Empirical Risk Minimization

When consistent learning is not possible, the best the agent can do is to minimize the *training error* (??), which is also called the **empirical risk**.

Notice the dilemma following from Theorem (??). A *larger* $\mathcal{H}$ will

- allow to achieve a smaller training error (we are choosing among more hypotheses)
- loosen the bound on the discrepancy (??) between error and the training error

Given a training set T, the dilemma is usually solved *empirically*, e.g., by cross-validating different $\mathcal{H}$ on T and then using the best $\mathcal{H}$ to learn from T.

Classification with Noise

Real-world concepts are often not "crisp" subsets of X as assumed by our current assumption.

Consider a "soft" alternative assuming that $x \in X$ belongs to class $y \in Y$ *with probability* $P(y \mid x)$. An appropriate replacement for the prescription (??) of the unit rewards ($R = \{0, 1\}$) is then ($k \in \mathbb{N}$)

$$\begin{aligned} r_1 &= 0 \\ r_{k+1} &= \begin{cases} 0 \text{ with probability } P(y_{k+1} \mid x_k) \\ -1 \text{ otherwise} \end{cases} \end{aligned} \tag{1}$$

Such rewards are probabilistic and we have already considered that. The agent can resort to empirical risk minimization using a class $\mathcal{H}$ of "crisp" hypotheses.

Learning a Probability Distribution

But consider an alternative. Instead of learning a binary-policy hypothesis h_k from training set T_k ($k > 1$), learn from T_k an estimate p_k of the distribution $P(x, y)$ and use the policy

$$y_k = \arg\max_{y \in Y} p_k(y \mid x_{k-1}) \tag{2}$$

where $p_k(y \mid x_k) = p_k(x_k, y)/p_k(x_k)$ and $p_k(x_k) = \sum_{y \in Y} p_k(x_k, y)$.

This approach is appropriate for example when we do not know which class $\mathcal{H}$ contains a low-error hypothesis but we know the class of distributions (e.g., normal) containing P.

Being able to learn a distribution allows us to design agents for agent-environment interactions beyond classification.

Learning a Probability Distribution (cont'd)

For example, let $V_1, V_2, \ldots, V_n$ be a set of discrete random variables distributed by some $P(V_1, V_2, \ldots, V_n)$. Let observations x_k be sampled from P but conveyed to the agent with *missing values* for some of the variables, i.e. only some of the n values are given to the agent.

Given x_k, the agent then predicts through y_{k+1} the most probable values of the rest of the variables according to its current model, i.e., $(k \in \mathbb{N})$

$$y_{k+1} = \arg\max_{\{x_k^i\}_{i\in I}} p_{k+1}(\{x_k^i\}_{i\in I} \mid \{x_k^j\}_{j\in J}) \quad (3)$$

where J (I, respectively) contains the indexes of the observed (unobserved) variables.

Example: predicting occluded pixels given the surrounding pixels in images.

Learning a Probability Distribution (cont'd)

Computing (2) or (3) is called **maximum aposteriori probability** (MAP) inference. (2) can be viewed as a special case of (3), in which the argument of maximization is fixed to the class variable.

Conceptually, (2) and (3) are the same problem, requiring the agent to learn a model of a joint distribution from samples from the distribution, possibly with missing values.

More precisely, the agent receives observations in the following way. Let $\mathcal{V}$ be a set of discrete random variables jointly distributed by P. Each observation *is sampled i.i.d from P* and then an arbitrary subset of its components is set to value '?' (indicating 'missing value').

From such observations the agent should learn an estimate p of P, i.e., for each $k > 1$, p_k is estimated from $x_{<k}$.

Dimensionality Problem

The task can be accomplished with a variant of the EM algorithm. At each $k+1$ ($k \in \mathbb{N}$), first set $p := p_k$, and then loop over two steps

1. Fill in missing values: Estimate the most probable values of unobserved components in all of $x_{\leq k}$ by MAP inference using p, yielding $\widehat{x}_{\leq k}$ with no missing values.
2. Re-estimate p by *relative frequencies*: for each value tuple v of $\mathcal{V}$ set $p(v) := m/k$ where m is the number of times v occurs in $\widehat{x}_{\leq k}$.

until p converges. Then set $p_{k+1} = p$.

The problem with the relative-frequency estimate is that when the dimension n grows linearly, to keep the accuracy of the estimate unchanged, the number of samples k must *grow exponentially*.

Independence to the Rescue

The dimensionality problem vanishes in the special case where the n variables are pairwise independent, so P **factorizes** (i.e., is equal to a product of smaller factors) as

$$P(V_1, V_2, \ldots, V_n) = P_1(V_1)P_1(V_2)\ldots P_1(V_n) \tag{4}$$

Then instead of estimating P of dimension n, the agent estimates $P_1, P_2, \ldots P_n$, each of dimension 1. *This is trivial: e.g.* $P(v) \approx$ *the proportion of value* v *among all non-missing values* $x^i_{\leq k}$.

The problem with assumption (4) is that it is too strong making it irrelevant to real-life machine learning problems.

However, a weaker form of independence can be defined and exploited.

Conditional Independence

Conditional Independence

Let P be a joint probability distribution of a set of random variables $\mathcal{V}$. Let $A, B \in \mathcal{V}$ and $\mathcal{E} \subseteq \mathcal{V}$. We say that **$A$ and B are conditionally independent given $\mathcal{E}$** (under P) if $P(A, B \mid \mathcal{E}) = P(A \mid \mathcal{E})P(B \mid \mathcal{E})$. We denote this as $A \perp\!\!\!\perp_P B \mid \mathcal{E}$.

We will drop the set delimiters $\{\}$ in the conditional part when there is only one variable in the condition, i.e. will we write $A \perp\!\!\!\perp_P B \mid C$ rather than $A \perp\!\!\!\perp_P B \mid \{ C \}$ to denote that A is conditionally independent of B given C.

It is obvious from the definition that $A \perp\!\!\!\perp_P B \mid \mathcal{E}$ implies

- $B \perp\!\!\!\perp_P A \mid \mathcal{E}$ (i.e., $\perp\!\!\!\perp_P$ is symmetric)
- $P(A|B, \mathcal{E}) = P(A|\mathcal{E})$ (hint: use the chain rule)

Example: Conditional Independence

Three random variables:

T	outdoor temperature
I	ice-cream sales
H	heart-attack rate

I and H are not independent:

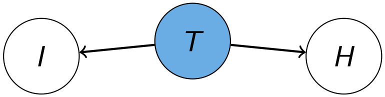
$$P(I, H) \neq P(I)P(H)$$

but they are *conditionally independent*:

$$P(I, H \mid T) = P(I \mid T)P(H \mid T)$$

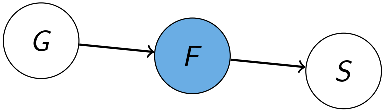
Conditional Independence in Cause-Effect Graphs

Heart attack rate and ice-cream sales independent if temperature known:



$$P(I, H \mid T) = P(I \mid T)P(H \mid T)$$

Son and grandfather's high IQ independent if same known for father:



$$P(S, G \mid F) = P(S \mid F)P(G \mid F)$$

In both cases: *any vertex is conditionally independent of all of its non-descendants given all its parents.*

This principle motivates the framework of *Bayesian networks*, which are a special case of probabilistic graphical models.

Bayes Graph

Denote $\mathrm{par}_G(V)$ the set of all parents of vertex V in an oriented graph G.

Bayes Graph

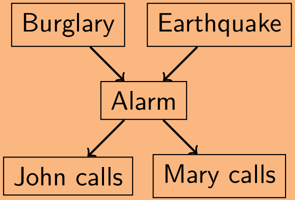
A **Bayes Graph** for a set $\mathcal{V}$ of random variables is an acyclic directed graph G with vertex set $\mathcal{V}$. A Bayes G is **correct** for a distribution P on $\mathcal{V}$ if $\forall V, V' \in \mathcal{V} : V \perp\!\!\!\perp_P V' \mid \mathrm{par}_G(V)$ whenever V' is not a descendant of V in G.

(Exercise problem)

So a Bayes Graph is similar to cause-effect graphs but edges *need not correspond to cause-effect directions*. A Bayes graph for P indicates pairs of variables conditionally independent under P: a variable is conditionally independent of all its non-descendants if exactly all its parents are given.

There may by multiple BG's for one P.

Example: Bayes Graph for Binary Variables (from AIMA)



From this Bayes graph, we can infer:

- $P(B, E) = P(B)P(E)$
- $P(J \mid X, A) = P(J \mid A)$ for all of $X \in \{ B, E, M \}$
- $P(M \mid X, A) = P(M \mid A)$ for all of $X \in \{ B, E, J \}$

By the chain rule of probability

$$P(B, E, A, M, J) = P(J \mid B, E, A, M)P(M \mid B, E, A)P(A \mid B, E)P(B, E)$$

but this simplifies using the inferred equalities:

$$P(B, E, A, J, M) = P(J \mid A)P(M \mid A)P(A \mid B, E)P(B)P(E)$$

(See more in a tutorial.)

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Probability Factorization by a Bayes Graph

The following theorem is a general statement of the factorization shown in the example. Its validity follows directly from the definition.

Theorem 1

Let G be a Bayes Graph correct for distribution P on variables $V_1, V_2, \ldots, V_n$. Then

$$P(V_1, V_2, \ldots, V_n) = \prod_{i=1}^{n} P\left(V_i \mid \mathrm{par}_G(V_i)\right) \tag{5}$$

Similarly to (4), the Theorem enables to express a high-dimensional distribution as a product of low-dimensional distributions provided that the variables V_i's have a low number of parents in G. *This assumption is more realistic than pairwise independence.*

Conditional Probability Table

Let us abbreviate $V = 1$ ($V = 0$, respectively) as v ($\neg v$) for any binary random variable V.

To store an estimate of $P(B, E, A, J, M)$ from the example, we need an array of size $2^5 = 32$ to store a probability for each value combination of the 5 variables. *More precisely, we need to store only 31 parameters as the 32 of them sum to 1.*

To specify its factorization $P(J \mid A)P(M \mid A)P(A \mid B, E)P(B)P(E)$, we need a *conditional probability table (CPT)* for each of the factors. E.g. for $P(A \mid B, E)$:

$P(a \mid B, E)$	E	B
0.001	0	0
0.940	0	1
0.290	1	0
0.950	1	1

(Exercise problem)

Bayes Network

Bayes Network

A **Bayes Network** for a distribution P on a set $\mathcal{V}$ of random variables consists of a Bayes graph G for $\mathcal{V}$, and a conditional probability table for each $V \in \mathcal{V}$ containing a number from $[0;1]$ (i.e., a probability value) for each assignment of values to random variables $\text{par}_G(V)$.

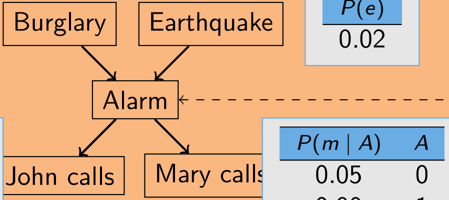
The probabilities in the CPT of any V specify $P(V \mid \text{par}_G(V))$. So due to (5), a BN fully specifies P.

There are in general multiple BN's specifying the same P. More edges mean more parameters.

Bayes Network Example

$P(b)$
0.01

$P(e)$
0.02



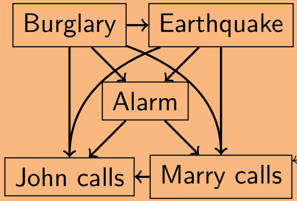
$P(a \mid B, E)$	B	E
0.00	0	0
0.29	0	1
0.94	1	0
0.95	1	1

$P(j \mid A)$	A
0.01	0
0.70	1

$P(m \mid A)$	A
0.05	0
0.90	1

10 parameters, less than 31 in the full joint probability table.

A Different Graph for the Same Example



$P(m \mid B, E, A)$	B	E	A
0.00	0	0	0
0.30	0	0	1
0.05	0	1	0
0.25	0	1	1
(. . . 4 more rows)			

This Bayes graph does not imply any conditional independence. For each vertex, all non-descendants are parents. Joint distribution calculated as

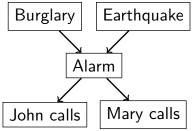
$$P(B, E, A, M, J) = P(J \mid B, E, A, M)P(M \mid B, E, A)P(A \mid B, E)P(E \mid B)P(B)$$

CPT's for the BN with this BG have $2^4 + 2^3 + 2^2 + 2 + 1 = 31$ parameters, same as the full joint probability table.

Computing Marginal Probabilities from a Bayes Network

So far we know how to compute the full joint distribution from CPT's:

$$P(B, E, A, J, M) = P(J \mid A)P(M \mid A)P(A \mid B, E)P(B)P(E)$$



A straightforward way to compute marginals, e.g. $P(A, J)$ is to *sum out* the remaining variables. Think why it is good below to push the sums as far right as possible! *(implemented in a tutorial)*

$$\begin{aligned} P(A, J) &= \sum_B \sum_E \sum_M P(J \mid A)P(M \mid A)P(A \mid B, E)P(B)P(E) \\ &= P(J \mid A) \sum_M P(M \mid A) \sum_B \sum_E P(A \mid B, E)P(B)P(E) \end{aligned}$$

B under a $\sum$ means summing over b and $\neg b$. Same for other variables.

Computing Conditional Probabilities from a Bayes Network

Conditional probabilities are just fractions of marginals, e.g.

$$P(A, J \mid B, E) = \frac{P(A, J, B, E)}{P(B, E)}$$

(exercise problem)

Instead of calculating the denominator, we can evaluate the numerator for all assignments to A, J and normalize, since $\sum_A \sum_J P(A, J \mid B, E) = 1$.

$$\alpha \left[P(\neg a, \neg j, B, E) + P(\neg a, j, B, E) + P(a, \neg j, B, E) + P(a, j, B, E)\right] = 1$$

After computing the summands, we compute $\alpha = 1/P(B, E)$ from the equation above. Then we can get the conditional probability for any $\langle A, J \rangle$; e.g. for $\langle \neg a, j \rangle$

$$P(\neg a, j \mid B, E) = \alpha \cdot P(\neg a, j, B, E)$$

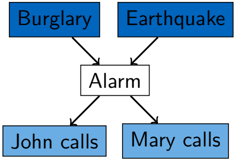
FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Evidence and Query Variables

In BN terminology, the variables whose joint conditional probability is computed are called *query* variables; those in the condition part are *evidence* variables.

Example query: *probability that neither John nor Mary will call during a burglary and no earthquake*:

$$P(\underbrace{\neg j, \neg m}_{\text{query}} \mid \underbrace{b, \neg e}_{\text{evidence}})$$



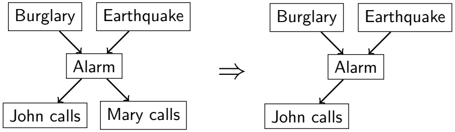
In (3), the query variables include *all* unobserved variables and evidence includes *all* observed variables. But BN's enable more general queries: query and evidence can be arbitrary subsets of all variables.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Removing Irrelevant Variables

Consider
$P(J \mid b) = \alpha P(b) \sum_E P(E) \sum_A P(A \mid b, E) P(J, A) \sum_M P(M \mid A)$

$\sum_M P(M \mid A) = 1$ so it can be left out, i.e. remove the corresponding vertex from the BN.



In general, *any vertex that is not an ancestor to a query variable or evidence variable of a query can be removed from the graph when computing the query*.

d-Separation

Consider the Bayes Graph

$$\begin{tikzcd} \boxed{A} \arrow[r] & \boxed{B} \arrow[r] & \boxed{C} \arrow[r] & \boxed{D} \end{tikzcd}$$

correct for some $P(A, B, C, D)$.

Are A and D independent under P if B is observed? I.e., does G imply

$$A \perp\!\!\!\perp_P D \mid B \text{ ?}$$

Yes, but this does not immediately follow from the definition because $\mathrm{par}_G(D) = \{ C \}$ is not observed.

The *d-separation* criterion serves to decide all cases of independence implied by a Bayes Graph.

d-Separation (cont'd)

Given some evidence $\mathcal{E}$, we say that variables A and B are **d-separated** in the Bayes Graph G by $\mathcal{E}$ if on every undirected path between A and B in G, there is

- either a vertex $V \in \mathcal{E}$ such that the (directed) edges adjacent to V on the path are
  - either *diverging*, i.e., $\longleftarrow \boxed{\text{V}} \longrightarrow$
  - or *linear*, i.e., $\longrightarrow \boxed{\text{V}} \longrightarrow$.
- or a vertex $V \notin \mathcal{E}$ such that $D \notin \mathcal{E}$ *also for all descendants* D of V in G, and the edges adjacent to V on the path are
  - *converging*, i.e., $\longrightarrow \boxed{\text{V}} \longleftarrow$.

d-Separation (cont'd)

Theorem 2

Let G be a Bayes graph for a distribution P of a set $\mathcal{V}$ of random vars. Let further $A, B \in \mathcal{V}$ and $\mathcal{E} \subseteq \mathcal{V}$. If A and B are d-separated in G by $\mathcal{E}$ then $A \perp\!\!\!\perp_P B \mid \mathcal{E}$.

Proof (not trivial) can be found in Verma & Pearl, 1998.

(exercise problem)

D-separation can be checked by an efficient algorithm that does not enumerate all paths between the inspected pair of nodes.

Checking d-Separation

To determine if A and B are d-separated in G by $\mathcal{E}$,

1. Extract from G the **ancestral graph** G_{anc} by keeping only vertices in $\{A, B\} \cup \mathcal{E}$ and all their ancestors (edges between kept vertices are kept.)
2. **Moralize** G_{anc} by putting an undirected edge between (i.e., "marrying") each pair of parents of any vertex; then replace in G_{anc} all directed edges by an undirected edge.
3. Delete from G_{anc} all vertices from $\mathcal{E}$ along with their edges.

A and B are d-separated in G by $\mathcal{E}$ iff A and B are not connected in the resulting graph.

(Implemented in a tutorial).