USI/SUPSI
Istituto
Dalle Molle
di Studi
sull'Intelligenza
Artificiale
IDSIA

Advanced Probabilistic Modelling

Definitions and Fundamentals

Marco Scutari

Dalle Molle Institute for
Artificial Intelligence (IDSIA)

Academic Year 2025–2026

Machine learning studies the algorithms and the statistical tools that allow computer systems to perform specific, well-defined tasks without explicit instructions. It is a sub-field of artificial intelligence.

Broadly speaking, in order to do this:

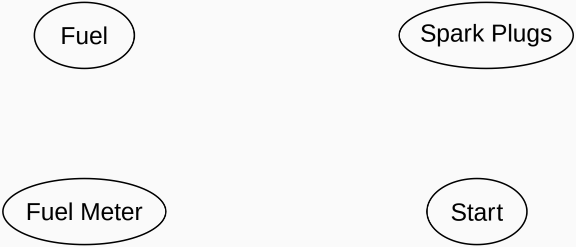
1. We need a working model of the world that describes the task and its context in a way a computer can understand.
2. We need a goal: how do we measure the performance of the model? Because that is what we optimise for! Usually it is the ability to predict new events.
3. We encode our knowledge of the world drawing information from training data, experts or both: this is called learning.
4. The computer system uses the model as a proxy of reality and to perform inference and decide if/how to perform the assigned task as new inputs come in.

Identify the Variables to Include in the Model

The first step in building a machine learning model is to choose which variables to include. Which aspects of/entities in the world do we need the model to represent for the computer to carry out the assigned task? This is known as feature selection.

- Each aspect of the world or entity is modelled with one random variable.
- We should use a small enough number of variables because if we have too many:
  - it is difficult it is to construct the model;
  - it is difficult to interpret and to troubleshoot it;
  - the model requires too much computing power to learn and to run.
- We must choose which are the relevant events that make up the sample space of each variable, again taking care of not having too many.

An example: The Car Start Problem



	Realistic	Pragmatic
Fuel	0%–100%	Yes, No
Spark Plugs	Work, Fault	Work, Fault
Start	Yes, No	Yes, No
Fuel Meter	0%–100%	Empty, Half, Full

The second step is choosing which class of machine learning models to select from.

- Generative models: we have a set of variables $X_1, \ldots, X_N$ describing various components of a complex phenomenon, and we are interested in building a mechanistic model of that phenomenon to understand it. Therefore, we want to show how the various parts interact with each other. In order to do so we choose to model their joint probability $\mathrm{P}(X_1, \ldots, X_N)$.

- Discriminative models: we have one particular variable (say, X_1) that is closely tied with our model task, and a number of other variables ($X_2, \ldots, X_N$) which we believe can be used to predict it. We do not care about how the X_i are related to each other, so we just model $\mathrm{P}(X_1 \mid X_2, \ldots, X_N)$.

Model Relationships Between Variables

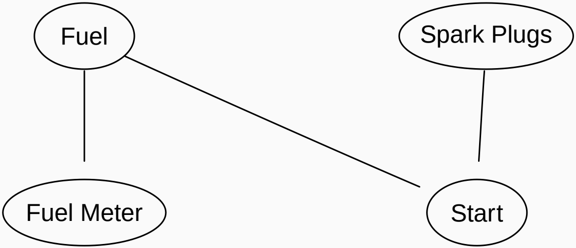
How do we decide whether there is a relationships between variables?

We never have perfect knowledge of what we are modelling: hence we use the language of probability, and we say that two variables are associated if the occurrence of an event in one variable affects the probability of an event occurring in another variable, possibly conditional on other variables.

How can we acquire information on what we are modelling:

- consulting domain experts;
- using probability and statistics to extract it from data;
- a combination of both.

The Car Start Problem, with Edges



- The `Fuel Meter` measures the amount of `Fuel`?
- The `Spark Plugs` ignite the `Fuel` to `Start` the car?
- If there is `Fuel` in the car, it can start even if the `Fuel Meter` is wrong and displays 0%?

Car Start: Probabilistic versus Causal Construction

In probability associations are symmetric: the derivation of Bayes' theorem makes it really clear that

$$\mathrm{P}(X_1 \mid X_2)\,\mathrm{P}(X_2) = \mathrm{P}(X_1, X_2) = \mathrm{P}(X_2 \mid X_1)\,\mathrm{P}(X_1).$$

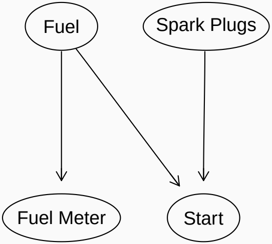
However, it feels more natural to choose the conditioning variables such that they affect the conditioned variables instead of the other way round.

But what does that mean from a modelling point of view? It means that we are giving arcs a causal interpretation and that we choose arc directions to go from cause (nodes) to effect (nodes).

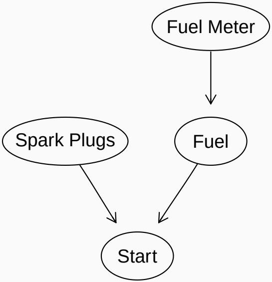
How do we do that?

The Car Start Problem, with Arcs

$$\mathrm{P}(\texttt{Start}, \texttt{Fuel Meter}, \texttt{Fuel}, \texttt{Spark Plugs})$$
$$= \mathrm{P}(\texttt{Start} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \times$$
$$\mathrm{P}(\texttt{Fuel Meter} \mid \texttt{Fuel})\, \mathrm{P}(\texttt{Fuel}) \times$$
$$\mathrm{P}(\texttt{Spark Plugs})$$



$$\mathrm{P}(\texttt{Start}, \texttt{Fuel Meter}, \texttt{Fuel}, \texttt{Spark Plugs})$$
$$= \mathrm{P}(\texttt{Start} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \times$$
$$\mathrm{P}(\texttt{Fuel Meter})\, \mathrm{P}(\texttt{Fuel} \mid \texttt{Fuel Meter}) \times$$
$$\mathrm{P}(\texttt{Spark Plugs})$$



The criterion to identify causes and effect is intervention. Consider:

- If we fill the tank with fuel, the fuel meter goes up.
- If we tamper with the fuel meter to make is say `Full`, the fuel tank does not magically refill itself.

Hence, `Fuel` is the cause and `Fuel Meter` is the effect and the most intuitive arc direction is $\texttt{Fuel} \to \texttt{Fuel Meter}$.

What the probability $\mathrm{P}(\texttt{Fuel Meter} \mid \texttt{Fuel})$ tells us is just that if the fuel meter says `Full` there probably is fuel in the tank, whereas if the fuel meter says `Empty` there may be no fuel in the tank (assuming the fuel meter works reliably).

Car Start: The Conditional Probabilities

Spark Plugs

Work	Fault
?	?

Fuel

Yes	No
?	?

Start

	Spark Plugs = Work	
	Fuel = Yes	Fuel = No
Yes	?	?
No	?	?

Fuel Meter

	Fuel = Yes	Fuel = No
Empty	?	?
Half	?	?
Full	?	?

	Spark Plugs = Fault	
	Fuel = Yes	Fuel = No
Yes	?	?
No	?	?

After we decide that the first model is good to go, we need to:

1. choose which distribution to use for each node;
2. fill in the values of its parameters by asking domain experts, estimating them from data or a combination of the two.

The number of parameters gives the complexity of the model, rather than the number of nodes or the number of arcs.

A more general way of using a model is to interrogate it: we have some evidence on some of the variables (that is, we assume we know their values), and we would like to know the the probability of some event.

For instance: say that $\texttt{Fuel Meter} = \texttt{Half}$. How does $\mathrm{P}(\texttt{Start} = \texttt{Yes})$ change after we introduce this evidence in the model?

Predicting `Start` from all the other variables is a particular case in which we have evidence on all the other variables.

Armed with patience, we start by writing

$$
\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}) = \\
&\qquad \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}) + \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{No})
\end{aligned}
$$

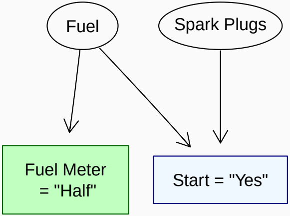
and then, recursively,

$$
\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}) = \\
&\qquad \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Work}) + \\
&\qquad\qquad \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Fault})
\end{aligned}
$$

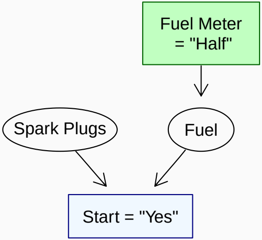
$$
\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Work}) \\
&= \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Work}, \texttt{Fuel.Meter} = \texttt{Full}) + \\
&\quad \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Work}, \texttt{Fuel.Meter} = \texttt{Half}) + \\
&\quad \mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel} = \texttt{Yes}, \texttt{Spark.Plugs} = \texttt{Work}, \texttt{Fuel.Meter} = \texttt{Empty})
\end{aligned}
$$

Car Start: The Principled (Probabilistic) Way

$$
\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel Meter} = \texttt{Half},\\
&\qquad \texttt{Fuel}, \texttt{Spark Plugs}) =\\
&= \mathrm{P}(\texttt{Start} = \texttt{Yes} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \times\\
&\quad \mathrm{P}(\texttt{Fuel Meter} = \texttt{Half} \mid \texttt{Fuel}) \times\\
&\quad \mathrm{P}(\texttt{Fuel})\, \mathrm{P}(\texttt{Spark Plugs})
\end{aligned}
$$

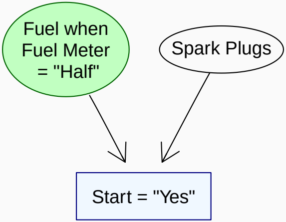


$$
\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel Meter} = \texttt{Half},\\
&\qquad \texttt{Fuel}, \texttt{Spark Plugs})\\
&= \mathrm{P}(\texttt{Start} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \times\\
&\quad \mathrm{P}(\texttt{Fuel} \mid \texttt{Fuel Meter} = \texttt{Half}) \times\\
&\quad \frac{\mathrm{P}(\texttt{Fuel.Meter} = \texttt{Half})}{\cancel{\mathrm{P}(\texttt{Fuel})}} \times\\
&\quad \cancel{\mathrm{P}(\texttt{Fuel})}\, \mathrm{P}(\texttt{Spark Plugs})
\end{aligned}
$$



Car Start: The Principled (Probabilistic) Way

$$\begin{aligned}
&\mathrm{P}(\texttt{Start} = \texttt{Yes}, \texttt{Fuel}, \texttt{Spark Plugs} \mid \\
&\qquad \texttt{Fuel Meter} = \texttt{Half}) \\
&= \mathrm{P}(\texttt{Start} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \times \\
&\qquad \mathrm{P}(\texttt{Fuel} \mid \texttt{Fuel Meter} = \texttt{Half}) \times \\
&\qquad \cancel{\frac{\mathrm{P}(\texttt{Fuel.Meter} = \texttt{Half})}{\mathrm{P}(\texttt{Fuel.Meter} = \texttt{Half})}} \times \\
&\qquad \mathrm{P}(\texttt{Spark Plugs})
\end{aligned}$$



This leaves three variables, of which $\texttt{Start}$ is fixed to $\texttt{Yes}$: hence we have to consider $\mathrm{P}(\texttt{Start} = \texttt{Yes})$ under four scenarios:

$$\begin{aligned}
&\texttt{Fuel} = \texttt{Yes} \mid \texttt{Fuel Meter} = \texttt{Half}, \texttt{Spark Plugs} = \texttt{Work} \\
&\texttt{Fuel} = \texttt{Yes} \mid \texttt{Fuel Meter} = \texttt{Half}, \texttt{Spark Plugs} = \texttt{Fault} \\
&\texttt{Fuel} = \texttt{No} \mid \texttt{Fuel Meter} = \texttt{Half}, \texttt{Spark Plugs} = \texttt{Work} \\
&\texttt{Fuel} = \texttt{No} \mid \texttt{Fuel Meter} = \texttt{Half}, \texttt{Spark Plugs} = \texttt{Fault}
\end{aligned}$$

and sum the corresponding $\mathrm{P}(\texttt{Start} = \texttt{Yes} \mid \text{scenario})\, \mathrm{P}(\text{scenario})$.

Car Start: The Principled (Algorithmic) Way

Exhaustive enumeration obviously:

1. does not scale (try that with 20 variables!);
2. is only feasible in the first place if all variables are discrete.

Each of the steps in the previous slide corresponds to both

- an (probabilistic) application of Bayes theorem
- a (graphical) manipulation of arcs and nodes.

We can use the fact that arcs represent probabilistic associations to perform symbolic computations though graphical operations!

We can also threat this model like a hierarchical model and adapt the literature on Monte Carlo simulations.

Now we want to automate the whole process, so that the computer system itself will (ideally) do all the work.

A model that promises to do this is Bayesian networks (BNs):

- They combine graphs and probability as we did earlier, but in a rigorous fashion.
- There are automated reasoning algorithms for that use the graphical part of the model to guide a computer system in manipulating probability distributions, computing probabilities of and predicting events of interest.
- It is possible to learn them automatically from data.
- They can be used as causal models.
- As far as models, go they are very green: they recycle large amounts of results from classical statistics.

Bayesian Networks in R: The bnlearn Package

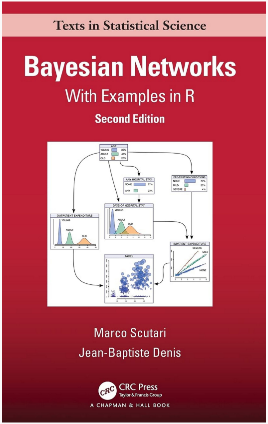
The most comprehensive R package for working with Bayesian networks is **bnlearn**, which you should install by

```r
install.packages("bnlearn")
```

The reference website for **bnlearn** is:

http://www.bnlearn.com

And there is a reference book too!



Bayesian networks (BNs) are defined by:

- a network structure, a directed acyclic graph $\mathcal{G}$, in which each node corresponds to a random variable X_i;
- a global probability distribution over $\mathbf{X} = \{X_1, \dots, X_N\}$ which can be factorised into smaller local probability distributions according to the arcs present in the graph.

The main role of the network structure is to express the conditional independence relationships among the variables in the model through graphical separation, thus specifying the factorisation of the global distribution:

$$\mathrm{P}(\mathbf{X}; \boldsymbol{\Theta}) = \prod_{i=1}^{N} \mathrm{P}(X_i \mid \Pi_{X_i}; \Theta_{X_i}) \quad \text{where} \quad \Pi_{X_i} = \{\text{parents of } X_i\}.$$

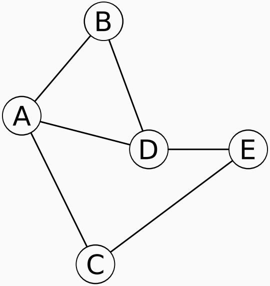
The first component of a BN is a graph. A graph $\mathcal{G}$ is a mathematical object with:

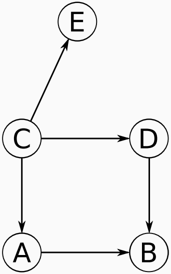
- a set of nodes;
- a set of arcs A which are identified by pairs for nodes.

Given the nodes, a graph is uniquely identified by the arc set. An arc can be:

- undirected if the arc has no direction, for instance $A - B$;
- directed if the arc has a specific direction, for instance $A \rightarrow B$.

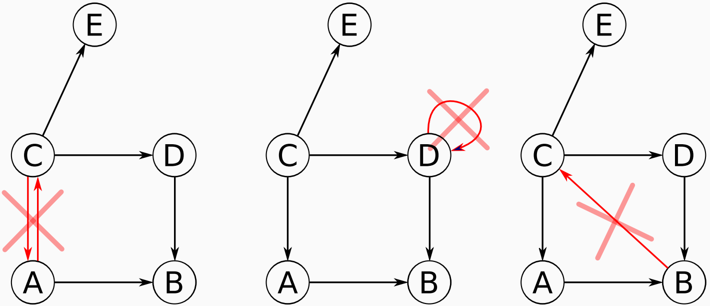
The assumption is that there is at most one arc between each pair of nodes.





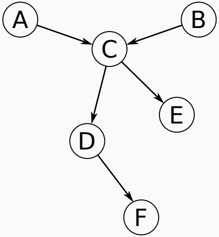
BNs use a specific kind of graph called a directed acyclic graph (DAG), that:

- contains only directed arcs;
- does not contain any loop (an arc $D \to D$ from a node to itself);
- does not contain any cycle (a sequence of arcs like $B \to C \to D \to B$ that starts and ends in the same node).



How the DAG Maps to the Probability Distribution

DAG



$\Longrightarrow$

Graphical separation

$A \perp\!\!\!\perp_G B$
$A \perp\!\!\!\perp_G D \mid C$
$B \perp\!\!\!\perp_G D \mid C$
$A \perp\!\!\!\perp_G E \mid C$
$B \perp\!\!\!\perp_G E \mid C$
$D \perp\!\!\!\perp_G E \mid C$
$C \perp\!\!\!\perp_G F \mid D$
$\ldots$

$\Longrightarrow$

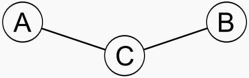
Probabilistic independence

$A \perp\!\!\!\perp_P B$
$A \perp\!\!\!\perp_P D \mid C$
$B \perp\!\!\!\perp_P D \mid C$
$A \perp\!\!\!\perp_P E \mid C$
$B \perp\!\!\!\perp_P E \mid C$
$D \perp\!\!\!\perp_P E \mid C$
$C \perp\!\!\!\perp_P F \mid D$
$\ldots$

Formally, the DAG is an independence map of the probability distribution of $\mathbf{X}$, with graphical separation ($\perp\!\!\!\perp_G$) implying probabilistic independence ($\perp\!\!\!\perp_P$).

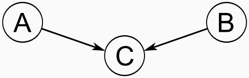
Graphical Separation in DAGs: Fundamental Connections

separation (undirected graphs)

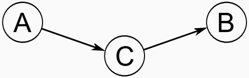


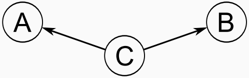
$$\mathbf{A} \perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \mathrm{P}(\mathbf{A} \mid \mathbf{C})\, \mathrm{P}(\mathbf{B} \mid \mathbf{C})\, \mathrm{P}(\mathbf{C})$$

d-separation (directed acyclic graphs)



$$\mathbf{A} \not\perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \mathrm{P}(\mathbf{C} \mid \mathbf{A}, \mathbf{B})\, \mathrm{P}(\mathbf{A})\, \mathrm{P}(\mathbf{B})$$





$$\mathbf{A} \perp\!\!\!\perp \mathbf{B} \mid \mathbf{C}$$
$$\begin{aligned} \mathrm{P}(\mathbf{A}, \mathbf{B}, \mathbf{C}) &= \\ &= \mathrm{P}(\mathbf{B} \mid \mathbf{C})\, \mathrm{P}(\mathbf{C} \mid \mathbf{A})\, \mathrm{P}(\mathbf{A}) \\ &= \mathrm{P}(\mathbf{A} \mid \mathbf{C})\, \mathrm{P}(\mathbf{B} \mid \mathbf{C})\, \mathrm{P}(\mathbf{C}) \end{aligned}$$

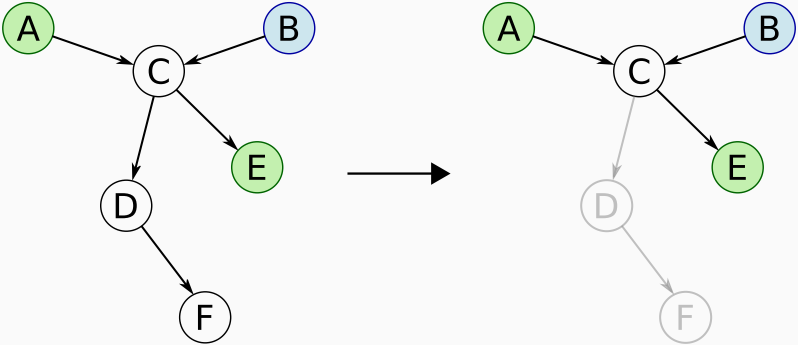
Graphical Separation in DAGs: General Case

Now, in the general case we can extend the patterns from the fundamental connections and apply them to every possible path between $\mathbf{A}$ and $\mathbf{B}$ for a given $\mathbf{C}$; this is how d-separation is defined.

> *If $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are three disjoint subsets of nodes in a directed acyclic graph $\mathcal{G}$, then $\mathbf{C}$ is said to d-separate $\mathbf{A}$ from $\mathbf{B}$, denoted $\mathbf{A} \perp\!\!\!\perp_G \mathbf{B} \mid \mathbf{C}$, if along every path between a node in $\mathbf{A}$ and a node in $\mathbf{B}$ there is a node v satisfying one of the following two conditions:*
>
> 1. *v has converging edges (that is, there are two edges pointing to v from the adjacent nodes in the path) and none of v or its descendants (that is, the nodes that can be reached from v) are in $\mathbf{C}$.*
> 2. *v is in $\mathbf{C}$ and does not have converging edges.*

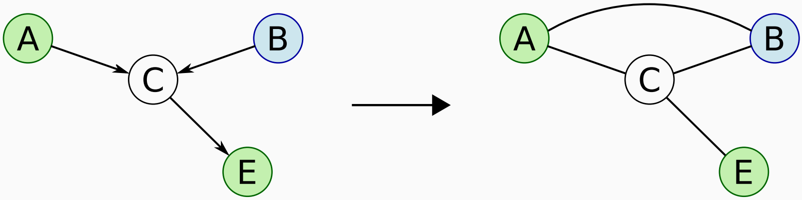
This definition clearly does not provide a computationally feasible approach to assess d-separation; but there are other ways.

A Simple Algorithm to Check D-Separation



Say that we want to check whether A and E are d-separated by B. First, we can drop all the nodes that are not ancestors (that is, parents, parents' parents, etc.) of A, E and B since each node only depends on its parents.

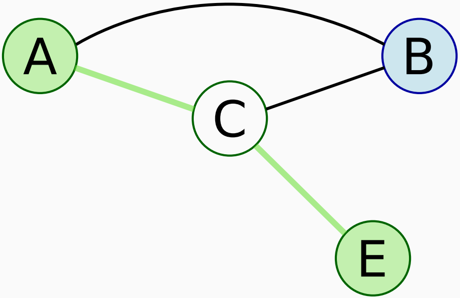
A Simple Algorithm to Check D-Separation



We then transform the subgraph into its moral graph by

1. connecting all the nodes that have one child in common; and
2. removing all arc directions to obtain an undirected graph.

This transformation makes the dependence between parents explicit by "marrying" them and of makes it possible for us to use the classic definition of graphical separation.



Finally, we can just perform a depth-first or breadth-first search and see if we can find an open path between A and E, that is, a path that is not blocked by B.

D-Separation Example: The DAG We Created Earlier

The last graph is an undirected graph: if there is a path from A to E there is a path from E to A. This means that d-separation is symmetric:

$$\mathsf{A} \not\perp\!\!\!\perp_G \mathsf{E} \mid \mathsf{B} \Longleftrightarrow \mathsf{E} \not\perp\!\!\!\perp_G \mathsf{A} \mid \mathsf{B}$$

Which must be the case because independence is also symmetric,

$$\mathrm{P}(\mathsf{A}, \mathsf{E} \mid \mathsf{B}) = \mathrm{P}(\mathsf{E}, \mathsf{A} \mid \mathsf{B}) \neq \mathrm{P}(\mathsf{A} \mid \mathsf{B})\, \mathrm{P}(\mathsf{E} \mid \mathsf{B}),$$

and d-separation implies probabilistic independence.

NOTE: d-separation does not necessarily require a separating set. Or, to put it in another way, the separating set can be empty. In that case we are checking whether variables are marginally independent because there is no path at all that connects them.

If we use d-separation as our definition of graphical separation, assuming that the DAG is an independence map leads to the general formulation of the decomposition of the global distribution

$$\mathrm{P}(\mathbf{X}) = \prod_{i=1}^{N} \mathrm{P}(X_i \mid \Pi_{X_i})$$

into the local distributions for the X_i given their parents Π_{X_i}. If X_i has two or more parents it depends on their joint distribution, because each pair of parents forms a convergent connection centred on X_i and we cannot establish their independence. This decomposition is preferable to that obtained from the chain rule,

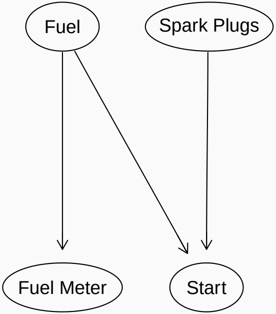
$$\mathrm{P}(\mathbf{X}) = \prod_{i=1}^{N} \mathrm{P}(X_i \mid X_{i+1}, \ldots, X_N),$$

because the conditioning sets are typically smaller.

Another result along the same lines is the local Markov property, which can be combined with the chain rule above to get the decomposition into local distributions.

> *Each node X_i is conditionally independent of its non-descendants (the nodes X_j for which there is no path from X_i to X_j) given its parents.*

Compared to the previous decomposition, it highlights the fact that parents are not completely independent from their children in the BN: a trivial application of Bayes' theorem to invert the direction of the conditioning shows how information on a child can change the distribution of the parent.



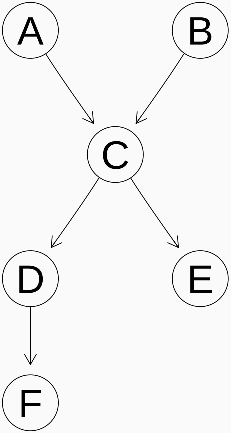
The parent sets:

$\texttt{Fuel} = \{\}$
$\texttt{Fuel Meter} = \{\texttt{Fuel}\}$
$\texttt{Spark Plugs} = \{\}$
$\texttt{Start} = \{\texttt{Fuel}, \texttt{Spark Plugs}\}$

The corresponding decomposition:

$$\begin{aligned} &\mathrm{P}(\texttt{Start}, \texttt{Fuel Meter}, \texttt{Fuel}, \\ &\qquad \texttt{Spark Plugs}) = \\ &\mathrm{P}(\texttt{Start} \mid \texttt{Fuel}, \texttt{Spark Plugs}) \\ &\mathrm{P}(\texttt{Fuel Meter} \mid \texttt{Fuel}) \times \\ &\qquad \mathrm{P}(\texttt{Fuel})\,\mathrm{P}(\texttt{Spark Plugs}) \end{aligned}$$

The Local Markov Property: The DAG We Created Earlier

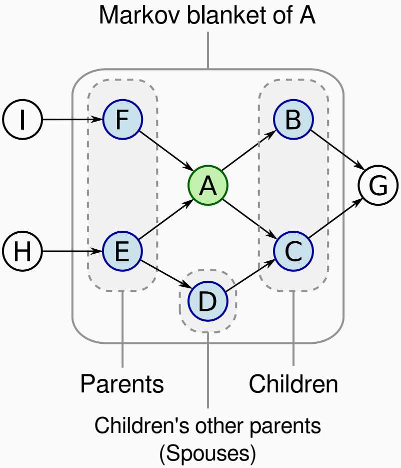


The parent sets:

$\mathsf{A} = \{\}$
$\mathsf{B} = \{\}$
$\mathsf{C} = \{\mathsf{A}, \mathsf{B}\}$
$\mathsf{D} = \{\mathsf{C}\}$
$\mathsf{E} = \{\mathsf{C}\}$
$\mathsf{F} = \{\mathsf{D}\}$

The corresponding decomposition:

$$\mathrm{P}(\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{F}) = \mathrm{P}(\mathsf{A})\,\mathrm{P}(\mathsf{B})\,\mathrm{P}(\mathsf{C} \mid \mathsf{A}, \mathsf{B})\,\mathrm{P}(\mathsf{D} \mid \mathsf{C})\,\mathrm{P}(\mathsf{E} \mid \mathsf{C})\,\mathrm{P}(\mathsf{F} \mid \mathsf{D})$$

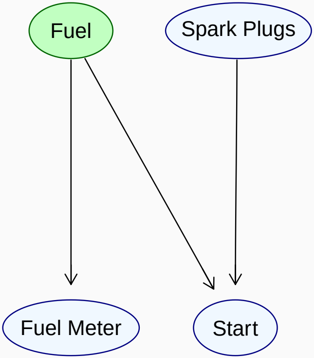


We can easily use the DAG to solve the feature selection problem. The set of nodes that graphically isolates a target node from the rest of the DAG is called its Markov blanket and includes:

- its parents;
- its children;
- other nodes sharing a child.

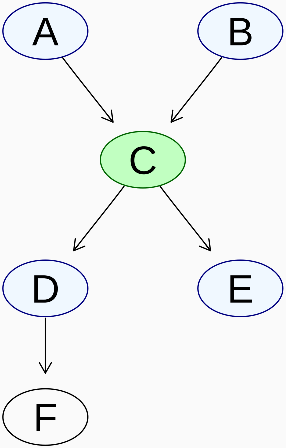
Since $\perp\!\!\!\perp_G$ implies $\perp\!\!\!\perp_P$, we can restrict ourselves to the Markov blanket to perform any kind of inference on the target node, and disregard the rest.

Markov Blanket: Car Start



The parents, children and spouses of `Fuel`:
{}
{`Fuel Meter`, `Start`}
{`Spark Plugs`}

The Markov blanket of `Fuel`:
{`Fuel Meter`, `Spark Plugs`, `Start`}



Printing the parents, children and spouses of `C`:

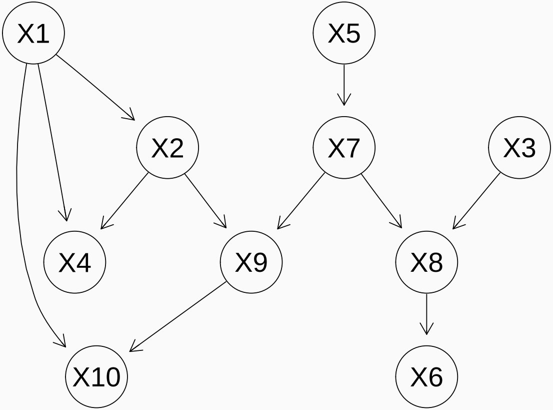
`{A, B}`
`{D, E}`
`{}`

The Markov blanket of `C`:

`{A, B, D, E}`

Different DAGs, Same Distribution

A DAG uniquely identifies a factorisation of $\mathrm{P}(\mathbf{X})$; the converse is not necessarily true. Consider this DAG:



Different DAGs, Same Distribution

The decomposition into local distributions is:

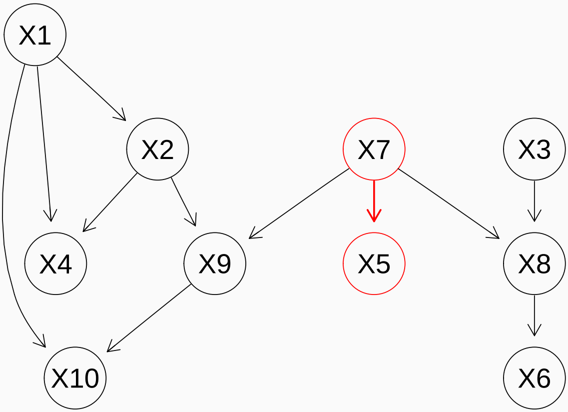
$$
\begin{aligned}
&\mathrm{P}(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}) = \\
&\quad \mathrm{P}(X_1)\,\mathrm{P}(X_3)\,\underbrace{\mathrm{P}(X_5)}_{X_5}\,\mathrm{P}(X_6 \mid X_8)\,\mathrm{P}(X_2 \mid X_1)\,\underbrace{\mathrm{P}(X_7 \mid X_5)}_{X_5 \to X_7} \\
&\quad \mathrm{P}(X_4 \mid X_1, X_2)\,\mathrm{P}(X_8 \mid X_3, X_7)\,\mathrm{P}(X_9 \mid X_2, X_7)\,\mathrm{P}(X_{10} \mid X_1, X_9).
\end{aligned}
$$

However, look at $X_5 \to X_7$: $\mathrm{P}(X_7 \mid X_5)\,\mathrm{P}(X_5) = \mathrm{P}(X_5 \mid X_7)\,\mathrm{P}(X_7)$ by Bayes' theorem. Then

$$
\begin{aligned}
&\mathrm{P}(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}) = \\
&\quad \mathrm{P}(X_1)\,\mathrm{P}(X_3)\,\underbrace{\mathrm{P}(X_7)}_{X_7}\,\mathrm{P}(X_6 \mid X_8)\,\mathrm{P}(X_2 \mid X_1)\,\underbrace{\mathrm{P}(X_5 \mid X_7)}_{X_7 \to X_5} \\
&\quad \mathrm{P}(X_4 \mid X_1, X_2)\,\mathrm{P}(X_8 \mid X_3, X_7)\,\mathrm{P}(X_9 \mid X_2, X_7)\,\mathrm{P}(X_{10} \mid X_1, X_9).
\end{aligned}
$$

Different DAGs, Same Distribution

The DAG that gives this new, equivalent decomposition is:



Next let's look at $X_8 \to X_6$.

$$\begin{aligned} \mathrm{P}(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}) = \\ \mathrm{P}(X_1)\,\mathrm{P}(X_3)\,\mathrm{P}(X_7)\,\underbrace{\mathrm{P}(X_6 \mid X_8)}_{X_8 \to X_6}\,\mathrm{P}(X_2 \mid X_1)\,\mathrm{P}(X_5 \mid X_7) \\ \mathrm{P}(X_4 \mid X_1, X_2)\,\underbrace{\mathrm{P}(X_8 \mid X_3, X_7)}_{X_8 \leftarrow X_3, X_8 \leftarrow X_7}\,\mathrm{P}(X_9 \mid X_2, X_7)\,\mathrm{P}(X_{10} \mid X_1, X_9). \end{aligned}$$

We cannot reverse the $X_8 \to X_6$ as we did with $X_5 \to X_7$ without changing the probability distribution. If we try, we get

$$\mathrm{P}(X_6 \mid X_8)\,\mathrm{P}(X_8 \mid X_3, X_7) = \mathrm{P}(X_8 \mid X_6)\,\mathrm{P}(X_6)\frac{\mathrm{P}(X_8 \mid X_3, X_7)}{\mathrm{P}(X_8)},$$

which does not simplify because X_8 has other parents (X_3, X_7).

Different DAGs, Same Distribution

Finally, let's look at X_1, X_2 and X_4.

$$
\begin{aligned}
&\mathrm{P}(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}) = \\
&\quad \underbrace{\mathrm{P}(X_1)}_{X_1} \mathrm{P}(X_3) \mathrm{P}(X_5) \mathrm{P}(X_6 \mid X_8) \underbrace{\mathrm{P}(X_2 \mid X_1)}_{X_1 \rightarrow X_2} \mathrm{P}(X_7 \mid X_5) \\
&\quad \underbrace{\mathrm{P}(X_4 \mid X_1, X_2)}_{X_1 \rightarrow X_4, X_2 \rightarrow X4} \mathrm{P}(X_8 \mid X_3, X_7) \mathrm{P}(X_9 \mid X_2, X_7) \mathrm{P}(X_{10} \mid X_1, X_9).
\end{aligned}
$$

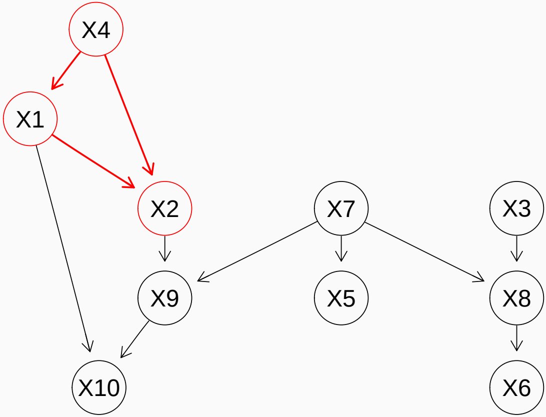
By Bayes' theorem we can say

$$
\begin{aligned}
\mathrm{P}(X_1) \mathrm{P}(X_2 \mid X_1) \mathrm{P}(X_4 \mid X_1, X_2) &= \mathrm{P}(X_1, X_2, X_4) = \\
&\underbrace{\mathrm{P}(X_4)}_{X_4} \underbrace{\mathrm{P}(X_2 \mid X_4)}_{X_4 \rightarrow X_2} \underbrace{\mathrm{P}(X_1 \mid X_2, X_4)}_{X_1 \leftarrow X_2, X_1 \leftarrow X_4}
\end{aligned}
$$

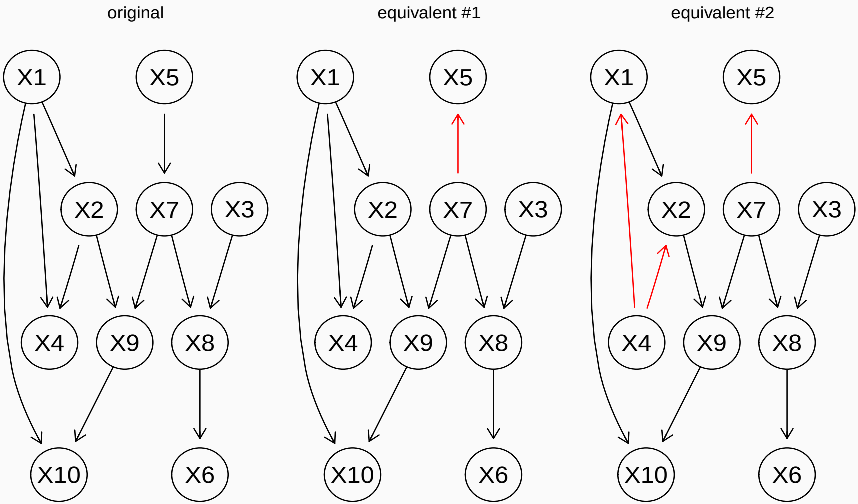
which gives us another DAG again.

Different DAGs, Same Distribution

The DAG that gives this last equivalent decomposition is:



Comparing These Different DAGs



To sum it up: we can reverse a number of arcs without changing the dependence structure of $\mathbf{X}$. Since the fundamental connections $A \rightarrow C \rightarrow B$ and $A \leftarrow C \rightarrow B$ are probabilistically equivalent, we can reverse the directions of their arcs as we like as long as we do not create any new v-structure ($A \rightarrow C \leftarrow B$, with no arc between A and B).

This means that we can group DAGs into equivalence classes that are uniquely identified by the underlying undirected graph and the v-structures. The directions of other arcs can be either:

- uniquely identifiable because one of the directions would introduce cycles or new v-structures (compelled arcs);
- completely undetermined.

The result is a completed partially directed graph (CPDAG).

It is important to note that even though $A \rightarrow C \leftarrow B$ is a convergent connection, it is not a v-structure if A and C are connected by $A \rightarrow B$ or $B \rightarrow A$. In that case, we are no longer able to identify which nodes are the parents in the connection.

For instance:

$$\underbrace{\mathrm{P}(A)\,\mathrm{P}(B \mid A)\,\mathrm{P}(C \mid A, B)}_{A \rightarrow C \leftarrow B,\; A \rightarrow B} = \mathrm{P}(A)\,\frac{\mathrm{P}(B, A)}{\mathrm{P}(A)}\,\frac{\mathrm{P}(C, A, B)}{\mathrm{P}(A, B)} =$$

$$= \mathrm{P}(A)\,\mathrm{P}(C, B \mid A) = \underbrace{\mathrm{P}(A)\,\mathrm{P}(B \mid C, A)\,\mathrm{P}(C \mid A)}_{C \rightarrow B \leftarrow A,\; A \rightarrow C}.$$

Therefore, the fact that the two parents in a v-structure are not connected is crucial in the identification of the correct CPDAG.

From this description we can tell different groups of arcs apart:

Directed arcs:

$$\begin{array}{ccc} X_8 & \to & X_6 \\ X_1 & \to & X_2 \\ X_5 & \to & X_7 \\ X_1 & \to & X_4 \\ X_2 & \to & X_4 \\ X_3 & \to & X_8 \\ X_7 & \to & X_8 \\ X_2 & \to & X_9 \\ X_7 & \to & X_9 \\ X_1 & \to & X_{10} \\ X_9 & \to & X_{10} \end{array}$$

Undirected arcs:

None.

Compelled arcs:

$$\begin{array}{ccr} X_1 & \to & X_{10} \\ X_2 & \to & X_9 \\ X_3 & \to & X_8 \\ X_7 & \to & X_8 \\ X_7 & \to & X_9 \\ X_8 & \to & X_6 \\ X_9 & \to & X_{10} \end{array}$$

V-structures:

$$\begin{array}{ccccc} X_1 & \to & X_{10} & \leftarrow & X_9 \\ X_3 & \to & X_8 & \leftarrow & X_7 \\ X_2 & \to & X_9 & \leftarrow & X_7 \end{array}$$

Which in the corresponding CPDAG become:

Directed arcs:

$$\begin{array}{ccc} X_1 & \rightarrow & X_{10} \\ X_2 & \rightarrow & X_9 \\ X_3 & \rightarrow & X_8 \\ X_7 & \rightarrow & X_8 \\ X_7 & \rightarrow & X_9 \\ X_8 & \rightarrow & X_6 \\ X_9 & \rightarrow & X_{10} \end{array}$$

Undirected arcs:

$$\begin{array}{ccc} X_1 & - & X_2 \\ X_1 & - & X_4 \\ X_2 & - & X_1 \\ X_2 & - & X_4 \\ X_4 & - & X_1 \\ X_4 & - & X_2 \\ X_5 & - & X_7 \\ X_7 & - & X_5 \end{array}$$

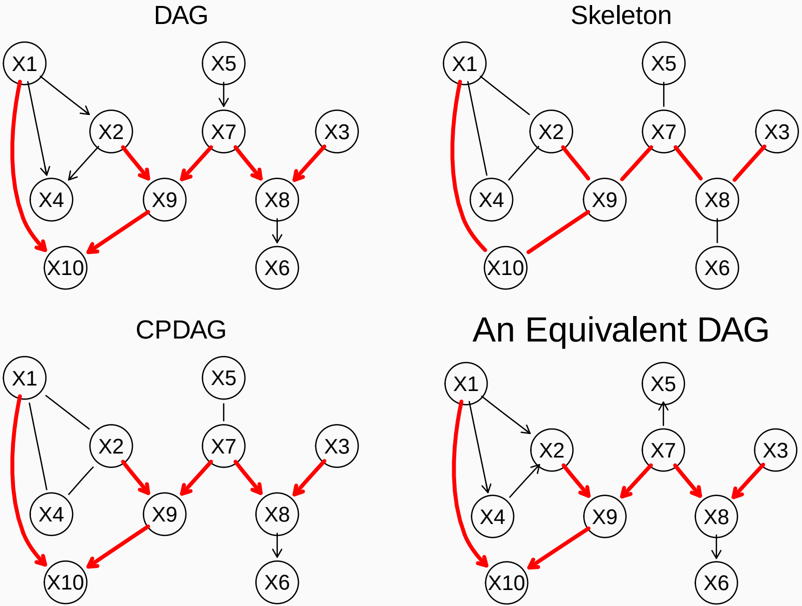
Compelled arcs:

$$\begin{array}{ccc} X_1 & \rightarrow & X_{10} \\ X_2 & \rightarrow & X_9 \\ X_3 & \rightarrow & X_8 \\ X_7 & \rightarrow & X_8 \\ X_7 & \rightarrow & X_9 \\ X_8 & \rightarrow & X_6 \\ X_9 & \rightarrow & X_{10} \end{array}$$

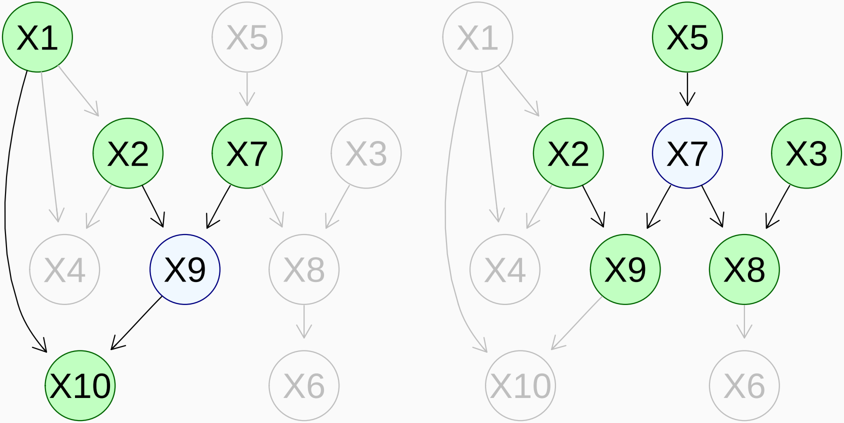
V-structures:

$$\begin{array}{ccccc} X_1 & \rightarrow & X_{10} & \leftarrow & X_9 \\ X_3 & \rightarrow & X_8 & \leftarrow & X_7 \\ X_2 & \rightarrow & X_9 & \leftarrow & X_7 \end{array}$$

DAG, CPDAG and Equivalent DAGs



Two More Examples of Markov Blankets



We can also check that Markov blankets are symmetric: if A is in the Markov blanket of B, then B is in the Markov blanket of A.

In which Markov blankets is X_9 in?

X1	X10	X2	X3	X4	X5	X6	X7	X8	X9
TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE

Which nodes are in the Markov blanket of X_9?

X1	X10	X2	X3	X4	X5	X6	X7	X8	X9
TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE

This is a consequence of the fact that if A is a parent of B, then B is a child of A; and if A is a spouse of B, then B is a spouse of A.

Relevant Functions in **bnlearn**

- creating DAGs: `empty.graph()`, `set.arc()`, `drop.arc()`, `reverse.arc()`.
- model string representations: `modelstring()`, `model2network()`.
- nodes in a DAG: `nodes()`, `parents()`, `children()`, `spouses()`, `nbr()`, `mb()`.
- arcs in a DAG: `arcs()`, `path.exists()`, `dsep()`, `directed.arcs()`, `undirected.arcs()`, `compelled.arcs()`.
- DAG transformation: `subgraph()`, `moral()`, `cpdag()`.
- plotting: `graphviz.plot()`, `graphviz.compare()`.

Summary and Remarks

- BNs are one of the oldest instances of machine learning models.
- BNs are a probabilistic model that use DAGs to make computations systematic in a rigorous way.
- BNs allow computer systems to perform automatically all the computations we did by hand at the beginning of this lecture.
- At the same time, BNs using DAGs means that they provide a qualitative, intuitive way to reason about complex phenomena.

Next:

- What probability distributions do we use to construct a BN?

Thanks!

Any questions?