Machine Learning Department
School of Computer Science
Carnegie Mellon University

Bayesian Networks

+

Reinforcement Learning:
Markov Decision Processes

Matt Gormley
Lecture 21
Apr. 4, 2022

Reminders

- Homework 7: HMMs
  - Out: Fri, Apr. 1
  - Due: Tue, Apr. 12 at 11:59pm

GRAPHICAL MODELS: DETERMINING CONDITIONAL INDEPENDENCIES

What Independencies does a Bayes Net Model?

- In order for a Bayesian network to model a probability distribution, the following must be true:

  Each variable is conditionally independent of all its non-descendants in the graph given the value of all its parents.

- This follows from $P(X_1, \ldots, X_T) = \prod_{t=1}^{T} P(X_t \mid \text{parents}(X_t))$

$$= \prod_{t=1}^{T} P(X_t \mid X_1, \ldots, X_{t-1})$$

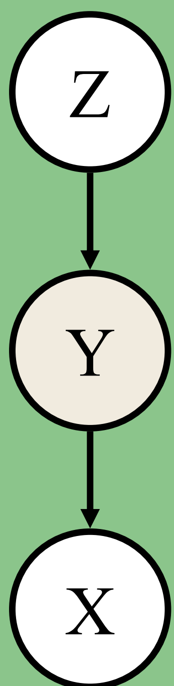
- But what else does it imply?

Slide from William Cohen

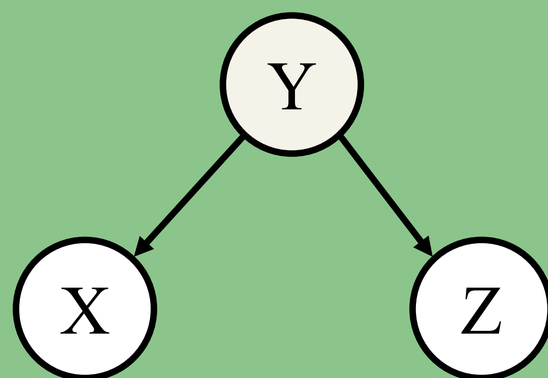
What Independencies does a Bayes Net Model?

Three cases of interest…

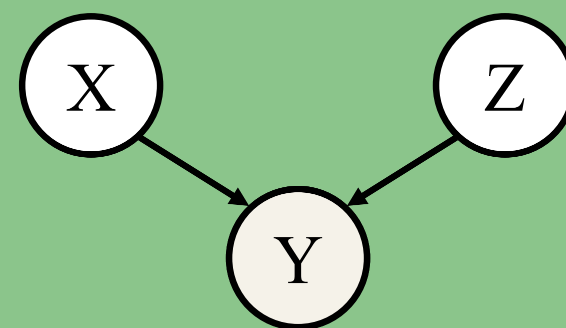
Cascade



Common Parent



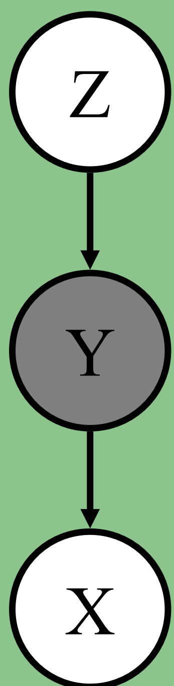
V-Structure



What Independencies does a Bayes Net Model?

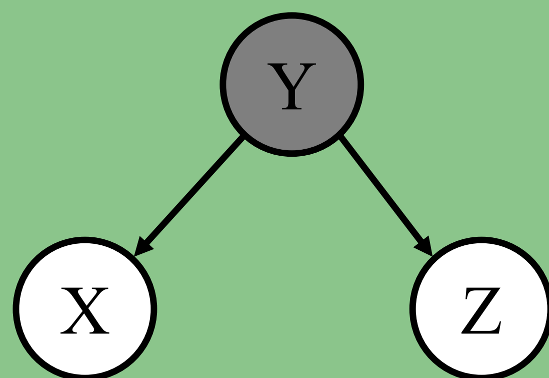
Three cases of interest…

Cascade



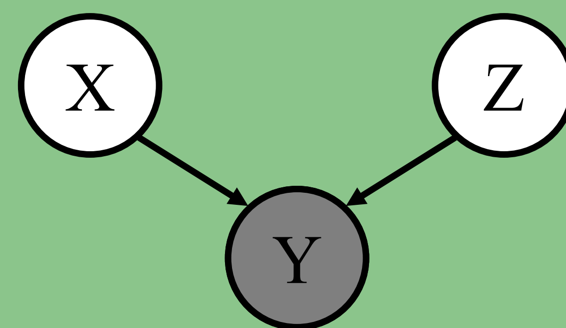
$$X \perp\!\!\!\perp Z \mid Y$$

Common Parent



$$X \perp\!\!\!\perp Z \mid Y$$

V-Structure



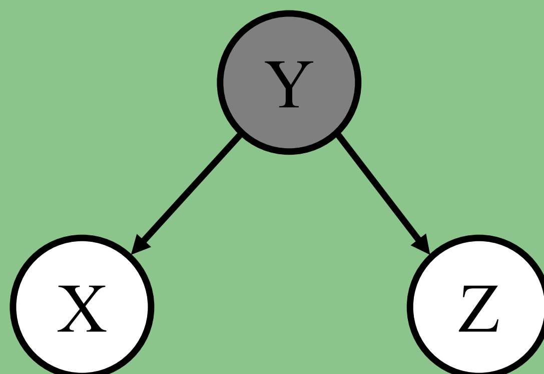
$$X \not\perp\!\!\!\perp Z \mid Y$$

Knowing Y
decouples X and Z

Knowing Y
couples X and Z

Whiteboard

Common Parent



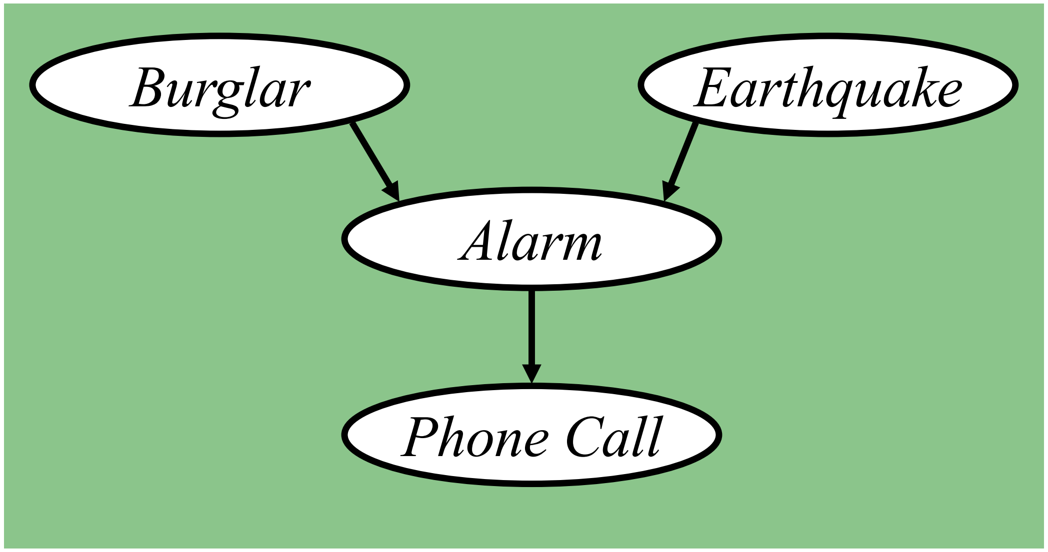
$$X \perp\!\!\!\perp Z \mid Y$$

Proof of conditional independence

(The other two cases can be shown just as easily.)

The "Burglar Alarm" example

- Your house has a twitchy burglar alarm that is also sometimes triggered by earthquakes.
- Earth arguably doesn't care whether your house is currently being burgled
- While you are on vacation, one of your neighbors calls and tells you your home's burglar alarm is ringing. Uh oh!



Quiz: True or False?

$$Burglar \perp\!\!\!\perp Earthquake \mid PhoneCall$$

Slide from William Cohen

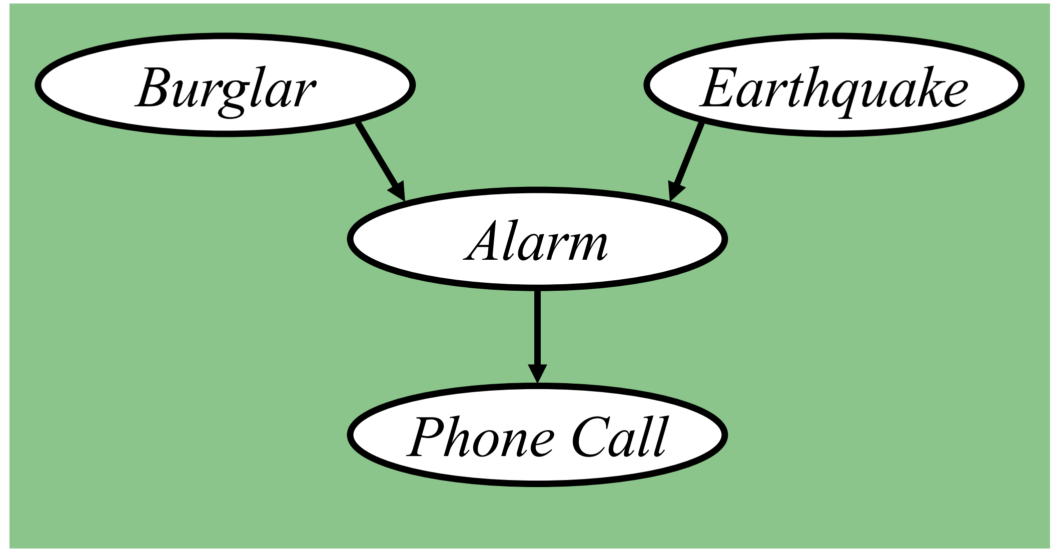
The "Burglar Alarm" example

- After you get this phone call, suppose you learn that there was a medium-sized earthquake in your neighborhood. Oh, whew! Probably not a burglar after all.
- Earthquake "explains away" the hypothetical burglar.
- But then it must **not** be the case that

  $$Burglar \perp\!\!\!\perp Earthquake \mid PhoneCall$$

  even though

  $$Burglar \perp\!\!\!\perp Earthquake$$

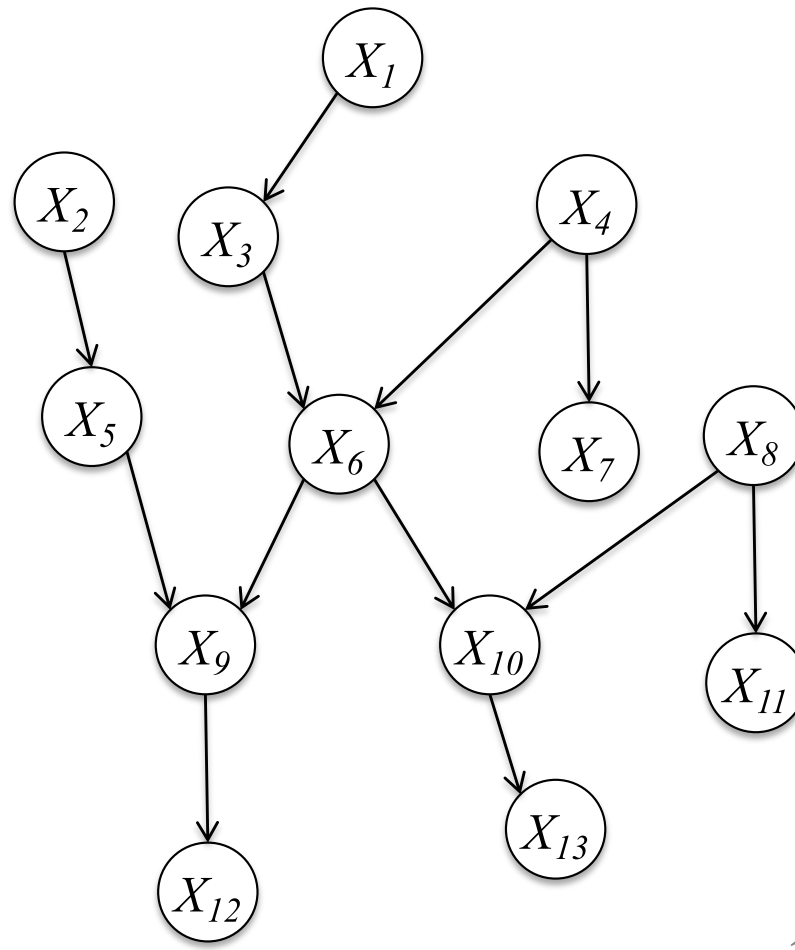


Slide from William Cohen

Markov boundary

Def: the **co-parents** of a node are the parents of its children

Def: the **Markov boundary** of a node is the set containing the node's parents, children, and co-parents.

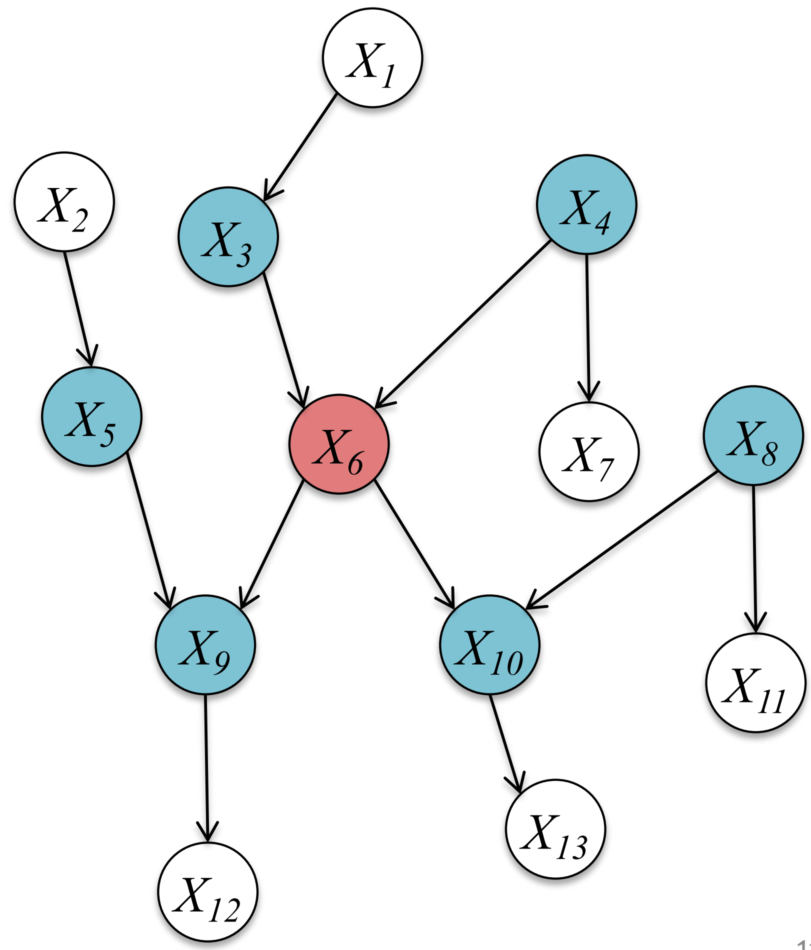


Markov boundary

Def: the **co-parents** of a node are the parents of its children

Def: the **Markov boundary** of a node is the set containing the node's parents, children, and co-parents.

Example: The Markov boundary of X_6 is $\{X_3, X_4, X_5, X_8, X_9, X_{10}\}$



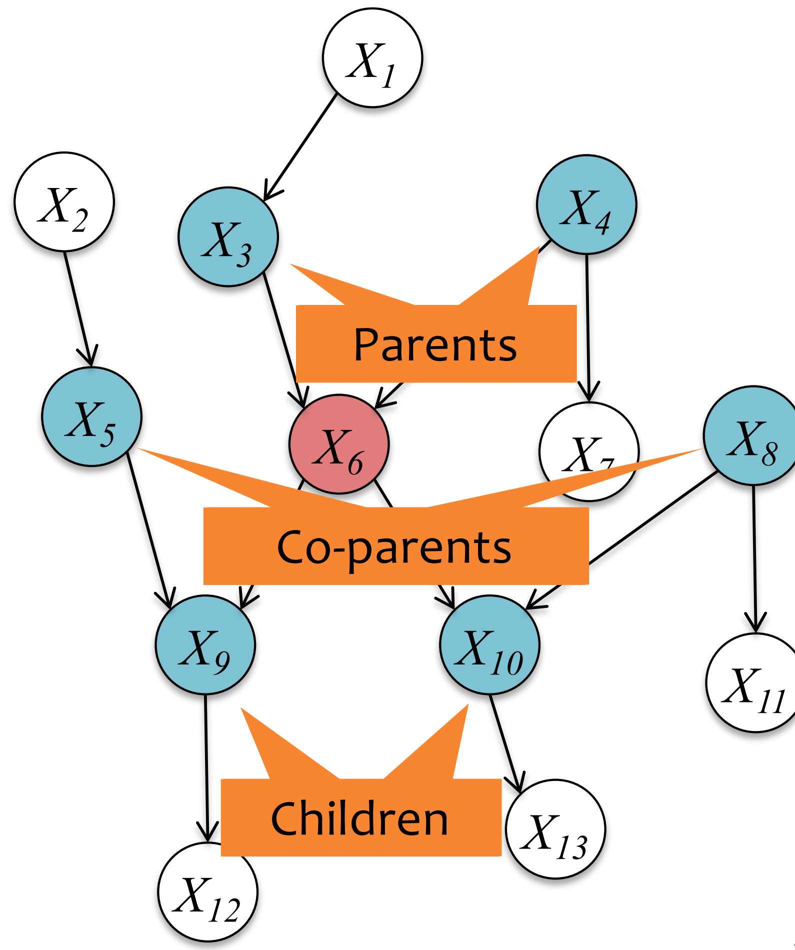
Markov boundary

Def: the **co-parents** of a node are the parents of its children

Def: the **Markov boundary** of a node is the set containing the node's parents, children, and co-parents.

Theorem: a node is **conditionally independent** of every other node in the graph given its **Markov boundary**

Example: The Markov boundary of X_6 is $\{X_3, X_4, X_5, X_8, X_9, X_{10}\}$

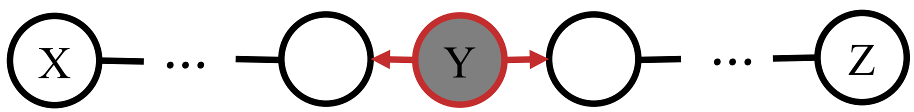


D-Separation

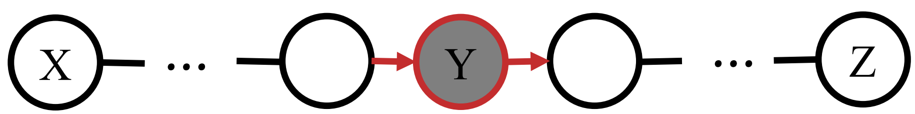
<u>Definition #1:</u>
Variables X and Z are **d-separated** given a **set** of evidence variables E (variables that are observed) iff every path from X to Z is "blocked".

A path is "blocked" whenever:

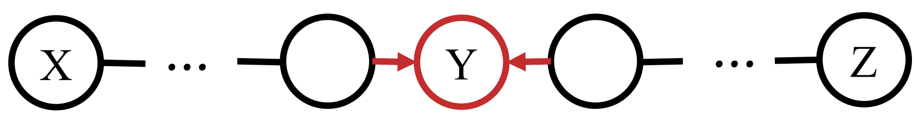
1. ∃Y on path s.t. Y ∈ E and Y is a "common parent"



2. ∃Y on path s.t. Y ∈ E and Y is in a "cascade"



3. ∃Y on path s.t. {Y, descendants(Y)} ∉ E and Y is in a "v-structure"



If variables X and Z are **d-separated** given a **set** of variables E
Then X and Z are **conditionally independent** given the **set** E

D-Separation

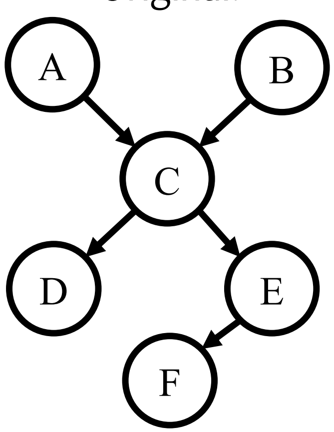
If variables X and Z are **d-separated** given a **set** of variables E
Then X and Z are **conditionally independent** given the **set** E

Definition #2:
Variables X and Z are **d-separated** given a **set** of evidence variables E iff there does **not** exist a path between X and Z in the **undirected ancestral moral** graph **with E removed**.

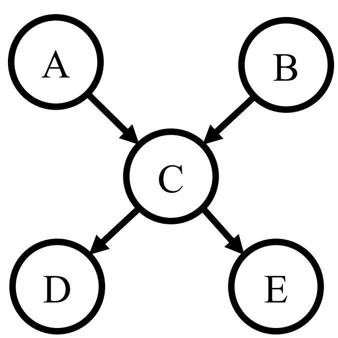
1. **Ancestral graph**: keep only X, Z, E and their ancestors
2. **Moral graph**: add undirected edge between all pairs of each node's parents
3. **Undirected graph**: convert all directed edges to undirected
4. Givens Removed: delete any nodes in E

Example Query: $A \perp\!\!\!\perp B \mid \{D, E\}$

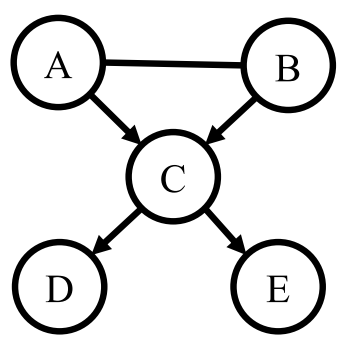
Original:



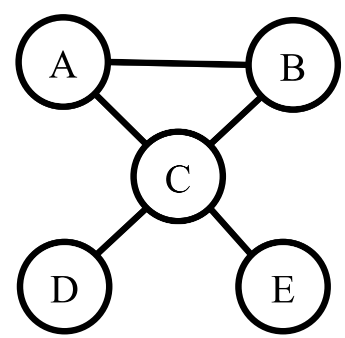
Ancestral:



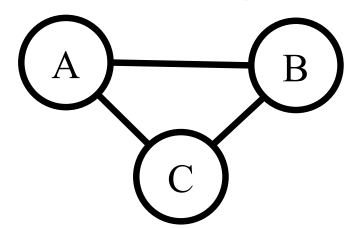
Moral:



Undirected:



Givens Removed:



⇒A and B connected
⇒ not d-separated

SUPERVISED LEARNING FOR BAYES NETS

Recipe for Closed-form MLE

1. Assume data was generated i.i.d. from some model (i.e. write the generative story)

   $x^{(i)} \sim p(x|\boldsymbol{\theta})$

2. Write log-likelihood

   $\ell(\boldsymbol{\theta}) = \log p(x^{(1)}|\boldsymbol{\theta}) + \ldots + \log p(x^{(N)}|\boldsymbol{\theta})$

3. Compute partial derivatives (i.e. gradient)

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_1 = \ldots$

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_2 = \ldots$

   …

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_M = \ldots$

4. Set derivatives to zero and solve for $\boldsymbol{\theta}$

   $\partial\ell(\boldsymbol{\theta})/\partial\theta_m = 0$ for all $m \in \{1, \ldots, M\}$

   $\boldsymbol{\theta}^{MLE}$ = solution to system of M equations and M variables

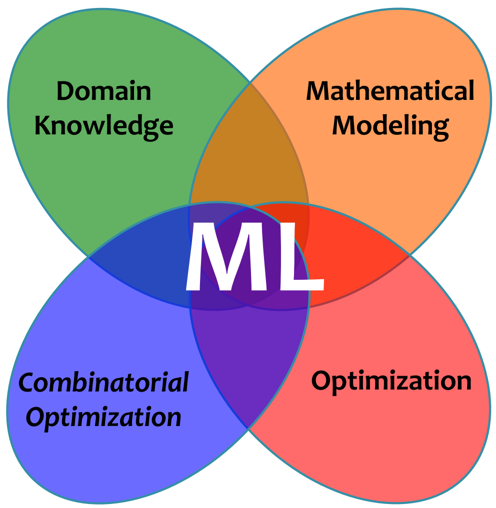
5. Compute the second derivative and check that $\ell(\boldsymbol{\theta})$ is concave down at $\boldsymbol{\theta}^{MLE}$

Machine Learning

The **data** inspires the structures we want to predict

Our **model** defines a score for each structure

It also tells us what to optimize



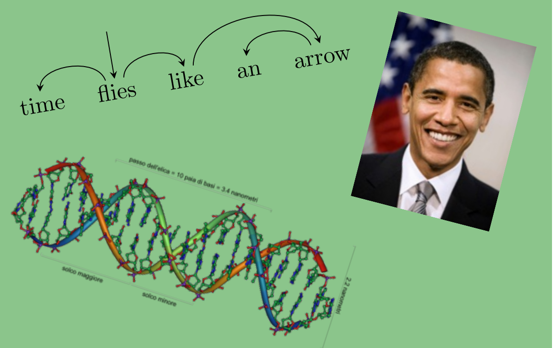
Inference finds {best structure, marginals, partition function} for a new observation

Learning tunes the parameters of the model

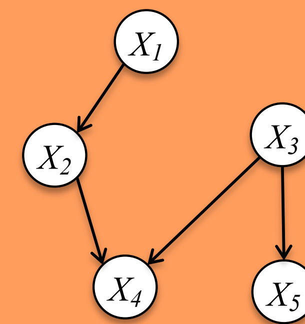
(**Inference** is usually called as a subroutine in learning)

Machine Learning

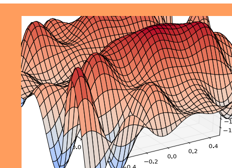
Data



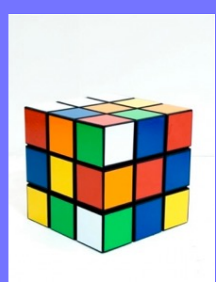
Model



Objective

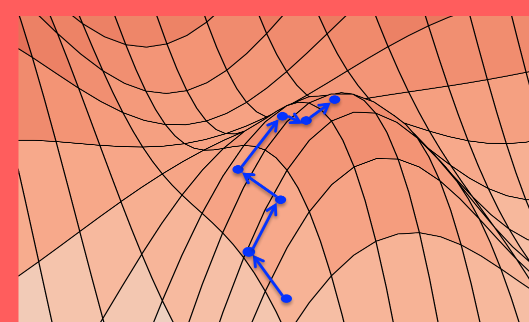


Inference

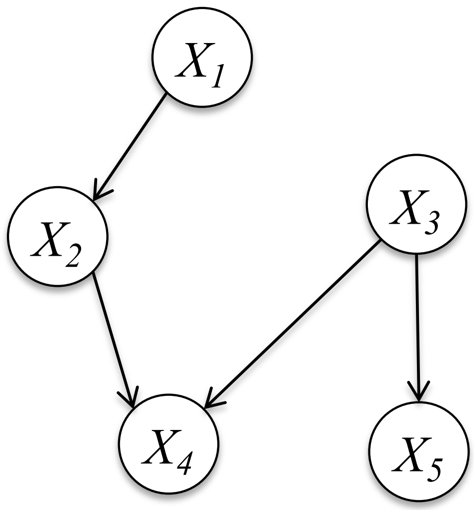


(**Inference** is usually called as a subroutine in learning)

Learning

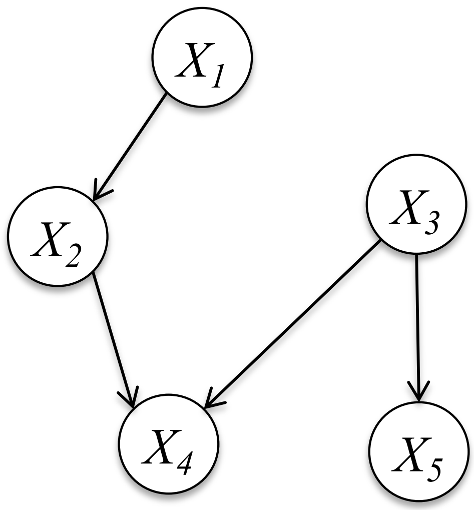


Learning Fully Observed BNs



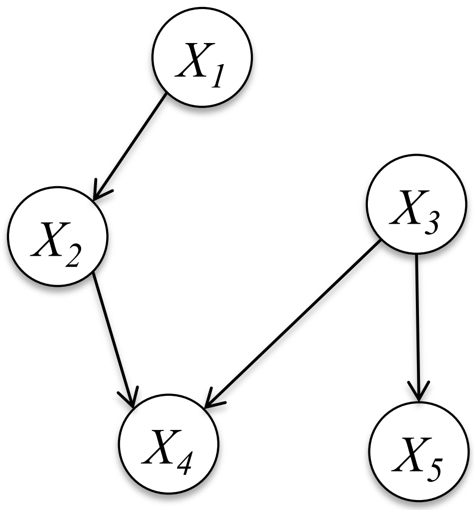
$$p(X_1, X_2, X_3, X_4, X_5) =$$
$$p(X_5|X_3)p(X_4|X_2, X_3)$$
$$p(X_3)p(X_2|X_1)p(X_1)$$

Learning Fully Observed BNs



$$p(X_1, X_2, X_3, X_4, X_5) = p(X_5|X_3)p(X_4|X_2, X_3) p(X_3)p(X_2|X_1)p(X_1)$$

Learning Fully Observed BNs



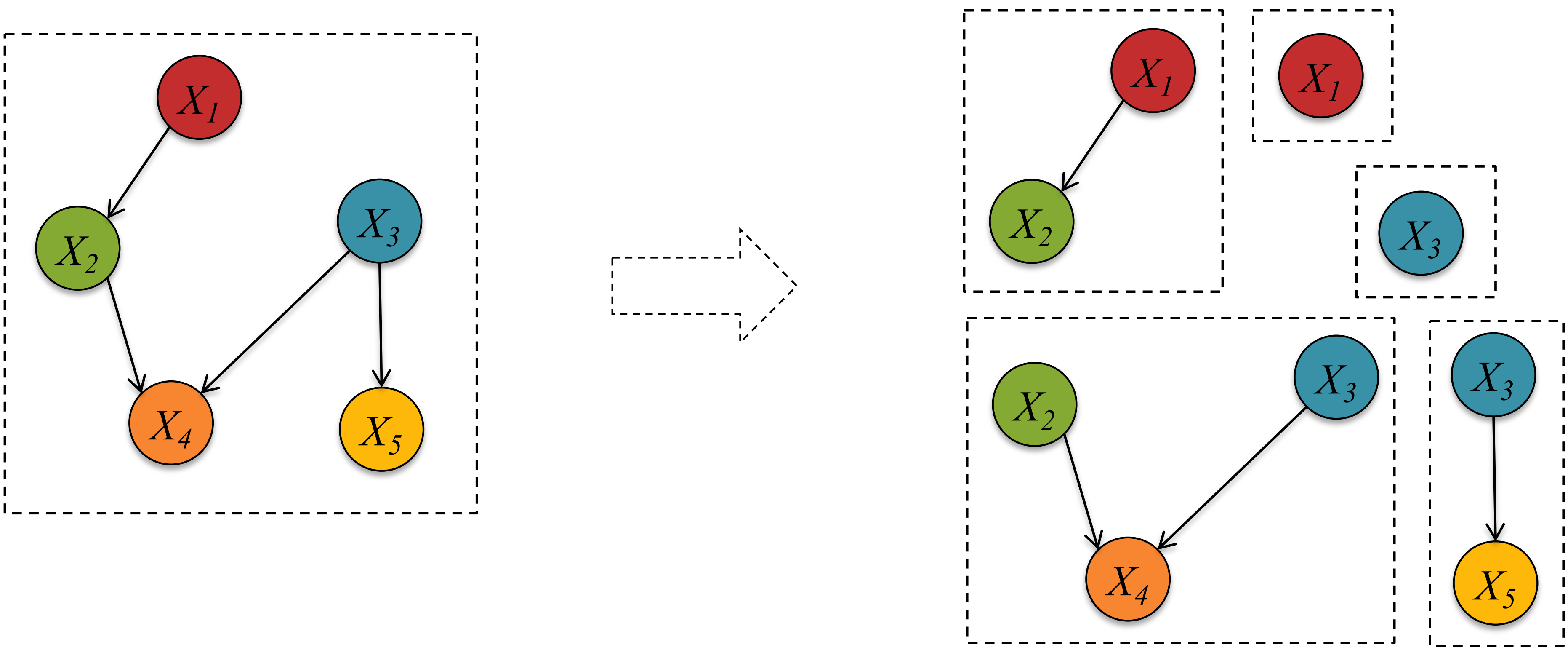
$$p(X_1, X_2, X_3, X_4, X_5) =$$
$$p(X_5|X_3)p(X_4|X_2, X_3)$$
$$p(X_3)p(X_2|X_1)p(X_1)$$

How do we learn these conditional and marginal distributions for a Bayes Net?

Learning Fully Observed BNs

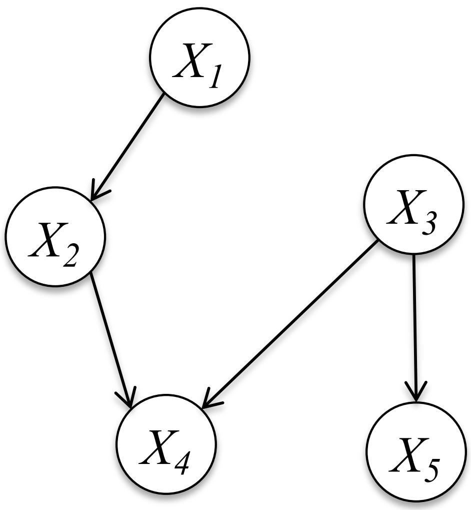
Learning this fully observed Bayesian Network is **equivalent** to learning five (small / simple) independent networks from the same data

$$
\begin{aligned}
p(X_1, X_2, X_3, X_4, X_5) = \\
p(X_5|X_3)p(X_4|X_2, X_3) \\
p(X_3)p(X_2|X_1)p(X_1)
\end{aligned}
$$



Learning Fully Observed BNs

How do we **learn** these conditional and marginal distributions for a Bayes Net?



$$\begin{aligned}\boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(X_1, X_2, X_3, X_4, X_5) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(X_5|X_3, \theta_5) + \log p(X_4|X_2, X_3, \theta_4) \\ &\qquad + \log p(X_3|\theta_3) + \log p(X_2|X_1, \theta_2) \\ &\qquad + \log p(X_1|\theta_1)\end{aligned}$$

$$\theta_1^* = \underset{\theta_1}{\operatorname{argmax}} \log p(X_1|\theta_1)$$

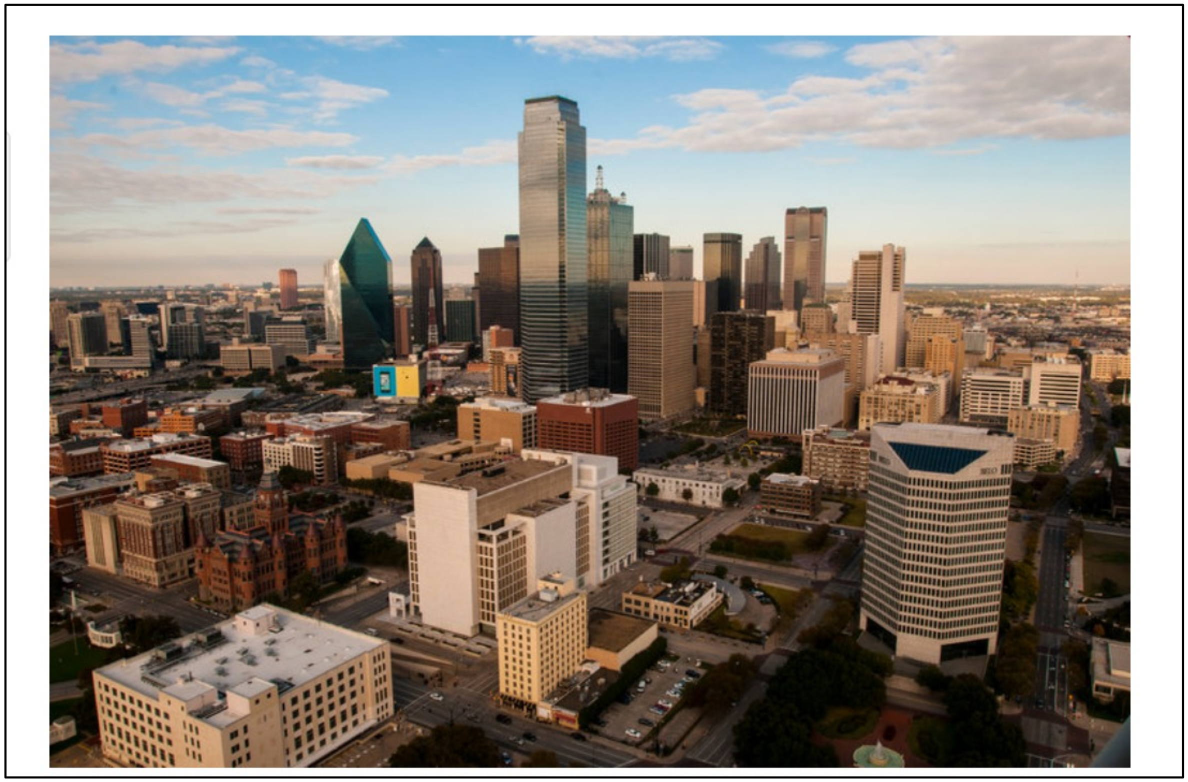
$$\theta_2^* = \underset{\theta_2}{\operatorname{argmax}} \log p(X_2|X_1, \theta_2)$$

$$\theta_3^* = \underset{\theta_3}{\operatorname{argmax}} \log p(X_3|\theta_3)$$

$$\theta_4^* = \underset{\theta_4}{\operatorname{argmax}} \log p(X_4|X_2, X_3, \theta_4)$$

$$\theta_5^* = \underset{\theta_5}{\operatorname{argmax}} \log p(X_5|X_3, \theta_5)$$

Example: Tornado Alarms



1. Imagine that you work at the 911 call center in Dallas
2. You receive six calls informing you that the Emergency Weather Sirens are going off
3. What do you conclude?

Example: Tornado Alarms

Hacking Attack Woke Up Dallas With Emergency Sirens, Officials Say

By ELI ROSENBERG and MAYA SALAM APRIL 8, 2017

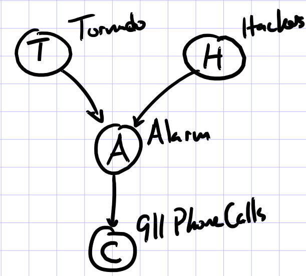
Warning sirens in Dallas, meant to alert the public to emergencies like severe weather, started sounding around 11:40 p.m. Friday, and were not shut off until 1:20 a.m. Rex C. Curry for The New York Times

1. Imagine that you work at the 911 call center in Dallas
2. You receive six calls informing you that the Emergency Weather Sirens are going off
3. What do you conclude?

Figure from https://www.nytimes.com/2017/04/08/us/dallas-emergency-sirens-hacking.html

Learning Fully Observed BNs

Ex: Tornado Alarms



$H \sim \text{Bernoulli}(\eta)$
$T \sim \text{Bernoulli}(\tau)$
$A \sim \text{Bernoulli}(\alpha_{H,T})$
$C \sim \text{Uniform}(\{1,\dots,6\}) + A * \text{Uniform}(\{1,\dots,6\})$

parameters

no parameters

integer

Dataset

i	T	H	A	C
1	0	0	0	2
2	0	0	0	6
3	0	0	0	4
⋮	1	0	0	3
	1	0	0	1
	1	0	1	10
	1	0	1	7
	0	1	0	2
	0	1	1	12
	0	1	0	5
⋮	1	1	1	10
12	1	0	0	2

What are the MLEs?

$\hat{\eta} = 1/3$

$\hat{\tau} = 1/2$

$\hat{\alpha} =$

	H=0	H=1
T=0	0	1/3
T=1	2/3	1

MLEs in Closed Form

$$\ell(\eta,\tau,\alpha) = \log \prod_{i=1}^{12} p(t^{(i)}, h^{(i)}, a^{(i)}, c^{(i)} \mid \eta, \tau, \alpha)$$

$$= \sum_{i=1}^{12} \log p(t^{(i)} \mid \tau) + \log p(h^{(i)} \mid \eta) + \log p(a^{(i)} \mid t^{(i)}, h^{(i)}, \alpha) + \log p(c^{(i)} \mid a^{(i)})$$

$$\hat{\eta}, \hat{\tau}, \hat{\alpha} = \operatorname{argmax} \ell(\eta, \tau, \alpha)$$

$$\hat{\eta} = \operatorname*{argmax}_{\eta} \sum_{i=1}^{12} \log p(h^{(i)} \mid \eta) = \#(T=1)/N$$

$$\hat{\tau} = \operatorname*{argmax}_{\tau} \sum_{i=1}^{12} \log p(t^{(i)} \mid \tau) = \#(H=1)/N$$

$$\hat{\alpha} = \operatorname*{argmax}_{\alpha} \sum_{i=1}^{12} \log p(a^{(i)} \mid t^{(i)}, h^{(i)}, \alpha)$$

$$\hat{\alpha}_{t,h} = \frac{\#(A=1, T=t, H=h)}{\#(T=t, H=h)}$$

INFERENCE FOR BAYESIAN NETWORKS

A Few Problems for Bayes Nets

Suppose we already have the parameters of a Bayesian Network…

1. How do we compute the probability of a specific assignment to the variables?
   $P(T=t, H=h, A=a, C=c)$

2. How do we draw a sample from the joint distribution?
   $t,h,a,c \sim P(T, H, A, C)$

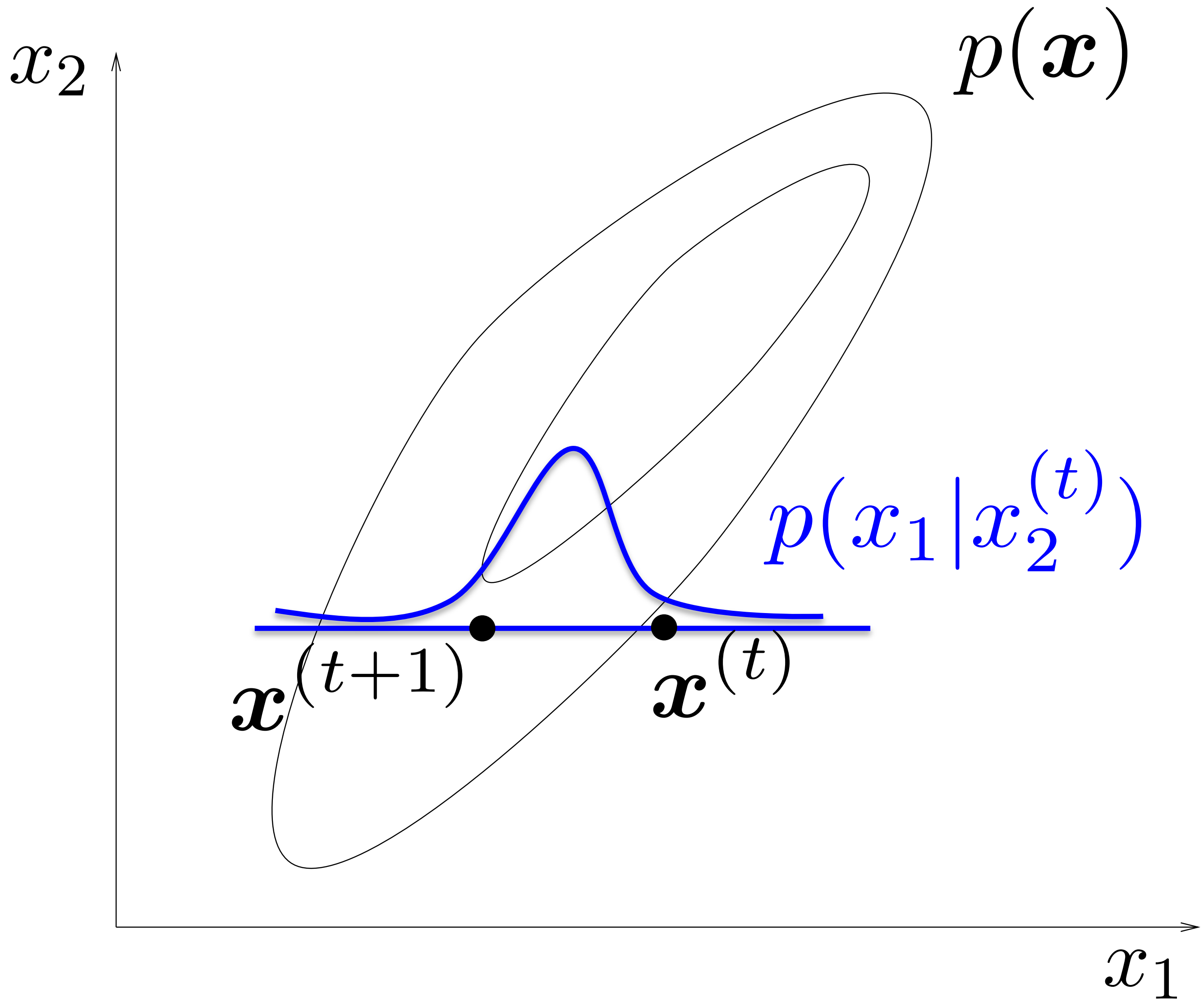
3. How do we compute marginal probabilities?
   $P(A) = \ldots$

4. How do we draw samples from a conditional distribution?
   $t,h,a \sim P(T, H, A \mid C = c)$

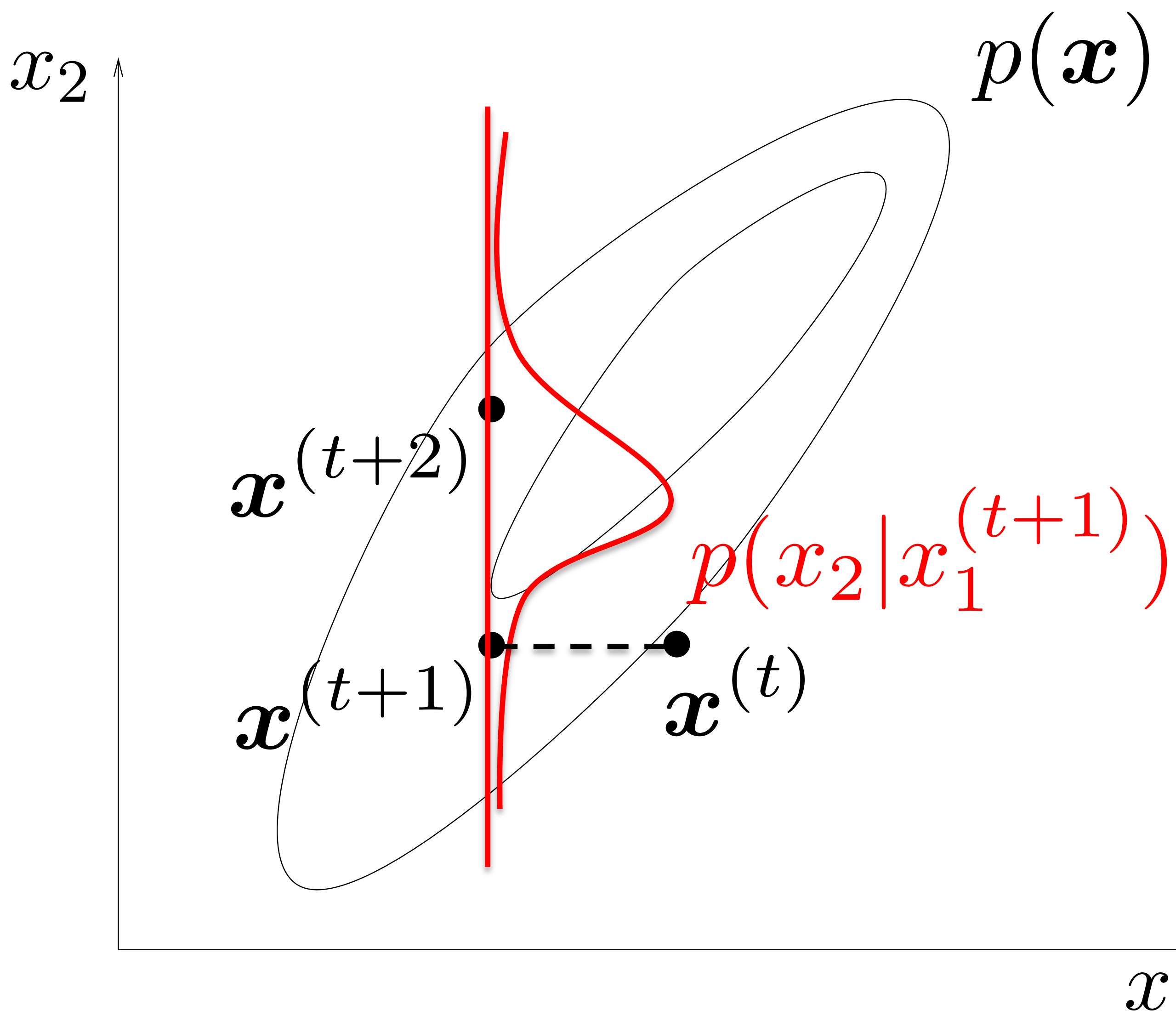
5. How do we compute conditional marginal probabilities?
   $P(H \mid C = c) = \ldots$

Can we use samples ?

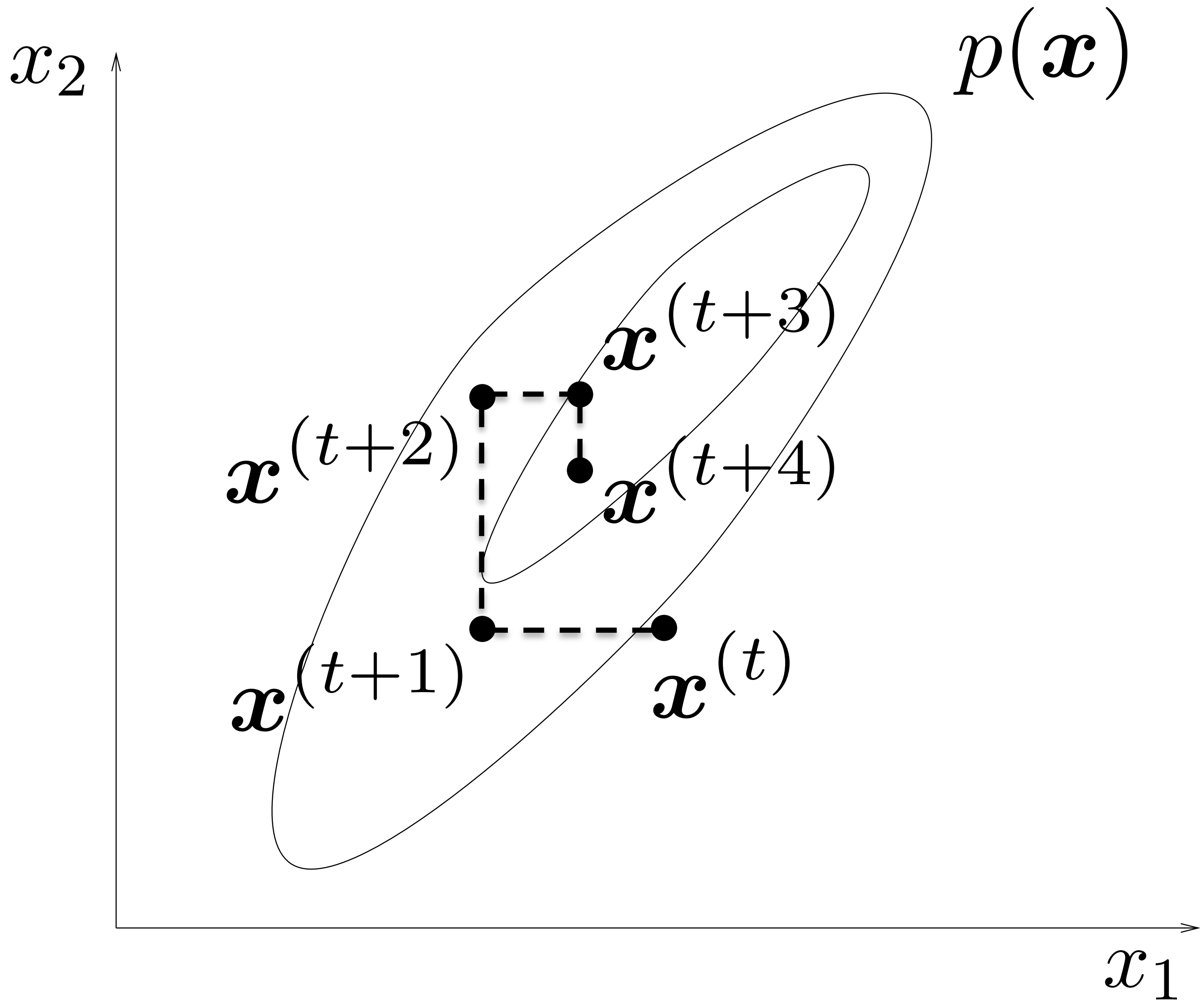
Gibbs Sampling



Gibbs Sampling



Gibbs Sampling



Gibbs Sampling

Question:
How do we draw samples from a conditional distribution?
$y_1, y_2, \ldots, y_J \sim p(y_1, y_2, \ldots, y_J \mid x_1, x_2, \ldots, x_J)$

(Approximate) Solution:

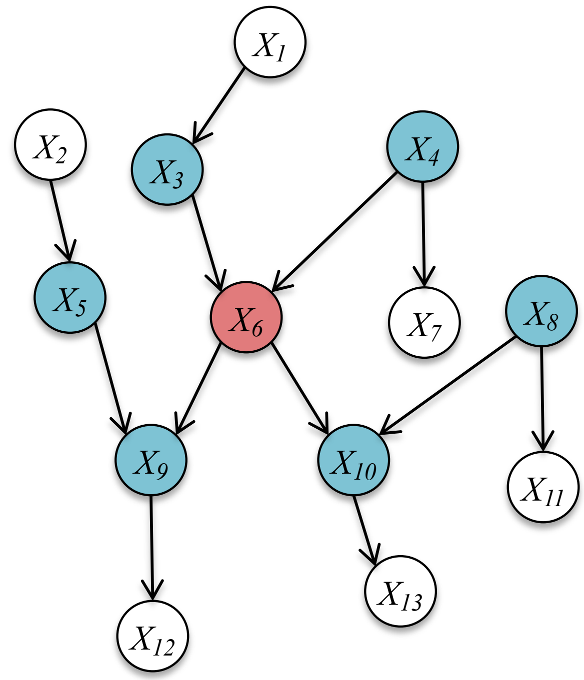
- Initialize $y_1^{(0)}, y_2^{(0)}, \ldots, y_J^{(0)}$ to arbitrary values
- For $t = 1, 2, \ldots$:
  - $y_1^{(t+1)} \sim p(y_1 \mid y_2^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - $y_2^{(t+1)} \sim p(y_2 \mid y_1^{(t+1)}, y_3^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - $y_3^{(t+1)} \sim p(y_3 \mid y_1^{(t+1)}, y_2^{(t+1)}, y_4^{(t)}, \ldots, y_J^{(t)}, x_1, x_2, \ldots, x_J)$
  - …
  - $y_J^{(t+1)} \sim p(y_J \mid y_1^{(t+1)}, y_2^{(t+1)}, \ldots, y_{J-1}^{(t+1)}, x_1, x_2, \ldots, x_J)$

Properties:

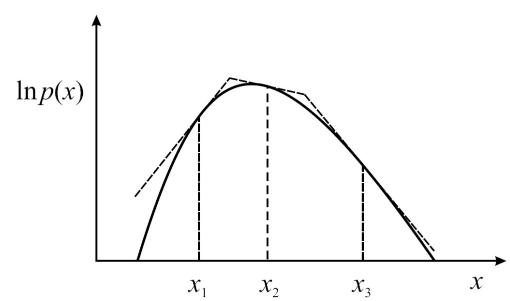
- This will eventually yield samples from
  $p(y_1, y_2, \ldots, y_J \mid x_1, x_2, \ldots, x_J)$
- But it might take a long time -- just like other Markov Chain Monte Carlo methods

Gibbs Sampling

Full conditionals only need to condition on the **Markov boundary**



- Must be "easy" to sample from conditionals
- Many conditionals are log-concave and are amenable to adaptive rejection sampling



Learning Objectives

Bayesian Networks

You should be able to…

1. Identify the conditional independence assumptions given by a generative story or a specification of a joint distribution
2. Draw a Bayesian network given a set of conditional independence assumptions
3. Define the joint distribution specified by a Bayesian network
4. User domain knowledge to construct a (simple) Bayesian network for a real-world modeling problem
5. Depict familiar models as Bayesian networks
6. Use d-separation to prove the existence of conditional indenpendencies in a Bayesian network
7. Employ a Markov boundary to identify conditional independence assumptions of a graphical model
8. Develop a supervised learning algorithm for a Bayesian network
9. Use samples from a joint distribution to compute marginal probabilities
10. Sample from the joint distribution specified by a generative story
11. Implement a Gibbs sampler for a Bayesian network

LEARNING PARADIGMS

Learning Paradigms

Paradigm	**Data**	
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$	$\mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$	
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$	
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$	
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector	

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$

Learning Paradigms

Paradigm	Data
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N}$ $\quad\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ $\quad\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ and can query $y^{(i)} = c^*(\cdot)$ at a cost

Learning Paradigms

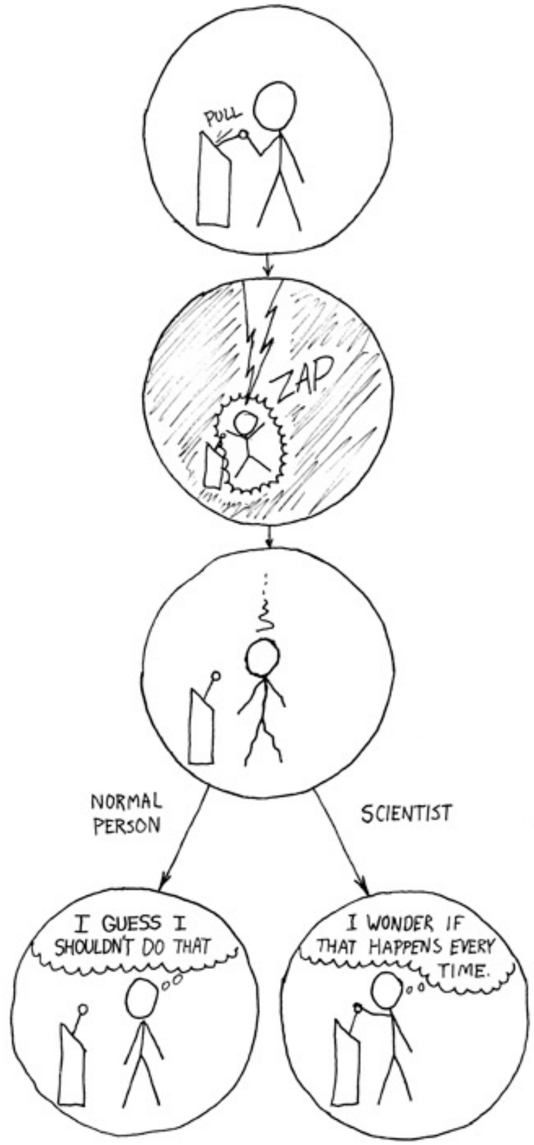
Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N \qquad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N \qquad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots\}$

Learning Paradigms

Paradigm	**Data**
Supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$ and $y = c^*(\cdot)$
$\hookrightarrow$ Regression	$y^{(i)} \in \mathbb{R}$
$\hookrightarrow$ Classification	$y^{(i)} \in \{1, \ldots, K\}$
$\hookrightarrow$ Binary classification	$y^{(i)} \in \{+1, -1\}$
$\hookrightarrow$ Structured Prediction	$\mathbf{y}^{(i)}$ is a vector
Unsupervised	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ $\quad \mathbf{x} \sim p^*(\cdot)$
Semi-supervised	$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N_1} \cup \{\mathbf{x}^{(j)}\}_{j=1}^{N_2}$
Online	$\mathcal{D} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), (\mathbf{x}^{(3)}, y^{(3)}), \ldots\}$
Active Learning	$\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$ and can query $y^{(i)} = c^*(\cdot)$ at a cost
Imitation Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}), (s^{(2)}, a^{(2)}), \ldots\}$
Reinforcement Learning	$\mathcal{D} = \{(s^{(1)}, a^{(1)}, r^{(1)}), (s^{(2)}, a^{(2)}, r^{(2)}), \ldots\}$

REINFORCEMENT LEARNING

Reinforcement Learning



Source: https://www.xkcd.com/242/

RL: Examples

Source: https://www.cnet.com/news/boston-dynamics-robot-dog-spot-finally-goes-on-sale-for-74500/

Source: https://twitter.com/alphagomovie

Source: https://techobserver.net/2019/06/argo-ai-self-driving-car-research-center/

Source: https://www.wired.com/2012/02/high-speed-trading/

AlphaGo

Source: https://www.youtube.com/watch?v=WXuK6gekU1Y&ab_channel=DeepMind

History of Reinforcement Learning

- Roots in the psychology of animal learning (Thorndike,1911).
- Another independent thread was the problem of optimal control, and its solution using dynamic programming (Bellman, 1957).
- Idea of temporal difference learning (on-line method), e.g., playing board games (Samuel, 1959).
- A major breakthrough was the discovery of Q-learning (Watkins, 1989).

What is special about RL?

- RL is learning how to map states to actions, so as to maximize a numerical reward over time.
- Unlike other forms of learning, it is a multistage decision-making process (often Markovian).
- An RL agent must learn by trial-and-error. (Not entirely supervised, but interactive)
- Actions may affect not only the immediate reward but also subsequent rewards (Delayed effect).

Elements of RL

- A policy
  - A map from state space to action space.
  - May be stochastic.
- A reward function
  - It maps each state (or, state-action pair) to a real number, called reward.
- A value function
  - Value of a state (or, state-action pair) is the total expected reward, starting from that state (or, state-action pair).

Example: Robot in a Room

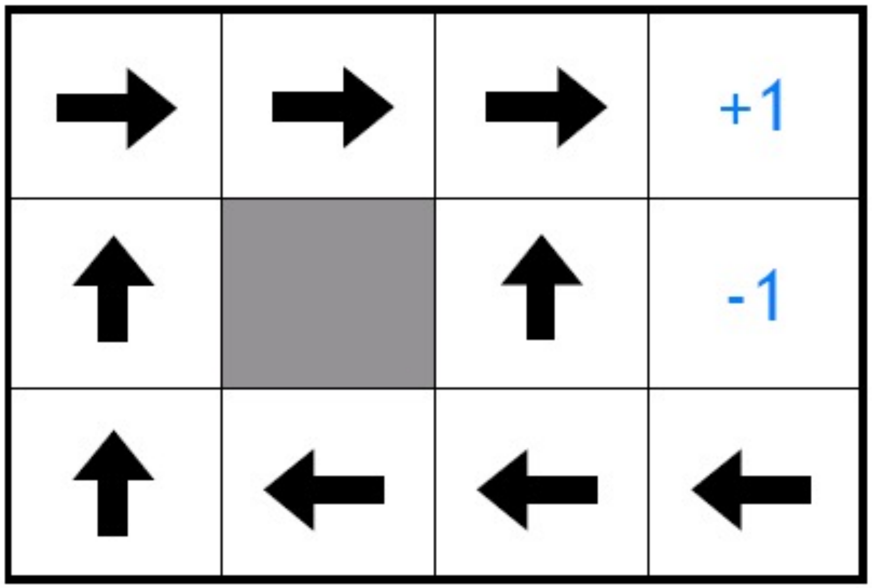
actions: UP, DOWN, LEFT, RIGHT

UP

80% move UP
10% move LEFT
10% move RIGHT

- reward +1 at [4,3], -1 at [4,2]
- reward -0.04 for each step

Example: Robot in a Room

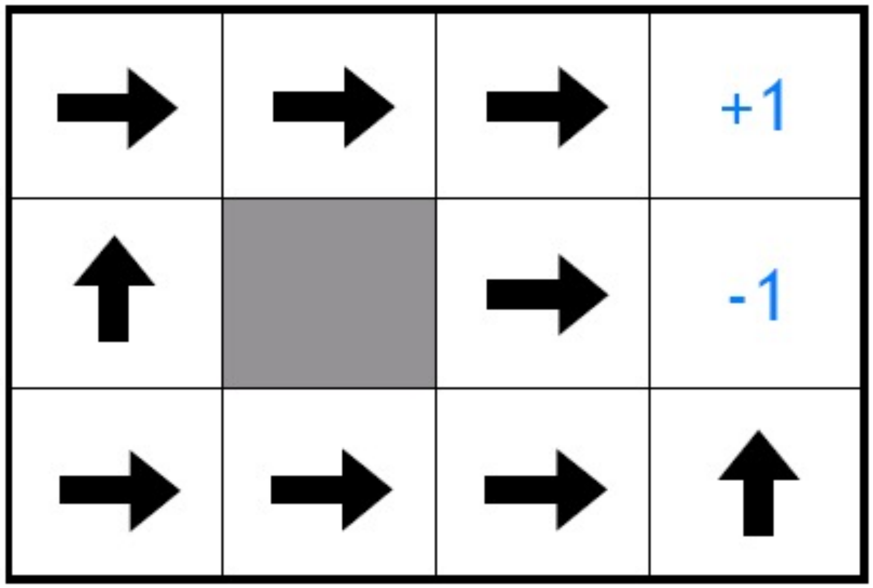


Question:
Is this policy optimal: yes or no? Briefly justify your answer.

Answer: (*Hint*: both yes and no are acceptable answers, I'm interested in your justification.)

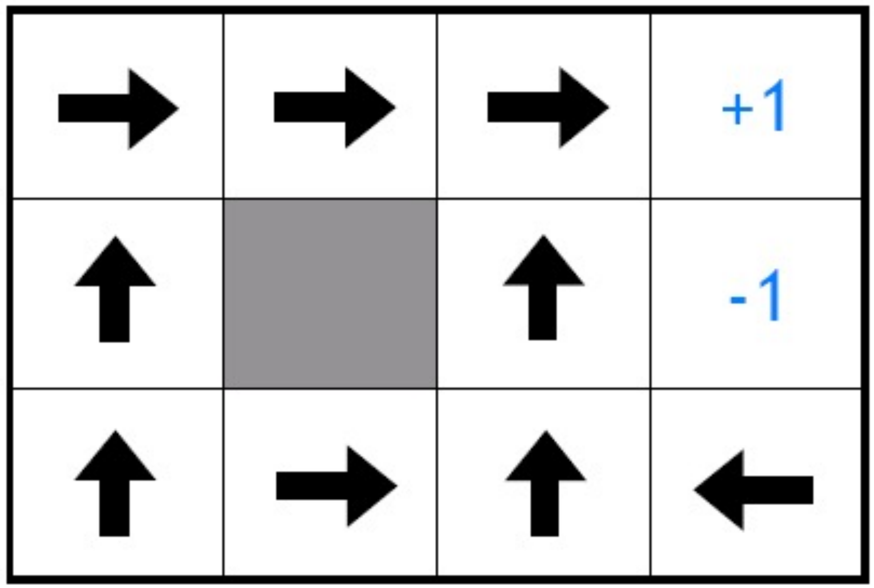
Example: Robot in a Room

- Reward for each step -2



Example: Robot in a Room

- Reward for each step: -0.1



The Precise Goal

- To find a policy that maximizes the Value function.
  - transitions and rewards usually not available
- There are different approaches to achieve this goal in various situations.
- Value iteration and Policy iteration are two more classic approaches to this problem. But essentially both are dynamic programming.
- Q-learning is a more recent approaches to this problem. Essentially it is a temporal-difference method.