CPSC 440: Advanced Machine Learning

More DAGs 2

Mark Schmidt

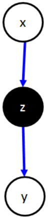
University of British Columbia

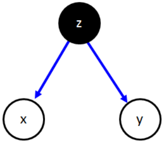
Winter 2021

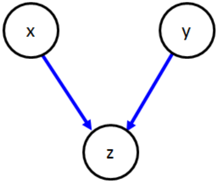
Last Time: D-Separation

- D-separation can be used to "read" conditional independence from graph.
  - Can be derived by considering DAG as "inheritance of genes".

- 3 Cases that can "block" a path between nodes:

Case 1: 

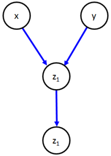
Case 2: 

Case 3: 

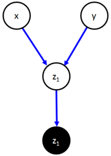
- Case 1: Observing a variable in a "chain" blocks a path.
- Case 2: Observing a parent in a "fork" blocks a path.
- Case 3: Not observing a child in a "v-structure" blocks a path.
  - We say that variables are "d-separated" if every path between them is blocked.

D-Separation Case 3: Common Child

- Case 3: x and y share a child z_1:
  - If there exists an unobserved grandchild z_2:

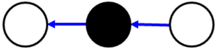
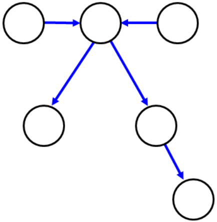
    

    We have $x \perp y$: the path is still blocked by not knowing z_1 or z_2.
  - But if z_2 is observed:

    We have $x \not\perp y \mid z_2$: grandchild creates dependence even with unobserved parent.
- Case 3 needs to consider descendants of child.

D-Separation Summary (MEMORIZE)

- We say that A and B are d-separated (conditionally independent) if *all undirected paths* P from A to B are "blocked" because *at least one* of the following holds:
  1. P includes a "chain" with an observed middle node (e.g., Markov chain):
  
  2. P includes a "fork" with an observed parent node (e.g., naive Bayes):
  
  3. P includes a "v-structure" or "collider" (e.g., probabilistic PCA):
  
  where the "child" and all its descendants are unobserved.

Alarm Example

- Case 1:
  - Earthquake $\not\perp$ Call.
  - Earthquake $\perp$ Call $|$ Alarm.
- Case 2:
  - Alarm $\not\perp$ Stuff Missing.
  - Alarm $\perp$ Stuff Missing $|$ Burglary.

Alarm Example

- Case 3:
  - Earthquake $\perp$ Burglary.
  - Earthquake $\not\perp$ Burglary | Alarm.
    - "Explaining away": knowing one parent can make the other less/more likely.
- Multiple Cases:
  - Call $\not\perp$ Stuff Missing.
  - Earthquake $\perp$ Stuff Missing.
  - Earthquake $\not\perp$ Stuff Missing | Call.

Discussion of D-Separation

- D-separation lets you say if conditional independence is implied by assumptions:

$$(A \text{ and } B \text{ are d-separated given } E) \Rightarrow A \perp B \mid E.$$

- However, there might be extra conditional independences in the distribution:
  - These would depend on specific choices of the $p(x_j \mid x_{\text{pa}(j)})$.
  - Or some *orderings* of the chain rule may reveal different independences.
  - So lack of d-separation does not imply dependence.

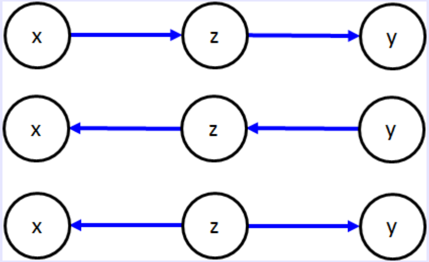
- Instead of restricting to $\{1, 2, \ldots, j-1\}$, consider general parent choices.
  - x_2 could be a parent of x_1.

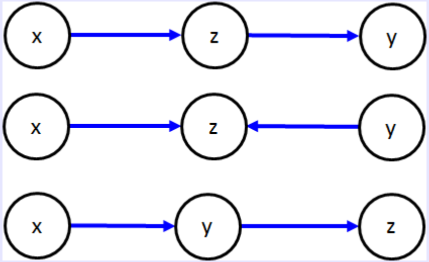
- As long the graph is acyclic, there exists a valid ordering (chain rule makes sense).

(all DAGs have a "topological order" of variables where parents are before children)

Non-Uniqueness of Graph and Equivalent Graphs

- Note that some graphs imply same conditional independences:
  - Equivalent graphs: same v-structures and other (undirected) edges are the same.
  - Examples of 3 *equivalent* graphs (left) and 3 non-equivalent graphs (right):



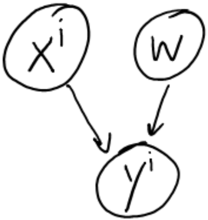


Discussion of D-Separation

- So the graph is not necessarily unique and is not the whole story.
- But, we can already do a lot with d-separation:
  - Implies every independence/conditional-independence we've used in 340/440.
- Here we start blurring distinction between data/parameters/hyper-parameters...

Tilde Notation as a DAG

- When we write
$$y^i \sim \mathcal{N}(w^T x^i, 1),$$
this can be interpretd as a DAG model:

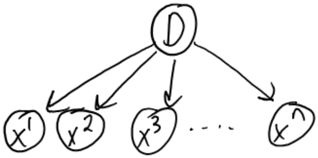


- "The variables on the right of $\sim$ are the parents of the variables on the left".
  - In this case, w only depends on X since we know y.

- Note that we're now including both data and parameters in the graph.
  - This allows us to see and reason about their relationships.

IID Assumption as a DAG

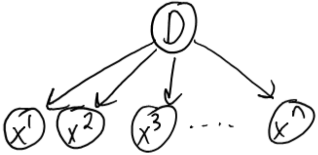
- During week 1, our first independence assumption was the IID assumption:



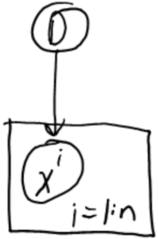
- Training/test examples come independently from data-generating process D.
- But D is unobserved, so knowing about some x^i tells us about the others.
  - This why the IID assumptions lets us learn.
- We'll use this understanding later to relax the IID assumption.
  - Bonus: using this to ask "when does semi-supervised learning make sense?"

Plate Notation

- Graphical representation of the IID assumption:

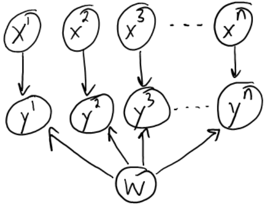


- It's common to represent repeated parts of graphs using plate notation:

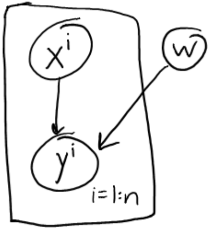


Tilde Notation as a DAG

- If the x^i are IID then we can represent linear regression as



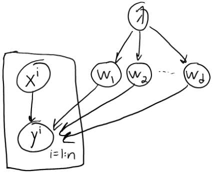
or



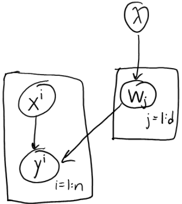
- From d-separation on this graph we have $p(y \mid X, w) = \prod_{i=1}^{n} p(y^i \mid x^i, w)$.
- We often omit the data-generating distribution D.
  - But if you want to learn then you should remember that it's there.
- Note that plate reflects parameter tieing: that we use same w for all i.

Tilde Notation as a DAG

- When we do MAP estimation under the assumptions
$$y^i \sim \mathcal{N}(w^T x^i, 1), \quad w_j \sim \mathcal{N}(0, 1/\lambda),$$
we can interpret it as the DAG model:



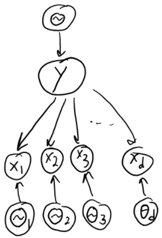
- Or introducing a second plate using:



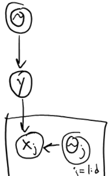
Other Models in DAG/Plate Notation

- For naive Bayes we have

$$y^i \sim \text{Cat}(\theta), \quad x^i \mid y^i = c \sim Cat(\theta_c).$$



- Or in plate notation as



Outline

Parameter Learning in General DAG Models

- The log-likelihood in DAG models is separable in the conditionals,

$$\log p(x \mid \Theta) = \log \prod_{j=1}^{d} p(x_j \mid x_{\mathsf{pa}(j)}, \Theta_j)$$
$$= \sum_{j=1}^{d} \log p(x_j \mid x_{\mathsf{pa}(j)}, \Theta_j)$$

- If each $p(x_j \mid x_{\mathsf{pa}(j)})$ has its own parameters Θ_j, we can fit them independently.
  - Optimize $\log p(x_1 \mid \Theta_j)$, then $\log p(x_2 \mid x_1, \Theta_j)$ (if x_1 is a parent), and so on.
  - We've done this for: naive Bayes, Gaussian discriminant analysis, M-step for mixtures.

- Sometimes you want to have tied parameters ($\Theta_j = \Theta_{j'}$)
  - Homogeneous Markov chains, Gaussian discriminant analysis with shared covariance.
  - Not separable, and need to fit $p(x_j \mid x_{\mathsf{pa}(j)}, \Theta_j)$ and $p(x_{j'} \mid x_{\mathsf{pa}(j')}, \Theta_j)$ together.

Tabular Parameterization in DAG Models

- To specify distribution, we need to decide on the form of $p(x_j \mid x_{\mathsf{pa}(j)}, \Theta_j)$.
- For discrete data a default choice is the tabular parameterization:

$$p(x_j \mid x_{\mathsf{pa}(j)}, \Theta_j) = \theta_{x_j, x_{\mathsf{pa}(j)}} \quad \text{(one parameter per child/parent combo)},$$

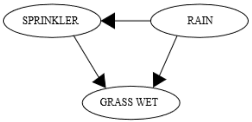
  as we did for Markov chains (but now with multiple parents).
- Intuitive: just need conditional probabilities of children given parents like

$$p(\text{"wet grass"} = 1 \mid \text{"sprinkler"} = 1, \text{"rain"} = 0),$$

  and MLE is just counting.

Tabular Parameterization Example

	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99



RAIN	
T	F
0.2	0.8

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

https://en.wikipedia.org/wiki/Bayesian_network

Some quantities can be directly read from the tables:

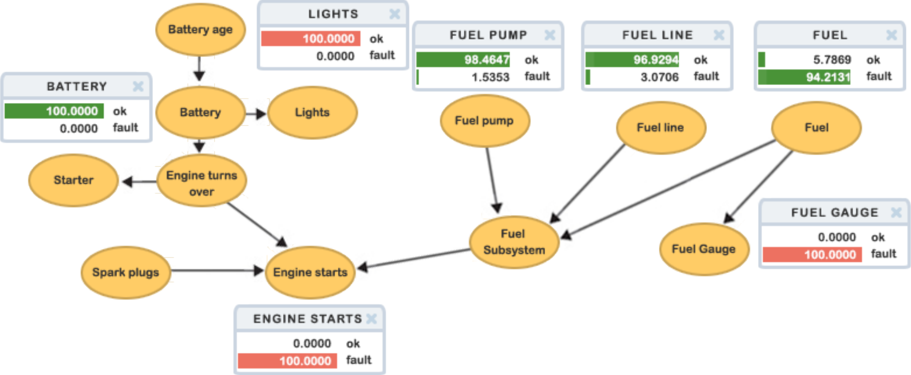
$$p(R = 1) = 0.2.$$

$$p(G = 1 \mid S = 0, R = 1) = 0.8.$$

Can calculate any probabilities using marginalization/product-rule/Bayes-rule (bonus).

Tabular Parameterization Example

Some companies sell software to help companies reason using tabular DAGs:



http://www.hugin.com/index.php/technology

Fitting DAGs using Supervised Learning

- But tabular parameterization requires too many parameters:
  - With binary states and k parents, need 2^{k+1} parameters.
- One solution is letting users specify a "parsimonious" parameterization:
  - Typically have a linear number of parameters.
  - For example, the "noisy-or" model: $p(x_j = 1 \mid x_{\text{pa}(j)}) = 1 - \prod_{k \in \text{pa}(j)}(1 - q_k)$.
    - "Estimate probability that each symptom leads to disease on its own".
    - Value q_k is "probability of seeing disease given symptom k".
- But if we have data, we can use supervised learning.
  - Write fitting $p(x_j \mid x_{\text{pa}(j)})$ as our usual $p(y \mid x)$ problem.
    - Predicting one column of X (child) given the values of some other columns (parents).

Fitting DAGs using Supervised Learning

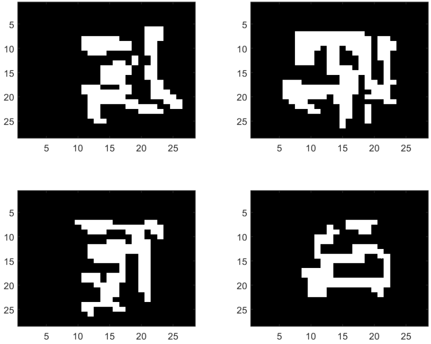
- For $j = 1 : d$:
  1. Set $\bar{y}^i = x^i_j$ and $\bar{x}^i = x^i_{\text{pa}(j)}$.
  2. Solve a supervised learning problem using $\{\bar{X}, \bar{y}\}$.
     - Gives you a model of $p(x_j \mid x_{\text{pa}(j)})$.
- Combine the d regression/classification models as the density estimator.

- We've turned unsupervised learning into supervised learning.

- We can use our usual tricks:
  - Linear models, non-linear bases, regularization, kernel trick, random forests, etc.
  - With least squares for continuos x_j it's called a Gaussian belief network.
  - With logistic regression for binary x_j it's called a sigmoid belief networks.
  - Don't need Markov assumptions to tractably fit these models.

MNIST Digits with Tabular DAG Model

- Recall our latest MNIST model using a tabular DAG:

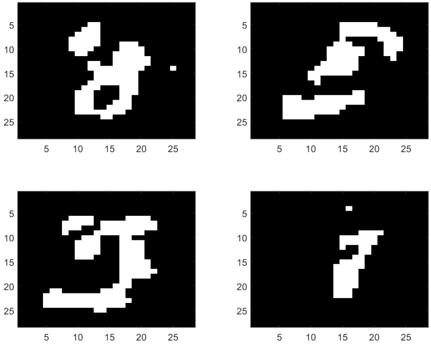


- This model is pretty bad because you only see 8 parents.

MNIST Digits with Sigmoid Belief Network

- Samples from sigmoid belief network:

(DAG with logistic regression for each variable)



where we use all previous pixels as parents (from 0 to 783 parents).

- Models long-range dependencies but has a linear assumption.

DAGs: Big Picture

- Setting the parameters of a DAG model:
  - Get the graph from an expert, or learn the graph (later).
  - Given the conditional probabilities from an expert, or learn them from data.
    - Counting or supervised learning, and EM if you have hidden/missing values.
- Inference in DAG models:
  - Can use Monte Carlo approximations with ancestral sampling:
    - Sample x_1 from $p(x_1)$, x_2 from $p(x_2 \mid x_{\text{pa}(2)})$, x_3 from $p(x_3 \mid x_{\text{pa}(3)})$,...
  - Can use dynamic programming for exact inference with discrete x_j.
    - Also works if all $p(x_j \mid x_{\text{pa}(j)})$ are Gaussian.

Summary

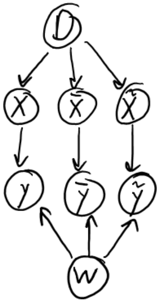
- D-separation allows us to test conditional independences based on graph.
  - Watch out for v-structures and ancestors of v-structures.
- Plate Notation lets us compactly draw graphs with repeated patterns.
  - There are fancier versions of plate notation called "probabilistic programming".
- Parameter learning in DAGs:
  - Can fit each $p(x_j \mid x_{\text{pa}(j)})$ independently.
  - Tabular parameterization, or treat as supervised learning.
- Next time: trying to discover the graph structure from data.

Does Semi-Supervised Learning Make Sense?

- Should unlabeled examples always help supervised learning?
  - No!
- Consider choosing unlabeled features $\bar{x}^i$ uniformly at random.
  - Unlabeled examples collected in this way will not help.
  - By construction, distribution of $\bar{x}^i$ says nothing about $\bar{y}^i$.
- Example where SSL is not possible:
  - Try to detect food allergy by trying random combinations of food:
    - The actual random process isn't important, as long as it isn't affected by labels.
    - You can sample an infinite number of $\bar{x}^i$ values, but they says nothing about labels.
- Example where SSL is possible:
  - Trying to classify images as "cat" vs. "dog.:
    - Unlabeled data would need to be images of cats or dogs (not random images).
    - Unlabeled data contains information about what images of cats and dogs look like.
    - For example, there could be clusters or manifolds in the unlabeled images.

Does Semi-Supervised Learning Make Sense?

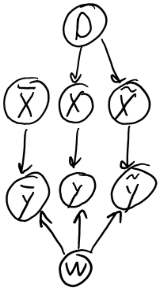
- Let's assume our semi-supervised learning model is represented by this DAG:



- Assume we observe $\{X, y, \bar{X}\}$ and are interested in test labels $\tilde{y}$:
  - There is a dependency between y and $\tilde{y}$ because of path through w.
    - Parameter w is tied between training and test distributions.
  - There is a dependency between X and $\tilde{y}$ because of path through w (given y).
    - But note that there is also a second path through D and $\tilde{X}$.
  - There is a dependency between $\bar{X}$ and $\tilde{y}$ because of path through D and $\tilde{X}$.
    - Unlabeled data helps because it tells us about data-generating distribution D.

Does Semi-Supervised Learning Make Sense?

- Now consider generating $\bar{X}$ independent of D:

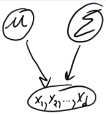


- Assume we observe $\{X, y, \bar{X}\}$ and are interested in test labels $\tilde{y}$:
  - Knowing X and y are useful for the same reasons as before.
  - But knowing $\bar{X}$ is not useful:
    - Without knowing $\bar{y}$, $\bar{X}$ is d-separated from $\tilde{y}$ (no dependence).

Other Models in DAG/Plate Notation

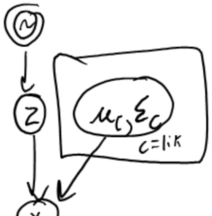
- In a full Gaussian model for a single x we have

$$x^i \sim \mathcal{N}(\mu, \Sigma).$$

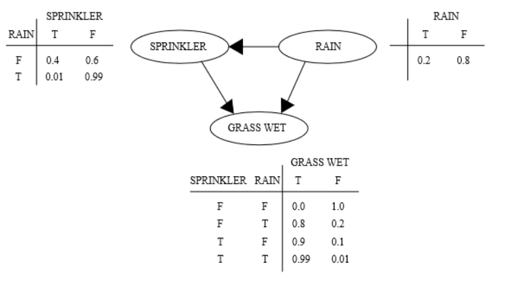


- For mixture of Gaussians we have

$$z^i \sim \mathsf{Cat}(\theta), \quad x^i \mid z^i = c \sim \mathcal{N}(\mu_c, \Sigma_c).$$



Tabular Parameterization Example



https://en.wikipedia.org/wiki/Bayesian_network

Can calculate any probabilities using marginalization/product-rule/Bayes-rule, for example:

$$p(G=1 \mid R=1) = p(G=1, S=0 \mid R=1) + p(G=1, S=1 \mid R=1) \qquad \left(p(a \mid c) = \sum_b p(a, b \mid c)\right)$$

$$= p(G=1 \mid S=0, R=1)p(S=0 \mid R=1) + p(G=1 \mid S=1, R=1)p(S=1 \mid R=1)$$

$$= 0.8(0.99) + 0.99(0.01) = 0.81.$$

Dynamic Bayesian Networks

- Dynamic Bayesian networks are a generalization of Markov chains and DAGs:
    - At each time, we have a set of variables x^t.
    - The initial x^0 comes from an "initial" DAG.
    - Given x^{t-1}, we generate x^t from a "transition" DAG.

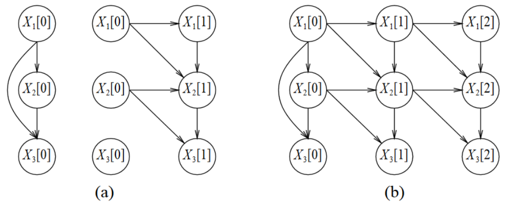


Figure 1: (a) A prior network and transition network defining a DPN for the attributes X_1, X_2, X_3. (b) The corresponding "unrolled" network.

https://www.cs.ubc.ca/~murphyk/Papers/dbnsem_uai98.pdf

- Can be used to model multiple variables over time.
    - Unconditional sampling is easy but inference may be hard.