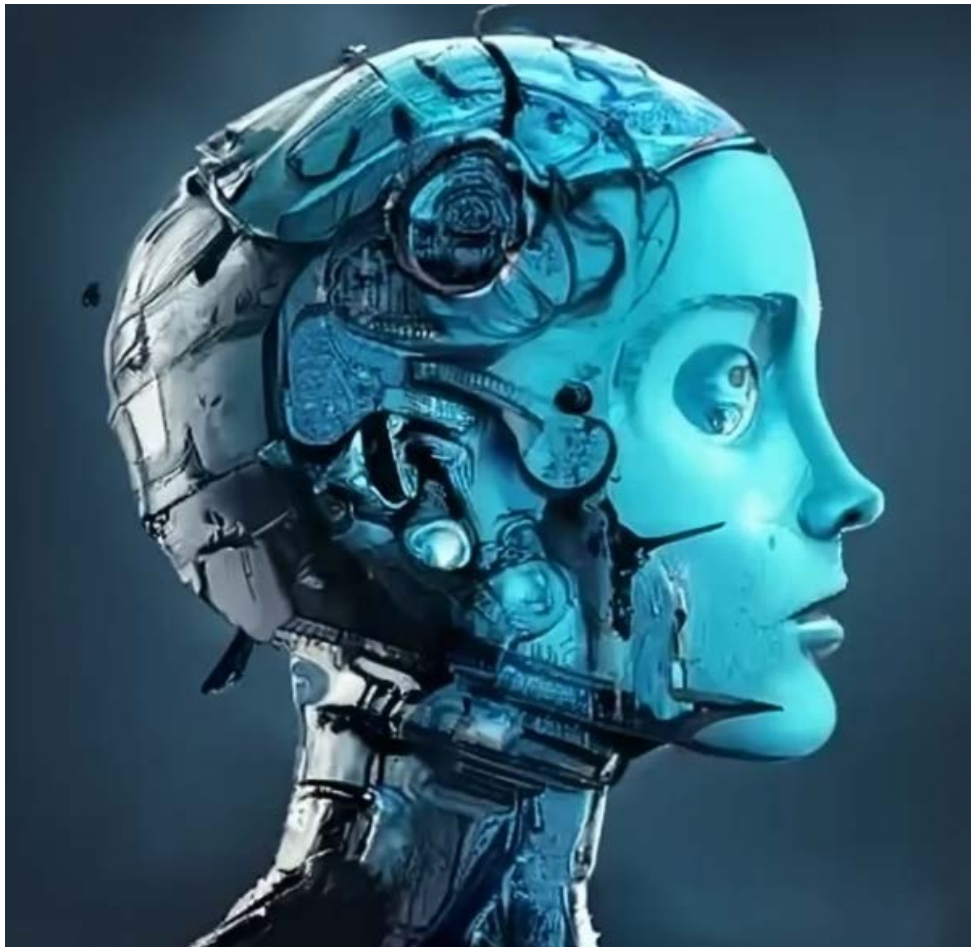
Machine Intelligence

Bayes Theorem and Applications

Dr. Cory Merkel



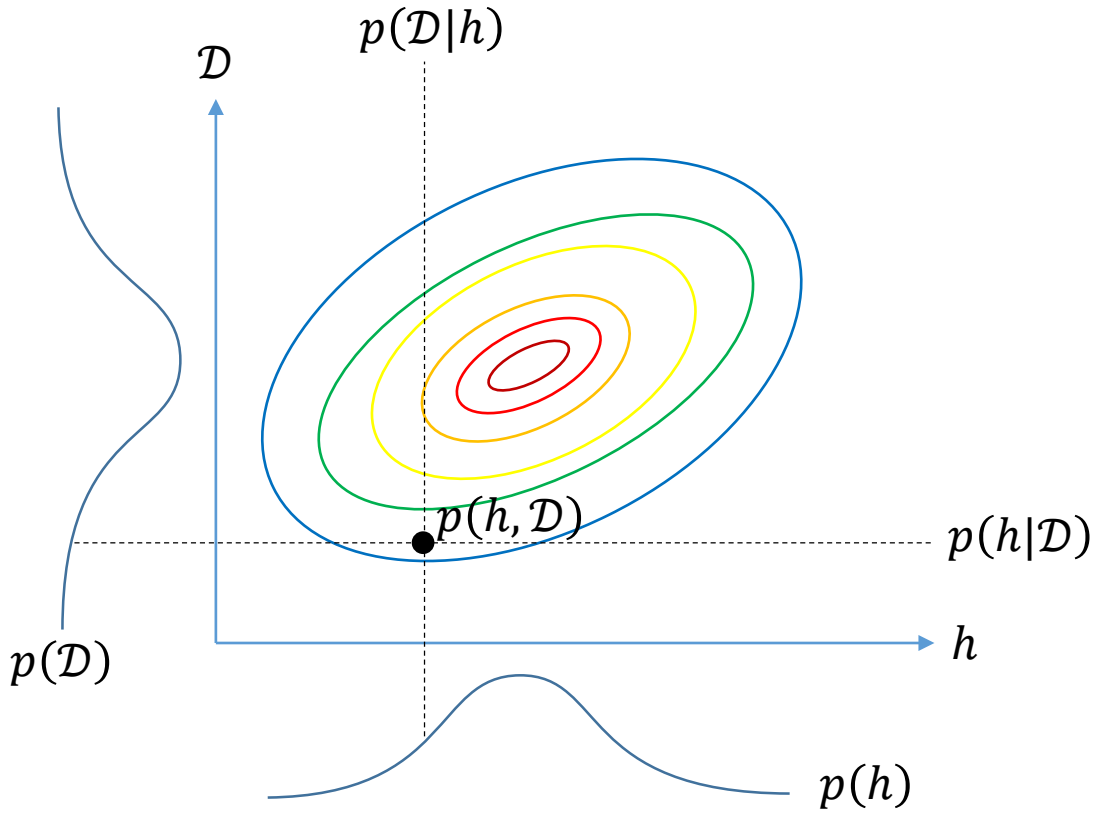
Bayes Theorem

- The joint probability for two random variables is

$$p(h, \mathcal{D}) = p(h|\mathcal{D})p(\mathcal{D}) = p(\mathcal{D}|h)p(h)$$

- Therefore:

$$\underbrace{p(h|\mathcal{D})}_{\text{posterior}} = \frac{\overbrace{p(\mathcal{D}|h)}^{\text{likelihood}}\overbrace{p(h)}^{\text{prior}}}{\underbrace{p(\mathcal{D})}_{\text{marginal or "evidence"}}}$$



Maximum a-posteriori (MAP) Hypothesis

- What is the most likely hypothesis given the observed data and the prior probability distribution over the hypothesis space $\mathcal{H}$?

$$h_{MAP} \equiv \arg\max_{h \in \mathcal{H}} p(h|\mathcal{D})$$

$$= \arg\max_{h \in \mathcal{H}} \frac{p(\mathcal{D}|h)p(h)}{p(\mathcal{D})}$$

$$= \arg\max_{h \in \mathcal{H}} p(\mathcal{D}|h)p(h)$$

No need for denominator since it is independent of h!

Maximum a-posteriori (MAP) Hypothesis

- Notice that if $p(h)$is constant (equal probability of all hypotheses then

$$\arg\max_{h \in \mathcal{H}} p(\mathcal{D}|h)p(h) = \arg\max_{h \in \mathcal{H}} p(\mathcal{D}|h) = h_{ML}$$

- So, in this case, our MAP hypothesis is equal to our maximum likelihood hypothesis

Example: Breast Cancer

- Imagine that we have developed a new test for breast cancer.
  - The test is 98% accurate when the patient has cancer and 97% accurate when the patient does not have cancer
  - The population's breast cancer rate is 0.8%
- Given a patient with 1 positive test result, should we tell them that they have cancer?

$$h_{MAP} = \arg\max_{h \in \mathcal{H}} p(\mathcal{D}|h)p(h)$$

$$p(positive|cancer)p(cancer) = 0.98 \times 0.008 = 0.00784$$

$$p(positive|\neg cancer)p(\neg cancer) = 0.03 \times 0.992 = 0.0298$$

$$\therefore h_{MAP} = \neg cancer$$

- Note the probability of $cancer$ is $\frac{0.00784}{0.00784+0.0298} 0.21$

Bayes Optimal Classifier

- For a given classification problem with dataset $\mathcal{D}$, we may find either the MAP or ML hypothesis.
  - Note that there are many other (maybe infinite) hypotheses that are less likely than h_{MAP} or h_{ML}
  - But, if a lot of less likely hypotheses are telling us one answer, and only one most likely hypothesis is telling us another, shouldn't we believe the majority?

$$\hat{l}_{BO} = \arg\max_{\hat{l}_j \in c} \sum_{h_i \in \mathcal{H}} p(\hat{l}_j | h_i) p(h_i | \mathcal{D})$$

For simplicity we use the label $\hat{l}$ instead of the one-hot vector $\hat{\mathbf{y}}$, where $\hat{l}$ is the index of the maximum value of $\hat{\mathbf{y}}$.

- It can be shown that no other classification method can beat this on average

Example: Breast Cancer

- Suppose another breast cancer test was developed that is 99% accurate when a patient has cancer and 99.9% accurate when a patient does not.
- Now, a new patient comes in, and based on their history and prior studies, a doctor is 60% likely to choose test A and 40% likely to choose test B.

$$\hat{l}_{BO} = \arg\max_{\hat{l}_j \in c} \sum_{h_i \in \mathcal{H}} p(\hat{l}_j | h_i) p(h_i | \mathcal{D})$$

$$= \arg\max_{\hat{l}_j \in c} \left(p(\hat{l} = cancer | test\ A) \times 0.6 + p(\hat{l} = cancer | test\ B) \times 0.4, p(\hat{l} = \neg cancer | test\ A) \times 0.6 + p(\hat{l} = \neg cancer | test\ B) \times 0.4 \right)$$

$$p(cancer | test\ A) = 0.21,\ p(cancer | test\ B) = 0.89$$

This would lead to a 48.2% chance of cancer (using both tests).

Naïve Bayes Classifier

- Applying Bayes classification is hard in practice. For some new instance $\mathbf{u}$ that we want to classify:

$$\hat{l}_{MAP} = \arg\max_{\hat{l}_j \in c} p(l_j|\mathbf{u}) = \arg\max_{l_j \in c} p(\mathbf{u}|l_j)p(l_j) = \arg\max_{l_j \in c} p(u_1, u_2, \ldots, u_N|l_j)p(l_j)$$

- Evaluating these probabilities will be challenging because the number of possible pairs $(\mathbf{u}, l_j)$ is huge.
- Notice that if each u_i is independent of all other u_i given l_j then we can write $p(u_1, u_2, \ldots, u_N|l_j) = \prod_i p(u_i|l_j)$
  - In this case we call our classifier a naïve Bayes classifier:

$$\hat{l}_{NB} = \arg\max_{l_j \in c} p(l_j) \prod_i p(u_i|l_j)$$

Getting good estimates of these probabilities is much easier since the number of possible pairs (u_i, l_j) is much smaller.

Example: Tennis

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

- Suppose on a given day we have the following data: *outlook = sunny, temperature = cool, humidity = high, wind = strong*
- Given this and the past data will we play tennis?
  - We can use naïve Bayes for this

$$\hat{l}_{NB} = \arg\max_{l_j \in c} p(l_j) \prod_i p(u_i|l_j)$$

- We have two possible labels (yes and no)

$$p(l_{yes}) = \frac{9}{14} = 0.64, \qquad p(l_{no}) = \frac{5}{14} = 0.36$$

Example: Tennis

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

- We need to calculate each $p(u_i|l_j)$

e.g.:

$$p(sunny|l_{yes}) = 2/9$$

Now, we compute:

$$p(l_{yes})p(sunny|l_{yes})p(cool|l_{yes})p(high|l_{yes})p(strong|l_{yes}) = 0.0053$$
$$p(l_{no})p(sunny|l_{no})p(cool|l_{no})p(high|l_{no})p(strong|l_{no}) = 0.0206$$

Therefore, we have $\hat{l}_{NB} = l_{no}$

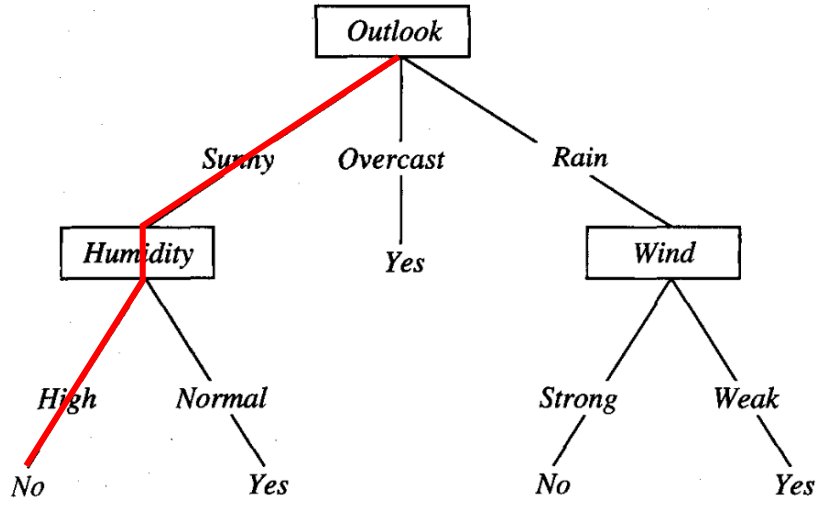
- We can also compute the probability of playing tennis:

$$p(l_{yes}) = \frac{p(l_{yes})p(sunny|l_{yes})p(cool|l_{yes})p(high|l_{yes})p(strong|l_{yes})}{p(sunny, cool, high, srong)} = \frac{0.0053}{0.0053 + 0.0206} = 0.2$$

Example: Tennis

- How does this compare with our decision tree model?

$\mathbf{u}$ =[*outlook = sunny, temperature = cool, humidity = high, wind = strong*]



It matches! But, with Bayes approach, we get more than just the classification, we also get the probability.

Bayesian Networks

- When we performed naïve Bayes, we calculated

$$\hat{l}_{NB} = \arg\max_{l_j \in c} p(l_j) \prod_i p(u_i|l_j)$$

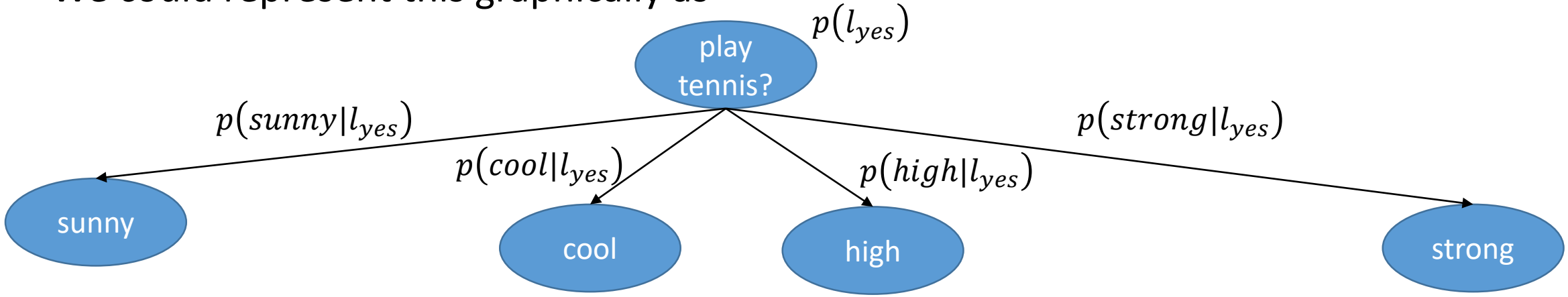
as the l corresponding to the maximum of

$$p(l_{yes})p(sunny|l_{yes})p(cool|l_{yes})p(high|l_{yes})p(strong|l_{yes})$$

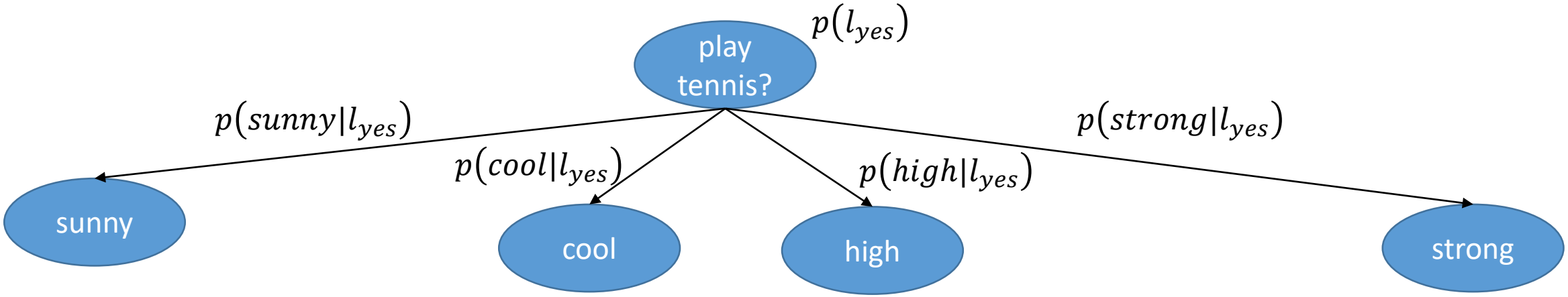
and

$$p(l_{no})p(sunny|l_{no})p(cool|l_{no})p(high|l_{no})p(strong|l_{no})$$

- We could represent this graphically as



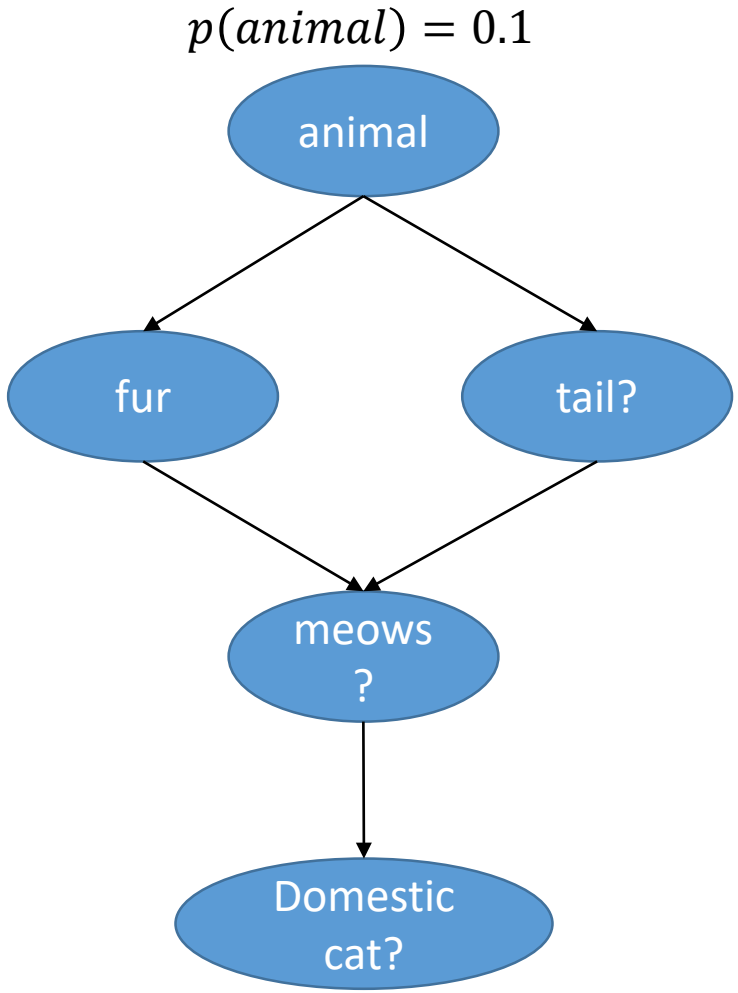
Bayesian Networks



- Bayesian networks represent conditional probabilities among a set of random variables
  - Acyclic directed graphs
  - Child nodes are independent of all non-descendent nodes given their parents
    - Ex: Whether or not it's cool out only depends on whether we play tennis, and not on whether it's sunny, high humidity, or strong wind (this is the naïve Bayes assumption we've made)

Bayesian Networks

- Suppose we have a dataset of images and audio, where 10% of the data correspond to animals.
- We might build a Bayes Net as shown to the right
- To get a full picture of the relationships between the variables, we need to know all of the conditional probabilities as well

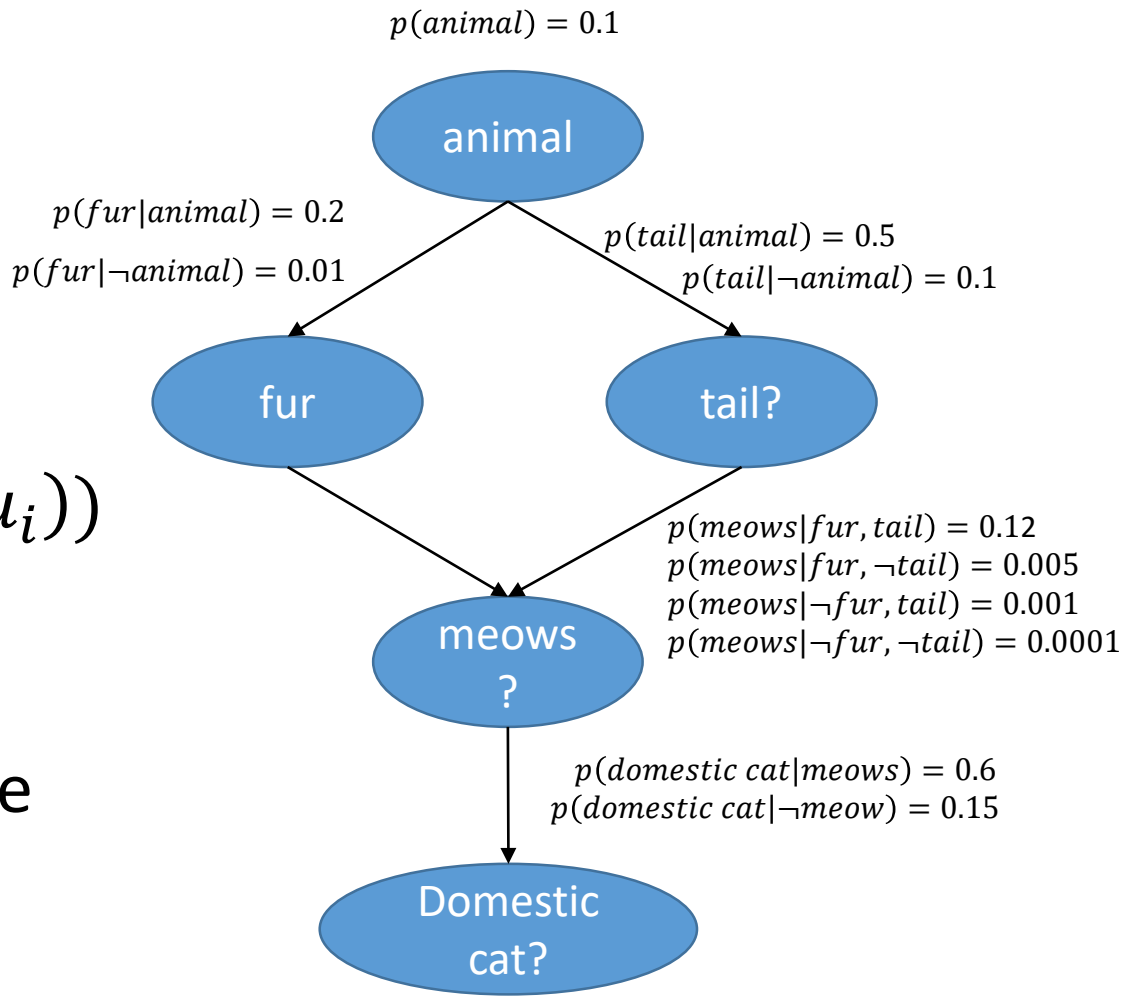


Bayesian Networks

- Using a Bayes net, we can calculate any joint probability as

$$p(u_1, u_2, \ldots, u_N) = \prod_{i=1}^{N} p(u_i | Parents(u_i))$$

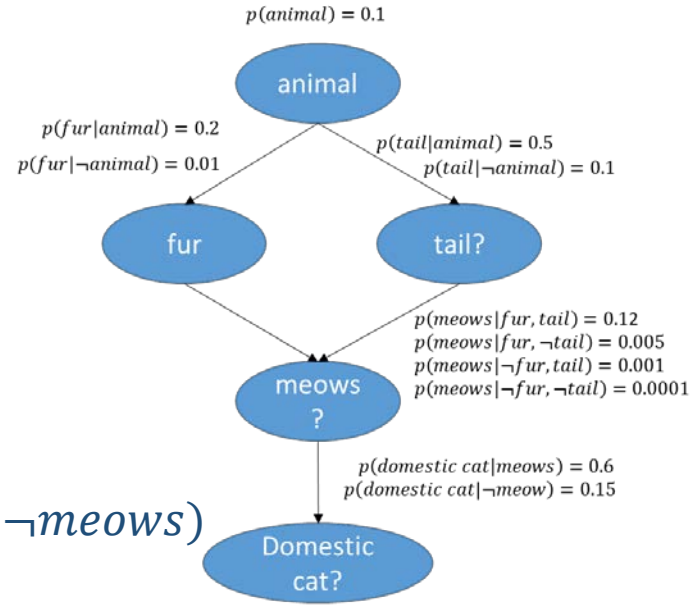
- Here, $Parents(u_i)$ is the set of variable that u_i depends on
  - E.g.: $Parents(meows) = fur, tail$



Notice that each node has a conditional probability table (CPT) associated with it, with rows corresponding to values of parent variables and columns for node variables:

	cat	$\neg cat$
$meows$	0.6	0.4
$\neg meows$	0.15	0.85

Bayesian Networks



- Let's compute
  - $p(animal, \neg fur, tail, \neg meows, \neg cat) =$
    $p(animal)p(\neg fur|animal)p(tail|animal)p(\neg meows|\neg fur, tail)p(\neg cat|\neg meows)$
    $= 0.1 \times 0.8 \times 0.5 \times 0.999 \times 0.85 = 0.034$

  - What if we know that it's an animal?

$(\neg fur|animal)p(tail|animal)p(\neg meows|\neg fur, tail)p(\neg cat|\neg meows) = 0.8 \times 0.5 \times 0.999 \times 0.85 = 0.34$

  - What if we want to know if something has fur regardless of whether or not it's an animal?
    - $P(B) = P(B, A) + P(B, \neg A)$

$p(fur) = p(fur, animal) + p(fur, \neg animal) = p(fur|animal)p(animal) + p(fur|\neg animal)p(\neg animal)$
$= 0.2 \times 0.1 + 0.01 \times 0.9 = 0.029$

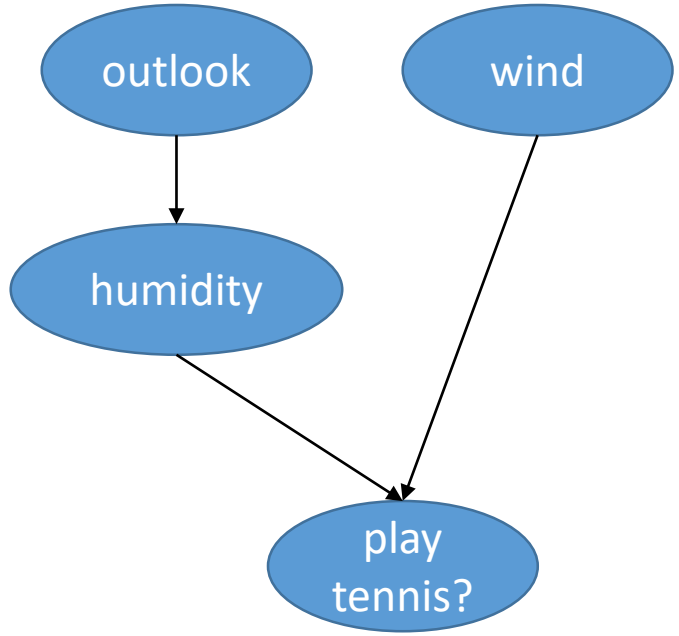
Bayesian Networks

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

- How can we learn the probabilities in the Bayes net?
- Suppose we came up with the following Bayes net structure:
  - We need to learn the following conditional probabilities:
    - $p(high|sunny), p(high|overcast), p(high|rain)$
    - $p(yes|high, strong), p(yes|high, weak), p(yes|normal, weak), p(yes|normal, strong)$
  - Based on the data, we get the following CPTs

	$high$	$\neg high$
$sunny$	3/5	2/5
$overcast$	2/4	2/4
$rain$	2/5	3/5

	yes	no
$high, strong$	1/3	2/3
$high, weak$	2/4	2/4
$normal, weak$	4/4	0/4
$normal, strong$	2/3	1/3



Bayesian Networks

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

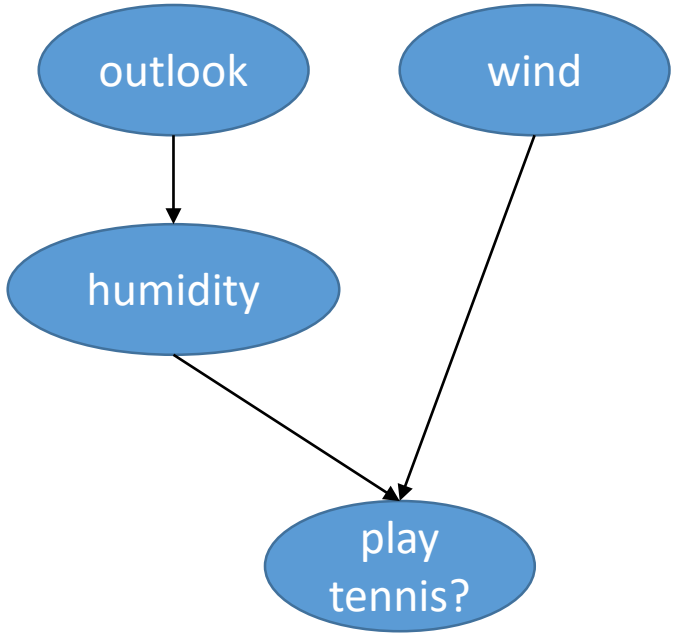
- Using our network, let's compute $p(yes, overcast, high, strong) = p(yes|high, strong)p(overcast)p(high|overcast)p(strong)$
- From our data, we have $p(overcast) = 4/14$ and $p(strong) = 6/14$, so

$$p(yes, overcast, high, strong) = \frac{1}{3} \times \frac{4}{14} \times \frac{1}{2} \times \frac{6}{14} = 0.02$$

- But, we have a case of $(yes, overcast, high, strong)$ in our dataset! What happened!?

	$high$	$\neg high$
$sunny$	3/5	2/5
$overcast$	2/4	2/4
$rain$	2/5	3/5

	yes	no
$high, strong$	1/3	2/3
$high, weak$	2/4	2/4
$normal, weak$	4/4	0/4
$normal, strong$	2/3	1/3



Gradient Ascent for Bayesian Networks

- It looks like our model (Bayes net) may not be representing our dataset as well as we would like.
- This could happen because:
  - The *structure* of our Bayes net is not correct
  - The CPTs don't have accurate values
- In general, we would like our Bayes net to reflect the data that we have observed
  - We want to maximize the likelihood of seeing the same data come from our model: $p(\mathcal{D}|h)$, where h is the hypothesis given by the Bayes net.
  - Instead of trying to estimate the CPTs from the dataset, could we *learn* them by maximizing $p(\mathcal{D}|h)$ (maximum likelihood estimation).

Gradient Ascent for Bayesian Networks

- Consider the CPT for a node $u_i \in \{a_{i1}, a_{i2}, \ldots\}$ with parents $Parents(u_i) \in \{a'_1, a'_2, \ldots\}$

		$u_i = a_{ij}$	
	…	…	…
$Parents(u_i) = a'_{ik}$	…	w_{ijk}	…
	…	…	…

- So $w_{ijk} \equiv p\left(u_i = a_{ij} \middle| Parents(u_i) = a'_{ik}\right)$
  - These are the conditional probabilities that we want to learn

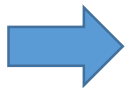
Gradient Ascent for Bayesian Networks

- We can use gradient ascent to find the w_{ijk} values by maximizing the log likelihood of the data:

$$\{w_{ijk}\}_{ML} = \arg\max_{\{w_{ijk}\}} \ln p(\mathcal{D}|\{w_{ijk}\})$$

- For gradient ascent, we need the gradient of $\ln p(\mathcal{D}|\{w_{ijk}\})$ w.r.t. each w_{ijk}

$$\frac{\partial \ln p(\mathcal{D}|\{w_{ijk}\})}{\partial w_{ijk}} = \sum_{\mathbf{u}\in\mathcal{D}} \frac{\partial \ln p(\mathbf{u}|\{w_{ijk}\})}{\partial w_{ijk}} = \sum_{\mathbf{u}\in\mathcal{D}} \frac{1}{p(\mathbf{u}|\{w_{ijk}\})} \frac{\partial p(\mathbf{u}|\{w_{ijk}\})}{\partial w_{ijk}}$$

After a bit of work: 

$$\frac{\partial \ln p(\mathcal{D}|\{w_{ijk}\})}{\partial w_{ijk}} = \sum_{\mathbf{u}\in\mathcal{D}} \frac{p(a_{ij}, a'_{ik}|\mathbf{u}, \{w_{ijk}\})}{w_{ijk}}$$

Gradient Ascent for Bayesian Networks

- So, we can update our conditional probabilities as

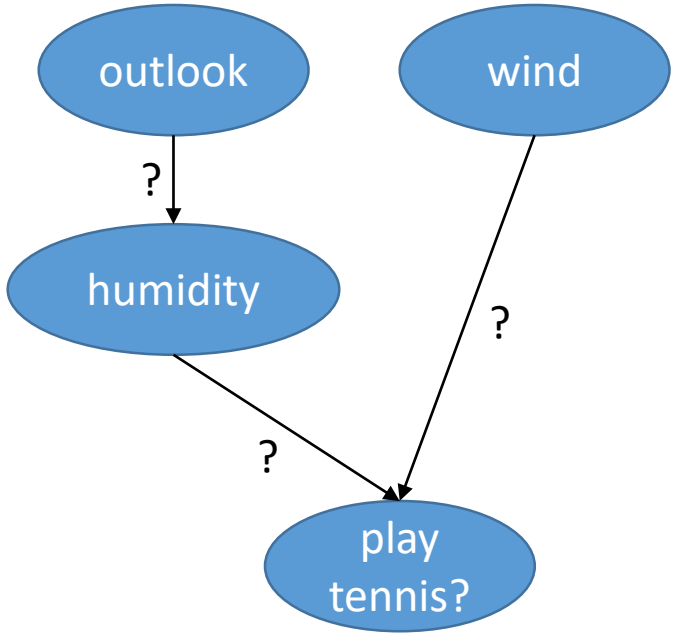
$$w_{ijk} := w_{ijk} + \alpha \sum_{\mathbf{u} \in \mathcal{D}} \frac{p\left(a_{ij}, a'_{ik} \middle| \mathbf{u}, \{w_{ijk}\}\right)}{w_{ijk}}$$

- After we update the w_{ijk} values in each iteration, we need to make sure that
  - Each one is between 0 and 1 and $\sum_j w_{ijk} = 1 \; \forall \; i, k$

Expectation Maximization

- What if we have cases where variables are missing?
- Here, we can use the expectation maximization algorithm (EM)
  - Foundational algorithm in machine learning that allows us to estimate a set of parameters θ of a probability distribution given only partial data

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain		Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal		Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild		Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



Expectation Maximization

- Suppose we have a dataset $\mathcal{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots\}$, where each component of each $\mathbf{x}$ is observed (known). Note that each $\mathbf{x}$ could have a different length in this case.
- Let $\mathcal{Z} = \{\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots\}$ be the unobserved components of each instance
- Our dataset $\mathcal{D} = \mathcal{X} \cup \mathcal{Z}$
  - E.g., our dataset for the tennis problem is and $\mathbf{x}^5 = [rain, normal, weak, yes], \mathbf{z}^5 = [temp^5]$

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain		Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal		Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild		Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Expectation Maximization

- To find estimates of the parameters θ, we can use maximum likelihood estimation

$$\theta_{ML} = \arg\max_{\theta} \ln p(\mathcal{D}|\theta)$$

- However, notice that $\ln p(\mathcal{D}|\theta)$ is itself a random variable since there are unknown values in $\mathcal{D}$. So, we should take the expected value

$$\theta_{ML} = \arg\max_{\theta} \mathbb{E}[\ln p(\mathcal{D}|\theta)]$$

- But, how do we compute $\mathbb{E}[\ln p(\mathcal{D}|\theta)]$ if we don't know θ?

Expectation Maximization

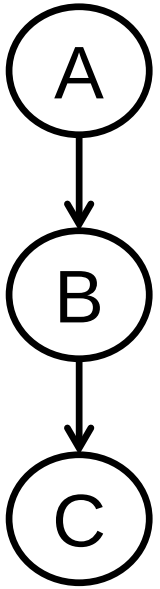
- Key idea: Iteratively find the maximum value by 1.) Finding the expectation given the current estimates of the parameters and the *observable* data (E step), 2.) Getting new estimates of the parameters by maximizing the expectation (M step):

Initialize θ
while not converged:
    E step: $Q(\theta'|\theta) \coloneqq \mathbb{E}_{\mathcal{D}}[\ln p(\mathcal{D}|\theta')\,|\theta, \mathcal{X}]$
    M step: $\theta \coloneqq \underset{\theta'}{arg\,max}\, Q(\theta'|\theta)$

Example

- Suppose we want to learn the following Bayes net
  - Note that one value is missing.
  - Use EM to find most reasonable value of x

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	*x*	0



- **<u>Initialization</u> (focus on set of probabilities to know everything else)**

P(A) =0.75

P(B|A) =

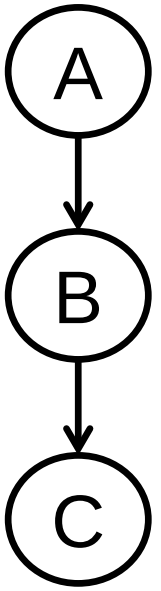
P(B|A') =

P(C|B) =

P(C|B') =

Example

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	*x*	0



- **Initialization**

Ignore missing data

P(A) =0.75

P(B|A) = 0.5

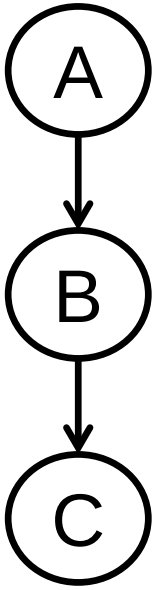
P(B|A') =

P(C|B) =

P(C|B') =

Example

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	*x*	0



- **Initialization**

Ignore missing data

P(A) =0.75

P(B|A) = 0.5

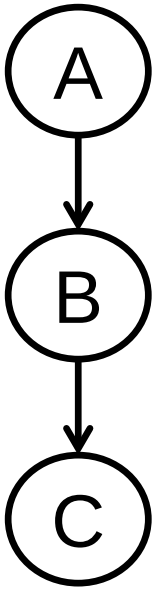
P(B|A') = 1

P(C|B) =

P(C|B') =

Example

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	*x*	0



- **Initialization**

Ignore missing data

P(A) =0.75

P(B|A) = 0.5

P(B|A') = 1

P(C|B) = 1

P(C|B') = 0 (never happening)

Example

Initialize θ
while not converged:
    E step: $Q(\theta'|\theta) \coloneqq \mathbb{E}_{\mathcal{D}}[\ln p(\mathcal{D}|\theta')\,|\theta, \mathcal{X}]$
    M step: $\theta \coloneqq \underset{\theta'}{arg\,max}\, Q(\theta'|\theta)$

- Our θ values are our probabilities
- The E step asks: What is the expected log-likelihood of our data for some set of probabilities. We compute this expectation based on our *current* estimates of the probabilities and the data that we observe
- The M step asks: What is the set of probabilities that maximizes this expectation?

Example: E step

- We have 2 possible choices for our data ($x = 0$ or $x = 1$), so

$$Q(\theta'|\theta) \coloneqq \mathbb{E}_{\mathcal{D}}[\ln p(\mathcal{D}|\theta')\,|\theta, \mathcal{X}] = p(x = 0|\theta, \mathcal{X}) \ln p(\mathcal{X}, x = 0|\theta') + p(x = 1|\theta, \mathcal{X}) \ln p(\mathcal{X}, x = 1|\theta')$$

$$p(x = 0|\theta, \mathcal{X}) = p(B'|A, C') = \frac{p(A, B', C')}{p(A, C')}$$

$$= \frac{p(A)p(B'|A)p(C'|B')}{p(A)p(B'|A)p(C'|B') + p(A)p(B|A)p(C'|B)}$$

$$= \frac{0.75 \times 0.5 \times 1}{0.75 \times 0.5 \times 1 + 0.75 \times 0.5 \times 0} = 1$$

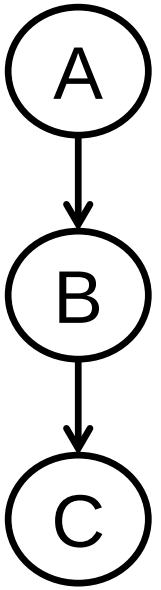
$$p(x = 1|\theta, \mathcal{X}) = 0$$

- So,

$$Q(\theta'|\theta) \coloneqq 1 \times \ln p(\mathcal{X}, x = 0|\theta') + 0 \times \ln p(\mathcal{X}, x = 1|\theta') = \ln p(\mathcal{X}, x = 0|\theta')$$

Example: M step

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	***0***	0

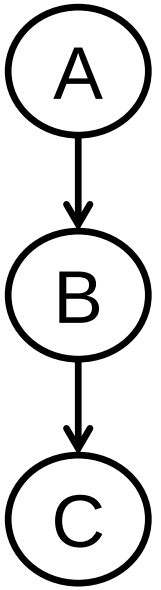


$$\theta := \arg\max_{\theta'} Q(\theta'|\theta) = \arg\max_{\theta'} \ln p(\mathcal{X}, x = 0|\theta')$$

- Now, we maximize the likelihood of the data, assuming $x = 0$
  - This can be done by counting the frequency of occurrences

Example: M step

	A	B	C
1)	0	1	1
2)	1	0	0
3)	1	1	1
4)	1	***0***	0



P(A) =0.75

P(B|A) = 0.33

P(B|A') = 1

P(C|B) = 1

P(C|B') = 0 (never happening)

Example: E step

- Next iteration:
- We *still* have 2 possible choices for our data ($x = 0$ or $x = 1$), so

$$Q(\theta'|\theta) \coloneqq \mathbb{E}_{\mathcal{D}}[\ln p(\mathcal{D}|\theta')\,|\theta, \mathcal{X}] = p(x = 0|\theta, \mathcal{X}) \ln p(\mathcal{X}, x = 0|\theta') + p(x = 1|\theta, \mathcal{X}) \ln p(\mathcal{X}, x = 1|\theta')$$

$$p(x = 0|\theta, \mathcal{X}) = p(B'|A, C') = \frac{p(A, B', C')}{p(A, C')}$$

$$= \frac{p(A)p(B'|A)p(C'|B')}{p(A)p(B'|A)p(C'|B') + p(A)p(B|A)p(C'|B)}$$

$$= \frac{0.75 \times 0.33 \times 1}{0.75 \times 0.33 \times 1 + 0.75 \times 0.33 \times 0} = 1$$

$$p(x = 1|\theta, \mathcal{X}) = 0$$

- So,

$$Q(\theta'|\theta) \coloneqq 1 \times \ln p(\mathcal{X}, x = 0|\theta') + 0 \times \ln p(\mathcal{X}, x = 1|\theta') = \ln p(\mathcal{X}, x = 0|\theta')$$

- Since the probabilities haven't changed, the EM algorithm has converged!