**Probability for Machine Learning:**

**Random Variables:**

A random variable is a variable that takes values based on the outcome of a random experiment.

* **Discrete variables:** takes countable values (e.g., number of heads in 10 coins tosses, dice roll 1–6).
* **Continuous variables:** takes any value in a range (e.g., height, weight, temperature).

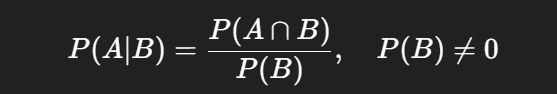
**Probability Distributions:**

* **Normal/Gaussian:** Bell curve, continuous. Most natural data (like height, errors) follow this.
* **Bernoulli:** One trial, 0 or 1 outcome (success/failure).
* **Binomial:** Repeated Bernoulli trials, counts successes (e.g., 5 heads in 10 tosses).
* **Poisson:** Discrete, counts events in fixed time/space. (Number of emails received in an hour.)
* **Uniform:** All outcomes equally likely (Picking a random card from a well-shuffled deck)
* **Exponential:** Models the time until an event happens (waiting time between events) (Time until the next customer arrives at a shop)

**Conditional Probability:**

Conditional Probability is the probability of an event happening given that another event has already occurred.

* **Notation:** *P(A∣B)* → Probability of A happening given B happened.
* **Formula:**



### **Example:**

Probability it rains today (A) given that it is cloudy (B).

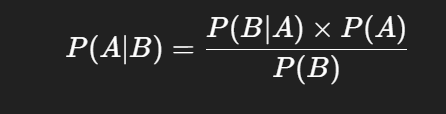
If it’s cloudy, the chance of rain is higher than usual.

* **Machine Learning:**

Used in Naive Bayes, where we calculate probability of a class given feature.

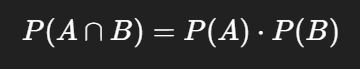
**Bayes Theorem:**

It tells us how likely something is, given that we know some related fact. Bayes’ Theorem is just a formula to calculate conditional probability in a smart way, using prior knowledge.



**Independence & dependence of events:**

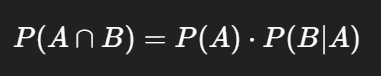
* **Independence:** Two events don’t affect each other.



**Example:**

Tossing a coin and rolling a die. The coin result doesn’t change the die result.

* **Dependence:** Two events affect each other.



**Example:**

Drawing two cards from a deck without replacement. The first card affects the probability of the second.

**Expectation (Mean):**

* Think of it as the average outcome of a random variable.
* If you repeat an experiment many times, the mean is what you expect on average.

**Variance:**

Measures how spread out the values are from the mean.

### **Law of Large Numbers (LLN):**

* **Idea:** If you repeat an experiment many times, the average of the results gets closer to the expected value (mean).
* **Example:**
  + Toss a fair coin 10 times → heads may be 4/10 (0.4)
  + Toss 1,000 times → heads ≈ 0.5 (closer to expected 50%)
* **Machine Learning:** Explains why more data improves model accuracy and reduces randomness.

### **Central Limit Theorem:**

* **Idea:** The distribution of the sample mean becomes approximately normal (bell-shaped), even if the original data is not normal, as sample size increases

**Example:**

* Heights of people in a town may be any distribution.
* Take averages of groups of 30 → those averages will form a normal distribution.

**Machine Learning:** Justifies using normal-based assumptions (e.g., Gaussian Naive Bayes, confidence intervals) even if raw data isn’t normal.

**Statistics for Machine Learning:**

**Descriptive stats:**

* **Mean (Average):** Add all values, divide by number of values.
* **Median:** The middle value when data is sorted.
* **Mode:** The most frequent value in data ([2, 3, 3, 5] → mode = 3)
* **Variance:** How spread out the data is from the mean.

**Standard Deviation:** Square root of variance → tells average distance from mean.

* **Correlation:** Shows relationship between two variables (range = -1 to +1, -1: strong negative, 1 increase, other decreases. +1 strong positive: 1 increases other increase, 0: no relation).

**Hypothesis Testing:**

* **Hypothesis Testing:**
* You always start with a **claim (assumption)** about the data → called the **Null Hypothesis (H₀)**.
* You also have an **Alternative Hypothesis (H₁)** which is the opposite claim.
* You collect data → calculate a **test statistic** → and see if the data supports H₀ or H₁.

**Example:**

* H₀: "The average height of students = 170 cm."
* H₁: "The average height ≠ 170 cm."

We test if our data provides strong enough evidence to **reject H₀**

* **P-Value:** The p-value tells you how likely your data is if H₀ were true.
* Small p-value (≤ 0.05) → your data is too rare under H₀ → reject H₀.
* Large p-value (> 0.05) → data fits H₀ fine → don’t reject H₀.
* Different tests are used to determine the p value (T-Test, Chi Square, ANOVA Test).
* **T-Test:** A t-test is a statistical test used to check if the means (averages) of two groups are significantly different or if a sample mean is significantly different from a known value.
* **One-sample t-test** → Compare one group’s mean to a known value.
* **Two-sample t-test** → Compare means of two independent groups (e.g., male vs female scores).
* **Paired t-test** → Compare means of the same group at two times (before vs after training).
* **Chi-Square Test:** Works on counts/frequencies of categorical data.

Two uses:

* **Goodness of fit**: Does observed frequency match expected?
* **Test of independence**: Are two categorical variables related?

**Example:**

* Do male and female students choose subjects differently? (independence test).
* Does your dataset’s class distribution match a uniform distribution? (goodness of fit).
* **ANOVA Test:** ANOVA (Analysis of Variance) is a statistical test used to check if the means of 3 or more groups are significantly different from each other.

**Example:**  
You want to see if students from Class A, Class B, and Class C have different average exam scores.

* Instead of doing multiple t-tests, you use ANOVA.
* If ANOVA result is significant → at least one class’s mean is different.
* But it won’t tell *which* one is different (you’d need further tests like Tukey test for that).

**In short:**

* **t-test** → compare 2 groups.
* **ANOVA** → compare 3 or more groups.

**Linear Algebra for Machine Learning:**

**Vectors & Matrices Basics:**

* **Scalar** → single number (e.g., 5).
* **Vector** → ordered list of numbers (e.g., [2, 3, 4]).
* **Matrix** → 2D array of numbers (rows × columns).
* **Tensor** → higher-dimensional generalization (3D, 4D… used in deep learning).
* **Shape** → describes dimensions (e.g., matrix 3×2 = 3 rows, 2 columns).
* **Operations**:
* Vector addition: [1,2] + [3,4] = [4,6].
* Scalar multiplication: 2 × [1,3] = [2,6].

**Matrix Operations:**

* **Addition** → add element by element (same shape).
* **Multiplication**:
* **Scalar × Matrix** → multiply each entry.
* **Matrix × Matrix** → row of first × column of second.
* **Transpose (Aᵀ)** → flip rows & columns.
* **Dot product (vectors)** → similarity measure: [1,2] · [3,4] = 1×3 + 2×4 = 11.
* **Cross product (vectors in 3D)** → gives a new vector perpendicular to both.
* **Identity matrix (I)** → diagonal of 1’s, leaves vector unchanged (A·I = A).
* **Zero matrix** → all zeros, wipes everything (A·0 = 0).

**Linear Transformations:**

* Matrices can transform vectors.

**Example:**

* Scaling: multiply by 2 → stretches vector.
* Rotation: special matrix rotates vector in space.
* **Geometric view**: a matrix maps points from one space to another.

**Determinant & Rank:**

* **Determinant** → number that tells if a square matrix is invertible (det = 0 → not invertible).
* **Rank** → number of independent rows/columns.
* Rank = dimension of the data (e.g., 2D data lying on a line → rank 1).

**Inverse & Pseudo-inverse:**

* **Inverse (A⁻¹)** → only exists if determinant ≠ 0.
* A·A⁻¹ = I.
* **Pseudo-inverse** (Moore-Penrose) → used when inverse doesn’t exist, especially in regression to solve equations like least squares.

**Norms & Distance:**

* **Norm** = size/length of a vector.
* L1 norm: sum of absolute values (used in L1 Regularization)
* L2 norm: Euclidean length (√(x₁² + x₂² + ...)). (used in L2 Regularization)
* **Distance**:
* Euclidean → straight-line distance (Used in k-NN, clustering, etc).
* Cosine similarity → measures angle (similarity in direction, not magnitude). (text embeddings, recommendation systems)

**Eigenvalues & Eigenvectors:**

* **Eigenvector** → direction that doesn’t change under a transformation, only scales.
* **Eigenvalue** → how much it scales.
* Example: Stretching rubber band → some directions remain the same, just stretched.
* Used in PCA (Principal Component Analysis) to find important directions in data and reduce dimentionality.

**Singular Value Decomposition (SVD):**

Think of your data as a rectangle made of numbers (a matrix).

SVD breaks this rectangle into three simpler parts:

* U → tells directions for the rows
* Σ (Sigma) → tells how important each direction is
* Vᵀ → tells directions for the columns

**Machine Learning:**

* It helps **reduce data size** without losing much info → faster ML, less noise.

**Example:**

* PCA uses SVD to find main features.
* Recommender systems (like Netflix) use it to guess missing ratings.
* Images can be compressed by keeping only top singular values.