Bayesian learning
(with a recap of Bayesian Networks)

Applied artificial intelligence (EDA132)
Lecture 05
2016-02-02
Elin A. Topp

Material based on course book, chapters 14.1-3, 20,
and on Tom M. Mitchell, "Machine Learning", McGraw-Hill, 1997

Bayesian networks

A simple, graphical notation for *conditional independence assertions* and hence for compact specification of full joint distributions

Syntax:

a set of nodes, one per random variable

a directed, acyclic graph (link ≈ "directly influences")

a conditional distribution for each node given its parents:

$\mathbf{P}(X_i \mid Parents(X_i))$

In the simplest case, conditional distribution represented as a

conditional probability table (CPT)

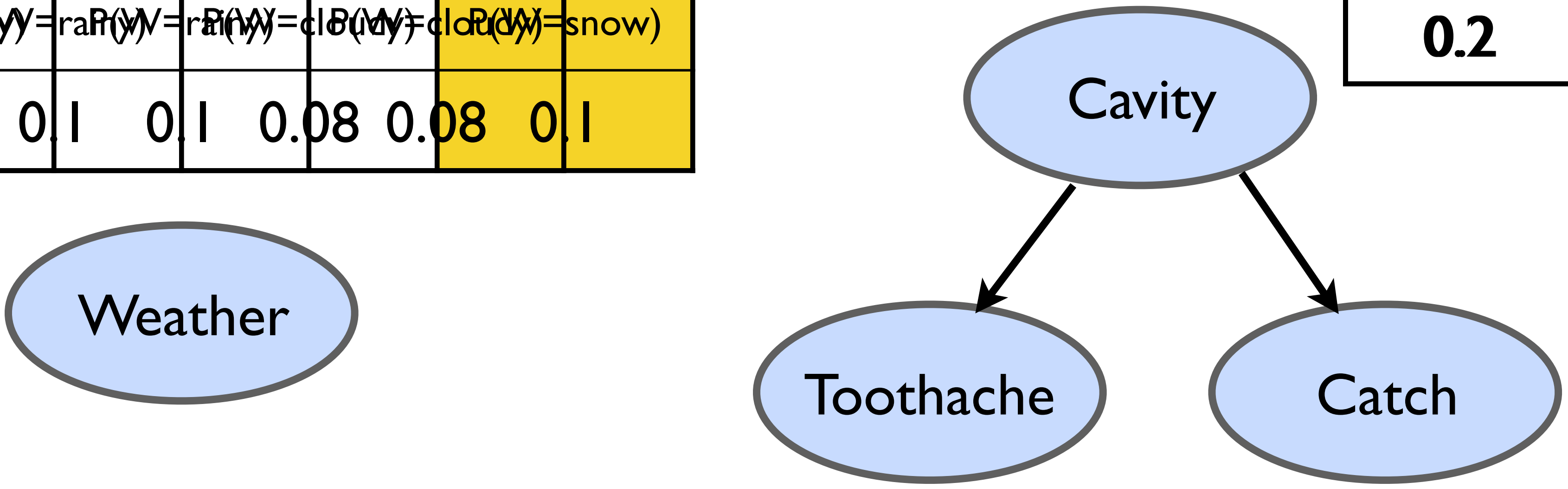
giving the distribution over X_i for each combination of parent values

Example

Topology of network encodes conditional independence assertions:

P(W=sunny)	P(W=rainy)	P(W=cloudy)	P(W=snow)
0.72	0.1	0.08	0.1

P(Cav)	P(¬Cav)
0.2	0.8



Cav	P(T|Cav)	P(¬T|Cav)
T	0.6	0.4
F	0.1	0.9

Cav	P(C|Cav)	P(¬C|Cav)
T	0.9	0.1
F	0.2	0.8

Weather is (unconditionally, absolutely) independent of the other variables

Toothache and *Catch* are conditionally independent given *Cavity*

We can skip the dependent columns in the tables to reduce complexity!

Example 2

I am at work, my neighbour John calls to say my alarm is ringing, but neighbour Mary does not call.

Sometimes the alarm is set off by minor earthquakes.

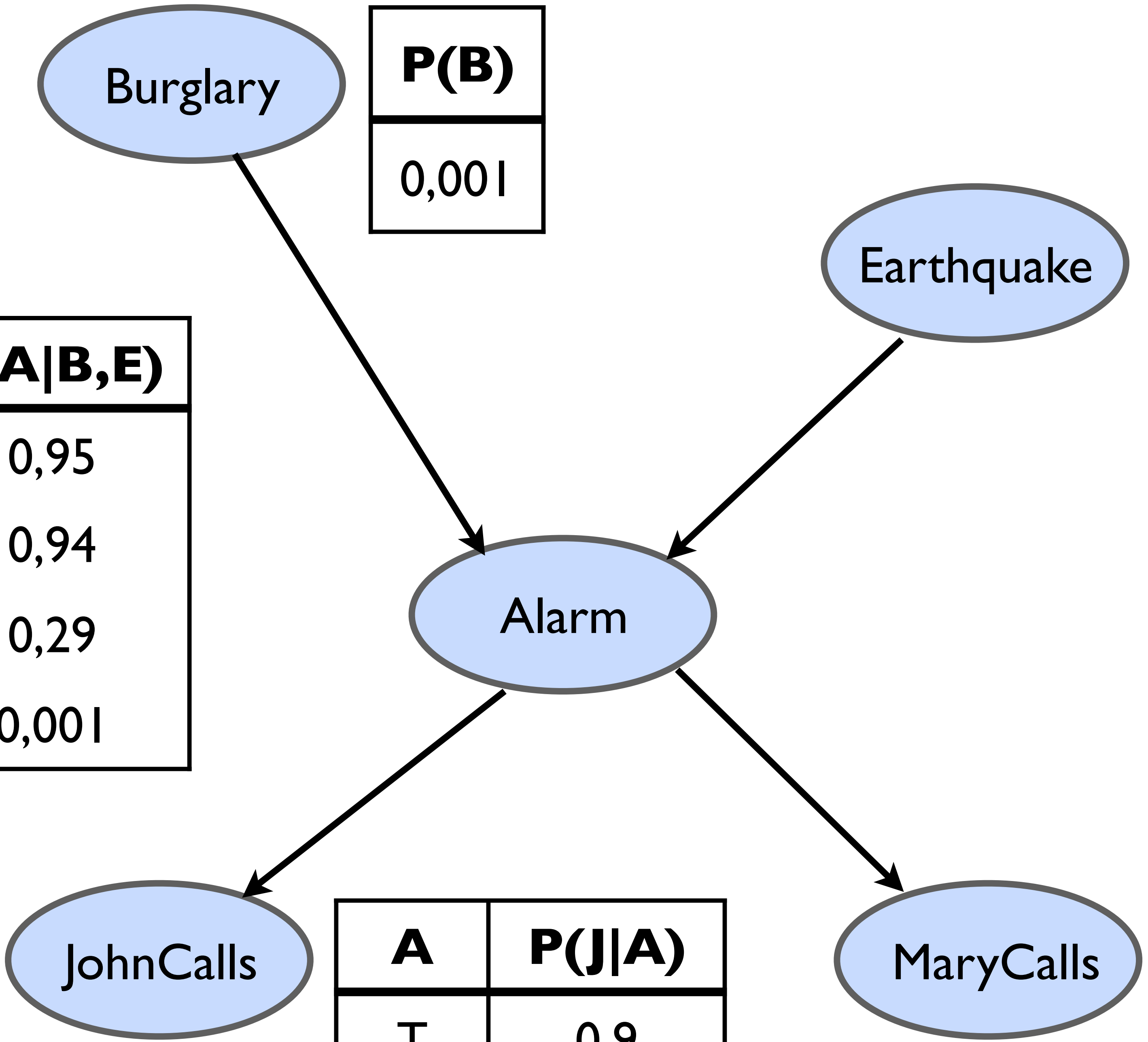
Is there a burglar?

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*

Network topology reflects "causal" knowledge:

> A burglar can set the alarm off
>
> An earthquake can set the alarm off
>
> The alarm can cause John to call
>
> The alarm can cause Mary to call

Example 2 (2)



P(B)
0,001

P(E)
0,002

B	E	P(A|B,E)
T	T	0,95
T	F	0,94
F	T	0,29
F	F	0,001

A	P(J|A)
T	0,9
F	0,05

A	P(M|A)
T	0,7
F	0,01

Global semantics

Global semantics defines the full joint distribution as the product of the local conditional distributions:

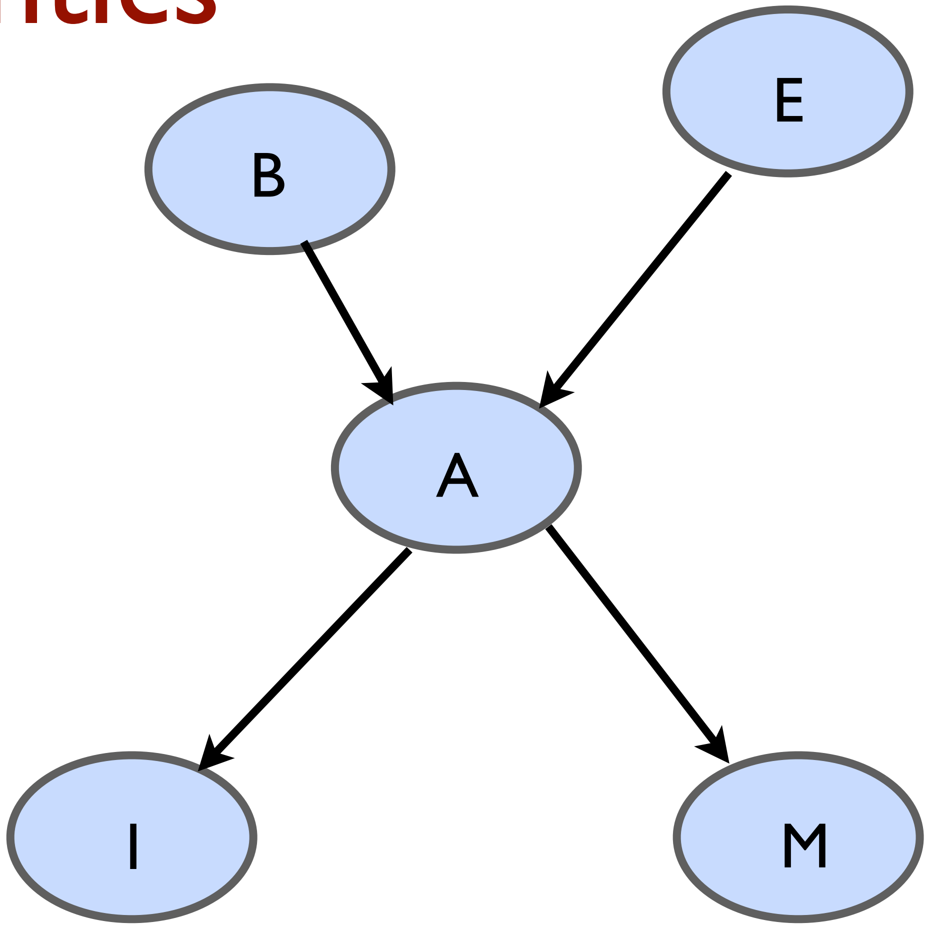
$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid parents(X_i))$$

E.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$= P(j \mid a)\, P(m \mid a)\, P(a \mid \neg b, \neg e)\, P(\neg b)\, P(\neg e)$

$= 0.9 * 0.7 * 0.001 * 0.999 * 0.998$

≈ 0.000628



Constructing Bayesian networks

We need a method such that a series of locally testable assertions of conditional independence guarantees the required global semantics.

1. Choose an ordering of variables $X_1, \ldots, X_n$

2. For $i = 1$ to n

    add X_i to the network

    select parents from $X_1, \ldots, X_{i-1}$ such that

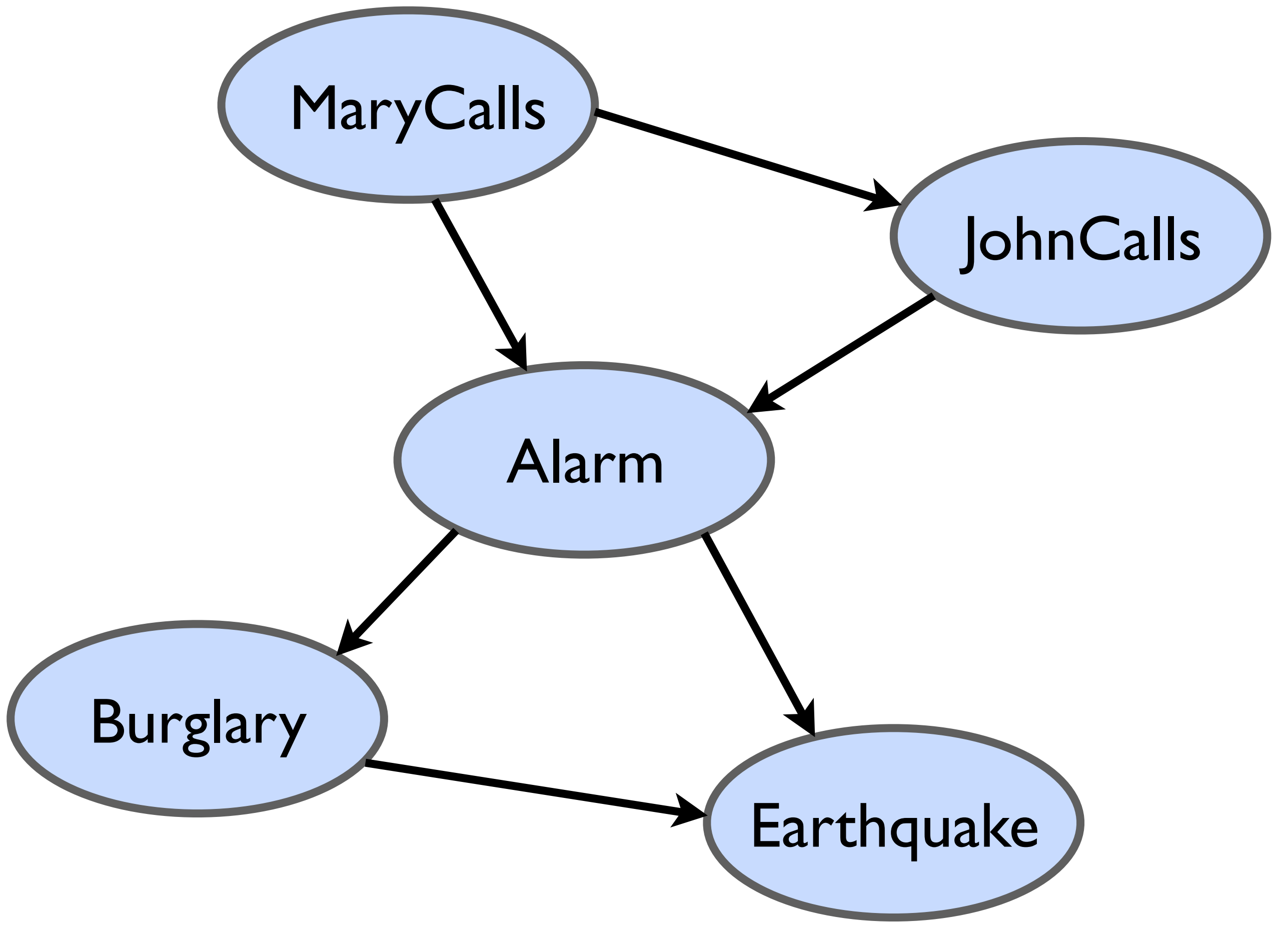
    $$\mathbf{P}(X_i \mid Parents(X_i)) = \mathbf{P}(X_i \mid X_1, \ldots, X_{i-1})$$

This choice of parents guarantees the global semantics:

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i \mid X_1, \ldots, X_{i-1}) \qquad \text{(chain rule)}$$

$$= \prod_{i=1}^{n} \mathbf{P}(X_i \mid Parents(X_i)) \qquad \text{(by construction)}$$

Construction example



Deciding conditional independence is hard in noncausal directions

(Causal models and conditional independence seem hardwired for humans!)

Assessing conditional probabilities is hard in noncausal directions

Network is less compact: $1 + 2 + 4 + 2 + 4 = 13$ numbers

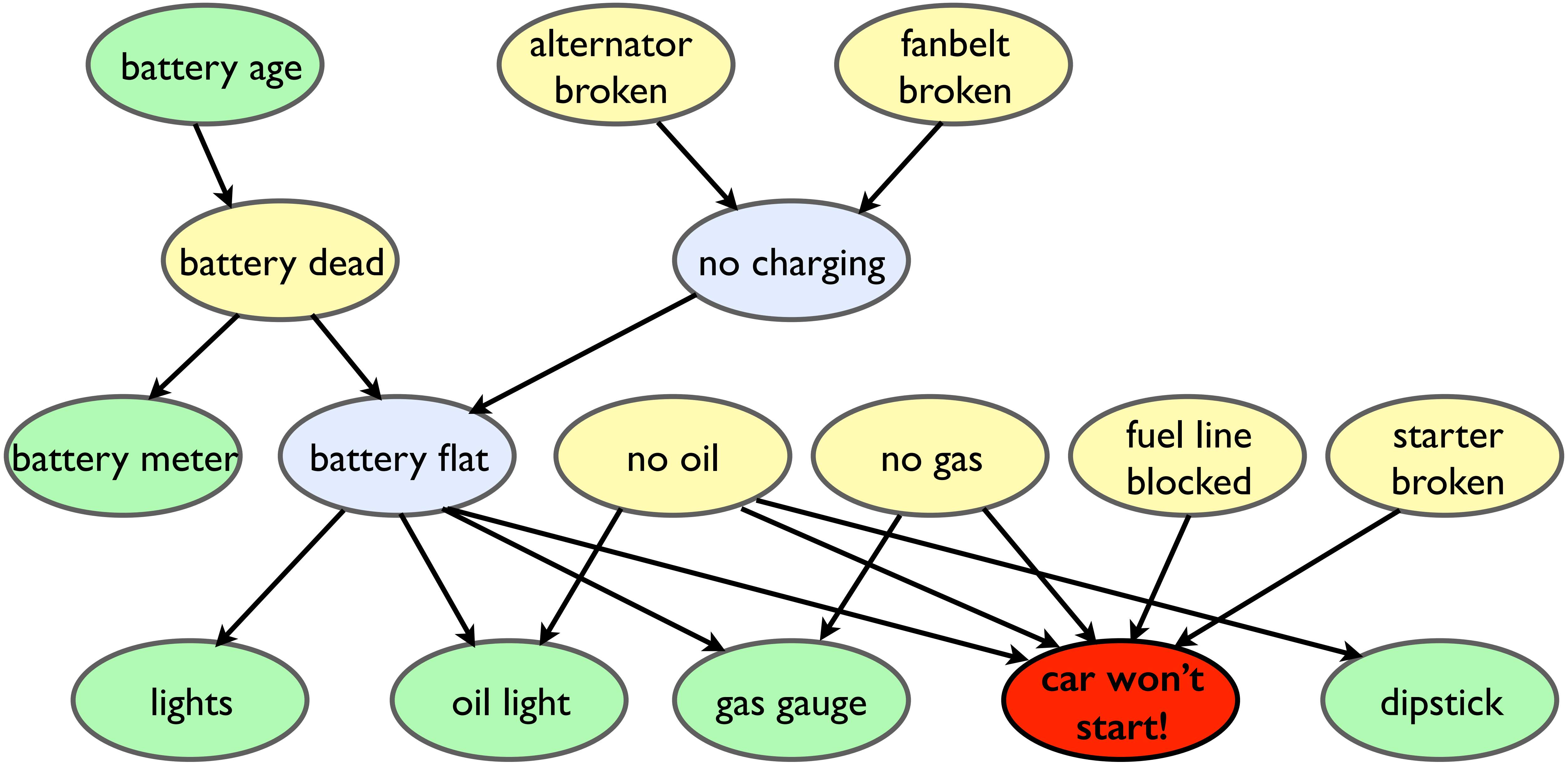
Hence: Choose preferably an order corresponding to the cause → effect "chain"

Locally structured (sparse) network

Initial evidence: The *** car won't start!

Testable variables (green), "broken, so fix it" variables (yellow)

Hidden variables (blue) ensure sparse structure / reduce parameters



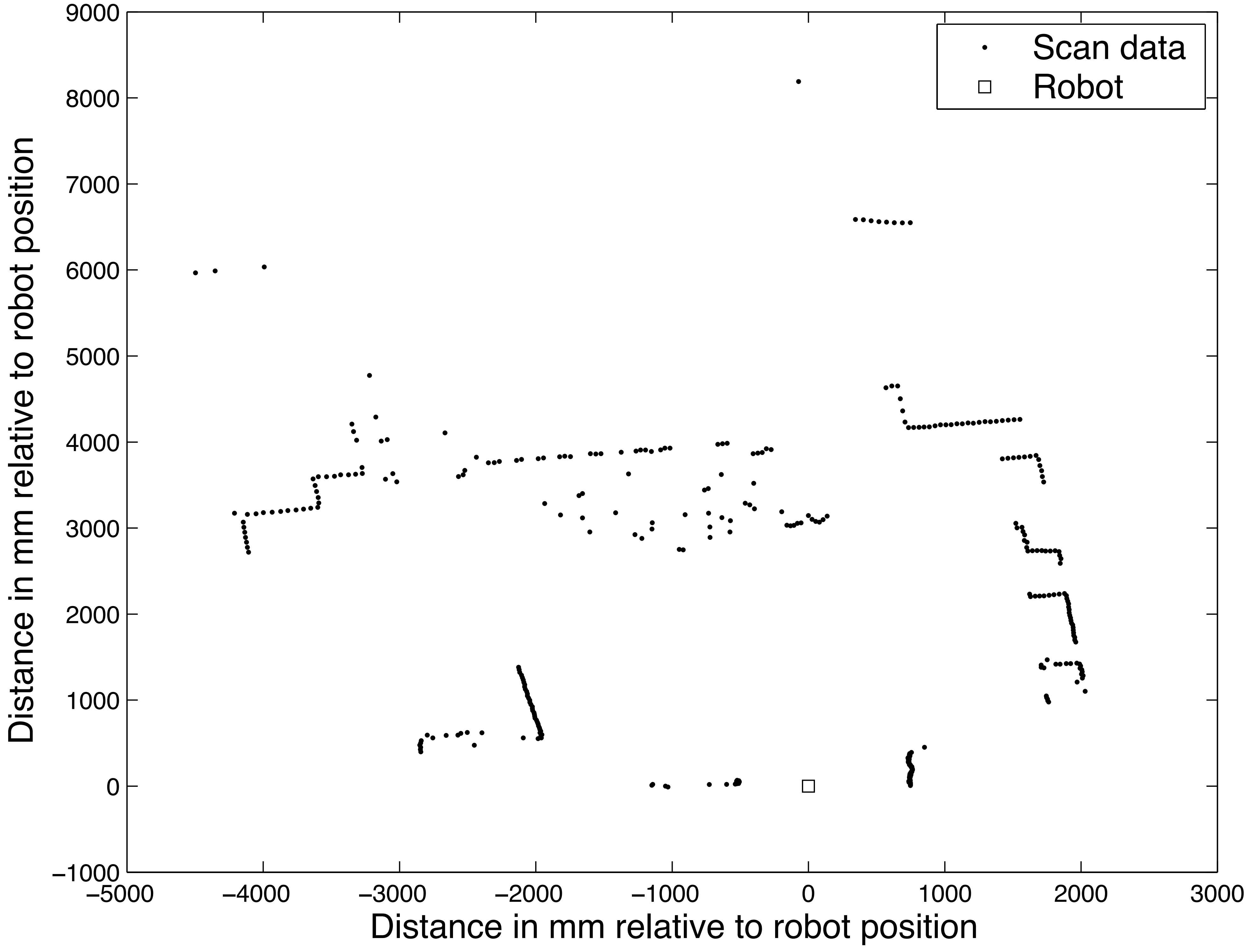
And now - learning.

How do we get the numbers into the network???

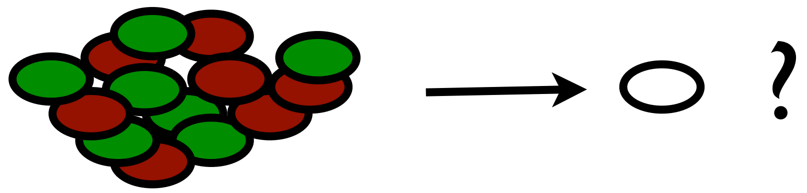
How do we determine the network structure?

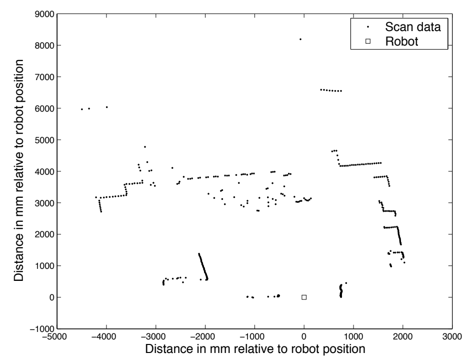
More general: How can we predict and explain based on (limited) experience?

A robot’s view of the world...



Predicting the next pattern type



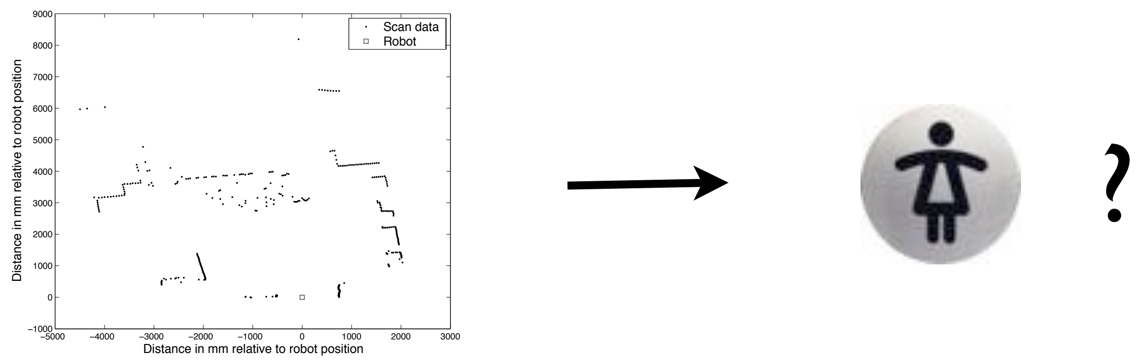


Images preprocessed into categories / collections according to the type of situation and possible numbers of "leg-like" patterns based on the knowledge of how many persons were in the room at a given time.
Labels for the image categories are lost, only numbers and pattern labels remain…

Hypotheses for types of pattern collection (i.e., images from a certain situation) are also available, with their *priors*:

h_1: only furniture	$P(h_1) = 0.1$
h_2: mostly furniture (75%), few persons	$P(h_2) = 0.2$
h_3: half furniture (50%), half persons	$P(h_3) = 0.4$
h_4: few furniture (25%), mostly persons	$P(h_4) = 0.2$
h_5: only persons	$P(h_5) = 0.1$

Maximum Likelihood

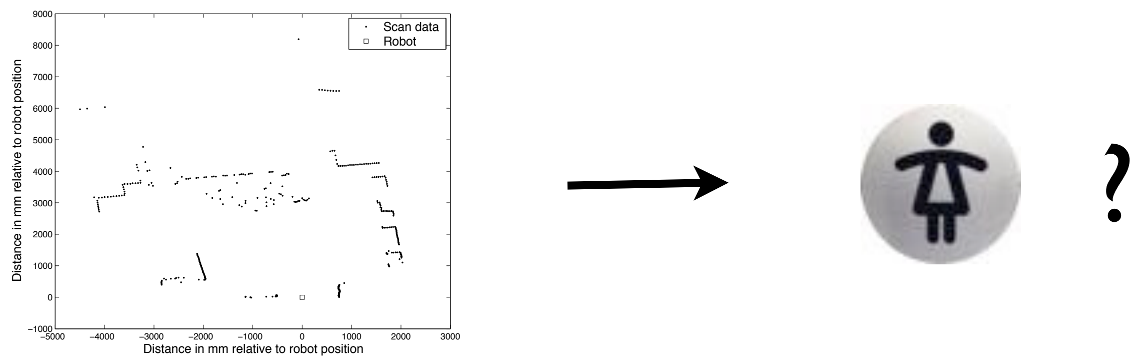


We can predict (probabilities) by maximizing the likelihood of having observed some particular data with the help of the ***Maximum Likelihood*** hypothesis:

$$h_{ML} = \underset{h}{\mathit{argmax}}\ P(D \mid h)$$

… which is a strong simplification disregarding the priors…

“Maximum A Posteriori” - MAP


Scan data
Robot
Distance in mm relative to robot position
Distance in mm relative to robot position
?

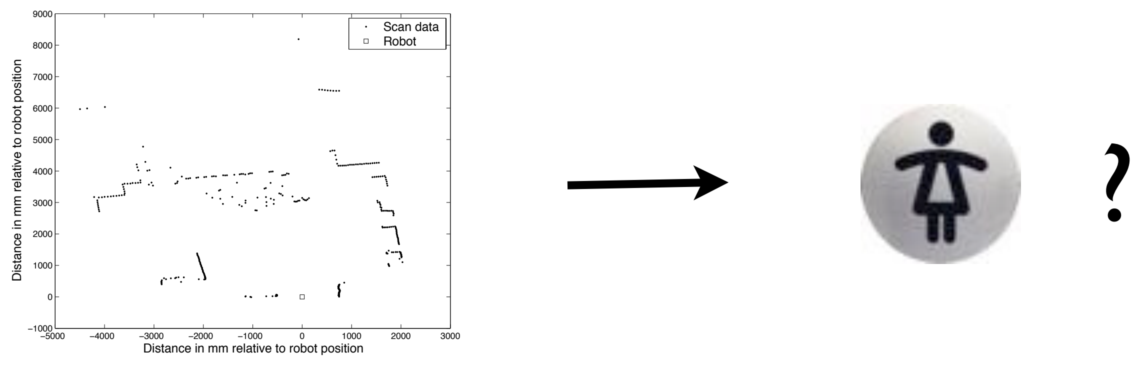
Finding the slightly more sophisticated ***Maximum A Posteriori*** hypothesis:

$$h_{MAP} = \underset{h}{argmax}\ P(h \mid D)$$

Then predict by assuming the MAP-hypothesis (quite bold)

$$\mathbf{P}(X \mid \mathbf{D}) = P(X \mid h_{MAP})$$

Optimal Bayes learner

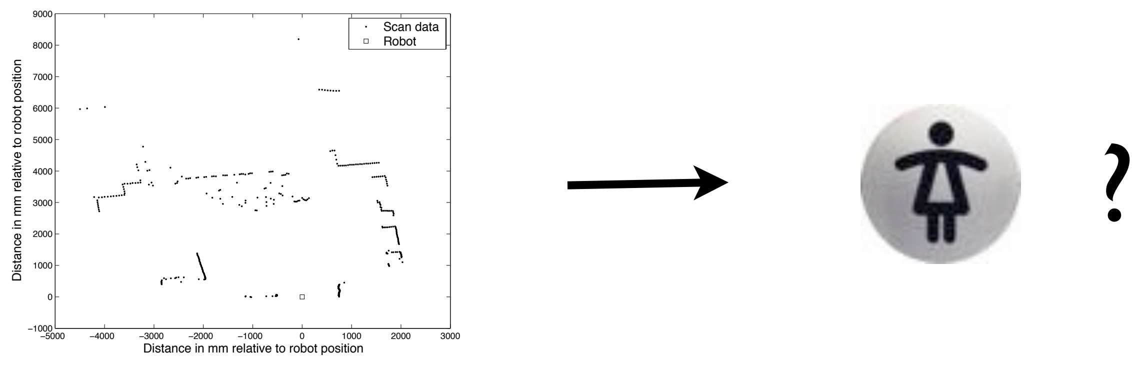


Scan data
Robot
Distance in mm relative to robot position
Distance in mm relative to robot position
?

Prediction for X, given some observations $\mathbf{D} = \langle d_0, d_1 \ldots\ldots d_n \rangle$

$\mathbb{P}(X \mid \mathbf{D}) = \sum_i \mathbb{P}(X \mid h_i)\, P(h_i \mid \mathbf{D})$ in first step, $P(h_i \mid \mathbf{D}) = P(h_i)$...

Learning from experience



Prediction for the first pattern picked, assuming e.g., h_3, and no observations are made:

$P(d_0 = \text{Furniture} \mid h_3) = P(d_0 = \text{Person} \mid h_3) = 0.5$

First pattern is of type *person*, now we know:

$P(h_1 \mid d_0) = 0$ (as $P(d_0 \mid h_1) = 0$), etc...

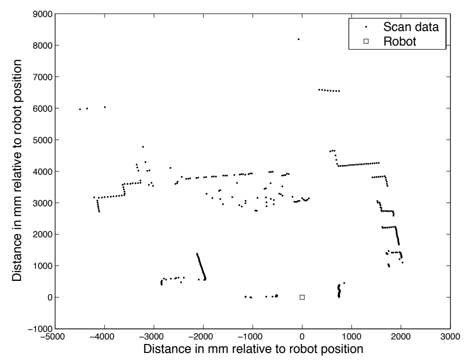
After 10 patterns that all turn out to be *Person*, assuming that outcomes for d_i are *i.i.d. (independent and identically distributed)*:

$P(\mathbf{D} \mid h_k) = \prod_i P(d_i \mid h_k)$

$\mathbb{P}(h_k \mid \mathbf{D}) = \mathbb{P}(\mathbf{D} \mid h_k)\, P(h_k) \,/\, \mathbb{P}(\mathbf{D}) = \alpha\ \mathbb{P}(\mathbf{D} \mid h_k)\, P(h_k)$

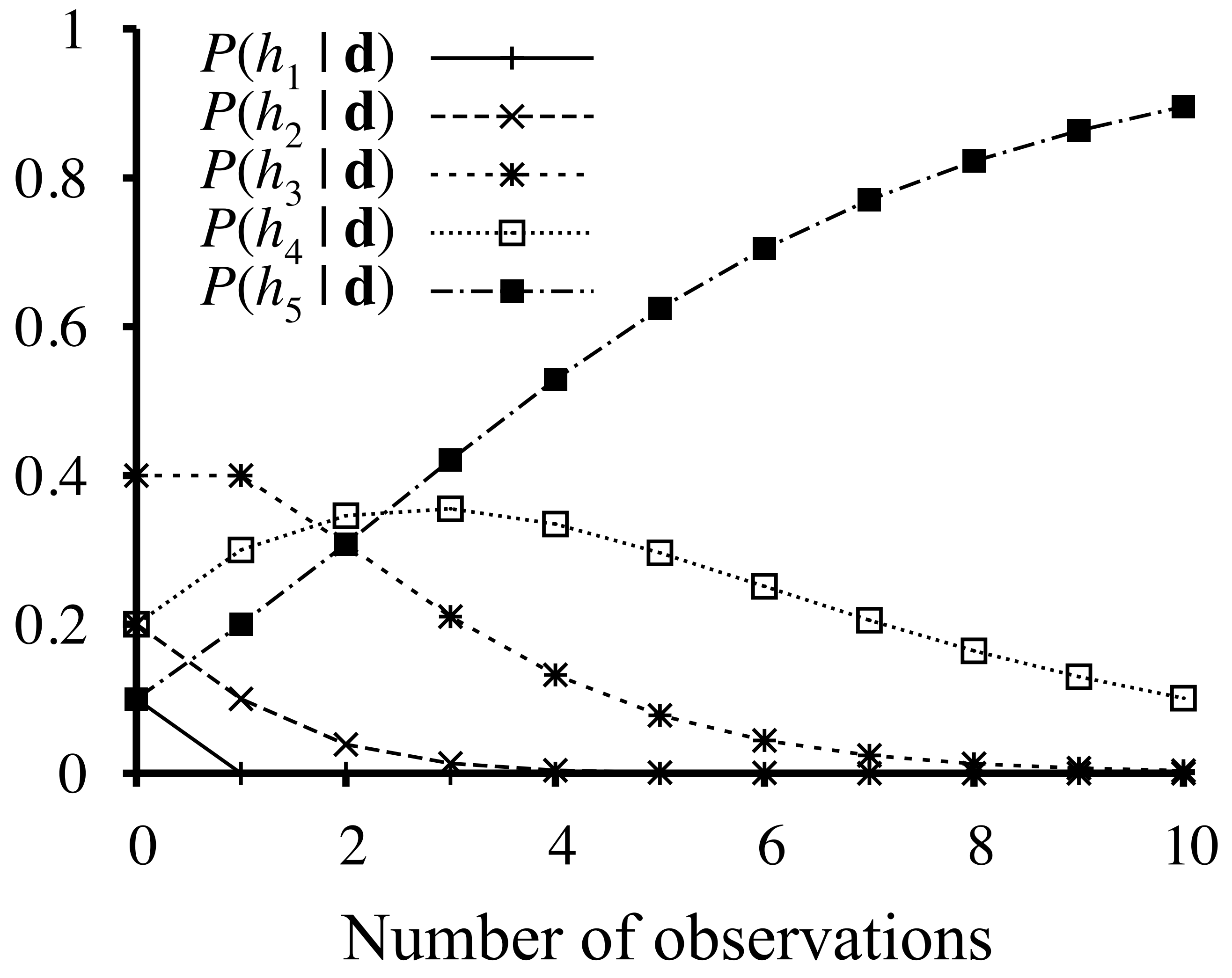
X

Posterior probabilities

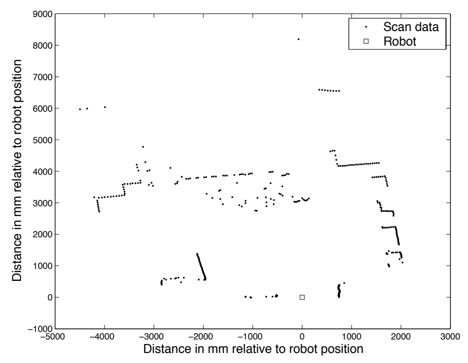


?

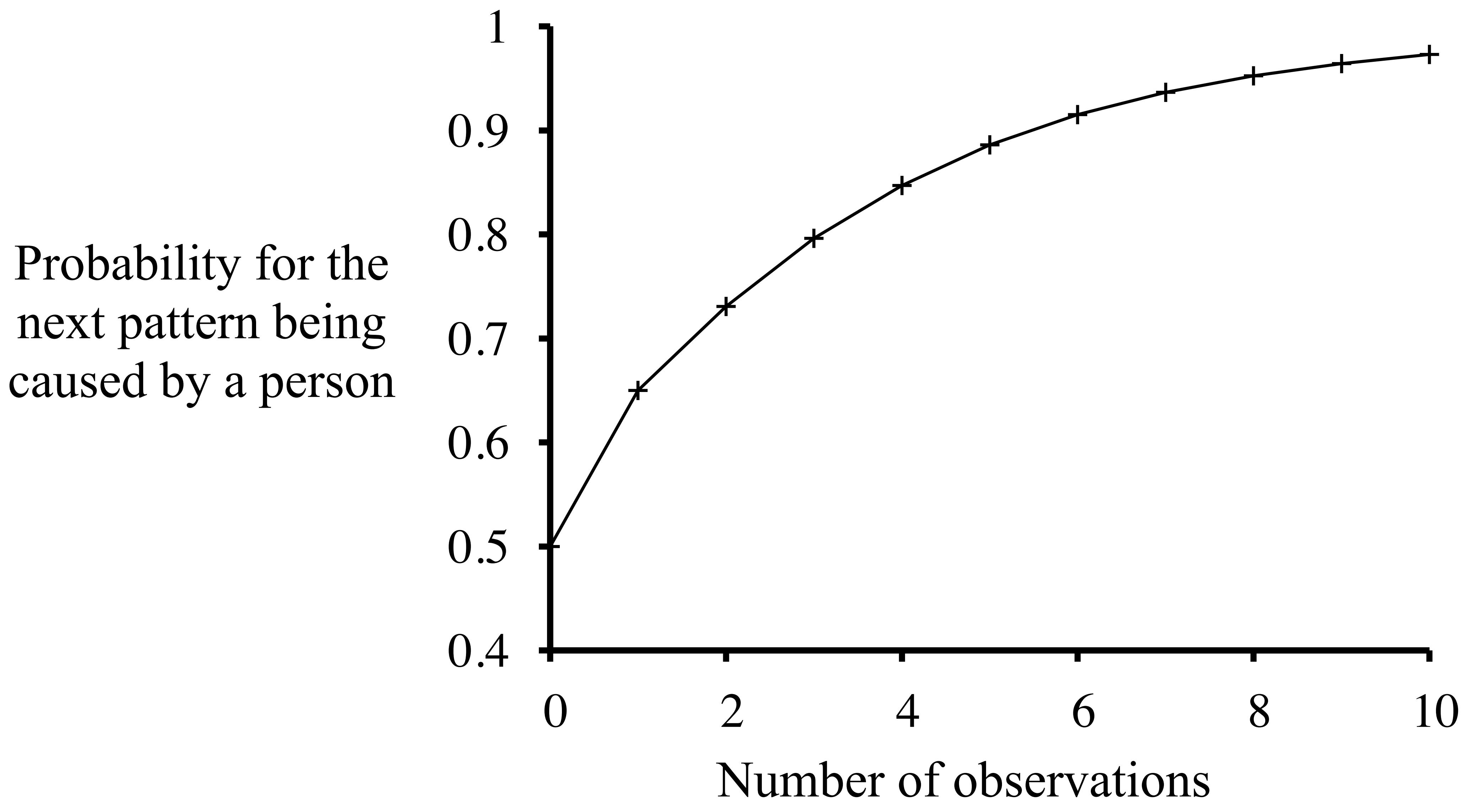
Posterior probability for hypothesis h_k after *i* observations



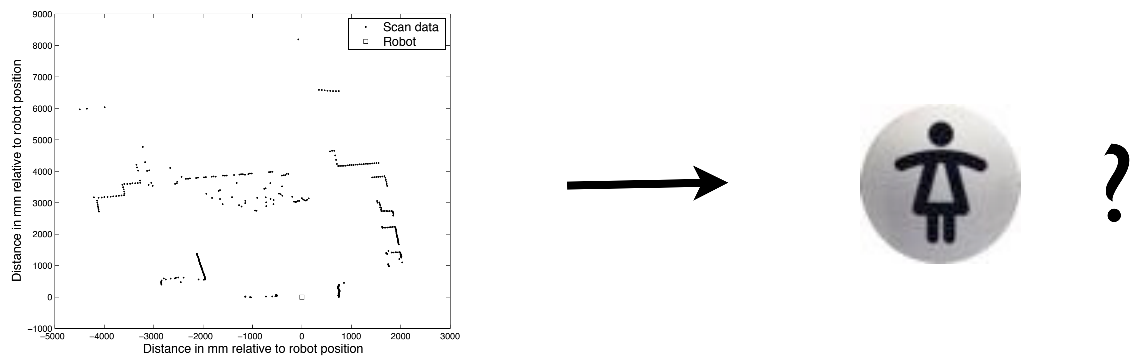
Prediction after sampling



?



Optimal learning vs MAP-estimating



Predict by assuming the MAP-hypothesis:

$\mathbb{P}(X \mid \mathbf{D}) = P(X \mid h_{MAP})$ with $h_{MAP} = \underset{h}{\mathit{argmax}}\ P(h \mid D)$

i.e., $P\_h_{MAP}(d_4 = \mathit{Person} \mid d_1 = d_2 = d_3 = \mathit{Person}) = P(X \mid h_5) = 1$

While the optimal classifier / learner predicts

$P(d_4 = \mathit{Person} \mid d_1 = d_2 = d_3 = \mathit{Person}) = \ldots = 0.7961$

However, they will grow closer! Consequently, the MAP-learner should not be considered for small sets of training data!

X

The Gibbs Algorithm

Optimal Bayes Learner is costly, MAP-learner might be as well.

Gibbs algorithm (surprisingly well working under certain conditions regarding the a posteriori distribution for H):

1. Choose a hypothesis *h* from H *at random,* according to the posterior probability distribution over H (i.e., rule out "impossible" hypotheses)
2. Use *h* to predict the classification of the next instance x.

Bayes' Rule

Bayes' Rule $P(a \mid b) = \dfrac{P(b \mid a)\, P(a)}{P(b)}$

or in distribution form:

$$\mathbb{P}(Y \mid X) = \frac{\mathbb{P}(X \mid Y)\, \mathbb{P}(Y)}{\mathbb{P}(X)} = \alpha\, \mathbb{P}(X \mid Y)\, \mathbb{P}(Y)$$

Useful for assessing *diagnostic* probability from *causal* probability

$$P(Cause \mid Effect) = \frac{P(Effect \mid Cause)\, P(Cause)}{P(Effect)}$$

And, if independence (at least conditional such) can be assumed:

Naive Bayes model: $\mathbb{P}(Cause, Effect_1, \ldots, Effect_n) = \mathbb{P}(Cause) \prod_i \mathbb{P}(Effect_i \mid Cause)$

Naive Bayes classifier

Each instance (pattern) with a value v_j from a fixed set V (= {furniture, person}) in a training set (all patterns registered and annotated) is described by several attributes $<a_1, \ldots, a_i, \ldots, a_n>$ (e.g., number of laser data points, curvature of the "arc", distance from first to last point)

Now we try to maximise:

$$v_{MAP} = \underset{v_j}{argmax}\, P(v_j \mid a_1, a_2, \ldots.\, a_n)$$

$$= \underset{v_j}{argmax}\, \frac{P(a_1, a_2, \ldots.\, a_n \mid v_j)\, P(v_j)}{P(a_1, a_2, \ldots.\, a_n)}$$

$$= \underset{v_j}{argmax}\, P(a_1, a_2, \ldots.\, a_n \mid v_j)\, P(v_j)$$

And (by assuming independence) end up with the Naive Bayes Classifier (corresponding in some sense to the MAP-hypothesis):

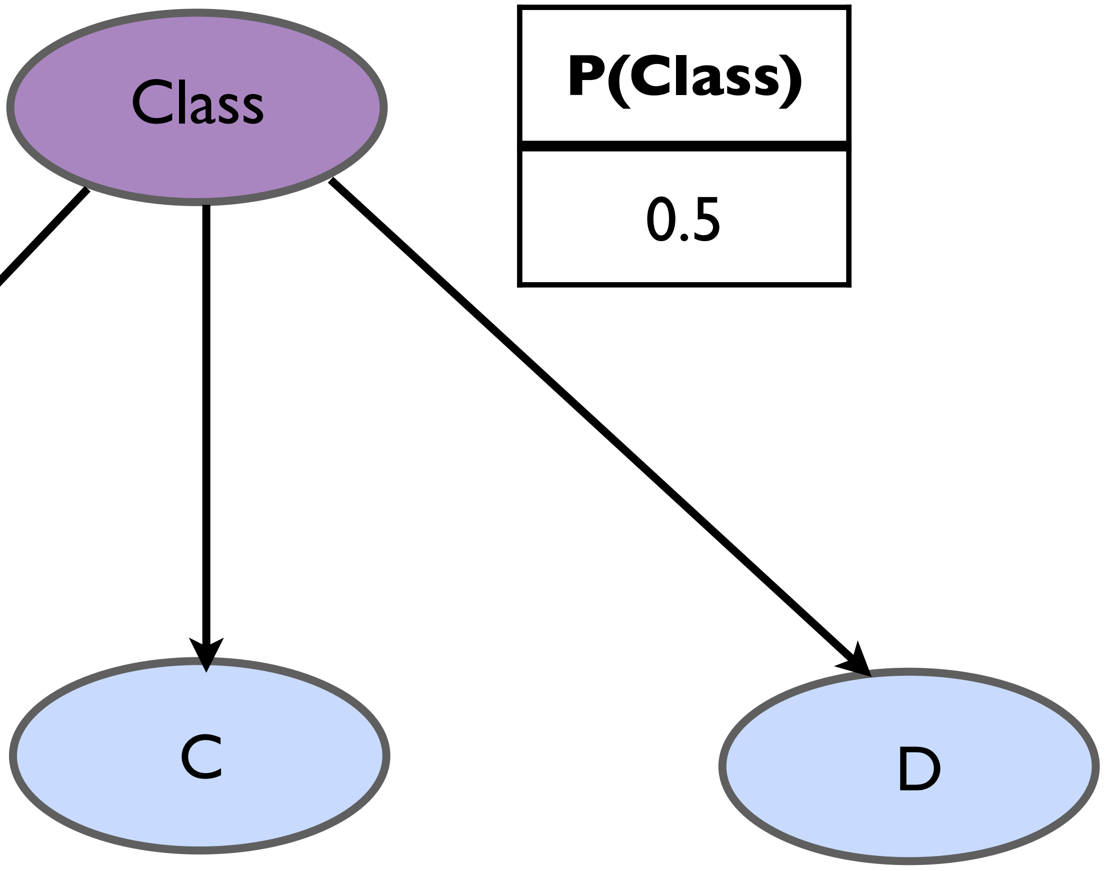
$$v_{NB} = \underset{v_j}{argmax}\, P(v_j) \prod_i P(a_i \mid v_j)$$

Or, finding the class for the pattern... (true model)

N = No of points,
n1 = "N<threshold", n2 = "N >= threshold"

C = Curvature,
c1 = "C=strong", c2 = "weak"

D = Distance first to last point,
d1 = "D<threshold", d2 = "D >= threshold



P(Class)
0.5

Class	P(N=n1|Class) = P(C = c1 | Class) = P(D = d1| Class)
Furniture	0.8
Person	0.3

Learning Bayesian Belief Networks

Two issues:

Learning the CPTs given a suitable structure AND all variables are observable:

Estimate the CPTs as for a Naive Bayes Classifier / Learner (relatively easy)

Learning the CPTs given a network structure with only partially observable variables:

Corresponds to learning the weights of hidden units in a neural network (ascent gradient or EM)

Learning the network structure

Difficult. Bayesian scoring method for choosing among alternative networks.

Expectation maximization - EM algorithm

A situation with some variables being sometimes unobservable, sometimes observable is quite common.

Use the observations that *are* available to predict in cases where there is not any observation.

Step 1: Estimate value for the hidden variable given some parameters (observed, initial...)

Step 2: Maximize parameters assuming this estimate

Excourse: Classifying text

Our approach to representing arbitrary text is disturbingly simple: Given a text document, such as this paragraph, we define an attribute for each word position in the document and define the value of that attribute to be the English word found in that position. Thus, the current paragraph would be described by 111 attribute values, corresponding to the 111 word positions. The value of the first attribute is the word "our", the value of the second attribute is the word "approach", and so on. Notice that long text documents will require a larger number of attributes than short documents. As we shall see, this will not cause us any trouble. (*)

$$v_{NB} = \underset{v_j \in \{\text{like, dislike}\}}{argmax} P(v_j) \prod_i^{111} P(a_i \mid v_j) = P(v_j) P(a_1 = \text{“our”} \mid v_j) * .... * P(a_{111} = \text{“trouble”} \mid v_j)$$

(*)[Tom M. Mitchell, "Machine Learning", p 180]

Naive Bayes Classifier for text

Given a test person who classified 1000 text samples into the categories "like" and "dislike" (i.e., the target value set *V*) and those text samples (*Examples*), the text from the previous slide is to be classified with the help of the Naive Bayes Classifier. This algorithm (from Tom M. Mitchell, "Machine Learning", p 183) assumes (and learns) the *m-estimate* for $P(w_k \mid v_j)$, the term describing the probability that a randomly drawn word from a document in class v_j will be the word w_k.

```
LEARN_NAIVE_BAYES_TEXT( Examples, V)
/* learn probability terms $P( w_k \mid v_j)$ and the class prior probabilities $P( v_j )$ */
1. Collect all words, punctuation, and other tokens that occur in Examples
    • Vocabulary ⟵ the set of all distinct words and other tokens occurring in any text document from Examples
2. calculate the required $P( v_j )$ and $P( w_k \mid v_j)$ terms
    • $docs_j$ ⟵ the subset of documents from Examples for which the target value is $v_j$
    • $P( v_j ) \longleftarrow | docs_j | / | Examples |$
    • $Text_j$ ⟵ a single document created by concatenating all members of $docs_j$
    • $n$ ⟵ total number of distinct word positions in $Text_j$
    • for each word $w_k$ in Vocabulary
        • $n_k$ ⟵ number of times word $w_k$ occurs in $Text_j$
        • $P( w_k \mid v_j) \longleftarrow ( n_k +1) / ( n + | Vocabulary |)$          /* m-estimate */


CLASSIFY_NAIVE_BAYES_TEXT( Doc)
/* Return the estimated target value for the document Doc. $a_i$ denotes the word found in ith position within Doc.
    • positions ⟵ all word positions in Doc that contain tokens found in Vocabulary
    • Return $v_{NB}$, where
```

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} \; P(v_j) \prod_{i \in positions} P(a_i \mid v_j)$$

Summary

Maximum likelihood hypothesis and MAP-hypothesis / learning

Optimal Bayes learner / classifier

Gibbs algorithm

Naive Bayes classifier

Learning Bayesian Belief Networks
- EM algorithm

(Example: The GeNIe network for interaction patterns)