Bayes rule: $P(X|y) = P(y|X) \cdot P(X) \cdot P(y)$

$= \alpha P(y|X) \cdot P(X)$

Marginalization:

$$P(\phi) = \sum_{w: w \models \phi} P(w)$$

Summing out:

$$P(X) = \sum_{y} P(X, y)$$

$$P(X) = \sum_{c \in E} P(X|e) \cdot P(e)$$

Normalization

$$P(X|E) = P(X, E) \cdot P(E)$$

$$= \alpha \cdot P(X, E)$$

because $\sum_{y \in Y} P(y|E) \cdot P(E) = 1$

Absolute independence: $P(X, Y) = P(X) \cdot P(Y)$

Conditional independence: $P(X, Y|Z) = P(X|Z) \cdot P(Y|Z)$

$P(X|Y, Z) = P(X|Z)$ etc...

Chain rule for Bayesian networks:

known from the network

$$P(x_1 - x_n) = \prod_i P(x_i \mid Parent(x_i))$$

From samples to probability

$$P(x_1 - x_n) = \lim_N \left(N(x_1 - x_n)/N \right)$$

rejection sampling: —> taking only those satisfying evidence

$$P(X|e) = P(X, e) \cdot P(e)$$

likelihood sampling:

Fix evidence variables and sample only remaining

We get:

$$P(z, e) = \prod_i (z_i \mid Parents(z_i))$$

We need weighting:

$$w(z, e) = \prod_j (e_j \mid parents(e_j))$$

$$P(X, e) = \alpha \cdot N(X, e) \cdot w(X, e)$$

Filtering:

Must define f such that $P(X_{t+1} | e_{1:t+1}) = f(e_{t+1}, P(X_t | e_{1:t}))$

$$P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$$

$P(X|Y) = \alpha \cdot P(Y|X) \cdot P(X)$

$$= \alpha \cdot P(e_{t+1} | X_{t+1}, e_{1:t}) \cdot P(X_{t+1} | e_{1:t})$$

$$= \alpha \cdot P(e_{t+1} | X_{t+1}) \cdot P(X_{t+1} | e_{1:t})$$

$$= \alpha \cdot P(e_{t+1} | X_{t+1}) \cdot \sum_{X_t} (X_{t+1} | X_t) \cdot \underbrace{P(X_t | e_{1:t})}_{f_{1:t}}$$

$$f_{1:t+1} = \alpha \cdot \text{Forward}(f_{1:t}, e_{t+1})$$

Prediction:

$$P(X_{t+h+1} | e_{1:t}) = \sum_{X_{t+h}} P(X_{t+h+1} | X_{t+h}) \cdot P(X_{t+h} | e_{1:t})$$

Konverguje k stacionární distribuci daného Markovského procesem

Smoothing:

$$P(X_k | e_{1:t}) = P(X_k | e_{1:k}, e_{k+1:t})$$

$$= \alpha \cdot P(e_{k+1:t} | X_k, e_{1:k}) \cdot P(X_k | e_{1:k})$$

$$= \alpha \cdot \underbrace{P(e_{k+1:t} | X_k)}_{b_{k+1:t}} \cdot \underbrace{P(X_k | e_{1:k})}_{f_{1:k}}$$

$$P(e_{k+1:t} \mid X_k) = \sum_{X_{k+1}} P(e_{k+1:t} \mid X_k, X_{k+1}) \cdot P(X_{k+1} \mid X_k)$$

$$= \sum_{X_{k+1}} P(e_{k+1:t} \mid X_{k+1}) \cdot P(X_{k+1} \mid X_k)$$

$$= \sum_{X_{k+1}} P(e_{k+1}, e_{k+2:t} \mid X_{k+1}) \cdot P(X_{k+1} \mid X_k)$$

$$= \sum_{X_{k+1}} P(e_{k+1} \mid X_{k+1}) \cdot P(\underbrace{e_{k+2:t} \mid X_{k+1}}_{b_{k+2:t}}) \cdot P(X_{k+1} \mid X_k)$$

Using backward alg.

$$b_{k+1:t} = \text{backward}(b_{k+2:t}, e_{k+1})$$

$$b_{t+1:t} = 1 = P(e_{t+1:t} \mid X_t) = P(1 \mid X_t)$$

Most likely explanation:

Viterbi alg. (going through graph)

$$\underset{X_{1:t}}{\operatorname{argmax}} \; P(X_{1:t} \mid e_{1:t})$$

$$\max_{x_1 \dots x_t} P(X_1, \dots, X_t, X_{t+1} \mid e_{1:t+1})$$

$$= \alpha \cdot P(e_{t+1} \mid X_{t+1}) \cdot \max_{x_t} \left(P(X_{t+1} \mid X_t) \cdot \max_{x_1 \dots x_{t-1}} P(x_1 \dots x_{t-1} \mid e_{1:t-1}) \right)$$

Bellman equations: $U(s) = R(s) + \gamma \cdot \max_a \sum_{s'} P(s'|a,s) \cdot U(s')$

bes: $\pi^*(s) = \arg\max_a \sum_{s'} P(s'|a,s) \cdot U(s')$

MDP:

Transition model: $P(s'|a,s)$

Reward: $R(s)$

Utility function: $U([s_0, s_1 \dots]) = R(s_0) + \gamma R(s_1) + \gamma^2 \cdot R(s_2) \dots$

Bellman: value-iteration

First set some default $U(s)$

$$U_{i+1}(s) = R(s) + \gamma \cdot \max_a \sum_{s'} P(s'|a,s) \cdot U_i(s')$$

Repeat until some treshold in max diff in one iteration

$$\pi^*(s) = \arg\max_\pi U^\pi(s)$$

Bellman: policy-iteration

1) Policy evaluation:

$$U(s) = R(s) + \gamma \cdot \sum_{s'} P(s'|s, \pi(s)) \cdot U(s')$$

2) Policy-improvement

$$\pi_{i+1}(s) = \arg\max_a \sum_{s'} P(s'|a,s) \cdot U^{\pi_i}(s')$$

↳ pokud existuje lepší akce než ta dřívější policy π_i

$$H(V) = -\sum_k P(v_k) \cdot \log_2(P(v_k))$$

$$B(q) = -q \cdot \log_2 q - (1-q) \cdot \log_2(1-q)$$

Information gain míra, jak moc se mi rozdělí množina

$$\text{Remainder}(A) = \sum_k \left(B\left(p_k / (n_k + p_k)\right) \cdot \left((n_k + p_k)/(n+h)\right) \right)$$

$$\text{Gain}(A) = B(p/(n+p)) - \text{Remainder}(A)$$

Solving optimal parameters w_0, w_1 analytically:

$$\frac{\partial \sum_j (y_j - h(x_j))^2}{\partial w_0} = 0$$

$$\frac{\partial \sum_j (y_j - h(x_j))^2}{\partial w_1} = 0$$

$$w_1 = \left(N \sum_j x_j y_j - \sum_j x_j \sum_j y_j\right) / \left(N \sum_j x_j^2 - \left(\sum_j x_j\right)^2\right)$$

$$w_0 = \left(\sum_j y_j - w_1 \cdot \sum_j x_j\right) / N$$

With gradient descent:

$$w_i = w_i - \alpha \cdot \frac{\partial \text{Loss}(w_i)}{\partial w_i}$$ → jdu směrem z kopce pro daný bod dokud nekonverguju

$$w_0 = w_0 + \alpha \cdot \left(\sum y_j - h(x_j)\right)$$

$$w_1 = w_1 + \alpha \cdot \left(\sum y_j - h(x_j)\right) \cdot x_j$$

Finding linear separator

$$w_i = w_i + \alpha \cdot (y - h(x)) \cdot x_i$$

Gradient descent method

$$w_{ij} = w_{ij} - \alpha \cdot \frac{\partial \, Loss(h_w)}{\partial \, w_{ij}}$$

Learning in feed-forward multilayer network
and how to measure errors in hidden layers?

Backpropagate!

- initialize random values to inner weights
- calculate the network output based on examples
- calculate and modify error for each node:

$$\Delta_j = g'(in_j) \cdot (y_j - a_j)$$

↳ g je aktivační funkce

- propagate Δ values back

$$\Delta_i = g'(in_i) \cdot \sum_j w_{ij} \cdot \Delta_j$$

- update the weights

Nearest neighbour model:

Minkowski distance measure

$$L^p(x_j, x_q) = \left(\sum_i |x_{j,i} - x_{q,i}|^p \right)^{1/p}$$

$$P(h_i|d) = \alpha \cdot P(d|h_i) \cdot P(h_i)$$

$$P(X|d) = \sum_i P(X|d,h_i) \cdot P(h_i|d) = \sum_i P(X|d) \cdot P(h_i|d)$$

$$P(d|h_\theta) = \prod_j P(d_j|h_\theta) = \theta^c \cdot (1-\theta)^l$$

Milkovski distance

$$L^p(x_j, x_q) = \left(\sum_i |x_{ji} - x_{qi}|^p\right)^{1/p}$$

Error-weights back propagation

- Start with some random weights
- Calculate output on the last level
- calculate the error at the output level: $\Delta_j = g'(in_j) \cdot (y_j - a_j)$
- propagate error to the previous layer: $\Delta_i = g'(in_i) \cdot \sum_j w_{ij} \cdot \Delta_j$
- update the weights

Finding the linear separator

$$w_i = w_i - \alpha \cdot (y - h(x)) \cdot x_i$$

$$Q(s,a) = Q(s,a) + \alpha \cdot \left(R(s) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$