Lecture 2: Decision Theory & Directed Graphical Models

Thibault Randrianarisoa

University of Toronto, Winter 2026

January 12, 2026

Table of contents

Decision theory

Why do we care about probabilities in the first place?

Answer: They help us make decisions. In some senses, making better decisions/actions is the only reason to ever think.

Pascal, 1670: When faced with a choice of actions, you should:

1. Determine the value (goodness) of all possible outcomes. $V(o) \quad \forall o$
   (This is subjective and usually hard to determine, but usually only the order of magnitude matters. If you get it a little wrong, no big deal)
2. Find the probability of each outcome under each action. $p(o|a) \quad \forall o \forall a$
   👉 That's what this course can help with
3. Choose the action with the highest expected value:

$$\underset{a}{\operatorname{argmax}}\, \mathbb{E}_{p(o|a)}[V(o)]$$

Where did $P(\text{outcome} \mid \text{action})$ come from?

That's what the rest of the course is about.

- In general, these numbers will also be expectations over joint distributions of many possible variables, like which infection we have, the details of our own physiology.
- We can always make the model more detailed to include more information.

But this is ultimately what we're going to do with these probabilities:
Use them to make informed decision to make our lives better.

Objections to Utility Theory I

Objection 1: I don't care about the average outcome, some outcomes are simply unacceptable. I want to make sure my plane never crashes, or my bridge never falls.

Answer: You can't guarantee anything, you can only make probabilities small. If someone dying is really bad, just give it a really high negative utility. But you might have to trade some chances of deaths vs. other outcomes anyway.

Objection 2: You can't compare the pain of being sick to the cost of medicine in dollars.

Answer: We have to. That is, we usually have to make tradeoffs, and so we have to compare different types of outcome on the same scale one way or another. We should be explicit about what we value so that we can discuss it and sanity-check it.

Objections to Utility Theory II

The World Health Organization estimated the relative quality people assigned to their own lives under different disabilities:

Condition	Life discount factor
Dementia	0.666
Blindness	0.594
Schizophrenia	0.528
AIDS, not on ART	0.505
Burns 20%-60% of body	0.441
Fractured femur	0.372
Moderate depression episode	0.350
Amputation of foot	0.300
Deafness	0.229
Infertility	0.180
Amputation of finger	0.102
Lower back pain	0.061

Objections to Utility Theory III

Objection 3: It's computationally expensive to compute conditional probabilities and expectations over all possible outcomes.

Answer: Agreed! That's what the tools in this course are designed to help with.

Decision making

Framework for understanding many of the procedures we consider.

- Suppose we have an input vector x and a corresponding target (output) value c with joint probability distribution: $p(x, c)$.
- Our goal is to predict the output label c given a new value for x.
- For now, we focus on classification so c is a categorical variable, but the same reasoning applies to regression (continuous target).

The joint probability distribution $p(x, c)$ provides a complete summary of uncertainties associated with these random variables.

Example: Cancer screening from chest X-ray

Based on the X-ray image, we would like to determine if the patient has cancer or not.

- The input vector x is pixel intensities, and the output c represents the presence of cancer, class $\mathcal{C}_1$, or absence of cancer, class $\mathcal{C}_2$.
- $\mathcal{C}_1$: cancer present
- $\mathcal{C}_2$: cancer absent

We can use an "*arbitrary*" encoding for these classes $\mathcal{C}_1$ and $\mathcal{C}_2$, e.g. take $c = 0$ corresponding to class $\mathcal{C}_1$, and $c = 1$ corresponds to $\mathcal{C}_2$.

Decision Problem

Suppose we estimated the joint distribution $p(x, c)$ using some ML method. Decide whether to give treatment to the patient or not.

Example (follow-up):

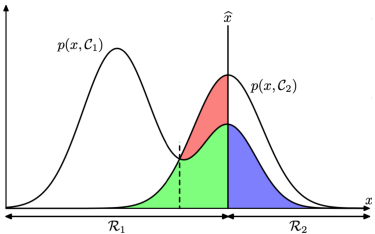
- Given a new X-ray image, our goal is to decide which of the two classes that image should be assigned to.
- We could compute conditional probabilities of the two classes, given the input image, for $k = 1, 2$:

$$p(\mathcal{C}_k|x) = \frac{p(x|\mathcal{C}_k)p(\mathcal{C}_k)}{p(x)} \quad \text{(Bayes' rule)}$$

- Intuitively, pick class with higher posterior probability.
- We now formalize in what sense this choice is optimal.

Decision rule: Divide the input space into regions $\mathcal{R}_1$, $\mathcal{R}_2$ (decision regions) such that all points x in $\mathcal{R}_k$ are assigned to class $\mathcal{C}_k$, $k = 1, 2$.

Criterion to optimize: Make as few misclassifications as possible.

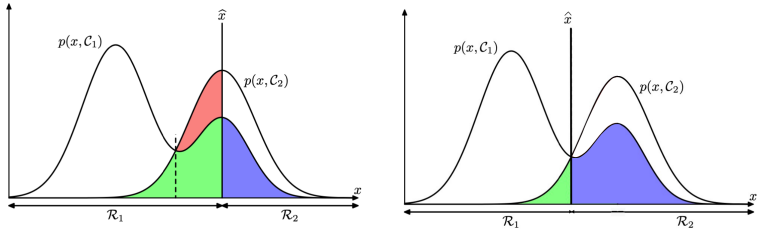


- Red + green regions: input belongs to class $\mathcal{C}_2$, but is assigned to $\mathcal{C}_1$.
- Blue region: input belongs to class $\mathcal{C}_1$, but is assigned to $\mathcal{C}_2$.

Probability of mistake:

Misclassification rate

Compare the following two decision rules:



- Blue + green area is always included in the p(mistake).
- On the left there are points $x \in \mathcal{R}_1$ for which $p(x, \mathcal{C}_2) > p(x, \mathcal{C}_1)$ (red part).
- Reduce the red area by moving the threshold $\hat{x}$ to the left.

Misclassification error

- Misclassification error:

$$p(\text{mistake}) = \underbrace{\int_{\mathcal{R}_1} p(x, \mathcal{C}_2) dx}_{\text{red+green}} + \underbrace{\int_{\mathcal{R}_2} p(x, \mathcal{C}_1) dx}_{\text{blue}}$$

  and the decision regions $\mathcal{R}_1$ and $\mathcal{R}_2$ are disjoint.
- Therefore, for a particular input x, if $p(x, \mathcal{C}_1) > p(x, \mathcal{C}_2)$, then we assign x to class $\mathcal{C}_1$.

Minimizing misclassification

Since $p(x, \mathcal{C}_k) = p(\mathcal{C}_k|x)p(x)$, in order to minimize the probability of making mistake, we assign each x to the class for which the posterior probability $p(\mathcal{C}_k|x)$ is largest. This minimizes the misclassification rate.

Expected loss

Simply minimizing the misclassification rate may not be desirable.

- We incorporate a **loss function** to measure the loss incurred by taking any of the available decisions.
- Suppose that for x, the true class is $\mathcal{C}_k$, but we assign x to class $\mathcal{C}_j$ and incur loss of L_{kj} ((k, j)-th element of a loss matrix).

Example of a loss matrix for the cancer example:

		Decision	
		cancer	healthy
Truth	cancer	0	1000
	healthy	1	0

Thus the expected loss is given by

New goal: Minimize expected loss

Therefore, we want to minimize

$$\mathbb{E}[L] = \sum_k \sum_j \int_{\mathcal{R}_j} L_{kj} p(x, \mathcal{C}_k) dx$$

Define $g_j(x) = \sum_k L_{kj} p(x, \mathcal{C}_k)$. Notice that $g_j(x) \geqslant 0$ and

Thus, minimizing $\mathbb{E}[L]$ is equivalent to choosing

Simplifying further

We can also use the product rule $p(x, \mathcal{C}_1) = p(\mathcal{C}_1|x)p(x)$ and reduce the problem to:

Discriminant rules

Find regions $\mathcal{R}_j$ such that the following is minimized:

$$\sum_k L_{kj} p(\mathcal{C}_k|x).$$

That is

$$\mathcal{R}_j = \left\{ x : \sum_k L_{kj} p(\mathcal{C}_k|x) < \sum_k L_{ki} p(\mathcal{C}_k|x) \text{ for all } i \neq j \right\}.$$

The reject option

In high-risk domains (e.g., medicine, finance), we may prefer to say "I don't know" rather than making an untrustworthy prediction.

- **Actions:** $\mathcal{A} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \} \cup \{R\}$, where action R represents the **reject** option.
- **Loss function:** Similar to the missclassification rate, we assume the cost of misclassification is 1, the cost of correct classification is 0, and the cost of rejecting is λ_r (where $0 < \lambda_r < 1$).

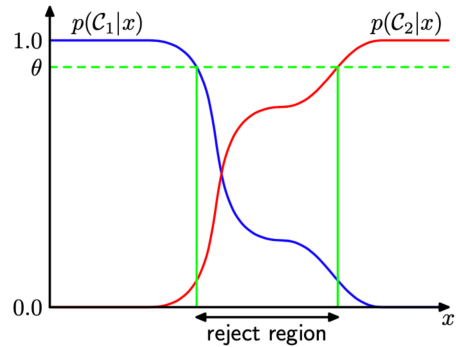
Optimal Policy

The optimal action is to pick the most probable class y^* only if its probability exceeds a threshold determined by the reject cost:

For the regions where we are relatively uncertain about class membership, we do not have to make a decision.

		Decision		
		cancer	healthy	Abstention
Truth	cancer	0	1000	10
	healthy	1	0	0.1

$$\mathcal{R}_R = \left\{ x : \sum_k L_{kR} p(\mathcal{C}_k|x) < \sum_k L_{ki} p(\mathcal{C}_k|x) \text{ for all } i \right\}$$



Missclassification rate: When the conditional class probabilities fall below λ_r, we refuse to make a decision.

Loss functions for regression

- Consider an input/target setup (x, t) where the target (output) is continuous $t \in \mathbb{R}$, and the joint density is $p(x, t)$.
- We aim to find a regression function $y(x) \approx t$ which maps inputs to the outputs.
- Consider the squared loss function L between $y(x)$ and t to assess the quality of our estimate $L(y(x), t) =$.

Goal:

What is the best function $y(x)$ that minimizes the expected loss?

$$\mathbb{E}[L] = \iint L(y(x), t)p(x, t)dxdt.$$

Minimizing expected loss: Best regression function

We add and subtract $\mathbb{E}[t|x]$ and write

The last term is zero since

Best regression function

- We showed that the expected loss is given by the sum of two **non-negative** terms

$$\mathbb{E}[L] = \iint (y(x) - \mathbb{E}[t|x])^2 p(x,t) dx dt + \iint (\mathbb{E}[t|x] - t)^2 p(x,t) dx dt.$$

- The second term does not depend on $y(x)$ thus choosing the best regression function $y(x)$ is equivalent to minimizing the first term on the right hand side.

- This term is always non-negative and exactly zero if

$$y(x) = \mathbb{E}[t|x].$$

- The second term is the expectation of the conditional variance of $t|x$. It represents the intrinsic variability of the target data and can be regarded as noise.

Summary: Decision making

- Depending on the application, one needs to choose an appropriate loss function.
- Loss function can significantly change the optimal decision rule.
- One can always use the reject option and not make a decision.
- In case of regression, the optimal map between x and t corresponds to the conditional expectation $\mathbb{E}[t|x]$.
- We focused on classification/regression but similar framework can be used to evaluate any statistical procedure (e.g. estimation).

Directed graphical models

Next:

- Graphical models notation
- Conditional independence on directed acyclic graphs (DAGs)
- DFS and pruning/deletion algorithm

Joint distributions

- The joint distribution of N random variables $(x_1, x_2, ..., x_N)$ is a very general way to encode knowledge about a system.
- Assume $x_i \in \{0, 1\}$ are binary, then it requires $2^N - 1$ parameters to specify the joint distribution

$$p(x_1, x_2, ..., x_N).$$

- This can be also written as

for any ordering of the variables, where $p(x_1|x_0) = p(x_1)$.

Powerful modelling principle

Exploit dependencies among variables and reduce the number of parameters!

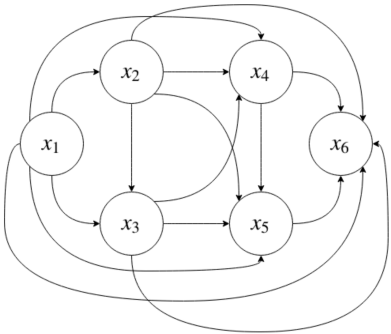
Conditional Independence

- Assume there are N random variables $x_1, x_2, ..., x_N$.
- For set $A \subset \{1, 2, ..., N\}$, we denote by $x_A = \{x_i : i \in A\}$.
- For disjoint A, B, C, if random variables x_A, x_B are conditionally independent given x_C, we write

$$x_A \perp x_B | x_C$$

- The following conditions are equivalent
  - $x_A \perp x_B | x_C$
  - $p(x_A, x_B | x_C) = p(x_A | x_C) p(x_B | x_C)$
  - $p(x_A | x_B, x_C) = p(x_A | x_C)$
  - $p(x_B | x_A, x_C) = p(x_B | x_C)$

Directed Acyclic Graphical Models (Bayes' Nets)



- A directed acyclic graphical model (DAG) encodes a particular form of factorization of the joint distribution.
- Variables are represented by nodes, and edges represent direct dependence.

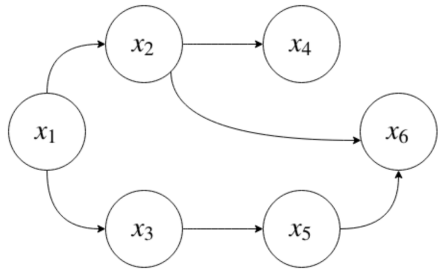
DAG induces the following factorization of the joint distribution:

$$p(x_1, ..., x_N) = \prod_{i=1}^{N} p(x_i | x_1, ..., x_{i-1}) =$$

where parents(x_i) is the set of nodes with edges pointing to x_i.

Example: Joint factorization induced by a DAG

Recall: In a DAGs $p(x_1, x_2, ..., x_N) = \prod_{i=1}^{N} p(x_i|\text{parents}(x_i))$.
Consider the following graph:

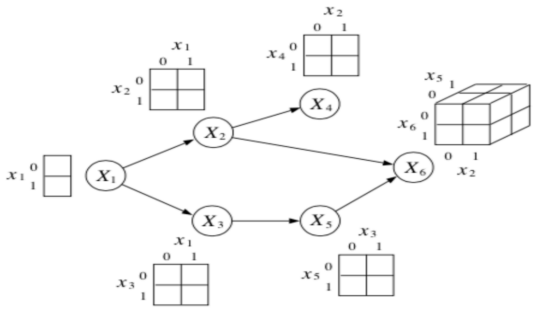


It induces the following factorization of the joint distribution:

$$p(x_1, x_2, ..., x_6) =$$

Conditional Probability Tables (CPT)

In our example, suppose each x_i is a binary random variable. How many parameters does it take to represent this joint distribution?



- For example, 2x2 CPT for the node x_4 corresponds to $p(x_4|x_2)$ requires 2 parameters.
- Each CPT with K_i parents requires 2^{K_i} parameters. In total, $\sum_i 2^{K_i} \leqslant N2^{\max K_i}$ parameters.
- If we allow all possible dependencies (fully-connected DAG), we need parameters.

This gives a big reduction in storage and computations; here 63 vs 13.

Conditional Independence in DAGs

D-separation (directed-separation) is a notion of connectedness in DAGs in which two sets of nodes may or may not be connected conditioned on a third set of nodes.

- Fix a DAG over N nodes $1, 2, ..., N$.
- This DAG defined factorization of the joint distribution $p(x_1, ..., x_N)$.
- This factorization implies some conditional independence that can be deducted from d-separation: if C d-separates A and B in the DAG then $x_A \perp x_B | x_C$.

We still have not defined d-separation...

Important reduction

- We have $x_A \perp x_B | x_C$ if and only if $x_a \perp x_b | x_C$ for all $a \in A, b \in B$.
- Also C d-separates A and B if and only if it d-separates each $a \in A$ and $b \in B$. We note this property $\text{dsep}_G(\mathbf{A}, \mathbf{C}, \mathbf{B})$.

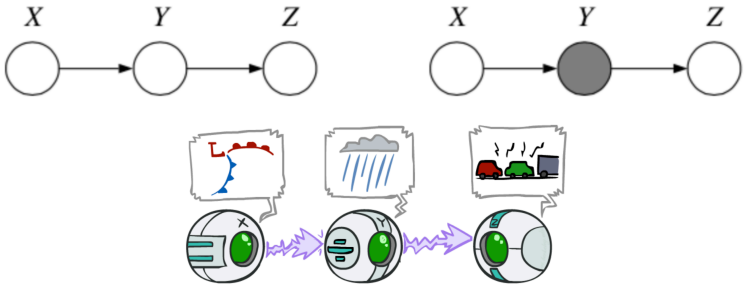
DFS Algorithm for checking independence

To check if an independence is true, we can cycle through each node in A, do a depth-first search to reach every node in B, and examine the path between them. If all of the paths are d-separated (i.e., conditionally independent), then

$$x_i \perp x_j \mid x_k$$

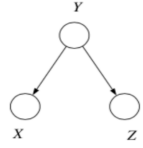
- It will be sufficient to consider triples of nodes.
- Let's go through some of the most common triples.

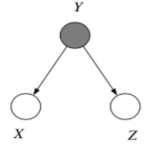
Causal Chain

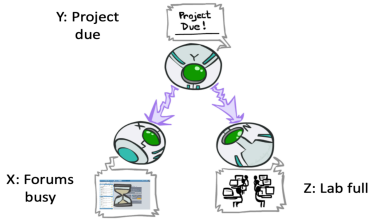


Common Cause

Where we think of y as the "common cause" of the two independent effects x and z.



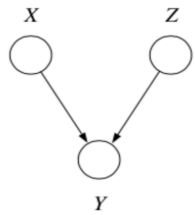


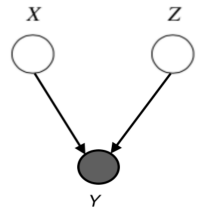


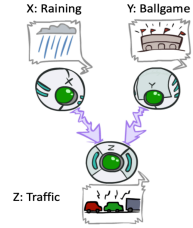
Question: When we condition on y, are x and z independent?
Answer: From the graph, we get

Thus, Y d-separates X and Z like in the previous case.

Explaining Away (Common Effect)







Question: When we condition on y, are x and z independent?
Answer: From the graph, we get

images credit Abbeel & Klein

Pruning / Edge Deletion method

The definition of d-separation calls for considering all paths connecting nodes in $\mathbf{X}$ with nodes in $\mathbf{Y}$. The number of such paths can be exponential, yet we can implement the test efficiently.

Efficient d-separation test

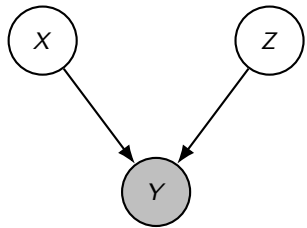
Testing whether $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$ in DAG G is equivalent to testing whether $\mathbf{X}$ and $\mathbf{Y}$ are **disconnected** in a new DAG G', obtained by pruning G as follows:

1. **Delete barren nodes:** Delete any leaf node W from DAG G as long as W does not belong to $\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}$.
   - *This process is repeated until no more nodes can be deleted.*
2. **Delete outgoing edges:** Delete all edges outgoing from nodes in $\mathbf{Z}$.

If $\mathbf{X}$ and $\mathbf{Y}$ are disconnected in the resulting graph G', then $\text{dsep}_G(\mathbf{X}, \mathbf{Z}, \mathbf{Y})$ holds.

Example I: Explaining Away (Using Pruning Algorithm)

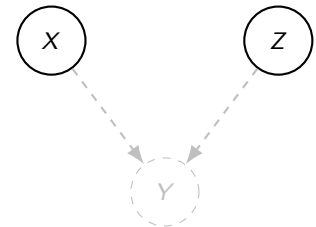
Case 1: Check $X \perp Z \mid Y$
(Evidence Y is shaded)



Algorithm Steps:
1. No barren leaves to prune.
2. Y is evidence. Delete outgoing from Y (none exist).

Result: X and Z are connected.

Case 2: Check $X \perp Z$
(No evidence)

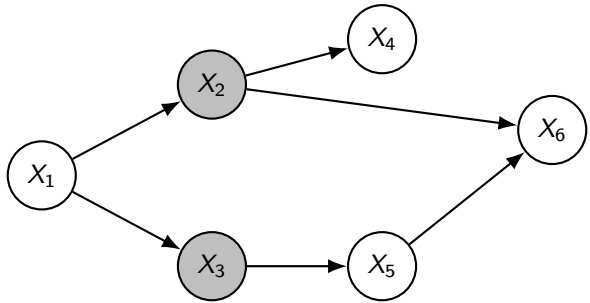


Algorithm Steps:
1. Y is a leaf and not in XYZ. **Prune Y.**

Result: X and Z are disconnected.

Example II: Large DAG

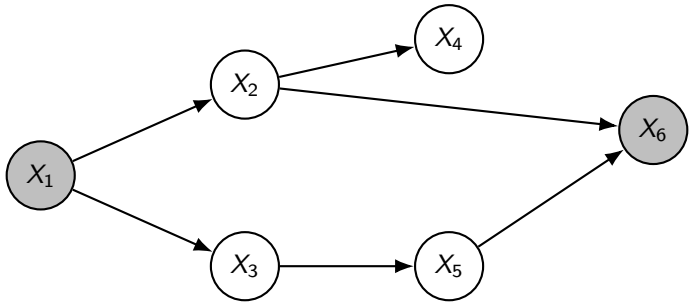
In the following graph, is $X_1 \perp X_6 \mid \{X_2, X_3\}$?



Example II: Solution (Applying Pruning Algorithm)

Example III

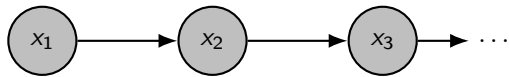
In the following graph, is $X_2 \perp X_3 \mid \{X_1, X_6\}$?



Example III: Solution (Applying Pruning Algorithm)

Example of a DAGM: Markov Chain

Markov chains are a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event.

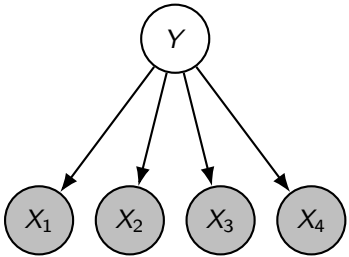


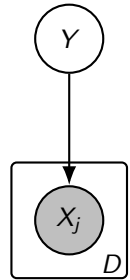
$$p(x) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2)p(x_4 \mid x_3)\dots$$

In other words, it is a model that satisfies the Markov property, i.e., conditional on the present state of the system, its future and past states are independent.

Plates

Because Bayesian methods treat parameters as random variables, we would like to include them in the graphical model. One way to do this is to repeat all the iid observations explicitly and show the parameter only once. A better way is to use **plates**, in which repeated quantities that are iid are put in a box.



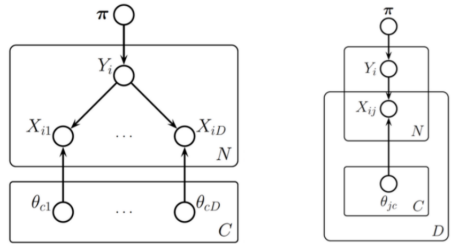


Plates denote replication of random variables

The rules of plates are simple: repeat every structure in a box a number of times given by the integer in the corner of the box (e.g. N or D), updating the plate index variable (e.g. n) as you go. Duplicate every arrow going into the plate and every arrow leaving the plate by connecting the arrows to each copy of the structure.

Nested Plates

Plates can be nested, in which case their arrows get duplicated also, according to the rule: draw an arrow from every copy of the source node to every copy of the destination node.



$$p(\pi)\left[\prod_{c=1}^{C}\prod_{j=1}^{D}p(\theta_{cj})\right]\prod_{i=1}^{N}\left[p(y_i \mid \pi)\prod_{j=1}^{D}p(x_{ij} \mid y_i, \theta_{j1}, \ldots, \theta_{jC})\right]$$

Plates can also cross (intersect), in which case the nodes at the intersection have multiple indices and get duplicated a number of times equal to the product of the duplication numbers on all the plates containing them.

Summary

- DAGs are great for encoding conditional independencies.
- They can reduce the number of parameters significantly.
- Conditional independence between two sets of variables on a DAG can be found using the **Pruning / Edge Deletion method**.
- Next lecture: ?