MOD-I Pattern Recognition.

Basics of Pattern Recognition.

What is Pattern Recognition?
- Science of identifying patterns & regularities in data
- Involves classifying I/P data into identifiable categories based on key features/ patterns.

Pattern →
Set of attributes/features - help in identifying/categorizing data
Ex: A handwritten digit like "5" is a pattern.
The strokes & curves are features.

Recognition →
labeling / classifying a given pattern into a known class.
Ex - Recognizing whether a shape is circle/pattern

classifier →
An algorithm/model that assigns labels to I/P pattern
Ex - Decision Tree, K-NN, SVM, Neural Networks

Types of Pattern Recognition →

Supervised learning → Training with labeled data. The system learns from ex.

unsupervised learning → No labels provided. The system groups data based on similarity.

Reinforcement learning → learning from feedback (reward / punishment) in an environment.

Steps in PR process:

1. Data collection → Gathering I/P data
2. Preprocessing → cleaning & transforming data (eg. noise removal, normalizn.)
3. Feature Extraction → Selecting imp. info. from raw data.
4. classification → Using algo. to assign categories to the data
5. Post-processing → Improving/interpreting the output (eg. majority voting)

Application of PR →

Handwriting recognition (OCR)
Face Recognition
Speech Recognition
Medical diagnosis
Fraud detection

Common Algorithms used →

K-Nearest Neighbor (K-NN)
Support Vector Mic (SVM)
Naive Bayes
Decision Trees
Artificial Neural Networks
Clustering (eg. K-Means - unsupervised learning)

Features selection vs. feature extraction

- selects subset of original features.
- ~~Reduces dimensionality~~ Remove irrelevant features.
- easy to interpret
- Techniques :- Filter (chi-sq.)
- less intensive (computn cost)
Ex:- Selecting top 10 features based on correlation scores.

- creates new features from original data.
- ~~create~~ transforming original features into a new set of features.
- Harder to interpret.
- Techniques - Autoencoders
- more intensive.
Ex:- Reducing 100 features to 10 using PCA.

MOD-II

① Bayesian Decision Theory

- fundamental statistical approach to the problem of pattern classification.
- uses probability & Bayes' Theorem
- to minimize risk (error) of making wrong decisions.
- Key components :-

Classes ($\omega_1, \omega_2, \ldots \omega_n$): The categories / labels.

Prior Probability $P(\omega)$: The prob. that a random sample belongs to class ω before seeing the data.

- Likelihood $P(x|\omega)$: The probability of observing a feature vector x given class ω.
- Posterior Probability $P(\omega|x)$: Prob. that the pattern belongs to class ω given feature x (calculated using Bayes theorem).

Bayes' Theorem

$$P(\omega_i|x) = \frac{P(x|\omega_i) \cdot P(\omega_i)}{P(x)}$$

where :-

$$P(x) = \sum_j P(x|\omega_j) \cdot P(\omega_j)$$

Bayesian Decision Rule (Min. Error)

- choose the class ω_i for which $P(\omega_i|x)$ is max

or, - choose ω_i if $P(\omega_i|x) > P(\omega_j|x)$, $\forall j \neq i$

② Classifiers :-

- used to assign i/p patterns (like images, text) to predefined categories / classes.
- most classifiers operate in supervised learning setting where the model is trained using labeled data.
- include K-NN, SVM, Decision Trees, Naive Bayes, Neural Networks.

- classifier creates decision boundary
diff. classes in the feature space.
- shape of this boundary depends on type of classifier.
- classifier effectiveness - measured using tools like confusion matrix, f1-score, recall, etc.

Discriminant Functions

- score fun. used to make decisions.
- is a formula / rule helps us decide which class a data point belongs to.
- helps in classifying objects / patterns by finding which class has highest score / value for a given I/P.
- ex:- If you're classifying emails as spam or not spam, a discriminant fun. will give a score for each class. whichever score is higher determines the final label.

- Decision Rule :-
Assign x to class w_i if $g_i(x) > g_j(x)$, $\forall j \neq i$

For bayesian classifier :-
$g_i(x) = P(w_i|x)$ or $\ln P(x|w_i) + \ln P(w_i)$

Decision Surface :

- boundary separates regions in the feature space where diff. decision rules apply.

Ex:- for 2D feature space, it could be a line (linear classifier) or curve (non-linear)

- created by discriminant functions.
- can be diff. shapes :- Line (2D), Plane (3D) or complex curve depending on classifier (linear / non-linear)
- Ex:- If you're classifying fruits as apples or Oranges based on color & size, the decision surface would be a line / curve that separates apples from oranges in a color-size graph.

(3) Normal Density & Discriminant functions

Normal Density

→ Gaussian Distribution.
→ Bell-shaped curve
→ Shows how data is spread out - Most values are around avg. (mean) & few are far away.
→ It is used when we assume that the data for each class follows a natural pattern (ex: cats vs dogs).
→ N.D. formula - helps us calculate probability that a data point belongs to a class based on its features.

$$P(x|\omega_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right]$$

where:-
μ_i : mean vector
Σ_i : covariance matrix
d : dimensionality of x.

→ ex:- Height of stu. in class follows normal curve. most are near avg., few are very tall/short.

Discriminant function for Gaussian

If covariance matrices are equal for all classes the discriminant becomes linear:-

$$g_i(x) = w_i^T x + w_{i0}$$

where:-
$w_i = \Sigma^{-1}\mu_i$
$w_{i0} = -\frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + \ln P(\omega_i)$

If covariance are different, it becomes a quadratic function of x.

(4) Discrete features in Detail:

When features are discrete (categorical) rather than continuous:-

✓ use Probability Tables:-

→ Each feature has a finite no. of values.

→ compute conditional probabilities directly.

$$P(x|\omega_i) = P(x_1, x_2, \ldots, x_n|\omega_i)$$

If assuming independence b/w features (Naive Bayes)

$$P(x|\omega_i) = \prod_{j=1}^{n} P(x_j|\omega_i)$$

Example

For spam detection using words (discrete feature)

x_1 : "free" appears / not

x_2 : "offer" appears / not

Learn probabilities like $P(x_1 = 1|\omega_{spam})$

use Naive Bayes classifier:-

$$g_i(x) = \ln P(\omega_i) + \sum_j \ln P(x_j|\omega_i)$$

MOD-III

(A) Parameter Estimation Methods

a) Maximum Likelihood Estimation (MLE)
b) Gaussian Mixture Models (GMM)
c) Expectation-Maximization (EM) Algorithm
d) Bayesian Estimation.

Parameter Estimation:

- process of estimating parameters (like mean, variance, etc.) of a probability distribution from observed data.
For ex:- Estimating mean (μ) and variance (σ^2) of a normal distribution.

Types of Estimation.

1. Point estimation → gives a single value estimate (eg. Sample mean).
2. Interval estimation → gives a range of values (confidence interval).

Common estimators

- Sample mean ($\bar{x}$) estimates pop^n mean (μ)
- Sample variance (s^2) estimates pop^n variance (σ^2)
- Sample proportion ($\hat{p}$) estimates pop^n proportion (p)

→ It helps us make decisions & predictions when we don't know the true values for the entire population.

a) Maximum Likelihood Estimation

- to estimate unknown parameters of a probability distribution by maximizing the likelihood that the observed data came from that distribution.

→ Likelihood - probability of data given certain parameters.

In MLE, we try to maximize this likelihood.

→ Steps → write the likelihood function.
→ Take the log (called log-likelihood)
→ Differentiate the log-likelihood
→ Solve the eq^n by setting derivative = 0
→ gives MLE estimate.

Common ex:- coin Toss Ex. (Bernoulli Trial)
- Suppose we toss a coin 10 times, & get 7 heads and 3 tails.
- Let p = probability of head
- Likelihood : $L(p) = p^7 \cdot (1-p)^3$
- Take log, differentiate, and solve → get MLE for $p = 7/10$.

MLE → widely used in:
ML, statistics, Data science, PR.

Properties of MLE:-
consistent, efficient,

Limitations:- can be difficult for complex distribution

Real-Life Analogy:-
Imagine guessing someone's weight by trying diff. values & seeing which guess best explains how their clothes fit = MLE is similar.

(b) **Gaussian Mixture Models** (GMM)

- way to group or cluster data that assumes data comes from a mixture of several normal (Gaussian) distributions.
- Each grp. (or cluster) is represented by bell-shaped curve (Gaussian distribution).
- Gaussian distribution is bell-shaped curve often seen in statistics.
  Defined by Mean (μ): center of curve
  Standard Deviation (σ): width of the curve.
- Why mixture? bcz. in real life, data is often a mix of diff. grps. GMM says:-
  Let's assume data comes from several Gaussian distributions combined together.
- Ex. - Imagine you're looking at the heights of people in a room that has:
  Adults, Teenagers, Kids.
  Each grp. will have a diff. avg. height
  So, GMM would try to separate the data into 3 groups each with its mean & std. deviation.

Application → Image Segmentation
Speech Recognition.

(c) **Expectation - Maximization method** (EM)

- to find missing / hidden info. in data by repeating 2 steps :- • Guessing
  • Improving the guess.
- It is mainly used in problems where some data is hidden / incomplete, like in GMM.

- Ice-cream Analogy.

Imagine 2 ice-cream m/c are making cones. You see only the cones but you don't know which m/c made which cone. You want to guess!

- The avg. size of cones made by each m/c.
- Which m/c likely made which cone.

But you don't know which cone came from which m/c - that's the hidden data.

EM helps you figure out that.

- 2 main steps in EM.

i) Expectation Step (E-Step) - "Guess who did what"
→ calculate probability.

ii) Maximization Step (M-Step) - "Update your guess"
→ recalculate best possible parameters.

- Repeat doing E-Step & M-Step again & again until the results stop changing much.

- Applications → GMM
→ Clustering
→ Image Processing

(d) Bayesian Estimation

→ Method of estimating unknown values (called parameters) using :- what we already know (prior knowledge)
New data we observe (evidence)
It uses Baye's Theorem to update our beliefs.

→ Weather Forecast (ex)

- You believe there's 70% chance it will rain tomorrow (prior belief)
- You now check the weather app, & it shows dark clouds (new evidence).

Using Bayesian estim", you update your belief based on this new evidence.

Now, you might say :- now I believe there's 90% chance it will rain.

→ Formula (Baye's Th.)

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}.$$

Prior → what you believe before seeing the data.
Likelihood → how likely is the new data, assuming your belief is true.
Posterior → updated belief after seeing the new data.
Evidence → A normalizing factor

→ Application :- Spam filters,
ML, Robotics & AI.

MOD-IV

(1) Hidden Markov Models (HMMs) for Sequential Classification :-

- Discrete Hidden Markov Models (DHMM)
- Continuous Density Hidden Markov Models (CDHMM)

(2) Hidden Markov Models (HMMs)

- Statistical model used to describe sequential data (like speech, handwriting, DNA, etc.) where the system is modeled as a Markov process with hidden (unobservable) states.

- Real world Ex :-
  - → Speech recognition.
  - → Bioinformatics (DNA sequences)
  - → Handwriting recognition.

- Components of HMM :

Symbol

N → No. of hidden states.

M → No. of possible observ^n. (for discrete models)

$A = \{a_{ij}\}$ → State transition probability matrix (State i to j)

$B = \{b_j(k)\}$ → observ^n. probability matrix.

$\pi = \{\pi_i\}$ → initial state distric^n.

Together, HMM is represented as :-

$\lambda = (A, B, \pi)$

3 main problems solved by this :-

1) Evaluation problem → Forward Algo.
2) Decoding Problem → Viterbi Algo.
3) Learning Problem → Baum-welch Algo.

(3) Discrete Hidden Markov Models (DHMM)

→ Discrete observn comes from a finite set (like words, phonemes, symbols).

→ Each state emits one of M discrete obs based on a probability distribution.

Ex:- Let's say M = 3 (Symbols: A, B, C).

If you're in State S_1 :-

Probability of A = 0.6
B = 0.3
C = 0.1

Then you emit one of them randomly based on this distribution.

→ Modelling steps :-

1. Define States & obs. symbols.
2. Initialize A, B and π.
3. use forward-Backward to compute req. probabilities.
4. Train using EM if model is unknown.

→ Applications → Text seq. classifn.
→ Speech recognition.
→ POS tagging.

(4) Continuous Density Hidden Markov Models (CDHMM)

* Observn. are continuous-valued vectors
* Each State emits observn. acc. to a continuous prob. distrib., usually a Gaussian / mixture of Gaussians (GMM)

Ex:- If you're in State S, your obs. is a feature vector like :-

$$x = [2.3, -1.5, 0.7]$$

This is drawn from:

$$P(x|S_1) \sim N(\mu_1, \Sigma_1)$$

Or more generally:

$$P(x|S_j) = \sum_{m=1}^{M} c_{jm} \cdot N(x|\mu_{jm}, \Sigma_{jm})$$

- c_{jm} → mixture weights
  μ_{jm}, Σ_{jm} :- mean & covariance of m-th Gaussian in state.

Real-world data like speech, images or sensor signals are not discrete — they are continuous.
CDHMMs provides a more realistic & powerful modelling tool for such data.

MOD-V

① Dimension Reduction Methods

Process of reducing the no. of I/P variables in a dataset while preserving as much information as possible.

Why ??
- removes irrevant features.
- reduces computational cost.
- helps in visualizing high-Dimension data.
- Improves classification.

2 main Approaches :-

Feature selection → choose a subset of original features

Feature Extraction → create new features from combination of old ones.

1) Fisher Discriminant Analysis
2) Principal component Analysis

② Fisher Discriminant Analysis

Goal :- find a new axis that maximizes class separability.

Key Idea :- Maximize b/w-class variance
Minimize within-class variance.

How it works :-
1. compute mean of each class & overall mean.
2. compute b/w-class scatter & within-class scatter
3. find projection w that maximizes the ratio :- $J(w) = \frac{w^T S_B w}{w^T S_W w}$

S_B → b/w-class scatter matrix
S_W → within-class scatter matrix

Pros

- Supervised
- Great for classification task
- Better class separation than PCA.

Cons

- Assumes normally distributed features
- Works best when class covariances are similar.

③ Principal component Analysis

Goal: Transform original data into a new coordinate system such that:-

- 1st new axis captures max. variance.
- Next axes capture remaining variance in decreasing order.

How it works?

1. Standardize data (mean = 0)
2. Compute covariance matrix.
3. Compute eigenvectors & eigenvalues
4. Select top k eigenvectors → new feature space.
5. Transform data into new space

Pros:

- Reduces overfitting
- Captures max. variance
- useful for visualizing data in 2D/3D.

Cons

- Principal comp. are not easily interpretable.
- Assumes linear relationship.

PCA vs LDA

• Unsupervised	• Supervised
• Max. variance	• Maximize class separation
• No class labels needed	• Needs class labels.
• Principal component	• Discriminant vectors.

Parzen-window methods

- Non-parametric technique
- to estimate the probability density function (PDF) of a random variable.

It helps us guess the shape of the data distribution, without assuming any specific shape.

Used in -
- pattern recognition.
- classification
- density estimation.

Imagine you are trying to find how people are spread across a park.
You don't know the exact distribⁿ.
So you place a small window around every person & count how many people fall within that window.
If many people fall into a window, you say "density is high here".
If few people fall in, you say "density is low here".

This is Parzen-window idea!

Adv.
- Doesn't assume any fixed shape (flexible)
- works for any distribution.

Disadv.
- Can be slow for large datasets.
- Choice of window size (h) is critical.
  - Too small → noisy
  - Too big → oversmooths.

(5) K-Nearest Neighbours method

→ K-NN – Supervised learning algo:
used for:- classification
- regression

→ Simplest algo in ML.

→ Ex:- Imagine you have a dataset of fruits.

color	Size	fruit
Red	Big	Apple
Yellow	Big	Banana
Red	Small	Cherry

Now you get a new fruit:-
Color: Red.
Size: Medium

using K-NN:
- You find its k-closest fruits in terms of color & size.
- Suppose k=3, and 2 of the 3 closest fruits are Apples → the new fruit is classified as Apple.

Dist. measures

1) Euclidean dist.:

$$d(p,q) = \sqrt{(p_1-q_1)^2 + (p_2-q_2)^2 + \cdots + (p_n-q_n)^2}$$

2) Manhattan Distance

3) Minkowski Distance.

Choosing K:
- Small K (like 1 or 3)
  Sensitive to noise & outliers.
- Large K (like 10 or 15):
  Smoother decision boundaries but may ignore local patterns.

MOD-VI

Non parametric techniques for density estimation:-

- Density Estimation → process of estimating the probability distribution (PDF) of a random variable based on given data.

Parametric	Non-parametric
- Assumes a fixed form - estimates parameters (mean, variance)	Makes no assumption about the distribution shape

Why Non-parametric → more flexible.
→ works better for unknown distribution.

Common Non-parametric methods:-

1. Histogram Estimation:-
   - divide the data range into eq-sized bins
   - count how many points fall into each bin
   - Estimate density by:-

     $$\text{Density} = \frac{\text{count in bin}}{n \times \text{Bin width}}$$

2. Parzen window method } module-(5)
3. k-NN

MOD-VII

① Linear Discriminant Function Based Classifier.

→ to classify data points by drawing a str. decision boundary (hyperplane) that separates classes.

Linear Discriminant function (LDF)

to assign a class label to an input vector x using a linear function.

Function form:- $g(x) = w^T x + w_0$

w: weight vector
x: input vector
w_0: bias (threshold)

Decision rule (for 2-class problem):

If $g(x) > 0$, assign to class 1
If $g(x) < 0$, assign to class 2

Ex:- If I/P has 2 features (x_1, x_2), the decision boundary is:-

$$w_1 x_1 + w_2 x_2 + w_0 = 0$$

→ this is the eq^n of a line.

Properties :→ Simple & fast
→ can't model non-linear rel^n
→ works best when data is linearly separable.

Simple mathematical tool, used to classify data into diff. categories by drawing a st line (2D) or a flat plane that seperates the data.

Ex:- You have 2 types of fruits - Apples, Bananas

Each fruit has 2 features - weight, color. U can plot them on a graph. If a str. line can separate apple from bananas, then you can use linear discrimin

② Perceptron

- Simplest type of neural network
- One of the earliest algo. used for classification
- tries to mimic how a human neuron works & is mainly used to classify data into 2 grps. (binary classification)
- A perceptron takes I/P, multiplies them with weights, adds them up, and then decides whether the O/P should be 1 (yes) or 0 (no).
- Formula :- $\text{Output} = \begin{cases} 1 & \text{if } (w \cdot x + b) > 0 \\ 0 & \text{otherwise} \end{cases}$

where :- x = I/P features (ht., wt.,)
w = weights (imp. of each I/P)
b = bias
$w \cdot x$ = weighted sum of I/P.

- Perceptron draws a st. line (2D) to separate 2 classes.
- Ex :- Want to separate cat (1) from Dog (0) Based on their height & weight. Perceptron will try to draw a line on a graph so that
  - cats are on one side
  - Dogs are on other side.

Adv:
- Simple
- fast
- Works well for linearly separable data.

Limitations
- Only works if data can be separated by a st. line
- Doesn't handle multi-class problems well.

(3) Support Vector Machines (SVM)

- powerful ML algo.
- used for classific. & regression.
- mostly used to classify data into 2 classes. (Spam or not spam, cat or dog)
- tries to find the best boundary that separates 2 classes with largest possible gap b/w them.
- Ex:- Imagine you have 2 types of shoes
  - Sports shoes
  - Party shoes.

You want to separate them on a graph based on 2 features: color & heel size.
SVM draws a line b/w the 2 types but not just any line.
It picks the best possible line that is far away from both classes.
That way, new shoes can be placed on the correct side of the line.

Adv.

- effective when clear margin of separation exists.
- can be used for both linear & non-linear classification.
- works well with high-D data.

Disadv.

- Slow with large datasets.
- Hard to choose the right kernel sometimes
- Doesn't perform perform well if classes overlap a lot.

Real-life uses of SVM:

- Handwriting recog.
- face detection.
- Spam email filtering
- Bioinformatics -

MOD-VIII

(1) Non-metric methods for pattern ~~recogn~~ classification

Non-metric methods

→ Classify methods that don't rely on dist. or geometric measurements.

→ Not based on metrices like Euclidean dist.

→ Useful when data is non-numeric.

→ Nominal data - data that represents categories not numbers.

→ Ex:- Colors: {R, G, B}
Weather: {Sunny, Rainy, Cloudy}
Gender: {Male, Female}
Blood Types: {A, B, AB, O}

→ Why Metric methods don't work?
You can't say "Red" is closer to "Blue" than "Green"
Thus non-metric method is needed.

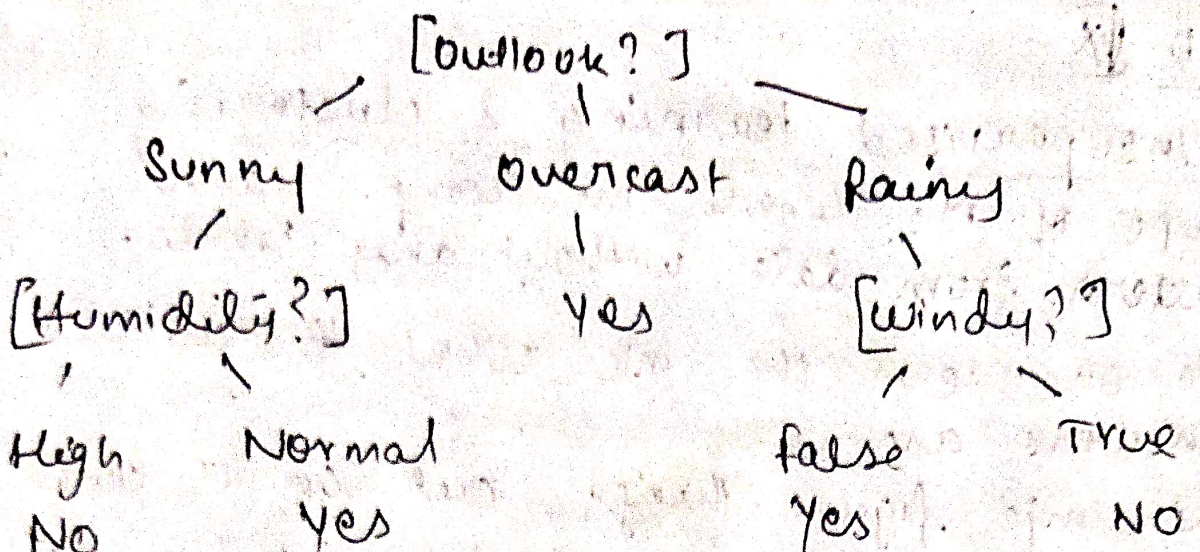
(2) Decision Trees

A tree-like str. where:

- Internal nodes - represent decision rules on attributes.
- Branches rep. outcomes of decisions
- Leaves rep. class labels (final decision)

Ex:- For a weather Dataset:-

Outlook	Temp.	Humidity	Windy	Play Tennis
Sunny	Hot	High	False	NO
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	false	Yes



Adv:

- Easy to visualize & interpret
- No need for feature scaling
- can handle non-numeric & categorical data directly.

Disadv.

- can become complex if not pruned.
- Sensitive to small changes in data
- Prone to overfitting.

MOD - VI

(1) Unsupervised learning & clustering

- Type of ML where the comp. learns pattern from data without any labels.
- No one tells the m/c what the correct answers are.
- The m/c figure things out on its own.
- Ex:- Imagine you ~~have~~ given a bunch of photos of animals, but no one tells you which one is a dog, cat, or rabbit.

Your task is to group similar-looking animals together. That's exactly what unsupervised learning does!

- Unsupervised Algorithms
  - K-Means clustering
  - Principal component Analysis (PCA)

Clustering :-

- an unsupervised learning method where we group similar data points into clusters.

Ex:- Given customer data (age, income), clustering can grp. them into segments like:- Low income, young age
High income, middle age
Retired, fixed income.

Application → Market segmentation
→ Doc. / Image classification
→ Anomaly detection.

(2) Criterion functions for clustering

- technique in unsupervised learning
- where we grp. similar data points together into clusters.
- Ex:- Grouping students by similar marks.
  Grouping customers based on shopping habits.
- Criterion fun. → like rule / formula
  → helps comp. measure how good a clustering is.

goals → points in the same cluster should be close together.
→ points in diff. clusters should be far apart.

common criterion functions:

- Within-cluster Distance (WCD) → Intra-cluster Distance
  - measures how close the points are inside each cluster.
  - make this as small as possible.
  - Smaller value - better grouping.
  - ex - group friends by hobbies.
- Between-cluster Distance (BCD) → Inter-cluster Distance
  - measures how far apart the clusters are from each other.
  - Goal - make this as large as possible.
  - Bigger dist. → better separation b/w clusters.
  - Ex - you want grp. of stu. with very diff. study habits in separate clusters.

- Sum of squared errors (SSE)

- Adds up the sq. dist. of all points from their cluster center.
- Minimize SSE
- used in k-Means clustering

(3) Algorithms for clustering

A) k-Means clustering

1. Choose a no. of clusters K
2. Initialize k random centroids
3. Assign each data point to nearest centroid
4. Update centroids by averaging points in each cluster.
5. Repeat steps 3-4 until convergence.

→ unsupervised learning algo.
→ used to grp. data points into 'K' clusters based on how similar they are
→ like putting similar items into same box.
→ Imagine you have a bag of colourful balls & you want to groups them by color, but the balls are not labeled.
→ K-means will automatically find groups based on color without you telling it.

→ K = no. of clusters you want. You choose this no. before the algo. runs.

for ex: K = 3

→ it will make 3 groups from the data.

→ Works?

1. Choose k
2. Place k random points on data
3. Assign each point to nearest centroid
4. Move the centroids to the centre of their assigned points
5. Repeat 3 & 4 until centroids don't move much.

→ Real-life use → Image compression.
→ Market Segmentation.

(4) Hierarchical Clustering

→ method of unsupervised learning
→ grp. data into tree-like str. called dendogram.
→ doesn't need you to choose the no. of clusters at the start like k-means.
Instead, it builds a hierarchy of clusters - from small to big or big to small
→ Ex - Organizing a family tree.
You 1st grp. close relatives
Then you merge them into bigger family groups
Eventually you get 1 big family tree.

Other methods

1> Density Based Spatial Clustering of Applications with Noise.

→ Groups points are are closely packed.

2> Mean Shift

3> Gaussian Mixture Models (GMM).