CPSC 340: Machine Learning and Data Mining

Markov Chains

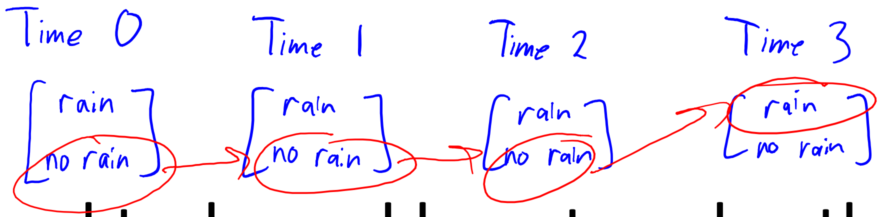
Fall 2015

Admin

- Assignment 6 due Friday.
  - Error in Q1.1 fixed: should be able to get to logistic loss.
- We will have office hours as usual next week.
- Final exam details:
  - December 15: 8:30-11 (WESB 100).
  - 4 pages of cheat sheet allowed.
  - 9 questions.
  - Practice questions and list of topics posted.

Last Time: Markov Chains

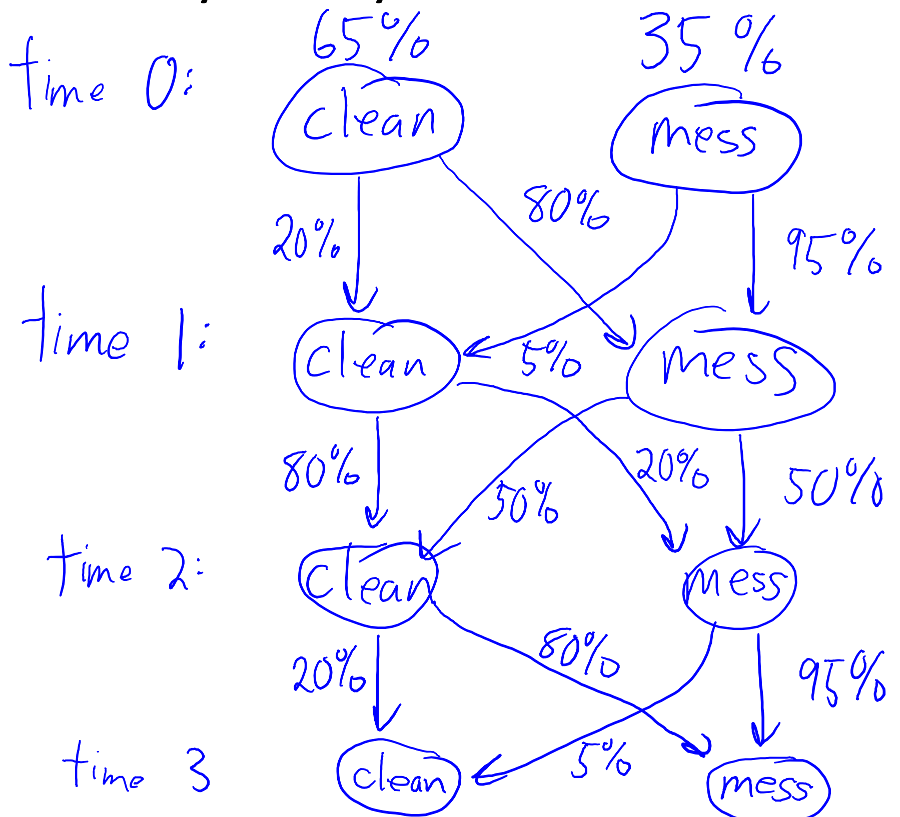
- Markov chains are common way to define probability of sequence.



- We discussed several tasks and how to solve them:
  1. Sampling: generate sequence following probability distribution.
     - Generate x_0 from p(x0), then generate x_t conditioned on x_{t-1} using $p(x_t \mid x_{t-1})$.
  2. Learning: estimate parameters given examples.
     - Count number of times we go from x_{t-1} to x_t in data, divided by number of times in x_{t-1}.
  3. Inference: computing probability of being in state ‘s’ at time ‘t’.
     - Matrix multiplication of $p(x_{t-1})$ and $p(x_t \mid x_{t-1})$ up to time ‘t’.
  4. Stationary distribution is steady-state after running for a long time.
     - Unique if probabilities positives, and obtained by normalized first “row” eigenvector.
  5. Decoding: compute most likely sequence of states.
     - Dynamic programming (“Viterbi decoding”).

Sequence of Most Probable ≠ Most Probable Sequence

- 2 roommates alternate cleaning duties for 4 days:
  - Roommate A cleans on days 0 and 2, roommate B cleans on days 1 and 3.
  - Roommate A prefers 'clean', but gets discourage if roommate B didn't clean.
  - Roommate B doesn't mind 'mess', especially if it's already messy.
- Assume the following probabilities:
  - $p(x_0 = \text{'clean'}) = 0.65$.
  - $p(x_1 = \text{'clean'} \mid x_0 = \text{'clean'}) = 0.20$.
  - $p(x_1 = \text{'clean'} \mid x_0 = \text{'mess'}) = 0.05$.
  - $p(x_2 = \text{'clean'} \mid x_1 = \text{'clean'}) = 0.80$.
  - $p(x_2 = \text{'clean'} \mid x_1 = \text{'mess'}) = 0.50$.
  - $p(x_3 = \text{'clean'} \mid x_2 = \text{'clean'}) = 0.20$.
  - $p(x_3 = \text{'clean'} \mid x_2 = \text{'mess'}) = 0.05$.



Sequence of Most Probable ≠ Most Probable Sequence

- Inference gives us probability of each state at each time.

$$\begin{array}{cc} \text{'mess'} & \text{'clean'} \end{array}$$

$$\mu_0 = \begin{bmatrix} 0.35 & 0.65 \end{bmatrix}$$

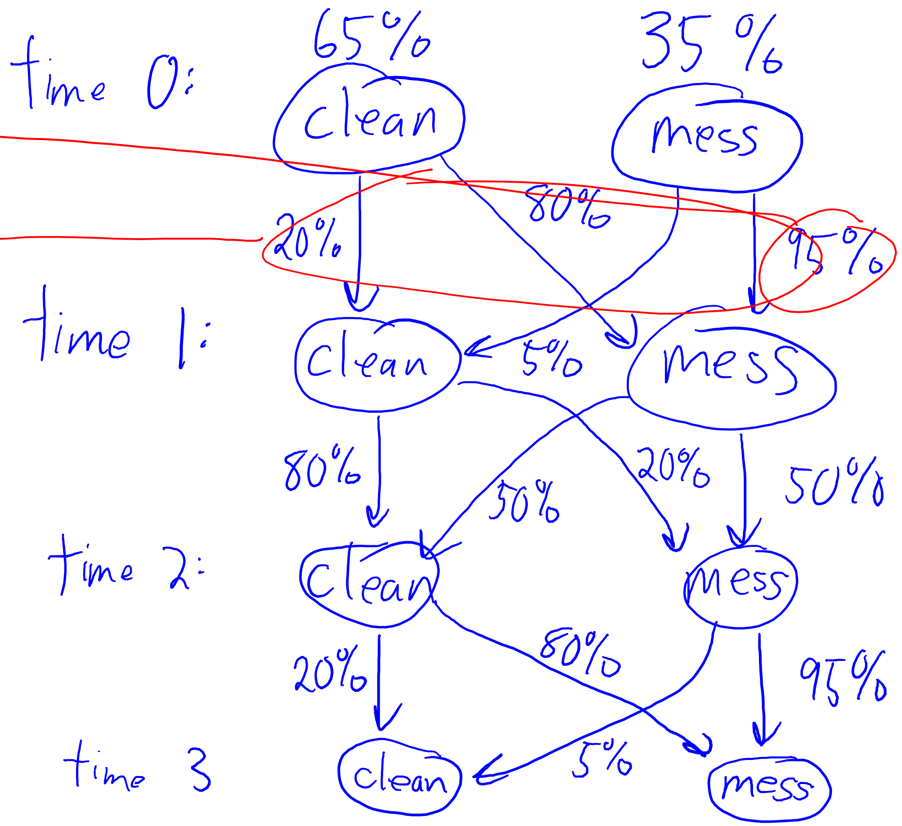
$$\mu_1 = \mu_0 P_1 = \begin{bmatrix} 0.35 & 0.65 \end{bmatrix} \begin{bmatrix} 0.95 & 0.05 \\ 0.80 & 0.20 \end{bmatrix}$$

$$= \begin{bmatrix} 0.8525 & 0.1475 \end{bmatrix}$$

$$\mu_2 = \underbrace{\mu_0 P_1}_{\mu_1} P_2 = \begin{bmatrix} 0.8525 & 0.1475 \end{bmatrix} \begin{bmatrix} 0.5 & 0.5 \\ 0.2 & 0.8 \end{bmatrix}$$

$$= \begin{bmatrix} 0.4557 & 0.5443 \end{bmatrix}$$

$$\mu_4 = \mu_0 P_1 P_2 P_3 = \mu_2 P_3 = \begin{bmatrix} 0.8684 & 0.1316 \end{bmatrix}$$



- Most likely at each time: ‘0:clean’, ‘1:mess’, ‘2:clean’, ‘3:mess’.

Sequence of Most Probable ≠ Most Probable Sequence

- Probability of sequence of most probable:

$$p(x_3='\text{mess}', x_2='\text{clean}', x_1='\text{mess}', x_0='\text{clean}') = \underbrace{p(x_3='\text{mess}' \mid x_2='\text{clean}')\,p(x_2='\text{clean}' \mid x_1='\text{mess}')\,p(x_1='\text{mess}' \mid x_0='\text{clean}')\,p(x_0='\text{clean}')}_{\text{Note: variables are not independent.}}$$

$$= (0.80)(0.50)(0.80)(0.65) = 0.2080.$$

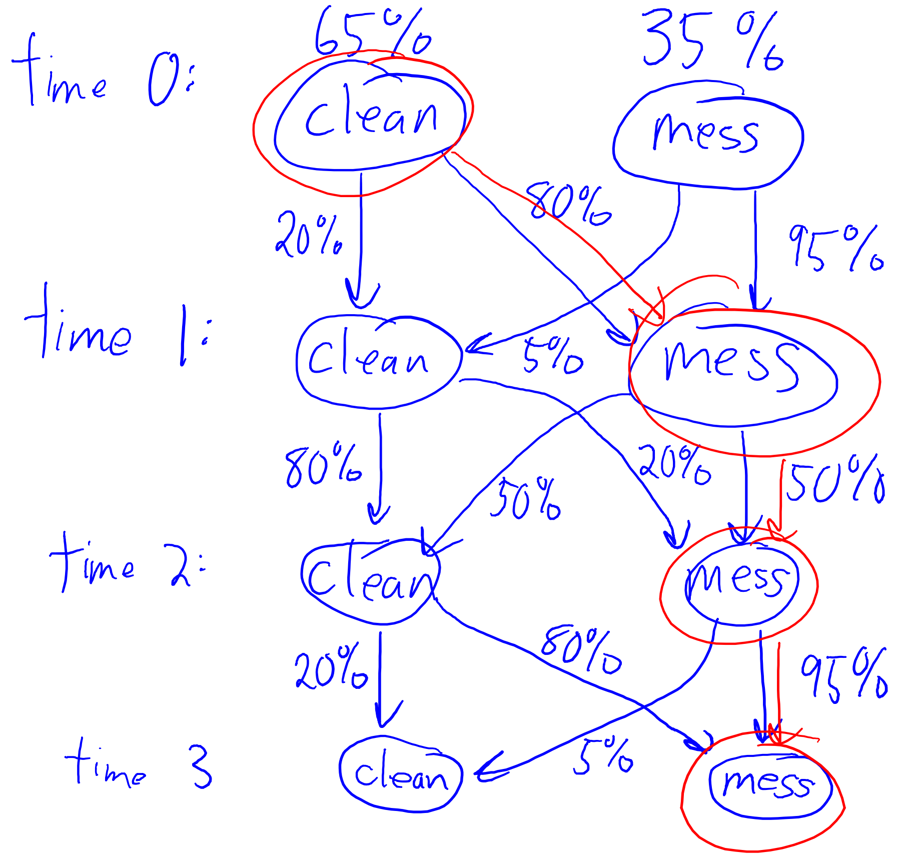
- Ignores probability of states co-occurring due to dependence.
  - Sequence of most probable states only happens 21% of the time.

Sequence of Most Probable ≠ Most Probable Sequence

- Decoding gives most probable sequence:
  - '0:clean', '1:mess', '2:mess', '3:mess'.

$$p(x_3='\text{mess}', x_2='\text{mess}', x_1='\text{mess}', x_0='\text{clean}')$$
$$=(0.95)(0.50)(0.95)(0.65)$$
$$=0.2933$$

  - Happens 29% of the time.
- Why the switch on day 2?
  - Many possible sequences of states.
  - Probability over all sequences that have '2:clean' higher than '2:mess'.
  - But no individual sequence has higher probability than decoding.



Should we use decoding or inference?

- Suppose someone asks us to predict a set of variables (x_1,x_2,x_3,x_4).

- Previously, we've only worried about prediction one variable (y):
  - No distinction between decoding/inference.
- Should you use decoding or inference?
  - If payoff is based on number of variables right: use inference.
  - If you get paid for getting the whole sequence right: use decoding.

http://collegebasketball.nbcsports.com/2015/03/15/2015-ncaa-tournament-printable-bracket/

Generalizations of Basic Markov Chain Model

- Standard Markov chain model is very limited.
- A variety of interesting extensions exist:
  - Multi-variable Markov chains:
    - x_t is vector ('rain','hot') instead of scalar, cost is exponential in number of 'variables'.
  - Higher-order Markov chains:
    - x_t depends on x_{t-1} *and* x_{t-2}, cost is exponential in length of 'history'.
  - Hidden Markov models: (Kalman filters)
    - We observe a measurement based on x_t but don't observe x_t directly.
    - E.g., tracking a player/plane/missile based on video/GPS/radar.
  - Conditional Markov models:
    - Supervised learning where we have Markov dependency in labels.
  - Belief networks.

Belief Networks

- We have a dataset with binary features:

$$X = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

- We want to model $p(x_i)$, probability of seeing a binary vector.
  - Why? Outlier detection, filling in missing values, scientific discovery, etc.
- We have seen two ways to do this:
  - Independent distribution (used by naïve Bayes): $p(x_i) = \prod_{j=1}^{d} p(x_{ij})$
  - Markov chains: $p(x_i) = p(x_{i1}) \prod_{j=2}^{d} p(x_{ij} \mid x_{ij-1})$
  - Today: generalization called belief networks.

Belief Networks

- Weird notation alert: we'll ignore 'i':
  - 'x' will be what we would normally call 'x_i'.
  - 'x_j' will be what we would normally call 'X_{ij}'.
- General representation of p(x) using product rule:

$$p(x) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots\cdots p(x_d \mid x_1, x_2, \ldots, x_{d-1})$$
$$= \prod_{j=1}^{d} p(x_j \mid x_{1:(j-1)})$$

- Problem: this has too many parameters: last term has 2^d values.]
- Solution 1: 'parsimonious' parameterization: $p(x_j) = \frac{1}{1+\exp(-x_j w_j^T x_{1:(j-1)})}$
- Solution 2: conditional independence.

Belief Networks

- Recall definition of conditional independence:

$$x_1 \perp x_3 \mid x_2 \iff p(x_3 \mid x_1, x_2) = p(x_3 \mid x_2)$$

- Naïve Bayes:

$$x_j \perp x_{1:(j-1)} \mid y \iff p(x_j \mid x_{1:(j-1)}, y) = p(x_j \mid y)$$

- Markov chain:

$$x_j \perp x_{1:(j-2)} \mid x_{j-1} \iff p(x_j \mid x_{1:(j-1)}) = p(x_j \mid x_{j-1})$$

- Belief networks:

$$x_j \perp x_{1:(j-1)\setminus\pi(j)} \mid x_{\pi(j)} \iff p(x_j \mid x_{1:(j-1)}) = p(x_j \mid \underbrace{x_{\pi(j)}}_{\text{"parents"}})$$

Belief Networks

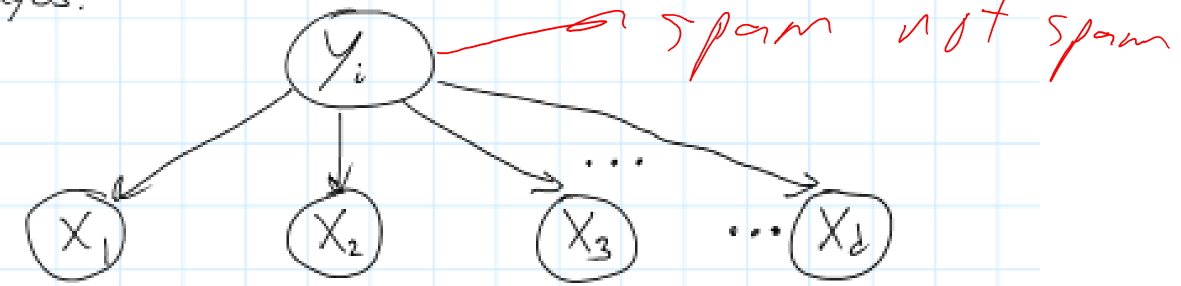
- Belief networks assume joint distribution factorized as:

$$p(x) = \prod_{j=1}^{d} p(x_j \mid x_{\pi(j)})$$

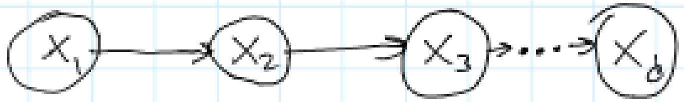
"parents" of variable j!

- Based on factorization we define a graph:
  - One vertex for each variable x_j.
  - We have an edge if 'i' is a parent of 'j'.
- Importance of graph:
  - Visual representation of assumptions.
  - Graph structure lets us test other conditional independencies.
  - Computational implications of graph structure (later in lecture).
- Also known as "Bayesian networks", "Causal networks", or "Directed acyclic graphical (DAG) models".

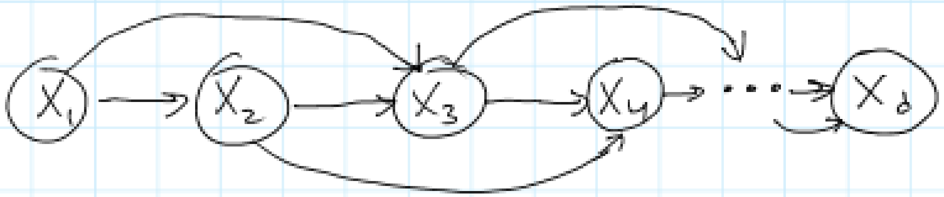
Naive Bayes:



Markov Chain:

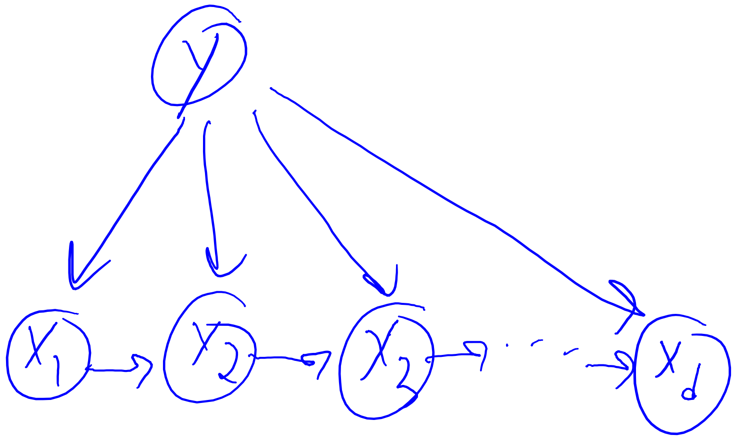


2nd Order Markov Chain:

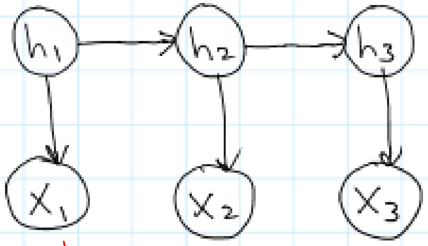


$$p(x_j \mid x_{j-1}, x_{j-2})$$

"Less-naive" Bayes:



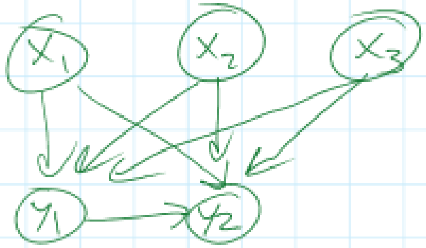
Hidden Markov Model: (Kalman filtering)



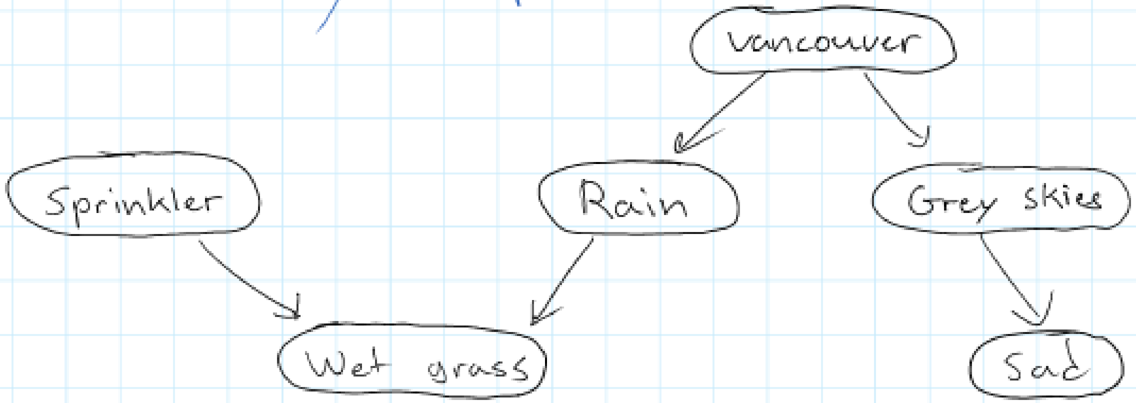
hidden (where submarine is)

Sonar signal at time "i"

Conditional DAG:



Conditionality Example



$x_j \to x_{j+1}$

Conditional Independence Properties

- We can use the graph to check whether or not

$$X_A \perp X_B \mid X_E$$

follows from conditional independence assumptions.

Conditional Independence Properties

- We can use the graph to check whether or not

$$X_A \perp X_B \mid X_E$$

follows from conditional independence assumptions.

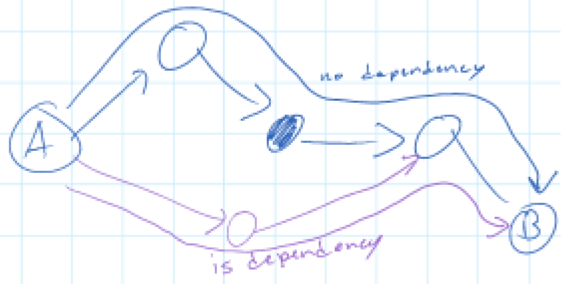
$X_A \perp X_B \mid X_E \iff$ A and B are "d-separated" given E

Conditional Independence Properties

- We can use the graph to check whether or not

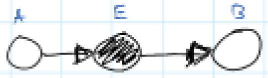
$$X_A \perp X_B \mid X_E$$

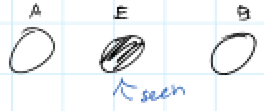
follows from conditional independence assumptions.



$X_A \perp X_B \mid X_E \iff$ A and B are "d-separated" given E

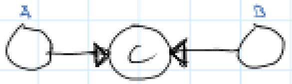
A and B are d-separated if for all paths 'p' between A and B, at least one of the following holds:

1. P includes a "chain": 



2. P includes a "fork": 

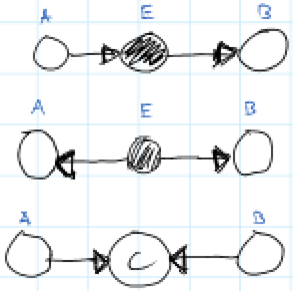
note: no E here

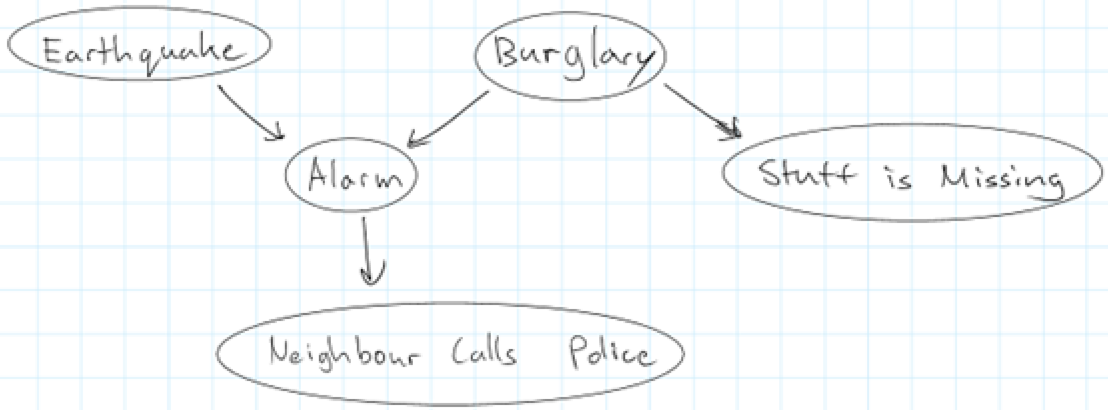
3. P contains a "collider": 

, where C and all its descendents are unobserved

D-separated if for all paths from A to B at least one of these "blocks" the path.

1. P includes a "chain":
2. P includes a "fork":
3. P contains a "collider":



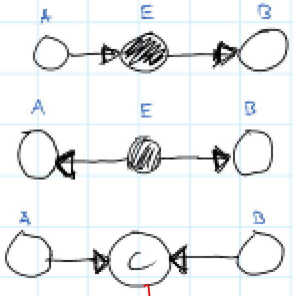


D-separated if for all paths from A to B at least one of these "blocks" the path.

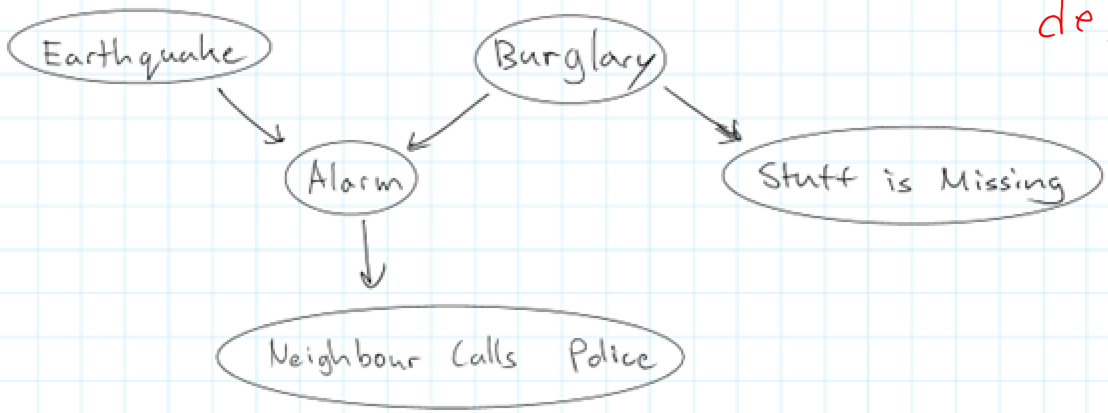
1. P includes a "chain":

2. P includes a "fork":

3. P contains a "collider":



no visible descendants

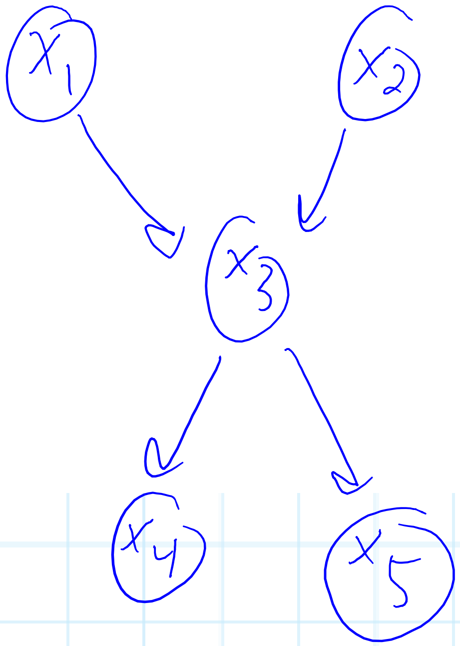


Testing conditional independence:

Earthquake $\not\perp$ call		"dependent"
Earthquake $\perp$ call | Alarm		"independent"
Alarm $\not\perp$ Stuff Missing		
Alarm $\perp$ Stuff Missing | Buglary		
Earthquake $\perp$ Buglary		
Earthquake $\not\perp$ Buglary | Alarm		"explaining away"
Earthquake $\not\perp$ Buglary | Call		
Buglary	Call	
Call	stuff is missing	

Sampling:

- similar to Markov chains



- Sample x_1
- Sample x_2
- Sample $x_3 | x_1, x_2$
- sample $x_4 | x_3$
- sample $x_5 | x_3$

Learning

parents

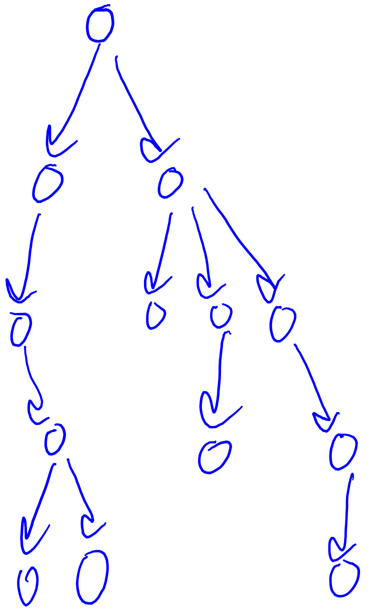
- fit each $p(x_i | x_{\pi(i)})$ independently
- "inference" computing $p(x)$
  computing $p(x_i | x_{\pi(i)})$ } easy
  similar to matrix multiplication
  computing $p(x_i | x_{i+1})$ } Hard in general
- decoding: hard in general

Decoding and "conditional" inference are easy for naive Bayes and Markov chains.

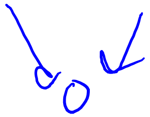
When is decoding and "conditional" inference hard?

Easy-case: "singly-connected" tree

- solved via dynamic programming.
- called "belief propagation".

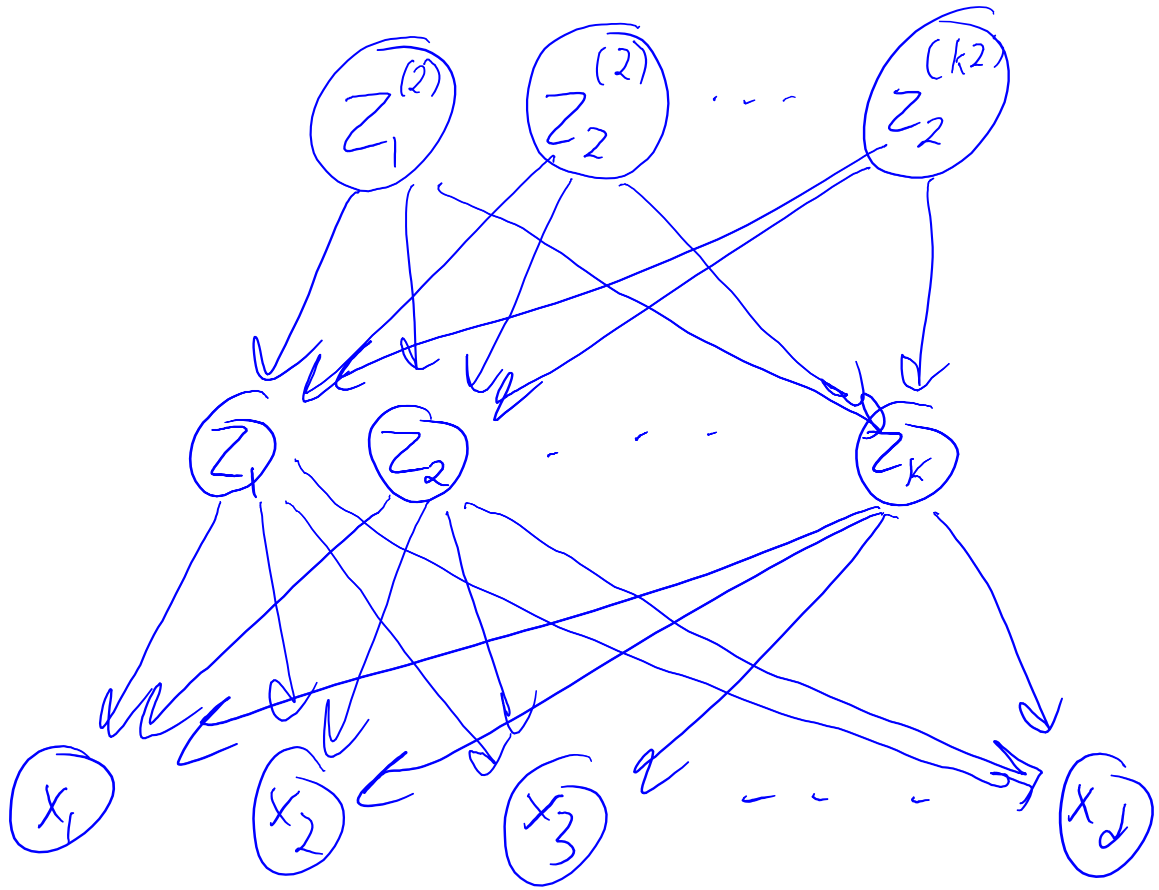


No "v"-structures:



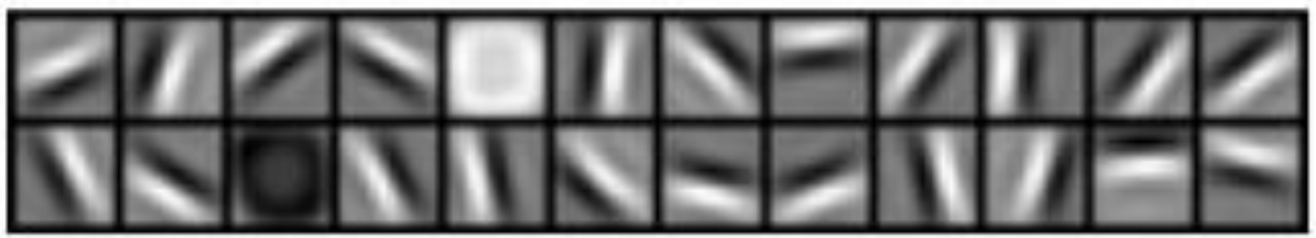
Otherwise, decoding and conditional inference may be hard.

Deep Belief Networks

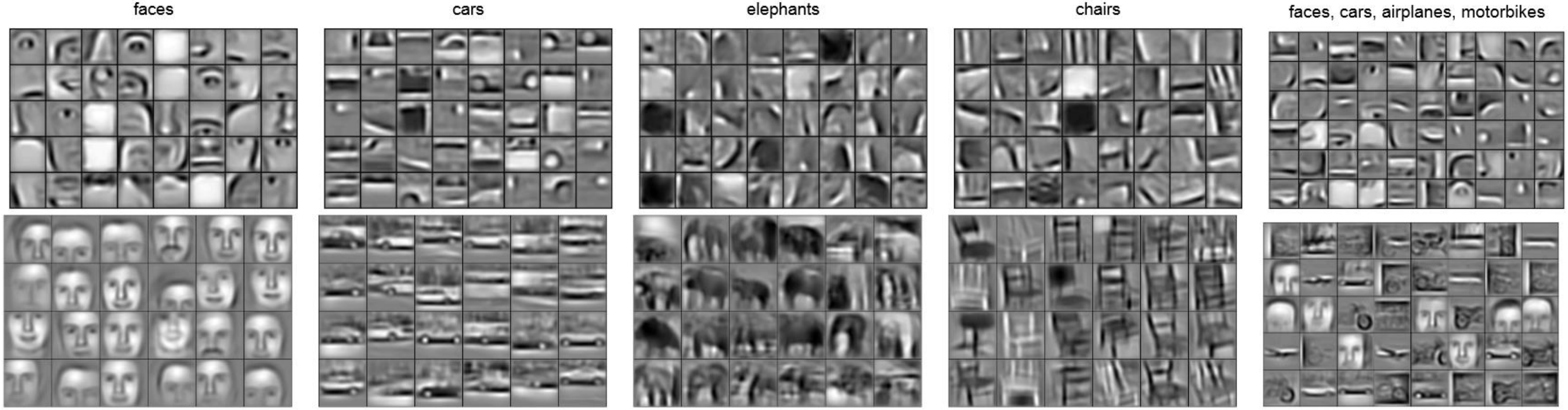


Cool Picture Motivation for Deep ~~Learning~~ Belief Networks

- First layer of z_i trained on 10 by 10 image patches:



- Visualization of second and third layers trained on specific objects:



http://www.cs.toronto.edu/~rgrosse/icml09-cdbn.pdf

Summary

- Belief networks represent conditional independence using graphs.
- Graphical representation of models like naïve Bayes and Markov.
- D-separation tests any conditional independence from graph.
- Sampling/inference and learning are easy.
- Decoding/conditional-inference and learning with hidden hard.
  - But easy if graph structure is ‘nice’.

- Next time:
  - Review of topics we’ve covered, overview of topics we didn’t.