Lecture 15: Learning in Bayesian Networks

University of BRISTOL

EMAT31530/March 2018/Raul Santos-Rodriguez

Have a look at ...

... Russell and Norvig (Ch. 14 and Ch. 15)

... The introduction to the book Graphical Models; Foundations of Neural Computing (ed. M. Jordan)

... David barber's Bayesian reasoning and Machine Learning:
`http://www.cs.ucl.ac.uk/staff/d.barber/brml/`

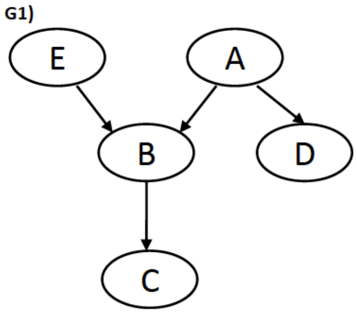
... Kevin Murphy's Toolbox:
`http://www.cs.ubc.ca/~murphyk/Software/`

Outline

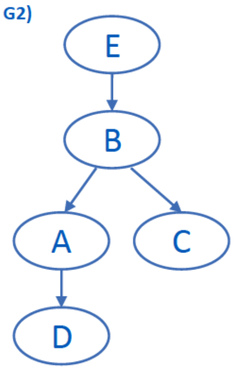
This lecture introduces the concept of learning in probabilistic graphical models. The objective is to discuss the following topics:

- Learning in Bayesian Networks
- Maximum likelihood
- Expectation-Maximization algorithm

Question



Are these two bayesian networks equivalent?

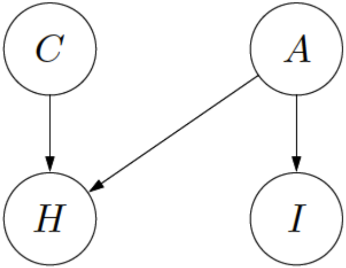


$$P(A,E,B,C,D) = P(A)P(E)P(B \mid A,E)P(C \mid B)P(D \mid A)$$
$$P(A,E,B,C,D) = P(E)P(B \mid E)P(A \mid B)P(C \mid B)P(D \mid A)$$

Review

So far, we have studied:

- Concept of Bayesian network
- Conditional independence
- Inference in Bayesian networks
- Dynamic Bayesian networks



Why learning?

Knowledge acquisition bottleneck

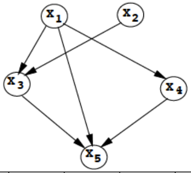
- Knowledge acquisition is an expensive process
- Often we don't have an expert

Data is cheap

- Amount of available information growing rapidly
- Learning allows us to construct models from raw data

Problem formulation

- Given:
  - A Bayesian network structure.



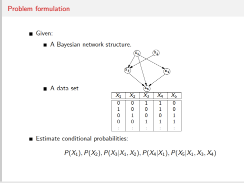
  - A data set

X_1	X_2	X_3	X_4	X_5
0	0	1	1	0
1	0	0	1	0
0	1	0	0	1
0	0	1	1	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

- Estimate conditional probabilities:

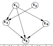
$$P(X_1), P(X_2), P(X_3|X_1, X_2), P(X_4|X_1), P(X_5|X_1, X_3, X_4)$$

Problem formulation



Problem formulation

- Given:
  - A Bayesian network structure.



  - A data set

X_1	X_2	X_3	X_4	X_5
0	0	1	1	0
1	0	0	1	0
0	1	0	0	1
0	0	1	1	1
⋮	⋮	⋮	⋮	⋮

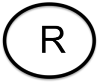
- Estimate conditional probabilities:

$$P(X_1), P(X_2), P(X_3|X_1, X_2), P(X_4|X_1), P(X_5|X_1, X_3, X_4)$$

We will first consider the supervised setting, where each data point (example) is a complete assignment to all the variables in the Bayesian network.

Example: one variable

Setting: Rating of a movie $\{1, 2, 3, 4, 5\}$

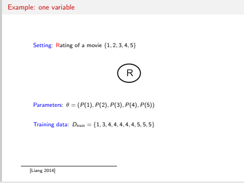


Parameters: $\theta = (P(1), P(2), P(3), P(4), P(5))$

Training data: $D_{train} = \{1, 3, 4, 4, 4, 4, 4, 5, 5, 5\}$

[Liang 2014]

Example: one variable



Example: one variable

Setting: Rating of a movie $\{1, 2, 3, 4, 5\}$

Parameters: $\theta = (P(1), P(2), P(3), P(4), P(5))$

Training data: $D_{train} = \{1, 3, 4, 4, 4, 4, 4, 5, 5, 5\}$

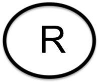
[Liang 2014]

Suppose you want to study how people rate movies. We will develop several Bayesian networks of increasing complexity, and show how to learn the parameters of these models. Let's start with the simplest model, which has one variable representing the movie rating. Here, there are 5 parameters, each one representing the probability of a given rating.

Example: one variable

We want to find θ using D_{train} but ...

... $P(R)$ is proportional to number of occurrences of R in D_{train}



$$D_{train} = \{1, 3, 4, 4, 4, 4, 4, 5, 5, 5\}$$

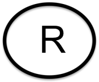
θ :

R	P(R)
1	?
2	?
3	?
4	?
5	?

Example: one variable

We want to find θ using D_{train} but ...

... $P(R)$ is proportional to number of occurrences of R in D_{train}



$$D_{train} = \{1, 3, 4, 4, 4, 4, 4, 5, 5, 5\}$$

θ :

R	P(R)
1	0.1
2	0
3	0.1
4	0.5
5	0.3

Example: two variables

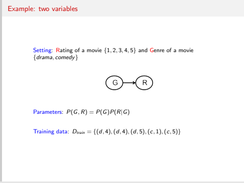
Setting: Rating of a movie $\{1, 2, 3, 4, 5\}$ and Genre of a movie $\{drama, comedy\}$

$$\begin{tikzcd} G \arrow[r] & R \end{tikzcd}$$

Parameters: $P(G, R) = P(G)P(R|G)$

Training data: $D_{train} = \{(d, 4), (d, 4), (d, 5), (c, 1), (c, 5)\}$

Example: two variables



Example: two variables

Setting: Rating of a movie $\{1, 2, 3, 4, 5\}$ and Genre of a movie $\{\mathit{drama}, \mathit{comedy}\}$

$$\begin{tikzcd} G \arrow[r] & R \end{tikzcd}$$

Parameters: $P(G, R) = P(G)P(R|G)$

Training data: $D_{train} = \{(d, 4), (d, 4), (d, 5), (c, 1), (c, 5)\}$

Let's improve the Bayesian network, since people don't rate movies completely randomly; the rating will depend on a number of factors, including the genre of the movie. This yields a two-variable Bayesian network. There should be up to $2 + 2 \times 5 = 12$ parameters in this model.

Example: two variables

We want to find θ using D_{train} but ...

... $P(G)$, $P(R|G)$ are proportional to number of occurrences of R, G in D_{train}

$$\begin{tikzcd} G \arrow[r] & R \end{tikzcd}$$

$$D_{train} = \{(d,4),(d,4),(d,5),(c,1),(c,5)\}$$

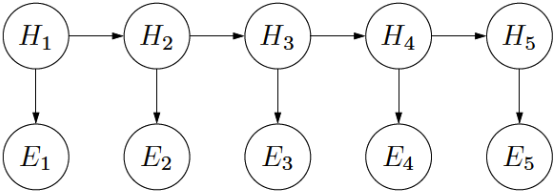
θ :

G	$P(G)$
d	3/5
c	2/5

G	R	$P(R\|G)$
d	4	2/3
d	5	1/3
c	1	1/2
c	5	1/2

Example: HMM

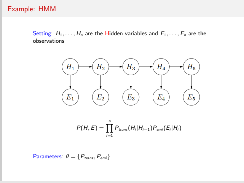
Setting: $H_1, \ldots, H_n$ are the Hidden variables and $E_1, \ldots, E_n$ are the observations



$$P(H, E) = \prod_{i=1}^{n} P_{trans}(H_i | H_{i-1}) P_{emi}(E_i | H_i)$$

Parameters: $\theta = \{P_{trans}, P_{emi}\}$

Example: HMM



Example: HMM

Setting: $H_1, \ldots, H_n$ are the Hidden variables and $E_1, \ldots, E_n$ are the observations

$$\begin{tikzcd}
H_1 \arrow[r] \arrow[d] & H_2 \arrow[r] \arrow[d] & H_3 \arrow[r] \arrow[d] & H_4 \arrow[r] \arrow[d] & H_5 \arrow[d] \\
E_1 & E_2 & E_3 & E_4 & E_5
\end{tikzcd}$$

$$P(H, E) = \prod_{i=1}^{n} P_{trans}(H_i | H_{i-1}) P_{emi}(E_i | H_i)$$

Parameters: $\theta = \{P_{trans}, P_{emi}\}$

The HMM is another model, which we saw was useful for speech recognition or object tracking. With K possible hidden states (values that H_t can take on) and D possible observations, the HMM has K^2 transition parameters and KD emission parameters.

Maximum likelihood

Maximum likelihood objective:

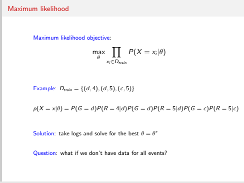
$$\max_{\theta} \prod_{x_i \in D_{train}} P(X = x_i | \theta)$$

Example: $D_{train} = \{(d,4),(d,5),(c,5)\}$

$$p(X = x|\theta) = P(G = d)P(R = 4|d)P(G = d)P(R = 5|d)P(G = c)P(R = 5|c)$$

Solution: take logs and solve for the best $\theta = \theta^*$

Question: what if we don't have data for all events?



Maximum likelihood

Maximum likelihood objective:

$$\max_{\theta} \prod_{x_i \in D_{train}} P(X = x_i|\theta)$$

Example: $D_{train} = \{(d,4),(d,5),(c,5)\}$

$p(X = x|\theta) = P(G = d)P(R = 4|d)P(G = d)P(R = 5|d)P(G = c)P(R = 5|c)$

Solution: take logs and solve for the best $\theta = \theta^*$

Question: what if we don't have data for all events?

So far, we have the idea of count-and-normalise, and hopefully this seems like a reasonable thing to do. But what's the underlying principle? Luckily, we can formalise our approach, as it is nothing more than a closed form solution to the maximum likelihood objective: try to find θ to maximize the likelihood of the data given the parameters.

Maximum likelihood vs Bayesian learning

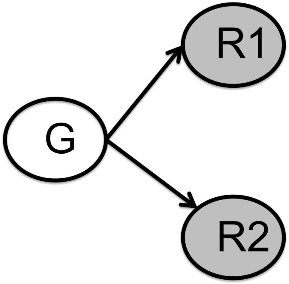
Maximum likelihood

- Assumes that θ is unknown but fixed parameter
- Finds θ^*, the value that maximizes the likelihood

Bayesian learning

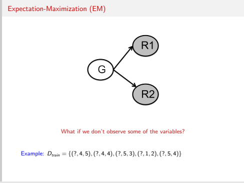
- Treats θ as a random variable
- Assumes a prior probability of θ : $p(\theta)$
- Tries to compute the posterior probability of θ : $p(\theta|D)$

Expectation-Maximization (EM)

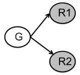


What if we don't observe some of the variables?

Example: $D_{train} = \{(?, 4, 5), (?, 4, 4), (?, 5, 3), (?, 1, 2), (?, 5, 4)\}$



Expectation-Maximization (EM)



What if we don't observe some of the variables?

Example: $D_{train} = \{(?,4,5),(?,4,4),(?,5,3),(?,1,2),(?,5,4)\}$

Until now, we have only described how to learn the parameters. However, one needs to specify two things to describe a BN: the graph topology (structure) and the parameters of each probability distribution. It is possible to learn both of these from data. However, learning structure is much harder than learning parameters. Also, learning when some of the nodes are hidden, or we have missing data, is much harder than when everything is observed. This gives rise to the 4 cases.

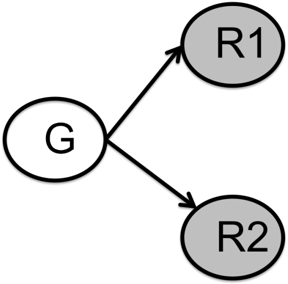
Structure	*Observability*	*Method*
Known	Full	Maximum Likelihood Estimation
Known	Partial	EM
Unknown	Full	Search through model space
Unknown	Partial	EM + search through model space

The EM algorithm is key for many of these tasks. In general, data collection is hard, and often we don't observe the value of every single variable. In this example, we only see the ratings (R_1, R_2), but not the genre G. Can we learn in this setting, which is clearly more difficult?

Expectation-Maximization (EM)

Variables: H is hidden, E is observed (to be e)

Example: $H = \{G\}$, $E = \{R_1, R_2\}$

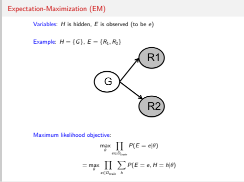


Maximum likelihood objective:

$$\max_\theta \prod_{e \in D_{train}} P(E = e | \theta)$$

$$= \max_\theta \prod_{e \in D_{train}} \sum_h P(E = e, H = h | \theta)$$

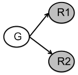
Expectation-Maximization (EM)



Expectation-Maximization (EM)

Variables: H is hidden, E is observed (to be e)

Example: $H = \{G\}$, $E = \{R_1, R_2\}$



Maximum likelihood objective:

$$\max_{\theta} \prod_{e \in D_{train}} P(E = e|\theta)$$
$$= \max_{\theta} \prod_{e \in D_{train}} \sum_{h} P(E = e, H = h|\theta)$$

If there were no hidden variables, then we would just use maximum likelihood. But since H is unobserved, we can simply replace the joint probability $P(H = h, E = e|\theta)$ with the marginal probability $P(E = e|\theta)$, which is just a sum over values h that the hidden variables H could take on.

Expectation-Maximization (EM)

Algorithm: Expectation Maximization

E-step:

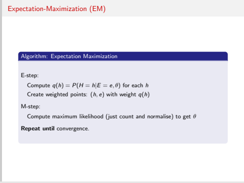
Compute $q(h) = P(H = h | E = e, \theta)$ for each h

Create weighted points: (h, e) with weight $q(h)$

M-step:

Compute maximum likelihood (just count and normalise) to get θ

Repeat until convergence.



Expectation-Maximization (EM)

Algorithm: Expectation Maximization

E-step:

Compute $q(h) = P(H = h|E = e, \theta)$ for each h

Create weighted points: (h, e) with weight $q(h)$

M-step:

Compute maximum likelihood (just count and normalise) to get θ

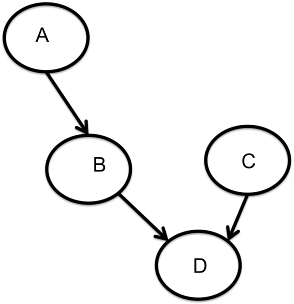
Repeat until convergence.

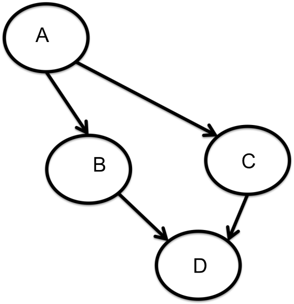
To get intuition for EM, consider K-means, which turns out to be a special case of EM (for Gaussian mixture models with variance tending to 0). In K-means, we had to somehow estimate the cluster centers, but we didn't know which points were assigned to which clusters. And in that setting, we took an alternating optimisation approach: find the best cluster assignment given the current cluster centers, find the best cluster centers given the assignments, etc.

The EM algorithm works analogously. EM consists of alternating between two steps, the **E-step** and the **M-step**. In the E-step, we don't know what the hidden variables are, so we compute the posterior distribution over them given our current parameters ($P(H|E = e, \theta)$). This can be done using any probabilistic inference algorithm. If H takes on a few values, then we can enumerate over all of them. These posterior distributions provide a weight $q(h)$ (which is a temporary variable in the EM algorithm) to every value h that H could take on. Conceptually, the E-step then generates a set of weighted full assignments (h, e) with weight $q(h)$. In the M-step, we take in our set of full assignments (h, e) with weights, and we just do maximum likelihood estimation, which can be done in closed form - just counting and normalising. If we repeat the E-step and the M-step over and over again, we are guaranteed to converge to a local optima. Just like the K-means algorithm, we might need to run the algorithm from different random initializations of θ and take the best one.

Model selection

Given a new dataset $\{A, B, C, D\}$, we can evaluate the probability of each model structure (using the parameters we learned by maximum likelihood) and pick the model with the highest $P(A, B, C, D|parameters)$.





For more information:
`http://research.microsoft.com/en-us/um/people/heckerman/tutorial.pdf`

Next lecture

In the next lecture, we will present a application area: text mining!