{P(data)} \Rightarrow BDe=\log(P(data|M))+\log(P(M))$$

Minimum Description Length // Bayesian Information Criterium

$$MDL(M|data)=\frac{r*\log(N)}{2}-\sum_{i=1}^{N}\log(P_{X_i}(X_i|\pi_i))$$

```
score(dag, data = survey, type = "bde") # Expert approach
score(dag.PC1, data = survey, type = "bde") # Expert approach + PC-Algorithm
score(dag.PC, data = survey, type = "bde") # Error: DAG should be directed
```

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



E	B	P(A | E,B)	
e	b	**0.9**	**0.1**
e	$\bar{b}$	**0.2**	**0.8**
$\bar{e}$	b	**0.9**	**0.1**
$\bar{e}$	$\bar{b}$	**0.01**	**0.99**

Score:

P(M) - “A priori” information

P(data|M) – Explain capability

Model complexity

Bayesian Dirichlet Equivalent

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

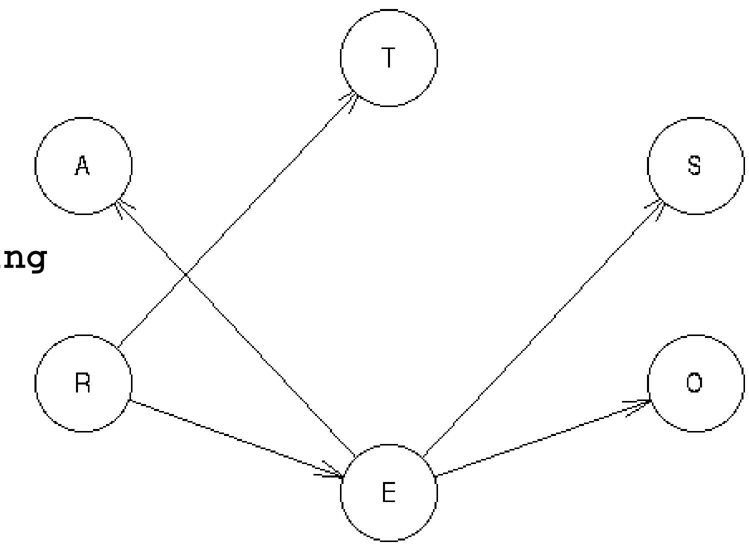
Minimum Description Length // Bayesian Information Criterium

$$MDL(M|data)=\frac{r*\log(N)}{2}-\sum_{i=1}^{N}\log(P_{X_i}(X_i|\pi_i))$$

```
dag.PC2 <- choose.direction(dag.PC, data=survey, c("R","T"), criterion="bde", debug=TRUE)
print(dag.PC2);plot(dag.PC2)
score(dag.PC2, data = survey, type = "bde") # Expert approach + PC-Algorithm
```

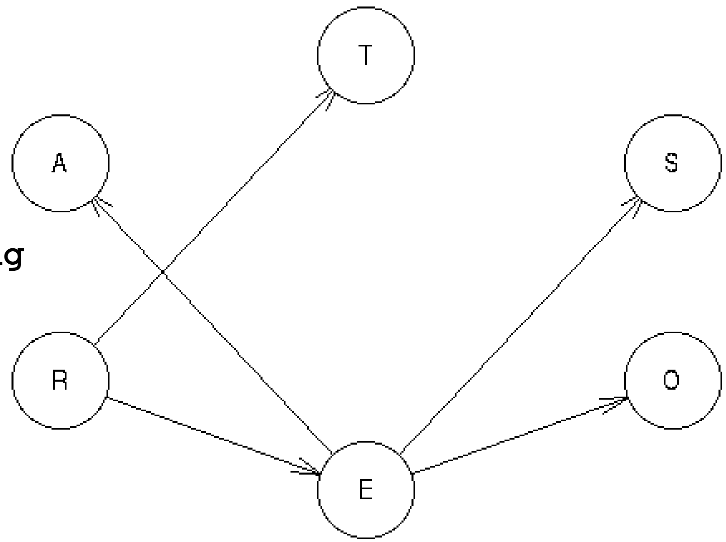
```
bnlearn implements the following score-based structure learning algorithms:
    Hill Climbing (hc);
    Tabu Search (tabu);
 (more details in http://www.bnlearn.com/examples/whitelist/)
# Loading the dataset:
survey <- read.table("survey.txt", header = TRUE)
dag.HC <- hc(survey, debug = TRUE)
plot(dag.HC)
print(dag.HC)
  B-N learned via Score-based methods

  model: [R][E|R][T|R][A|E][O|E][S|E]
  nodes:                                 6
  arcs:                                  5
    undirected arcs:                     0
    directed arcs:                       5
  average markov blanket size:           1.67
  average neighbourhood size:            1.67
  average branching factor:              0.83

  learning algorithm:                    Hill-Climbing
  score:                                 BIC (disc.)
  penalization coefficient:              3.107304
  tests used in the learning procedure:  40
  optimized:                             TRUE
```

HC-Algorithm



```
bnlearn implements the following score-based structure learning algorithms:
    Hill Climbing (hc);
    Tabu Search (tabu);
 (more details in http://www.bnlearn.com/examples/whitelist/)
# Loading the dataset:
survey <- read.table("survey.txt", header = TRUE)
dag.HC <- hc(survey, debug = TRUE)
plot(dag.HC)
print(dag.HC)
  B-N learned via Score-based methods

  model: [R][E|R][T|R][A|E][O|E][S|E]
  nodes:                                 6
  arcs:                                  5
    undirected arcs:                     0
    directed arcs:                       5
  average markov blanket size:           1.67
  average neighbourhood size:            1.67
  average branching factor:              0.83

  learning algorithm:                    Hill-Climbing
  score:                                 BIC (disc.)
  penalization coefficient:              3.107304
  tests used in the learning procedure:  40
  optimized:                             TRUE
```

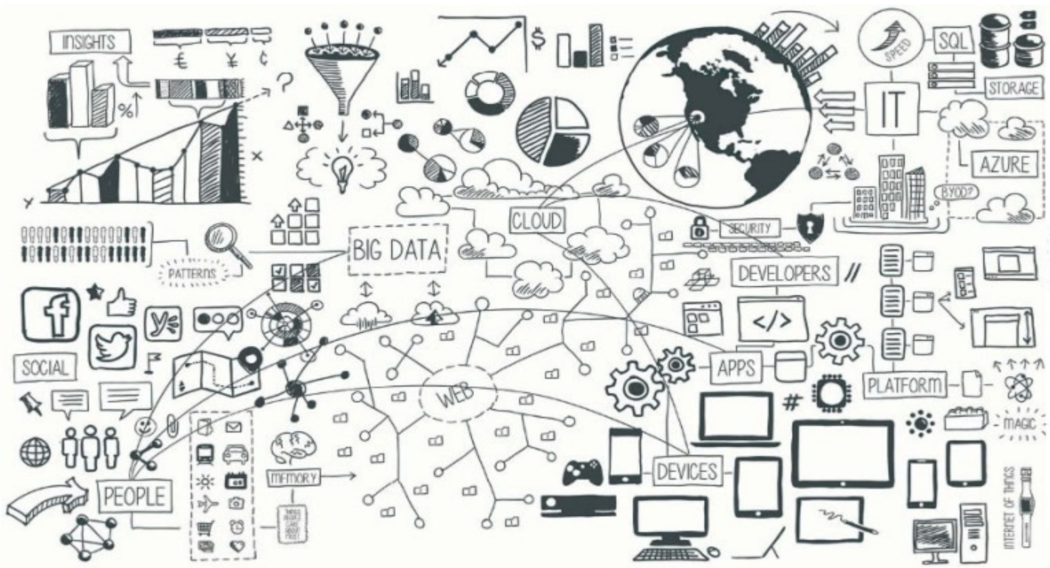
HC-Algorithm

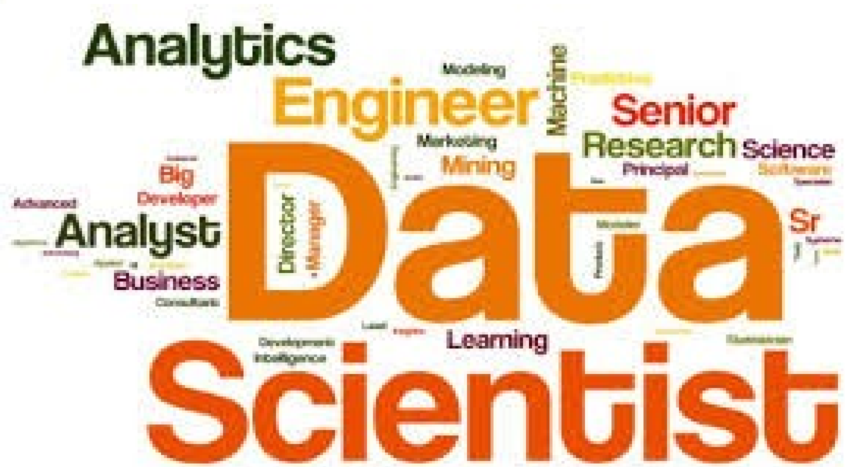



```
score(dag, data = survey, type = "bde") # Expert approach
score(dag.PC2, data = survey, type = "bde") # Expert approach + PC-Algorithm
score(dag.HC, data = survey, type = "bde") # HC-Algorithm
```

```
cpt.MLE <- bn.fit(dag, survey, method = "mle")
cpt.BAY <- bn.fit(dag, survey, method = "bayes")
str(cpt)
str(cpt.MLE)
str(cpt.BAY)
cpt$T                  || cpt.MLE$T                        || cpt.BAY$T
, , R = big            || , , R = big                      || , , R = big
       O               ||       O                          ||       O
T         emp  self    || T                emp        self || T                emp        self
  car    0.58 0.70     ||   car    0.58469945 0.69230769 ||   car    0.58452787 0.68553459
  train 0.24 0.21      ||   other 0.19945355 0.15384615 ||   other 0.19954494 0.15723270
  other 0.18 0.09      ||   train 0.21584699 0.15384615 ||   train 0.21592719 0.15723270
, , R = small          ||                                  ||
                       || , , R = small                    || , , R = small
       O               ||       O                          ||      O
T         emp  self    || T                emp        self || T                emp        self
  car    0.48 0.56     ||   car    0.54700855 0.75000000 ||   car    0.54655295 0.72549020
  train 0.42 0.36      ||   other 0.07692308 0.25000000 ||   other 0.07746979 0.25490196
  other 0.10 0.08      ||   train 0.37606838 0.00000000 ||   train 0.37597726 0.01960784

cpt.PC1.MLE <- bn.fit(dag.PC1, survey, method = "mle")
cpt.PC1.BAY <- bn.fit(dag.PC1, survey, method = "bayes")
cpt.PC2.MLE <- bn.fit(dag.PC2, survey, method = "mle")
cpt.PC2.BAY <- bn.fit(dag.PC2, survey, method = "bayes")
cpt.HC.MLE <- bn.fit(dag.HC, survey, method = "mle")
cpt.HC.BAY <- bn.fit(dag.HC, survey, method = "bayes")
```

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

$P(\mathbf{x})=P(x_1, \ldots, x_n)$

Variables discretas

PROBABILITY DISTRIBUTION

$\boldsymbol{P}$ on $\mathcal{X}$

TRAINING EXAMPLES

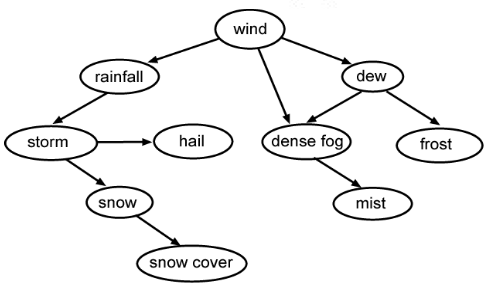
$\mathbf{x}_1, \ldots, \mathbf{x}_N$

$\mathbf{x}_1, \ldots, \mathbf{x}_N$

Parameter Learning: to obtain the conditional probabilities based on the edges of the graph.

Structure Learning: to obtain the edges of the graph.

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P_i(x_i | \pi_i)$$



Graphical Probabilistic Models

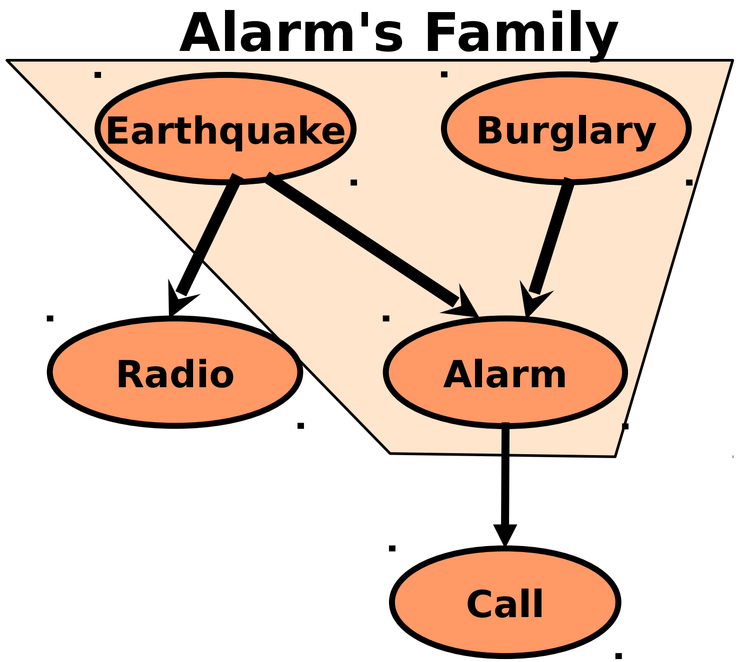
EL FAMILY

$\mathcal{H}$

OUTPUT

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.

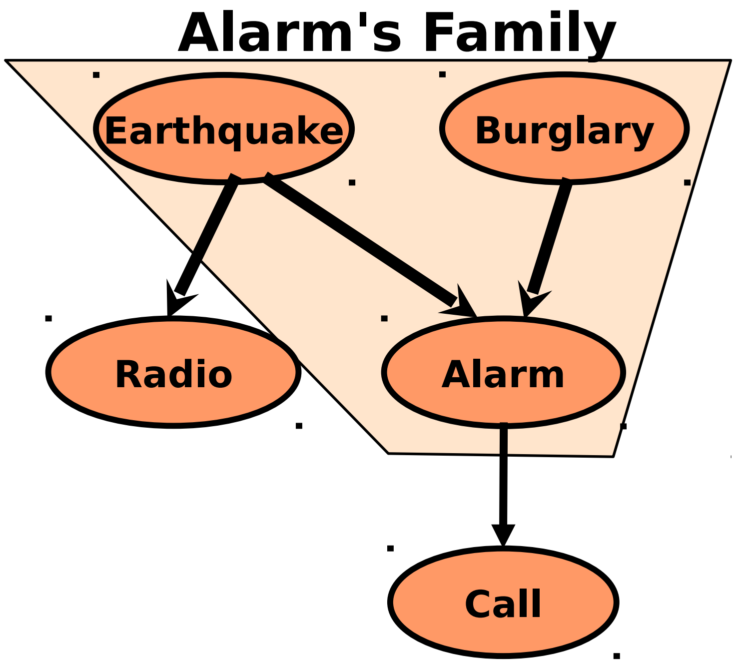


← **Multivariate Normal Distribution** $N\left(\boxed{\mu},\boxed{\Sigma}\right)$

Mean vector

Covariance Matrix

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



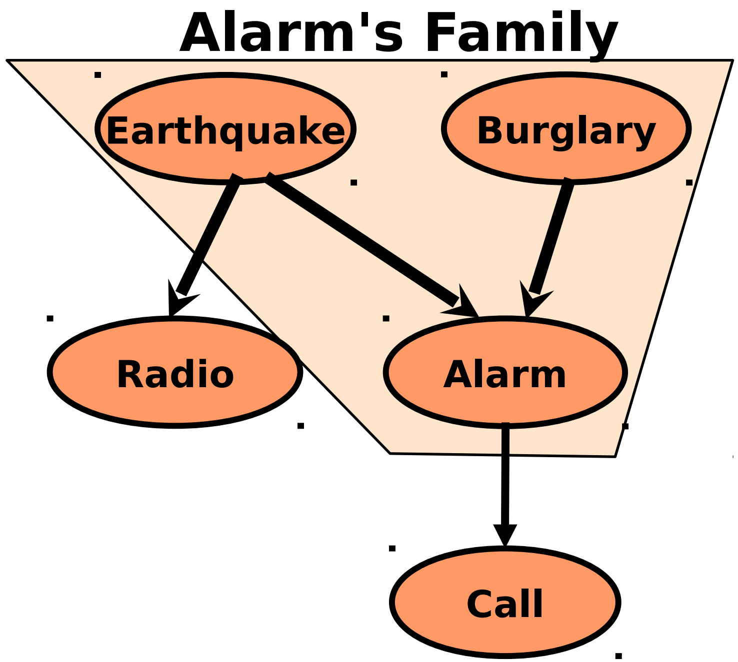
← **Multivariate Normal Distribution** $N(\mu, \Sigma)$

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

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** $N(\mu, \Sigma)$

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

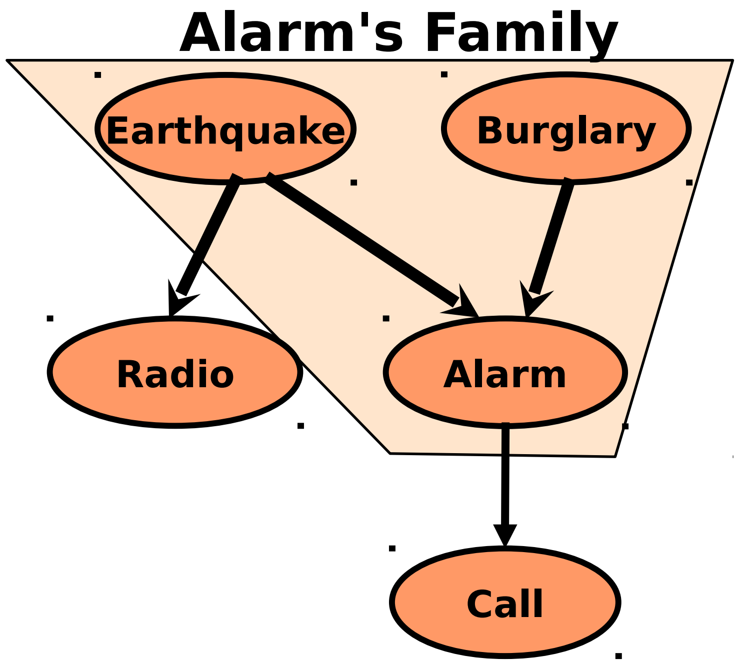
Factorization

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

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

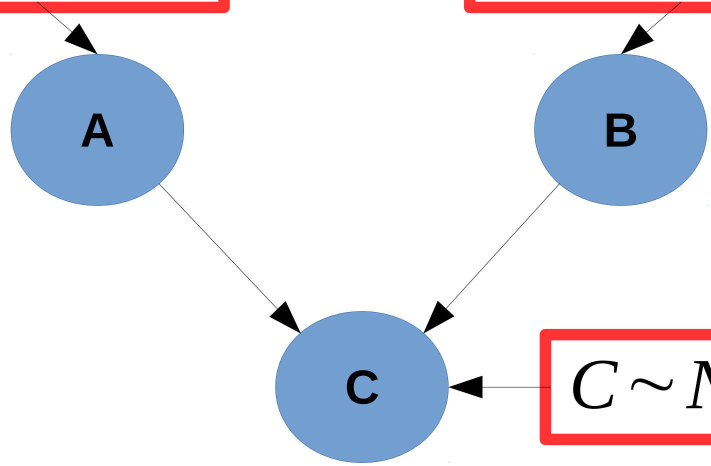
$$f(x|\pi_i)\sim N\left(\mu_i+\sum_{j=1}^{i-1}\boxed{\beta_{ij}}(x_j-\mu_j),\boxed{\nu_i}\right)$$

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

Conditional Variance

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

$$\boxed{A \sim N(2,1)} \qquad \boxed{B \sim N(1,1.5)}$$



$$\boxed{C \sim N(0.5+0.75(x_A-2)+1.32(x_B-1),0.4)}$$

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

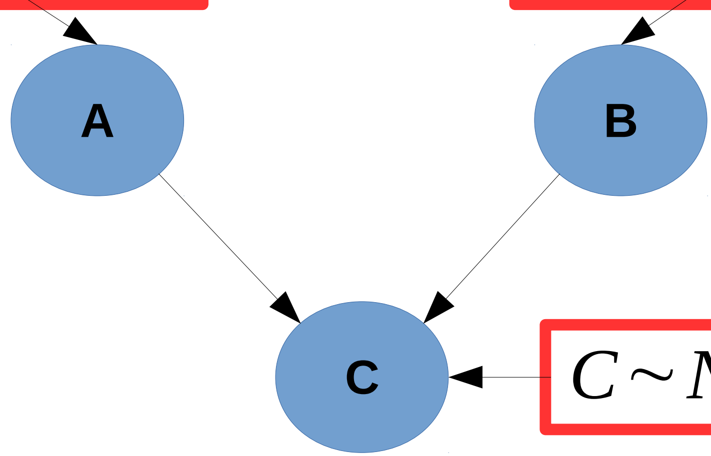
Conditional (In)dependence ⇐ **Regression Coefficients**

```
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.

$$\boxed{N(\mu_A, \nu_A)} \qquad \boxed{N(\mu_B, \nu_B)}$$



$$\boxed{C \sim N(\mu_C + \beta_{AC}(x_A - \mu_A) + \beta_{BC}(x_B - \mu_B), \nu_C)}$$

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

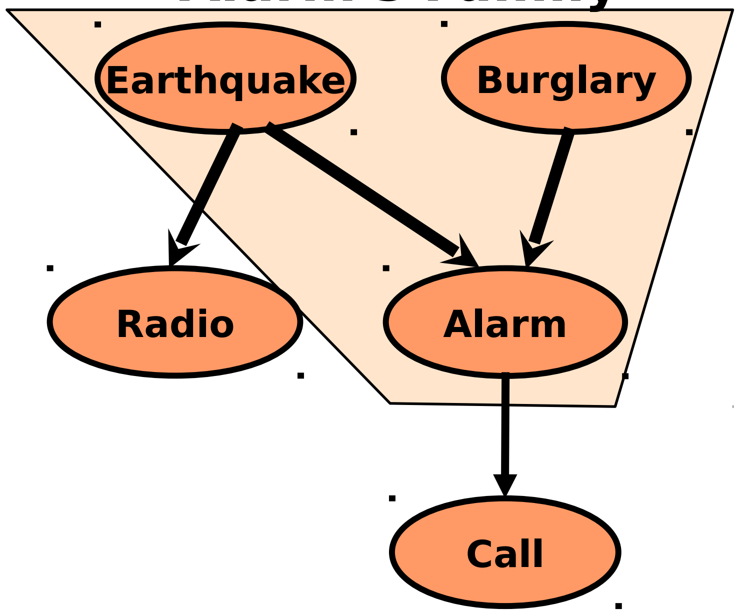
Conditional (In)dependence ⟵ **Regression Coefficients**

← 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.

Alarm's Family



← **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

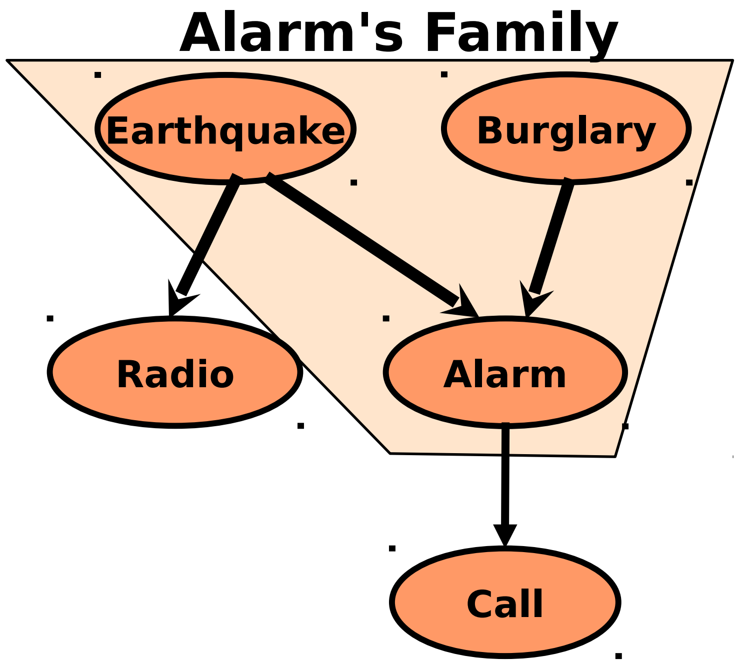
$$f(x|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \boxed{\beta_{ij}} (x_j - \mu_j), \nu_i\right)$$

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

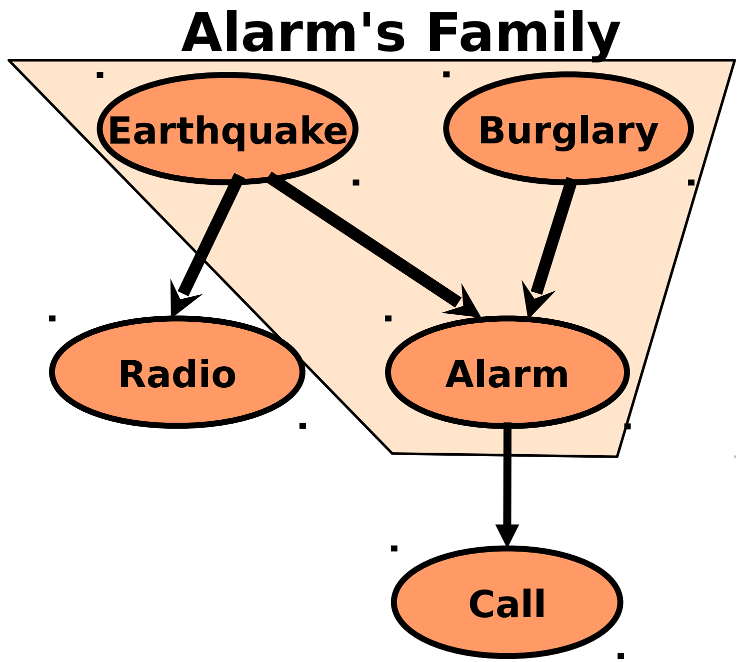
$$f(x|\pi_i) \sim N(\mu_i + \sum_{j=1}^{i-1} \boxed{\beta_{ij}} (x_j - \mu_j), \nu_i)$$

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)

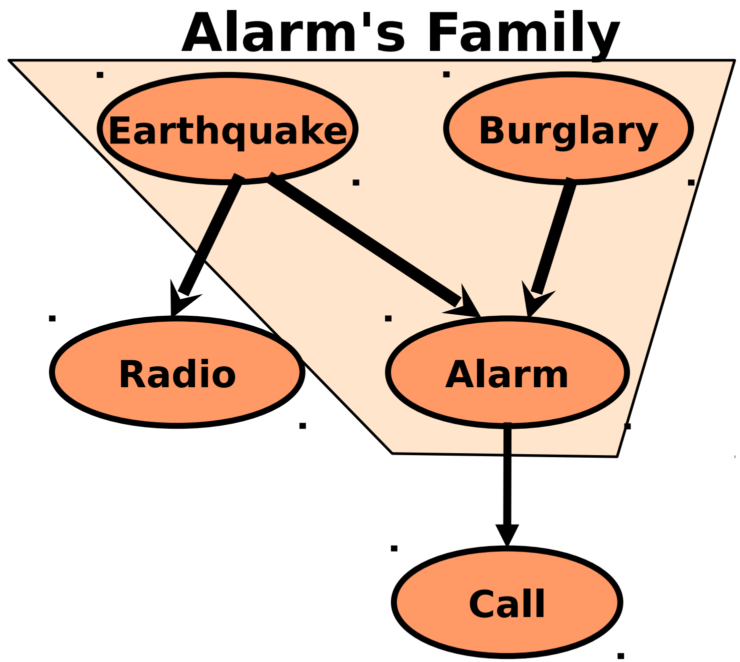
- **Nodes** – discrete and continuous
- **Links** – direct dependences

Joint Distribution is Conditional Gaussian.

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

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.

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

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

7. MÉTODOS DE LA EVALUACIÓN				
Descripción	Tipología	Eval. Final	Recuper.	%
Valoración de informes y trabajos escritos	Actividad de evaluación con soporte virtual	Sí	Sí	60,00

Calif. mínima	3,00
Duración	
Fecha realización	Durante el periodo de impartición de la asignatura.
Condiciones recuperación	
Observaciones	Evaluación de los trabajos de grupo e individuales entregados por el alumno.

M1970 – Machine Learning (L 16:00-18:00; X 16:00-18:00)

Mar	4	L	Redes Probabilísticas Discretas (2h-T)
	6	X	Redes Bayesianas: Creación e Inferencia (2h-L)
	11	L	Clasificacidores Bayesianos. Naive Bayes (2h-L)
	13	X	Redes Bayesianas: Aprendizaje (2h-T)
	18	L	Redes Bayesianas: Aprendizaje (2h-L)
	20	X	Redes Bayesianas: Aprendizaje (2h-L)
	25	L	Evaluación (2h)

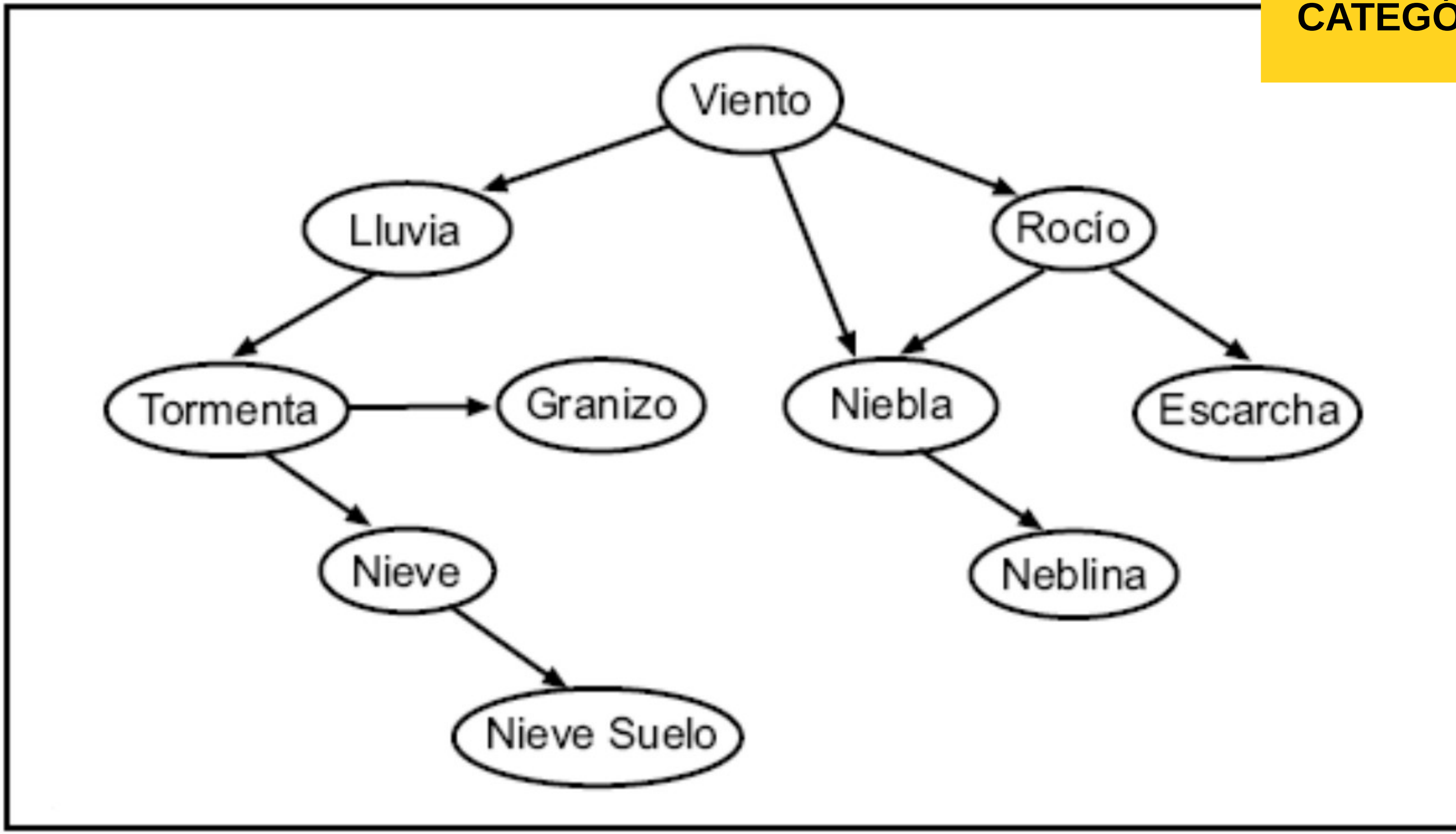
T01 – Redes Bayesianas

actividad recuperable será equivalente al de la actividad original.

Observaciones para alumnos a tiempo parcial

A nivel global, el valor de esta tarea se corresponde con el 30% de la nota final

CATEGÓRICO



Lluvia	nieve	granizo	tormenta	niebla	rocio	escarcha	nieveSuelo	neblina	viento
s	n	n	n	n	n	n	n	n	s
s	n	n	n	n	n	n	n	n	s
s	n	n	s	n	n	n	n	n	s
s	n	n	n	n	n	n	n	n	s