9/18/2025

Probability

Axiosms of probablity

Axiom 1: Non-negativity

For any event A ∈ SA \in \mathcal SA ∈ S:

P(A) ≥ 0

Probability can never be negative.

Axiom 2: Normalization

The probability of the entire sample space is **1**:

P(Ω)=1

Something in the sample space must happen.

Axiom 3: Countable Additivity

For any countable sequence of **mutually disjoint** events A1,A2,A3,…A_1, A_2, A_3, \dotsA1 ,A2 ,A3 ,…:

$$P\left(\bigcup_{i=1}^{n} A_i\right) = \sum_{i=1}^{n} P(A_i), \quad \text{if } A_i \cap A_j = \emptyset \text{ for } i \neq j.$$

Symbol	Meaning	Example
Ω (Omega)	The **sample space –** the set of *all possible outcomes*	Ω = {1, 2, 3, 4, 5, 6}
S	The **event space –** a set of subsets of Ω that we assign probabilities to	S = {∅, {1}, {2}, … , {6}, {1,3,5}, {2,4,6}, Ω}
α	A specific **event** (i.e., one element of S)	α = {1, 3, 5} → "the die shows an odd number"

Sample space:

Event Space: ($\mathbf{S}$)

The event space **S** is the collection of all events to which you are willing to assign a probability.

If you want to assign a probability **only** to the event *"getting 3"*, you also need to be able to assign a probability to its **complement** (i.e. *"not getting 3"*).

The complement of {3} is {1,2,4,5,6}.

So the **smallest valid event space** in that case would be:

S = { ∅ , {3} , {1,2,4,5,6} , Ω }

TIP:

- **We include both the event and its complement so that the event space is closed under complements (one of the required properties).**
- **every event in the event space must have its complement in the event space.**

Konnor
Probability theory requires that the event space satisfy three basic properties: • It contains the *empty event* $\emptyset$, and the *trivial event* Ω. • It is closed under union. That is, if $\alpha, \beta \in \mathcal{S}$, then so is $\alpha \cup \beta$. • It is closed under complementation. That is, if $\alpha \in \mathcal{S}$, then so is $\Omega - \alpha$.

General rule for building an event space (also called a σ–algebra)

An event space **S** over a sample space **Ω** must satisfy **all three** of the following properties:

Property	Meaning
1. ∅ ∈ S	The **empty set** must be in the event space
2. Closed under	If α ∈ S, then **Ω** \ **α** must also be in S

Property	Meaning
complements	
3. Closed under **(countable) unions**	If $\alpha_1, \alpha_2, \alpha_3, \ldots \in S$ then $\alpha_1 \cup \alpha_2 \cup \alpha_3 \cup \ldots$ is also in S

In a **finite** sample space (like a die), "countable unions" simply means:If two events are in **S,** then their **union** is also in **S**.

So for a finite Ω you can simplify as:

$$\emptyset \in S$$

$$\alpha \in S \rightarrow \Omega \setminus \alpha \in S$$

$$\alpha, \beta \in S \rightarrow (\alpha \cup \beta) \in S$$

Probablity Distrubution (Probablity Density factor)

A probability distribution P over (Ω, S) is a mapping from events in S to real values that satisfies
probability Distribution

the following conditions:

- $P(\alpha) \geq 0$ for all $\alpha \in S$.
- $P(\Omega) = 1$.
- If $\alpha, \beta \in S$ and $\alpha \cap \beta = \emptyset$, then $P(\alpha \cup \beta) = P(\alpha) + P(\beta)$

Probability Definition: there are two types of probablity definition

Aspect	Frequentist	Subjective (Bayesian)
Meaning	Long-run frequency	Degree of belief
Nature	Objective	Subjective
Experiment	Needs repeatable trials	Can be one-time events
Example	P(rolling 3 on a die) = 1/6	P(rain tomorrow) = 0.7

8/18/2025

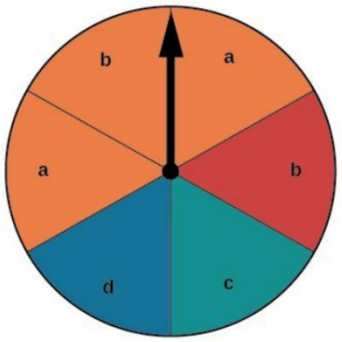
Addition rule of Probability

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

If the two events are mutually exclusive

$P(A \cap B) = 0$ therefor $P(A \cup B) = P(A) + P(B)$

Why subtract P(A n B) ?



There are a total of 6 sections, and 3 of them are orange. So the probability of spinning orange is 3/6=1/2. There are a total of 6 sections, and 2 of them have a b. So the probability of spinning a b is 2/6=1/3. If we added these two probabilities, we would be counting the sector that is both **orange** and b twice. To find the probability of spinning an orange or a b, we need to subtract the probability that the sector is both orange and has a b.

probability of disjoint union = sum of probabilities
(axiom 3 of probablity)

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i)$$

$$P(A) = \sum_{i=1}^{n} P(A \mid B_i)\, P(B_i)$$

General Rule (Inclusion–Exclusion Principle)

When you have more than one event, you need to be careful not to double-count the overlaps.
For **three** events, the probability of their union is:

$$P(A \cup B \cup C) = P(A) + P(B) + P(C) - P(A \cap B) - P(A \cap C) - P(B \cap C) + P(A \cap B \cap C)$$

Solution:

P(A u B u C) = P (A u (B u C))

P (A u (B u C)) = P(A) + P(BuC) - P(A n (B u C))

But P (BuC) = P(B) + P(C) - P(B n C)

:- P (A u (B u C)) = P(A) + P(B) + P(C) - P(B n C) - P(A n (B u C))

But P(A n (B u C)) = P((A n B) u (A n C))

P((A n B) u (A n C)) = P(A n B) + P(A n C) - P((A n B) n (A n C))

But: P((A n B) n (A n C)) = P(A n B n C)

:- P (A u (B u C)) = P(A) + P(B) + P(C) - P(B n C) - (P(A n B) + P(A n C) - P((A n B) n (A n C)))

:- P (A u (B u C)) = P(A) + P(B) + P(C) - P(B n C) - P(A n B) - P(A n C) + P(A n B n C))

Conditional Probability:

1. A ⊥ B | C = ?

(read: *"A is independent of B* **given C**"), it means that **once we know C**, information about B doesn't change the probability of A anymore.

Solution:

Normally, for any events (without independence):

$$P(A \cap B|C) = P(A|B,C)\ P(B|C)$$ # P(A|B,C) equivalent to P(A|B n C)

But **conditional independence** tells us that **A doesn't depend on B once C is known**, so: NB (if B does not depend on A the P(A | B) = P(A))

$$P(A|B,C) = P(A|C)\ P(A \mid B, C)$$

Plugging this into the general rule gives:

$$P(A \cap B|C) = P(A|C)\ P(B|C)$$

Some fundamental rules of probability:

- Conditional: $p(X \mid Y) = \frac{p(X,Y)}{p(Y)} = \frac{p(X,Y)}{\sum_x p(X=x,Y)}$
- Law of total probability: $p(Y) = \sum_x p(Y, X = x)$
- Probability chain rule: $p(X, Y) = p(Y)p(X \mid Y)$

Question:

This is A more general conditional version of Bayes' rule, where all our probabilities are conditioned on

some background event γ, also holds

Proof the left side is true

$$P(\alpha \mid \beta \cap \gamma) = \frac{P(\beta \mid \alpha \cap \gamma)P(\alpha \mid \gamma)}{P(\beta \mid \gamma)}$$

$P(\alpha \mid \beta \cap \gamma) = P(\alpha \cap \beta \cap \gamma) = P(\beta \cap \alpha \cap \gamma)$

$$\mathbf{P(\alpha \mid \beta \cap \gamma) = \frac{P(\beta \cap \alpha \cap \gamma)}{P(\beta \cap \gamma)}}$$

So lets take the $P(\beta \cap \alpha \cap \gamma)$

$P(\beta \cap \alpha \cap \gamma) = P(\beta \mid \alpha \cap \gamma)\, P(\alpha \cap \gamma)$

But $P(\alpha \cap \gamma) = P(\alpha \mid \gamma)\, P(\gamma)$

:- $\mathbf{P(\beta \cap \alpha \cap \gamma) = P(\beta \mid \alpha \cap \gamma)P(\alpha \mid \gamma)\, P(\gamma)}$

Now lets take the $P(\beta \mid \gamma)$

$\mathbf{P(\beta \cap \gamma) = P(\beta \mid \gamma)P(\gamma)}$

Now Plug both inside the original question

$$P(\alpha \mid \beta \cap \gamma) = \frac{\mathbf{P(\beta \mid \alpha \cap \gamma)P(\alpha \mid \gamma)\, P(\gamma)}}{\mathbf{P(\beta \mid \gamma)P(\gamma)}}$$

$$\mathbf{P(\alpha \mid \beta \cap \gamma) = \frac{P(\beta \mid \alpha \cap \gamma)P(\alpha \mid \gamma)}{P(\beta \mid \gamma)}}$$

8/20/2025

Random Variables

$$\sum_{i=1}^{k} P(X = x_i) = 1$$

Random variables depend on **chance or outcomes of an experiment**, while non-random variables are **fixed, constant, or deterministic**.

Intuition

Suppose you want the probability that someone **likes coffee given they live in Ethiopia**: (RV likeCoffe = ture or false)

P(like coffee | Ethiopia) = P(like coffe n Ethiopia) / P(Ethiopia) =>

P(Ethiopia) = $\sum_{i=1}^{k} P(likeCoffe = x_i, Ethiopia) = 1$

(Probability of Ethiopia for every possible values of liking coffee)
P(Ethiopia) = P(like coffe n Ethiopia) + P(doesn't like cofee n Ethiopia)

Two ways to classify random variables

1. By type of values they take:

1.1 Categorical (qualitative): Values are labels or categories

Example: Eye color = {Brown, Blue, Green}

1.2 Numerical / Real-valued (quantitative): Values are numbers

Example: Height = 1.65 m, 1.72 m

2. By how many values they can take:

2.1 Discrete: Can count the values (1, 2, 3, ...)

Example: Number of students in a class

2.2 Continuous: Can take any value in an interval (infinite possibilities)

Example: Weight = 50.2 kg, 50.25 kg, 50.251 kg ...

$$\sum_{i=1}^{k} P(X = x_i) = 1$$

Multinomial distribution

- Describes the number of times **each of several categories** occurs in *n* independent trials
- Example: Roll a 3-sided die 10 times → count how many times **1**, how many times **2**, and how many times **3** appear
- Random variable gives a **vector of counts**, one for each category

Think of it as **repeating a multi-category trial multiple times and counting how often each category happens**

Bernoulli distribution

Describes **a single trial** with **two possible outcomes**

Example: Toss one coin → Head (1) or Tail (0)

Random variable takes **only one value** (0 or 1)

Think of it as **one yes/no experiment**

Binomial distribution

Describes the **total number of successes in *n* independent Bernoulli trials**

Example: Toss a coin **10 times** → count how many Heads appear

Random variable takes values from **0 up to n**

Think of it as **repeating the Bernoulli experiment multiple times and counting how many "successes" you get**

This not Explains Koller page 21, clearly

X is a set containing random variables

x = {w1, w2}

x = {w1 = rainy, w2 = sunny}

Val(X) = all possible values for the random variables (rainy, sunny, stormy, foggy)

Y ⊆ X, Y is a subset of X

Y = {w1}+

x⟨Y⟩ takes the assignment x (value of x) and keeps only the values of Y

∴ **X** = the random variable (or set of variables)

∴ **Val(X)** = all *possible* assignments that X can take

∴ **x** ∈ **Val(X)** → **x** is *one specific assignment* (one possible value) of X

For two assignments **x** (over variables X) and **y** (over variables Y), we write **x ∼ y** if they **agree on the variables they have in common**.

That is: $x\langle X \cap Y\rangle = y\langle X \cap Y\rangle$

→ **Both assignments give the same values to the variables that are in the intersection** of X and Y.

1. X and Y are sets of random variables

X and Y are **not necessarily independent** — they are just sets of variables.

For example:

X = {Weather forecast from Station 1}

Y = {Weather forecast from Station 2}

2. x and y are assignments

x = one particular forecast from X (e.g., "rainy")

y = one particular forecast from Y (e.g., "rainy")

3. x ∼ y

The notation **x ∼ y** just means:

"x and y agree on the variables they share in common (X ∩ Y)"

In your TV station example:

Suppose both X and Y include "today's weather"

Then X ∩ Y = {today's weather}

$x\langle X \cap Y\rangle$ = "rainy" (Station 1)

$y\langle X \cap Y\rangle$ = "rainy" (Station 2)

Since they match → x ∼ y ✓

8/21/2025

⊨ is called the **semantic entailment symbol** (sometimes read as "models" or "entails").

In probability and logic, it means:

> **$P \models (\alpha \perp \beta \mid \gamma)$**
> → "In the probability distribution P, it is true that α is conditionally independent of β given γ."

So, ⊨ is like saying **"according to the model/distribution P, this statement holds."**

In logic:
M⊨φ means *the model M satisfies (or makes true) the formula φ.*

Expectation

Expectation

Defnition 1: is a weighted average of all possible outcomes.
If some outcomes are more likely, they "pull" the expectation toward them.

Defnition 2: if you randomly sample from a distribution what would you think "**average of the samples will likely be"**

8-5-2025

1. Representation (How we describe probability distributions with graphs)

Bayesian networks (BNs):

Definitions – A BN is a probabilistic model that represents variables and their conditional dependencies using a **directed acyclic graph (DAG)**.

Directed graphs – Nodes = random variables, Edges = direct causal/conditional relationships.

Independencies – BNs encode conditional independencies: a variable is independent of its non-descendants given its parents.

Markov Random Fields (MRFs):

Undirected vs directed models – Unlike BNs, MRFs use **undirected graphs** to represent mutual dependencies (good for things like image segmentation where relationships are symmetric).

Independencies – MRFs use the **Markov property**: a node is independent of others given its neighbors.

Conditional Random Fields (CRFs) – A type of MRF used for prediction tasks; models the conditional distribution P(Y|X)P(Y | X)P(Y|X), common in NLP and computer vision.

---

2. Inference (How we answer probability questions with these models)

Variable elimination

The **inference problem**: Given a graphical model, compute probabilities like marginals or conditionals.

Variable elimination: Systematically eliminate variables by summing them out.

Complexity: Depends on graph structure (can be exponential in the worst case).

Belief propagation

Junction tree algorithm – Convert graph into a tree-like structure to do exact inference efficiently.

Exact inference – Works for graphs without cycles.

Loopy belief propagation – Apply the same idea even with cycles (approximate, but works well in practice).

MAP inference (Maximum a Posteriori)

Find the **most probable assignment** of variables, not just probabilities.

Max-sum message passing – A variant of belief propagation for MAP.

Graph cuts – Efficient algorithm for MAP in vision problems.

Linear programming relaxations & dual decomposition – Advanced optimization techniques.

Sampling-based inference

Instead of exact math, **simulate samples** to estimate probabilities.

Monte Carlo sampling – General random sampling.

Forward sampling – Generate from the model step by step.

Rejection sampling – Generate then reject inconsistent samples.

Importance sampling – Weight samples for efficiency.

Markov Chain Monte Carlo (MCMC) – Build dependent samples that converge to the true distribution.

Applications – Useful when exact inference is impossible.

Variational inference

Approximate a complicated distribution with a simpler one.

Variational lower bounds – Replace hard integrals with optimization problems.

Mean Field – Assume independence between variables for tractability.

Marginal polytope relaxations – Approximation techniques for complex dependencies.

---

3. Learning (How we estimate model parameters/structure from data)

Directed models (Bayesian networks)

Maximum likelihood estimation (MLE) – Fit parameters to maximize likelihood of data.

Learning theory basics – Generalization, bias-variance tradeoff.

MLE for BNs – Straightforward if graph structure is known.

Undirected models (MRFs/CRFs)

Exponential families – Common parametric forms (e.g., logistic regression, CRFs).

MLE with gradient descent – Need iterative optimization (partition function is tricky).

Learning in CRFs – Estimate parameters for conditional models.

Latent variable models

Latent variables – Hidden/unobserved factors (e.g., clusters).

Gaussian Mixture Models (GMMs) – Example of a latent variable model.

Expectation Maximization (EM) – Standard algorithm for learning with hidden variables.

Bayesian learning

Bayesian paradigm – Treat parameters as random variables with priors.

Conjugate priors – Priors that make posterior math easy.

Examples – Normal-Normal, Beta-Binomial, etc.

Structure learning

Goal: Learn the **graph structure** (not just parameters) from data.

Chow-Liu algorithm – Efficient method for tree-structured networks.

AIC / BIC – Model selection criteria that trade off fit vs complexity.

Bayesian structure learning – Uses Bayesian methods to score possible structures.

Representation			
Graphical Structural Representation		Quantitative Representation	
baysian nw	marcove nw		

8-30-2025

Bayesian Network

The Scenario (Example)

Imagine we have a much simpler medical world with just **3 binary variables**:

1. HasFlu (F)
2. HasFever ®
3. Coughs (C)

And our simple Bayesian Network has this structure:
HasFlu → HasFever
HasFlu → Coughs

This means HasFlu is the parent of
both HasFever and Coughs.

Our Tiny "dataset.dat"

Let's say our training data has only **10 patient records**. Each number represents a patient's full state (like in your assignment).

[3, 1, 3, 0, 3, 1, 0, 1, 3, 0]

NB: Here the variables are represented using a 3 bit binary digit.

We need to convert these to binary to see the symptoms. Remember, the integer is encoded as a binary number where the rightmost bit is HasFlu.

Integer	Binary (F, R, C)	Meaning
0	000	**No Flu**, No Fever, No Cough
1	001	**Has Flu**, No Fever, No Cough

Integer	Binary (F, R, C)	Meaning
3	011	**Has Flu**, No Fever, **Has Cough**
0	000	**No Flu**, No Fever, No Cough
3	011	**Has Flu**, No Fever, **Has Cough**
1	001	**Has Flu**, No Fever, No Cough
0	000	**No Flu**, No Fever, No Cough
1	001	**Has Flu**, No Fever, No Cough
3	011	**Has Flu**, No Fever, **Has Cough**
0	000	**No Flu**, No Fever, No Cough

(Note: In this tiny example, HasFever never occurs, which is unrealistic but helps show how smoothing works.)

The "Empty CPTs" We Need to Fill

Our model has three variables. Their CPTs will look like this before learning:

1. CPT for HasFlu (it has no parents, so it's simple)

P(HasFlu = true)	
?	

2. CPT for HasFever (parent: HasFlu)

HasFlu	P(HasFever = true | HasFlu)
false	?
true	?

3. CPT for Coughs (parent: HasFlu)

HasFlu	P(Coughs = true | HasFlu)
false	?
true	?

Parameter Learning: Calculating the Numbers

Now, we use the data to calculate the ? values.

1. Learn P(HasFlu = true)

Count total patients: 10

Count patients with Flu (HasFlu=1)**:** Let's find all integers that have the first bit set: 1, 3, 1, 3, 1, 3. That's **6 patients**.

Apply Laplace Smoothing:

P(F = true) = (Count(F=true) + 1) / (Total Patients + 2) = (6 + 1) / (10 + 2) = 7/12 ≈ 0.583

Calculated Number for CPT: 0.583

2. Learn P(HasFever = true | HasFlu)

This has two rows. We calculate each separately.

Row 1: Given HasFlu = false **(0)**

- **Count patients with** HasFlu = false**:** The patients with integer 0. This happened **4 times**.
- **From those 4, count how many have** HasFever = true**:** Look at the binary for 0: 000. The fever bit is 0. This happened **0 times**.
- **Apply Smoothing:**

  P(R=true | F=false) = (0 + 1) / (4 + 2) = 1/6 ≈ 0.167

Row 2: Given HasFlu = true **(1)**

Count patients with `HasFlu = true`**:** We already know this is **6**.

-
-

From those 6, count how many have `HasFever = true`**:** Look at the data. Patients with flu are integers 1 and 3. Their binary is `001` and `011`. The fever bit is always 0. This happened **0 times**.

-
-

Apply Smoothing:
`P(R=true | F=true) = (0 + 1) / (6 + 2) = 1/8 = 0.125`

-

Calculated Numbers for CPT: `0.167` and `0.125`

3. Learn P(Coughs = true | HasFlu)

Again, two rows.

Row 1: Given `HasFlu = false` **(0)**

-

Count patients with `HasFlu = false`**: 4 times** (integer 0).

-
-

From those 4, count how many have `Coughs = true`**:** Look at the binary for 0: `000`. The cough bit is 0. This happened **0 times**.

-
-

Apply Smoothing:
`P(C=true | F=false) = (0 + 1) / (4 + 2) = 1/6 ≈ 0.167`

-

Row 2: Given `HasFlu = true` **(1)**

-

Count patients with `HasFlu = true`**: 6 times**.

o
o

From those 6, count how many have Coughs = true**:** Patients with flu are integers 1 and 3. Integer 1 (001) has no cough. Integer 3 (011) has a cough. So cough happened **3 times** (for each integer 3 in the list).

o
o

Apply Smoothing:
P(C=true | F=true) = (3 + 1) / (6 + 2) = 4/8 = 0.5

o

Calculated Numbers for
CPT: 0.167 and 0.5

The Final, Learned CPTs

After parameter learning, our previously empty CPTs are now filled with these **calculated numbers**:

1. CPT for HasFlu

P(HasFlu = true)
0.583

2. CPT for HasFever

HasFlu	P(HasFever = true | HasFlu)
false	0.167
true	0.125

3. CPT for Coughs

HasFlu	P(Coughs = true | HasFlu)
false	0.167
true	0.5

These numbers are the **parameters** of your model. They are all calculated directly from the data in dataset.dat using counting and Laplace smoothing. This entire process is **parameter learning**.

Representation

- the first task when drawing graphs or Bayesian NW

BN (Bayesian network):-

. indicating the direction of influence or causality (from cause to effect)

. edges are directed from cause --> effect

. edges are acyclic (no loops allowed)

. BN assums most relationships are conditionally independent with each other (that is why it reduces the complextity of having multiple dimensions since it leaves out independent variables and represents RVs with meaningfull relationship)

Example: Flu --> Caugh --> fatigue

What is Image Segmentation?

Image segmentation = dividing an image into meaningful parts (regions, objects, or boundaries).
Instead of treating every pixel separately, segmentation groups pixels with similar properties (color, texture, intensity, etc.) so we can analyze objects in the image.

The segmentation is then found by minimizing an **energy function** combining:

Unary costs → how well a pixel fits a label.

Pairwise costs → how smooth/consistent neighboring labels are.

When to Avoid MRF and Use CRF Instead?

Use CRF if you have labeled data for training, complex data (e.g., noisy Sentinel-2 images), or need to incorporate multiple features (e.g., color, texture, elevation) to improve accuracy.

TIP

Marginal Independence

Two variables **X** and **Y** are **marginally independent** if knowing one does **not** change the probability of the other.

Outfit and **Happiness** are both influenced by Weather.

Without knowing the Weather:

If you see someone wearing a raincoat (Outfit = raincoat), you can guess it's rainy, which also means lower Happiness.

So Outfit and Happiness are **dependent**.

$P(\text{Outfit},\text{Happiness}) \neq P(\text{Outfit}) \cdot P(\text{Happiness})$

Conditional Independence

Two variables **X** and **Y** are **conditionally independent given Z** if:

$$P(X,Y|Z)=P(X|Z)\cdot P(Y|Z)$$

This means: once we know **Z**, knowing **X** gives us no extra info about **Y**.

joint probability distribution (JPD) is the probability of all variables taking specific values at the same time.

Marginal probability = overall probability of observing today's satellite images (regardless of the true

weather). Probability of a variable regardless of others (can be computed from joint probabilities). probability of a variable, possibly estimated from data or computed from the model.

Prior = your initial guess about tomorrow's weather (say, 70% sunny, 30% rainy).
Likelihood = how well today's satellite images match each possibility.
Posterior = updated belief after combining prior + likelihood.

PDF as a **curve of likelihood** — the area under the curve between two points gives the probability.

· Factors / Potentials → score combinations of variables (like a table).
· A factor (or potential function) is a table or function that assigns a "score" to each combination of variable values.
· Purpose: Measures how compatible a combination of variables is. High score → more likely, low score → less likely.
· Cliques → groups of variables that are fully connected; each clique gets a factor.

: A clique is a subset of nodes in a graph where every node is connected to every other node in that subset.
joint probability distribution (JPD) is the probability of all variables taking specific values at the same time.

The core idea of PGMs is: instead of storing the giant joint probability table directly (which grows exponentially), we factorize it using the graph structure

==========

9-10-2025

INFRENCE

Variable Elimination: A method for exact inference in probabilistic graphicalmodels (e.g., Bayesian networks). It computes marginal probabilities by systematically eliminating variables one by one, summing out their contributions using factor operations (e.g., multiplication and summation). Efficient for sparse networks but can be computationally expensive with many variables.

Sum-Product / Max-Product Algorithm: A message-passing algorithm used in factor graphs for inference. The Sum-Product variant computes marginal probabilities by passing messages between nodes to sum out variables, while Max-Product finds the most probable configuration (MAP estimation) by maximizing over variables. Both are exact on trees but approximate on loopy graphs.

Junction Tree Algorithm: An exact inference algorithm for Bayesian networks, transforming the model into a junction tree (a tree of cliques) to handle loops. It uses message passing between cliques to compute marginals efficiently, making it suitable for complex, loopy networks like yours (A → C ← B → D).

Loopy Belief Propagation: An approximate inference method extending belief propagation to graphs with loops. It iteratively passes messages between nodes, even in cyclic structures, to estimate marginal probabilities. While not guaranteed to converge or be exact, it's computationally efficient and often works well in practice (e.g., in error-correcting codes or neural networks).

3.3s

what are these or what do they mean 1. factor graphs 2. MAP estimation 3. exact inference method 4. approximate inference method 5. message passing

1. **Factor Graphs**: A bipartite graphical representation of a probabilistic model, consisting of variable nodes and factor nodes. Factor nodes represent functions (e.g., probability distributions) that depend on connected variable nodes. They're used to visualize and perform inference in models like Bayesian networks or Markov random fields, making dependencies and computations clearer.
2. **MAP Estimation**: Stands for Maximum A Posteriori estimation. It's a method to find the most probable configuration of variables in a probabilistic model given evidence, by maximizing the posterior probability P(variables|evidence) P(\text{variables} | \text{evidence}) P(variables|evidence). It's widely used in

decision-making or prediction tasks, differing from marginal inference by focusing on a single best outcome.

3. **Exact Inference Method**: A technique that computes the exact probability distributions (e.g., marginals or conditionals) in a probabilistic model, such as variable elimination or the junction tree algorithm. It guarantees accurate results but can be computationally intensive, especially for large or loopy networks, often requiring exponential time in the worst case.
4. **Approximate Inference Method**: A technique that estimates probability distributions when exact methods are infeasible, using approximations like sampling (e.g., Monte Carlo) or iterative methods (e.g., loopy belief propagation). It's faster and scalable but may introduce errors, making it suitable for complex models with many variables.
5. **Message Passing**: A general inference strategy where nodes in a graphical model (e.g., factor graphs or junction trees) exchange information (messages) to update beliefs or probabilities. Used in algorithms like sum-product or belief propagation, it enables distributed computation and is key to both exact and approximate inference.

---

1. Chain (Head-to-Tail) (Causal or Evidence Trail)

Note: parent = cause , child = Évidence

A→B→C

Information flows along the chain.

Rule:

Without conditioning: A and C are dependent.

With conditioning on B: A⊥C|B

Example:

A=Rain → B=Clouds → C=Wet ground.

Rain and Wet ground are dependent, but if we know about Clouds, then Rain doesn't add extra info about Wet ground.

2- Fork (Tail-to-Tail) (Common Cause) (common Parent)

Note: parent = cause , child = Evidence

$$A \leftarrow B \rightarrow C$$

One parent causes two children.

Rule:

Without conditioning: A and C are dependent.

With conditioning on B: $A \perp C|B$

Example:

B=Genetic mutation

A=Heart disease

C=Diabetes.

Heart disease ←Genetic mutation →Diabetes.

Heart disease and Diabetes appear related, but once we know the mutation status, they're independent.

---

3. Collider (Head-to-Head) (Common Effect) (Decendants)

Note: parent = cause , child = Evidence

$$A \rightarrow C \leftarrow B$$

Two causes meet at a common effect.

Rule:

Without conditioning: $A \perp B$

With conditioning on C: A not $\perp$ B|C

Example (Explaining Away):

A=Burglary

B=Earthquake

C=Alarm.

Burglary --> Alarm. <-- Earthquake

Burglary and Earthquake are independent.

But if the Alarm goes off, and we know there was a Burglary, the probability of an Earthquake goes down (and vice versa).

Structure	Graph	Independence rule	Example
Head-to-Tail (Chain)	a→c→b	a⊥b|c	Rain → Clouds → Wet ground
Tail-to-Tail (Fork)	a←c→b	a⊥b|c	Mutation → Heart disease → Diabetes
Head-to-Head (Collider)	a→c←b	a⊥b, a not⊥ b|c	Burglary → Alarm ← Earthquake

Latent Variables

(also called Hidden Variables)

- A **latent variable** is a variable that **we don't directly observe** in the data.Instead, it's **inferred indirectly** from the relationships between the observed variables.
- They often represent **underlying causes** or **unmeasured factors**.

Psychology (IQ tests)

Observed: test scores in math, reading, logic.

Latent: "intelligence" (not directly measured, only inferred).

Explaining away" is one of the most **interesting Phenomena in Bayesian Networks,** and it happens in the **Head-to-Head (collider) structure**:

A: Burglary

B: Earthquake

C: Alarm goes off

Without evidence (alarm not observed):
Burglary and Earthquake are independent — one doesn't affect the other.

If alarm = ON:

If you learn that a **burglary happened**, that **explains the alarm** → the probability of an earthquake being the cause **goes down**.

If no burglary, then the earthquake becomes a more likely explanation.

Both causes compete to "explain" the effect → **explaining away**.

We move away from burglary to earthquake that is why it is called explaining away

Active Path and Blocked Path

· If we **don't observe WetGround** → Rain and Sprinkler are independent (path is blocked).

· If we **observe WetGround** → The path becomes active → Rain and Sprinkler become dependent (explaining away: "If WetGround is wet, was it Rain or Sprinkler?").

Season ← Weather → FluCases

Active path =

. information flows, variables can affect each other,

. children (Season, FluCases) are dependent.

. If you don't know the Weather: "If it's winter (Season), flu cases are probably high." (Season gives info about FluCases).

Blocked path =

. information doesn't flow, variables are independent given the evidence,

. children become independent given the cause.

.If you do know the Weather: “If Weather = cold, then FluCases are high, but Season adds no new info.” (Season is no longer useful once Weather is known).

D-separation (Directional Separation) is a criterion used to decide whether two sets of variables are **independent** given some observed variables (evidence).

Two variables (say X and Y) are **d-separated** by a set of nodes Z if **all paths** between X and Y are **blocked** once we consider Z.

If there is at least one **active path**, then XXX and YYY are **dependent** (not d-separated).

f PDF as a **curve of likelihood** — the area under the curve between two points gives the probability.