PATTERN RECOGNITION

- Automated analysis of data to identify patterns and regularities that can be used to classify or predict future events

- Pattern: A design / set of measurements or observations representing a physical object or event.

- Feature: Measurable characteristic of a pattern used for classification.

- Class: Group of patterns with similar characteristics. Classifiers assign patterns to classes.

- A good dataset is representative of real-world data, free of errors and inconsistencies (clean), balanced across different classes to avoid bias, and large enough to train a robust classifier.

① • Supervised Learning classifiers are trained on a labelled dataset, where each data point is associated with a known class.

Ex: SVM, NN, Decision Trees.

② • Unsupervised Learning classifiers are trained on an unlabelled dataset, where goal is to discover hidden patterns and structures in the data.

Ex: K-Means clustering, Hierachial, PCA

③ • Semi-supervised learning models are trained on a combo of labelled and unlabelled data.

Ex: Self-training (Train classifier on some labelled data to label unlabelled data)
Co-training (Uses 2 or more classifiers to train each other on diff views of the data)

④ • Reinforcement Learning models interact with the real-world based on the choices it makes, rewards or penalties will be given.

★ Paradigms of Pattern Recognition

① Statistical Pattern Recognition

• Treats it as a statistical problem, using probability and statistical features for classification

Ex: Facial Recognition using distances between facial features.

② Structural Pattern Recognition:

• Useful when shape and structure of patterns matter more than numbers.

Ex: Handwriting recognition using strokes and loops.

③ Neural Networks

- Inspired by the human brain; learns patterns from large datasets, making it very effective for image and speech recognition.

④ Template Matching

- Compares input data with predefined templates which works well for simple, fixed patterns.

Ex: Checking manufactured parts for defects.

⑤ Syntactic / Structural Recognition

- Describes patterns using relationships between components similar to grammar in a language.
- Used for complex structured data (Ex:- Protein sequence analysis)

⑥ Machine Learning Algorithms

- Learns from training data to make predictions. using algorithms like SVM, KNN, Decision Trees.

⑦ Hybrid Approaches

- Combines multiple paradigms for better accuracy.
- Uses strengths of different methods together

☆ Phases in Pattern Recognition System

① **Sensing** (Camera capturing image, Mic recording sound)

- Captures raw data from objects, depending on factors like resolution, sensitivity, bandwidth, distortion, etc.

② **Segmentation / Grouping** (Hard)

- Divides data into meaningful parts / groups related to components of an object.

③ **Feature Extraction**

- Selects important characteristics / features and converts raw data into a feature vector.
- Features must be similar within same class and different across different classes.

④ **Classification**

- Assigns objects to categories / classes based on feature vectors

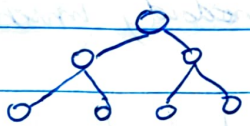
⑤ **Post-Processing**

- Makes final decisions or actions
- Optimizes performance (minimize error rate)

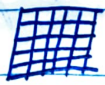
Ex: Triggering an alarm after face recognition.

Data Structures

- **Feature Vector** $v = [a \; b \; c \; d]$
  - Ordered list of numerical values (features) representing a pattern
  - Ex: Representing fruit by (weight, color intensity, diameter)
  - Simple, widely-used, compatible with many algorithms but may not capture complex relationships between features.
- **String** "Hello World"
  - Used for representing sequential data like text or time-series
  - Suitable for sequential data, can capture temporal relationships but requires specialized algos.
- **Trees and Graphs**



  - Used for representing structured data with hierarchical relationships
  - Can represent complex relationships but computationally expensive
- **Images / Matrices**



  - Directly represent image data as a matrix of pixel intensities, preserving spatial info.

- However, computationally intensive and can suffer from high dimensionality

✱ Representation of Clusters

- Way in which groups of similar data points are described and visualized.

① Centroid-based

- Each cluster is represented by its centroid (mean point) common in K-Means clustering. Simple and fast but sensitive to outliers.

② Medoid-based $E = |P_i - C_i|$

- Cluster represented by most central actual data point (medoid) which minimizes total distance to other points
- Robust to outliers but computationally expensive.

③ Density-Based (DBSCAN)

- Clusters are regions of high data density, visualized using heat maps or contours.
- Finds arbitrary-shaped clusters but sensitive to parameter choice.

④ Hierachical

- Clusters shown as a tree structure (dendrogram) which shows how small clusters merge into bigger ones.
- No need to predefine no. of clusters but hard to scale for large data.

⑤ Model-based

- Represented by statistical model using parameters like mean or variance (Gaussian), assuming data fits the model

★ Proximity Measures

- Quantify the similarity / dissimilarity between patterns.

– Euclidean Distance (Straight-line) $\left(d = \sqrt{\Sigma (x_i - y_i)^2}\right)$

- Most commonly used but sensitive to outliers and scale.

– Manhattan Distance (City-block)

$\left(d = \Sigma |x_i - y_i|\right)$ (Sum of absolute differences)

– Minkowski Distance

$\left(d = \left(\Sigma |x_i - y_i|^p\right)^{\frac{1}{p}}\right)$

$p = 1 \rightarrow$ Manhattan
$p = 2 \rightarrow$ Euclidean
$p = \infty \rightarrow$ Chebyshev

- **Mahalanobis Distance**

$$d(X,Y) = \sqrt{(X-Y)^T S^{-1} (X-Y)}$$

$X, Y \rightarrow$ feature vectors / $S \rightarrow$ covariance matrix of data

- **Cosine Similarity**

- Measures angle between two feature vectors, quantifying similarity in direction

$$\text{Cosine Similarity}(X,Y) = \frac{X \cdot Y}{\|X\| \, \|Y\|}$$

Small angle → High Sim
Larger angle → Low Sim

$X \cdot Y = \sum x_i \cdot y_i$ (dot)

$\|X\| = \sqrt{\sum x_i^2}$, $\|Y\| = \sqrt{\sum y_i^2}$ (magnitude)

- **Correlation Coefficient**

- Measures strength and direction of linear relationship between two variables (features)
(How strongly and in what direction one variable changes with another)

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}}$$

Note: Size of a pattern can mean feature dimension, sequence length, no. of data points, spatial size, or information content (complexity / richness)

- Nominal Attributes (Categorical with no order)
Data is qualitative, used only for classification.

Ex: Genders (male, female), Colors (red, green, blue)

- Binary Attributes (Boolean / dichotomous)
Only two possible values (0 or 1).

Ex: Pass / Fail, On / Off, Yes / No

- Ordinal Attributes (Categorical with order)
However, difference between values are not measurable.

Ex: Income Level: Low → Medium → High

- Numeric Attributes (Measurable, Quantitative)
Discrete (countable): No. of students, cars, pages
~~Continuous~~ Continuous (any real value): Time, temperature

— Feature Processing Techniques

① Normalization & Standardization

- Scales features to common range, prevents bias due to different units.

② Encoding

- Converts categorical data to numbers (One-hot encoding, Label encoding)

③ Aggregation: Combines data into summaries such as mean, total, count, etc.

④ Temporal / Spatial Abstraction: Extracts patterns from time or space (trends, cycles, etc.)

- ✱ Feature Selection

— Filter Methods: Select features based on statistical properties of data independent of algorithm chosen.

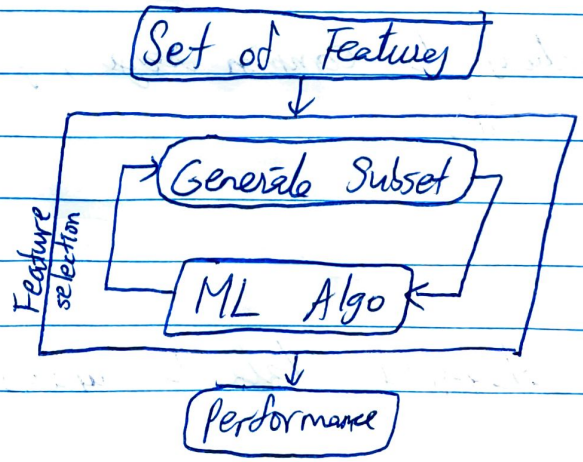
Ex- Correlation, Chi-square, Information Gain

- Fast and simple but may ignore feature interactions

— Wrapper Methods

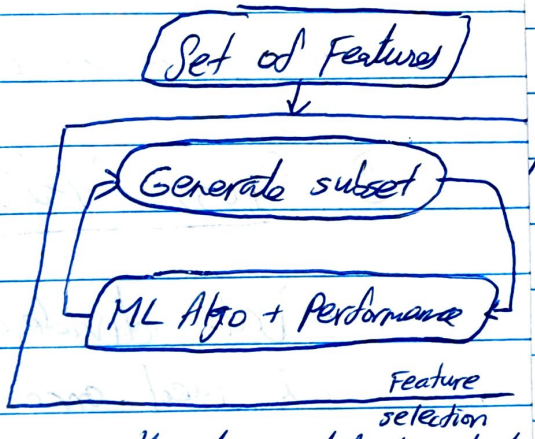
- Use a learning algorithm to evaluate feature subsets by trying diff combos and choosing the best performing set.

- High accuracy but computationally expensive.



- Forward: Start from empty feature set then add the feature maximizing performance one at a time.
Backward: Start from full feature set then remove one by one.

– Embedded / Intrinsic Methods



- Feature selection is part of the training process. (Decision Trees, Lasso)
- Balanced performance and efficiency, though model dependent.

(Pros):
- Improves model performance
- Reduces training time
- Avoids overfitting
- Better generalization

(Cons):
- Risk of removing useful features.
- Computationally costly
- Results depend on model.

★ Evaluation of Classifiers

① Holdout: Dataset split in training and test set and model is trained on training set and tested on test set.
However, some classes may appear only on test set leading to misclassification / poor training, which can be solved by stratification, keeping equal proportion of classes in each set.

② Repeated Holdout / Random Subsampling

- Holdout repeated K times, randomly sampling dataset each time. Final performance is average of all runs.
- However some datapoints may never appear in training set.

③ Cross - Validation

- Data divided into multiple folds where each fold is used once as test set. Results are averaged.

④ K - Fold CV : Data divided in K equal folds
Train on (k-1) fold, test on 1 fold.
Repeat K - times.

⑤ Stratified K-Fold CV : Maintains class proportions.
Suitable for imbalanced datasets

⑥ Leave-One-Out CV : Each datapoint used once as test set
Very accurate but very slow.

⑦ Leave - P - Out CV : Uses P samples as test set.
But computationally expensive for large P

⑧ Confusion Matrix : Means of displaying no. of accurate and inaccurate instances from models predictions.

(Predicted) \ (Actual)	T	F
T	TP	FP
F	FN	TN

TP → accurately predicted positive data
FP → wrongly predicted positive
FN → wrongly predicted negative
TN → accurately predicted negative data

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$ (Correctness)

$$\text{Recall} = \frac{TP}{FN + TP}$$ (Sensitivity) (Actual positives detected)

$$\text{Precision} = \frac{TP}{FP + TP}$$ (How many predicted positives actually correct)

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$ (Harmonic Mean)

$$\text{Specificity} = \frac{TN}{FP + TN}$$ (Actual negatives detected)

$$\text{Type-I Error} = \frac{FP}{TN + FP}$$ (Actual negatives NOT detected)

$$\text{Type-II Error} = \frac{FN}{TP + FN}$$ (Actual positives NOT detected)

☆ Evaluation of Clustering

- Checks how good the generated clusters are, based on high intra-cluster similarity (points in same cluster are close) and low inter-cluster similarity (diff clusters are well-seperated.

- Internal Evaluation uses only data and clustering result without any ground truth. Measures compactness and seperation

- External evaluation compares clusters with true labels using ground truth. (Purity)
- Cluster Cohesion measures how closely-related are objects in a cluster. (SSE)
- Cluster Separation measures how distinct / well separated a cluster is from others.
- The dataset we are working on must have clustering tendency and non-uniform distribution of points.
- Choosing optimal number of clusters (k) is crucial. If k is too high, each point will represent a cluster. If k is too low, can lead to wrong clusters being formed. Requires significant domain knowledge or data-driven approach

① Silhouette Score (Greater = better)

- Quantifies cohesiveness and separation between clusters.

$[-1, 1]$, ≈ 0 → cluster boundary.

$a(i)$ → intra-cluster distance (cohesive) (smaller = better)
$b(i)$ → inter-cluster distance (separation) (larger = better)

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$ ← i → data point

Note:
$a(i)$ → avg distance from point i to other points in same cluster
$b(i)$ → minimum of avg dist from i to point in other clusters.

Exp Cluster A = (1, 2, 3) $i=2$

Cluster B = (10, 11, 12)

$$b(i) = \frac{|10-2| + |11-2| + |12-2|}{3} = 9$$

$$a(i) = \frac{|1-2| + |2-2| + |3-2|}{3} = \frac{2}{3} = 0.67$$

$$S(i) = \frac{9 - 0.67}{\max(9, 0.67)} = 0.997 \simeq +1 \quad (\text{well clustered})$$

② Davies-Bouldin Index (DB) (lower = better)

- Evaluates clusters compactness and separation by comparing each cluster's average similarity-to-dissimilarity ratio to that of most similar neighbours.

$$DB = \frac{1}{n}\left(\sum_{i} \max_{j \neq i}(R_{ij})\right) \quad \text{where} \quad R_{ij} = \frac{S_i + S_j}{M_{ij}}$$

$n \rightarrow$ no. of clusters.

$S_i, S_j \rightarrow$ intra-cluster scatter to centroid of cluster

$M_{ij} \rightarrow$ distance between clusters centroids.

Exp Cluster 1 = (1, 2, 3) Cluster 3 = (15, 16, 17)

Cluster 2 = (8, 9, 10) Cluster 4 = (22, 23, 24)

$n=4$

$m_1 = 2, \; m_2 = 9, \; m_3 = 16, \; m_4 = 23$ (centroids)/(mean)

$$S_1 = \frac{|1-2| + |2-2| + |3-2|}{3} = \frac{2}{3} = 0.67$$

$$S_2 = S_3 = S_4 = 0.67$$

$$R_{12} = \frac{S_1 + S_2}{M_{12}} = \frac{1}{|2-9|} + \frac{4}{3} = \frac{4}{21} = R_{21}$$

$$R_{13} = \frac{S_1 + S_3}{M_{13}} = \frac{4}{3} + \frac{1}{|2-16|} = \frac{2}{21} = R_{31}$$

$$R_{14} = \frac{S_1 + S_4}{M_{14}} = \frac{4}{3} + \frac{1}{|2-23|} = \frac{4}{63} = R_{41}$$

$$R_{23} = \frac{S_2 + S_3}{M_{23}} = \frac{8}{3} + \frac{1}{|9-16|} = \frac{4}{21} = R_{32}$$

$$R_{34} = \frac{S_3 + S_4}{M_{34}} = \frac{4}{21} = R_{43}$$

$$R_{24} = \frac{S_2 + S_4}{M_{24}} = \frac{4}{21} + \frac{1}{13} = \frac{2}{21} = R_{42}$$

$$DB = \frac{1}{4}\left(\max(R_{12}, R_{13}, R_{14}) + \max(R_{21}, R_{23}, R_{24}) + \max(R_{31}, R_{32}, R_{34}) + \max(R_{41}, R_{42}, R_{43})\right)$$

$$= \frac{1}{4}\left(\frac{4}{21} + \frac{4}{21} + \frac{4}{21} + \frac{4}{21}\right) = \frac{4}{21} = 0.19$$

③ Adjusted Rand Index (ARI)

- Compares clustering with ground truth labels.

$$\left(ARI = \frac{RI - \text{Expected}(RI)}{\text{Max}(RI) - \text{Expected}(RI)}\right) \approx [-1, 1]$$

④ Calinski-Harabasz Index

- Measures ratio of between-cluster variance, within-cluster variance.

- Higher value → better clustering

(Module 1 - ARI)

Q) Given Data:

Ground Truth : Class A = {1, 2, 3}
Class B = {4, 5, 6}

Predicted clusters : Cluster 1 = {1, 2, 4}
Cluster 2 = {3, 5, 6}

Total points, $n = 6$

(Contingency Table)

	Cluster 1	Cluster 2	(a_i) Row Sum
Class A	2 (1, 2)	1 (3)	3
Class B	1 (4)	2 (5, 6)	3
Col. Sum (b_j)	3	3	6

Cell	n_{ij}	$^{n_{ij}}C_2$
(Class A) ∩ (Cluster 1)	2	$^2C_2 = 1$
(Class A) ∩ (Cluster 2)	1	$^1C_2 = 0$
(Class B) ∩ (Cluster 1)	1	$^1C_2 = 0$
(Class B) ∩ (Cluster 2)	2	$^2C_2 = 1$

$$\sum_{ij} {}^{n_{ij}}C_2 = 1 + 0 + 0 + 1 = 2 \quad \text{--- (1)}$$

$$\sum_{i} {}^{a_i}C_2 = {}^3C_2 + {}^3C_2 = 3 + 3 = 6 \quad \text{--- (2)}$$

(pairs in same class)

$$\sum_{j} {}^{b_j}C_2 = {}^3C_2 + {}^3C_2 = 6 \quad \text{--- (3)}$$

(pairs in same cluster)

$${}^nC_2 = {}^6C_2 = 15$$ (total pairs) --- (3)

~~ARI = $\sum {}^{n_{ij}}C_2$~~ -

$$ARI = \frac{\sum {}^{n_{ij}}C_2 - \frac{\sum {}^{a_i}C_2 \cdot \sum {}^{b_j}C_2}{{}^{n}C_2}}{\frac{1}{2}\left[\sum {}^{a_i}C_2 + \sum {}^{b_j}C_2\right] - \frac{\sum {}^{a_i}C_2 \cdot \sum {}^{b_j}C_2}{{}^{n}C_2}}$$

$$= \frac{2 - \frac{6 \times 6}{15}}{\left[\frac{1}{2} \times (6+6)\right] - \frac{6 \times 6}{15}} = \frac{-6}{15} \times \frac{15}{54} = -0.11$$

Note: ARI = 1 (Perfect match)
= ~0 (random like clustering)

So here, ARI = -0.11 ⇒ worse than Epstein (random)

a) (Calinski - Harabasz Index)

Given data

Dataset, $X = \{2, 4, 8, 10\}$

$K = 2$ clusters, $N = 4$ points

Cluster 1 = {2, 4}, Cluster 2 = {8, 10}

① Find centroids,

$C_1 = \frac{2+4}{2} = 3$ / $C_2 = \frac{8+10}{2} = 9$

$C_{overall} = \frac{2+4+8+10}{4} = 6$

② Compute B (Between - Cluster Sum of Squares)

$$B = \sum_{k=1}^{K} n_k \cdot \|C_k - C_{overall}\|^2$$

For cluster 1, $n_1 = 2$,

$B_1 = 2 \times (3-6)^2 = 18$

For cluster 2, $n_2 = 2$

$B_2 = 2 \times (9-6)^2 = 18$

$B = B_1 + B_2$

$= 36$

(3) Within - Cluster Sum of Squares, W

$$W = \Sigma \|x - C_k\|^2$$

For cluster 1, $W_1 = (2-3)^2 + (4-3)^2 = 2$

For cluster 2, $W_2 = (8-9)^2 + (10-9)^2 = 2$

$W = W_1 + W_2 = 4$

(4) $$CH = \frac{B}{W} \times \frac{N-K}{K-1}$$

$$= \frac{36}{4} \times \frac{4-2}{2-1} = 18$$

(Higher CH = better clustering) (B ↑, W ↓)

FEATURE EXTRACTION FOR PATTERNS

- Dimensionality reduction technique where raw data with many variables is transformed into smaller set of meaningful features that still represent the original data effectively.

- Large datasets usually contain many variables (features) and processing all of them can cause high computational cost, increased processing time and difficulty in analyzing patterns and risk of overfitting.

- A feature is an individual measurable property of characteristic of data. (column of data)

☆ Structural Pattern Recognition

- Identifies patterns based on structure and relationships between their components rather than only numerical features. It also takes into account the arrangement of components in space.

Ex: Letter A can be described as two slanted lines and one horizontal line, whose structure forms the pattern.

- While Structural Pattern Recognition focuses on structure of patterns, representing patterns as relationships Statistical Pattern Recognition focuses on numerical features, representing patterns as vectors.

① Graph Representation

Nodes → components of pattern
Edges → relationships between components.

② Tree Representation

Trees are used when patterns have hierachical relationships

Ex: 

③ String Representation

- Used when patterns occurs in sequences.

Ex: DNA sequence : A - T - G - C - A
(each is part of biological pattern)

- Applications of Structured Pattern Recognition include Character Recognition (OCR), Speech Recognition (Siri, Alexa), Biological Sequence Analysis (DNA, protein structure), Image Analysis (identifying objects or shapes in images)

- Feature extraction in SPR works in several steps:

① Identify the components / sub-patterns
Ex: Letter A contains left stroke, right stroke, crossbar.

② Once components are identified, represent using structures such as graphs, trees, strings, etc.

③ Capture spatial relationships (how components are arranged)

Ex: Distance between parts, position, angle between components

④ Use syntactic methods as well, such as rules/grammars similar to languages for string patterns.

Ex: Sentence = Subject + Verb + Object (patterns follow structural rules)

- Properties

(i) Invariance to Transformations (should be able to recognize patterns even if they change)

Rotation (↻), Scaling (⤡), Translation (↔)

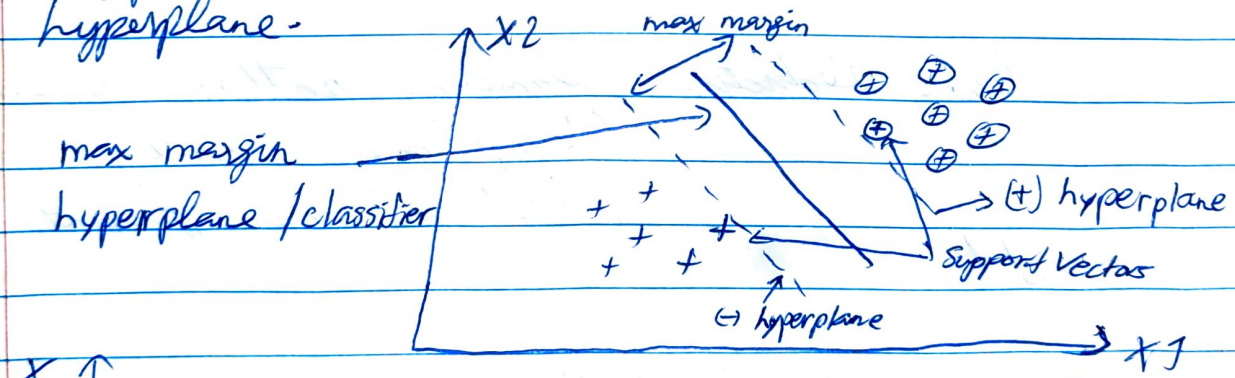
(ii) Complexity Reduction (simplify pattern representation)
(keep only important information)
(fast and more efficient PR)

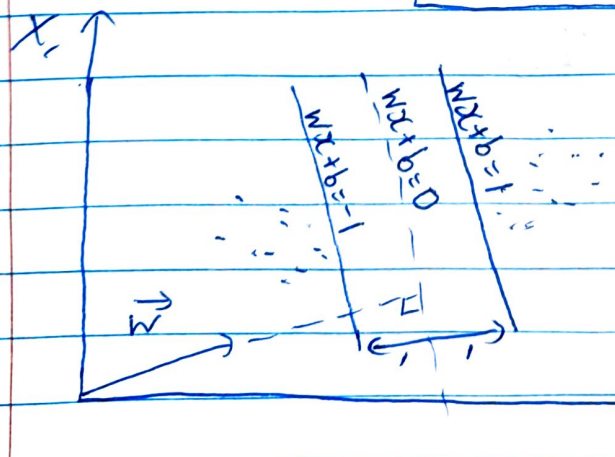
- Benefits

- Recognizes Complex Patterns
- Identifies Relationship
- Provides deeper insight
- Helps Decision Making
- Improves Accuracy

★ Support Vector Machines

- Supervised ML algorithm that can be used for both classification or regression challenges.
- In this algorithm, we plot each data item as a point in n-D space (n features) and perform classification by finding the optimal hyperplane that differentiates the two classes very well. (decision boundary)
- Here, we create a hyperplane that has the maximum margin (distance between data points) (↓)
- Data points/vectors closest to hyperplane are called Support Vectors, which affect the position of the hyperplane.





$$\min \frac{1}{2}\|w\| \qquad \max \frac{2}{\|w\|}$$

such that

$$(wx+b) \geq 1 \quad \forall x \in C_1$$
$$(wx+b) \leq -1 \quad \forall x \in C_2$$

☆ Clustering

- Distance-based unsupervised ML algorithm where data points close to each other are grouped in a given no. of clusters / groups.

① Hard Clustering: Each datapoint is assigned only a single cluster (K-Means, K-Medoid)

Ex- Data Point (Class 1), Data Point (Class 2)

② Soft Clustering: Each datapoint belongs to a cluster with a certain probability (Membership Value) (Fuzzy C-Means)

Ex- Data Point (0.03 ①, 0.97 ②), Data Point (0.51 ①, 0.49 ②)

— Fuzzy C-Means

- Unlike K-Means (Hard Clustering), FCM assigns membership values in [0,1] and sum of memberships of a point across all clusters is 1.

- Cluster boundaries are not sharp and datapoints can partially belong to multiple clusters

- As a result, FCM minimizes within-cluster distances to ensure compact yet fuzzy clusters

- Cluster Centers Update Formula,

$$V_i = \frac{\sum_{k=1}^{n} \gamma_{ik}^{m} x_k}{\sum_{k=1}^{n} \gamma_{ik}^{m}}$$

where V_i → centre of cluster i
x_k → k^{th} data point
γ_{ik} → membership of x_k in cluster i
m → fuzzification parameter ($m > 1$)
n → no. of data points.

High membership → high influence on the center

$$\gamma_{ki} = \left(\sum_{j=1}^{c} \left(\frac{d_{ki}^2}{d_{kj}^2} \right)^{\frac{1}{m-1}} \right)^{-1}$$

c → no. of clusters
$d_{ki} = \|x_k - V_i\|$ (distance between x_k and center V_i)

Membership $\propto \frac{1}{\text{Distance}}$

Closer the point to the cluster center → higher membership

Ex:

Clusters	(1,3)	(2,5)	(5,8)	(7,9)
1	0.8	0.7	0.2	0.1
2	0.2	0.3	0.8	0.9

($m=2$)

$$V_{11} = \frac{(0.8)^2 \times 1 + (0.7)^2 \times 2 + (0.2)^2 \times 4 + (0.1)^2 \times 7}{(0.8)^2 + (0.7)^2 + (0.2)^2 + (0.1)^2}$$

V_{11} ← cluster, data

$$= 1.568$$

$$V_{21} = \frac{(0.2)^2 \times 1 + (0.3)^2 \times 2 + (0.8)^2 \times 4 + (0.9)^2 \times 7}{(0.2)^2 + (0.3)^2 + (0.8)^2 + (0.9)^2}$$

$= 5.35$

$$V_{12} = \frac{(0.8)^2 \times 3 + (0.7)^2 \times 5 + (0.2)^2 \times 8 + (0.1)^2 \times 9}{(0.8)^2 + (0.7)^2 + (0.2)^2 + (0.1)^2}$$

$= 4.051$

$$V_{22} = \frac{(0.2)^2 \times 3 + (0.3)^2 \times 5 + (0.8)^2 \times 8 + (0.9)^2 \times 9}{(0.2)^2 + (0.3)^2 + (0.8)^2 + (0.9)^2}$$

$= 8.21$

$\therefore C_1 (1.568, 4.051)$

$C_2 (5.35, 8.21)$

$D_{11} = \sqrt{(1-1.568)^2 + (3-4.051)^2} = 1.1946$

D_{11} ↓ data cluster

$D_{21} = 1.0427$

$D_{31} = 4.6378$

$D_{41} = 7.348$

$D_{12} = 6.787$

$D_{22} = 4.639$

$D_{32} = 1.366$

$D_{42} = 1.829$

Now,

$$\gamma_{11} = \left(\frac{D_{11}^2}{D_{11}^2} + \frac{D_{11}^2}{D_{12}^2}\right)^{-1}$$

$= 0.97$

$$\gamma_{12} = \left(\frac{D_{12}^2}{D_{12}^2} + \frac{D_{12}^2}{D_{12}^2}\right)^{-1}$$

$= 0.03$

$\gamma_{21} = 0.95$

$\gamma_{22} = 0.05$

$\gamma_{31} = 0.08$

$\gamma_{32} = 0.92$

$\gamma_{41} = 0.06$

$\gamma_{42} = 0.94$

$\therefore$

Clusters	C_1	C_2
(1,3)	0.97	0.03
(2,5)	0.95	0.05
(4,8)	0.08	0.92
(7,9)	0.06	0.94

- Applications

- Image Segmentation
- Pattern Recognition
- Traffic Flow Analysis
- Medical diagnosis
- Risk Assessment

- Pros

- Works well for overlapping datasets
- A datapoint can belong to multiple clusters with different membership values

- Cons

- No. of clusters must be specified beforehand
- Low value of m gives better results but more iterations
- Uses Euclidean distance, may weigh factors unequally
- Performance depends on initial clusters centers & membership values

☆ CART :

- Predictive algorithm that models how target variable can be predicted based on input features

- Classification Trees, when target variable is categorical
  Regression Trees, when target variable is continuous

- Constructs binary tree structure where internal nodes represents decision based on feature value, and leaf node corresponds to class label / predicted value.

- At each node, CART selects feature and value to split the dataset into two subsets.

- The split maximizes the homogeneity/purity of the resulting subsets measured by Gini impurity for classification / MSE for regression.

- $Gini = 1 - \sum p_i^2$ (p_i → probability of each class i in the split)

  Gini Index $\in [0, 1]$
  Gini = 1 ⇒ items dispersed randomly
  Gini = 0.5 ⇒ items distributed evenly (~~pure~~)
  Gini = 0 ⇒ presence of single class (pure)

- For classification, goal is to split the data such the Gini impurity is minimized within each subset (how mixed up the data is)
- Continues until stopping criterion is reached, such as max tree depth or min instances in leaf node

$$\Delta Gini = Gini(\text{entire dataset}) - Gini(\text{split})$$

☆ K-Means Clustering

1. Define no. of clusters (k)
2. Initialize random no. of cluster centroids
3. Compute distance ~~from~~ matrix from k cluster centroid to all data sample
5. Assign the data samples to closest cluster

(5) Compute new cluster centroids by taking mean of the data samples of each cluster

(6) Check whether new cluster centroids are similar to previous centers

(7) If not, repeat, else stay.

Ex:

	$C_1(2,10)$	$C_2(5,8)$	$C_3(1,2)$	Cluster
$(2,10)$	0	$\sqrt{13}$	$\sqrt{65}$	C_1
$(2,5)$	5	$\sqrt{18}$	$\sqrt{10}$	C_3
$(8,4)$	$\sqrt{72}$	5	$\sqrt{53}$	C_2
$(5,8)$	$\sqrt{13}$	0	$\sqrt{52}$	C_2
$(7,5)$	$\sqrt{50}$	$\sqrt{13}$	$\sqrt{45}$	C_2
$(6,4)$	$\sqrt{52}$	$\sqrt{17}$	$\sqrt{29}$	C_2
$(1,2)$	$\sqrt{65}$	$\sqrt{52}$	0	C_3
$(4,9)$	$\sqrt{5}$	$\sqrt{2}$	$\sqrt{58}$	C_2

Now Cluster 1 $\Rightarrow$ $(2,10)$

Cluster 2 $\Rightarrow$ $(8,4), (5,8), (7,5), (6,4), (4,9)$

Cluster 3 $\Rightarrow$ $(2,5), (1,2)$

New centroids $\Rightarrow$ $C_1(2,10)$

$$C_2\left(\frac{8+5+7+6+4}{5}, \frac{4+8+5+4+9}{5}\right)$$

$C_2(6,6)$

$C_3(1.5, 3.5)$

✱ Feature Selection

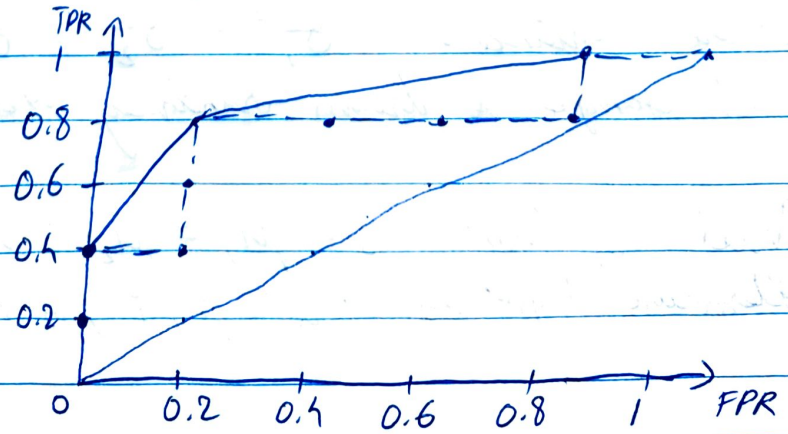
✱. ROC Curve (TPR (Sensitivity) - FPR)

- Graph used to evaluate performance of a classification model, by showing trade-off between sensitivity and specificity

- Perfect classifier ⇒ Top-left corner (0,1) of graph
  Curve near diagonal ⇒ poor model

Ex:

Tuple	Class	Prob	TP	FP	TPR	FPR
1	P	0.90	1	0	1/5	0/5
2	P	0.80	2	0	2/5	0/5
3	N	0.70	2	1	2/5	1/5
4	P	0.60	3	1	3/5	1/5
5	P	0.55	4	1	4/5	1/5
6	N	0.54	4	2	4/5	2/5
7	N	0.53	4	3	4/5	3/5
8	N	0.51	4	4	4/5	4/5
9	P	0.51	5	4	5/5	4/5
10	N	0.40	5	5	5/5	5/5



★ ANOVA (Analysis of Variance)

- Strong statistical technique used to show difference between two or more means or components through significance tests
- ANOVA works by comparing variance between groups and variance within groups.

$$\text{ANOVA} = \frac{\text{Variance Between Groups}}{\text{Variance Within Groups}}$$

The larger the variance between groups, the more significantly different the group means will be.

Ex: Comparing average marks of students from 3 different schools.

— **Assumptions**

- Each population has normal distribution
- Populations from which samples are drawn must have equal variance. $\sigma_1^2 = \sigma_2^2 = \sigma_3^2 = \ldots = \sigma_n^2$
- Each sample is drawn randomly and are independent

- Null Hypothesis, $H_0: \mu_1 = \mu_2 = \mu_3 = \ldots = \mu_n$
  Alternative Hypothesis, $H_a: \mu_1 \neq \mu_2 \neq \mu_3 \neq \ldots \neq \mu_n$

— One-Way ANOVA

- Used when there is one independent variable / factor.

Ex:-

A	B	C
2	3	4
4	5	6
6	7	8

(Schools) (One-way factor = Marks)

$c = 3$ (columns), $n = 9$ (samples)

① $H_0 : \bar{x}_A = \bar{x}_B = \bar{x}_C$

$H_a : \bar{x}_A \neq \bar{x}_B \neq \bar{x}_C$

② $\bar{x}_A = 4$, $\bar{x}_B = 5$, $\bar{x}_C = 6$

Grand Average of Means, $\bar{\bar{x}} = \frac{1}{3}(4+5+6) = 5$

③ SSC (Sum of Squares between Samples) (column-wise)

$(\bar{x}_A - \bar{\bar{x}})^2$	$(\bar{x}_B - \bar{\bar{x}})^2$	$(\bar{x}_C - \bar{\bar{x}})^2$
$(4-5)^2 = 1$	0	1
1	0	1
1	0	1
3	0	3

$SSC = 3 + 0 + 3 = 6$ $\quad (\Sigma(\bar{x}_i - \bar{\bar{x}})^2)$

④ SSE (Sum of Squares within samples) (row-wise)

$(A - \bar{x}_A)^2$	$(B - \bar{x}_B)^2$	$(C - \bar{x}_C)^2$
$(2-4)^2 = 4$	4	4
0	0	0
4	4	4
8	8	8

$SSE = \Sigma(x - \bar{x})^2 = 24$

Degree of freedom

(4)

Source of Variation	Sum of Squares	DOF	Mean SS
Between	SSC = 6	$v_1 = c-1 = 2$	$MSC = \frac{SSC}{c-1} = 3$
within	SSE = 24	$v_2 = n-c = 6$	$MSE = \frac{SSE}{n-c} = 4$

$$F = \frac{MSC}{MSE} = \frac{3}{4} = 0.75$$

(5) If $F_{calc} < F_{tabulated}$, then H_0 is accepted

If $F_{calc} > F_{tabulated}$, then H_0 is accepted
H_0 is rejected.

Considering 5% variation Fischer table,
F_{tab} $_{(2,6)}$ = 5.14 > F_{calc}, then H_0 is accepted

∴ No significant different between means of three groups

8 — Two-Way ANOVA

- Used when result is affected by two variables / factors.

Ex:

Days	A	B	C	D	
Monday	2	3	4	5	c = 4
Tuesday	4	4	6	6	r = 3
Wednesday	6	5	8	8	n = 12

Here variance determined for between students (A, B, C, D) and between days (M, T, W)

① min = 2, max = 8

Correction Factor = Preferably between [2, 8] in table.

CF = 5 (random)

Days	A	B	C	D	Total
Monday	2-5=-3	-2	-1	0	-6
Tuesday	-1	-1	1	1	0
Wednesday	1	0	3	3	7
Total	-3	-3	3	4	1 ← GT (Grand Total)

$$CF = \frac{(GT)^2}{n} = \frac{1}{12 \text{ samples}} = 0.08$$

② SSC (between columns)

$$SSC = \frac{A^2}{N_A} + \frac{B^2}{N_B} + \frac{C^2}{N_C} + \frac{D^2}{N_D} - CF$$

(D^2: column total; N_D: no. of elements in column)

$$= \frac{(-3)^2}{3} + \frac{(-3)^2}{3} + \frac{(3)^2}{3} + \frac{(4)^2}{3} - 0.08$$

$$SSC = 14.25$$

③ SSR (between rows)

$$SSR = \frac{(Mon)^2}{N_M} + \frac{(Tues)^2}{N_T} + \frac{(Wed)^2}{N_W} - CF$$

$$= \frac{(-6)^2}{4} + \frac{(0)^2}{4} + \frac{(7)^2}{4} - 0.08$$

$$SSR = 21.17$$

④ $SST\ (total) = (-3)^2 + (-2)^2 + (-1)^2 + (0)^2 + (-1)^2 + (-1)^2 + (1)^2 + (1)^2 + (1)^2 + (0)^2 + (3)^2 + (3)^2 - CF$

$SST = 36.92$

⑤ $SSE\ (error) = SST - (SSC + SSR)$

$SSE = 1.5$

Sum of Squares	DOF	Mean
$SSC = 14.25$	$c-1=3$	$MSC = 4.75$
$SSR = 21.17$	$r-1=2$	$MSR = 10.585$
$SSE = 1.5$	$(c-1)(r-1)=6$	$MSE = 0.25$
$SST = 36.92$	$n-1=11$	

$F_{col} = \frac{MSC}{MSE} = 19$ $\quad F_{row} = \frac{MSR}{MSE} = 42.85$

$F_{tab,col} = 4.76$ $\quad F_{tab,row} = 5.14$

$(3,6)$ $\quad (2,6)$

Since $F_{tab} < F_{calc}$, H_A is accepted

H_0 is rejected.

∴ Study hours vary significantly between students

Study hours also vary significantly between days

Kolmogorov - Smirnov (K-S) Test

- Non-parametric statistical test used to determine whether a dataset follows a specific probability distribution or whether two datasets come from same distribution.

Ex: Check whether dataset follows normal distribution.

- Maximum difference between distributions:

$$D = \max |F_o(x) - F_e(x)|$$

(F_o: observed, F_e: expected)

Ex: (obs)

Model	A	B	C	D	E	
No. of purchases	14	18	32	20	16	(=100)
Expected	20	20	20	20	20	

Test at $\alpha = 0.05$ that distribution of preference is same (uniform)

$H_0: F(x) = F_e(x)$

$H_a: F(x) \neq F_e(x)$

(1) Arrange in ascending order ~~of frequency~~

	(cdf) $S_n(x)$	$F_o(x)$	$\lvert S_n(x) - F_o(x) \rvert$
(A)	$\frac{14}{100}$	$\frac{20}{100}$	0.06
(A, B)	$\frac{32}{100}$	$\frac{40}{100}$	0.08
(A, B, C)	$\frac{46}{100}$	$\frac{60}{100}$	0.04

	$S_n(x)$	$F_0(x)$	$S_n(x) - F_0(x)$
(A, B, C, D)	$\frac{84}{100}$	$\frac{80}{100}$	0.04
(A, B, C, D, E)	$\frac{100}{100}$	$\frac{100}{100}$	0

$D_n = \max |S_n(x) - F_0(x)| = 0.08$

$D_{\substack{\alpha=0.05 \\ n=5}} = 0.565$ (from K-S table)

Since $D_\alpha > D_n$, H_0 : accepted

Note: If $n > 35$, $D_{\alpha=0.05} = \frac{1.36}{\sqrt{N}}$

Ex: Consider the following numbers as random sample from $U(0,1)$ distribution.

(obs) x = 0.41, 0.51, 0.02, 0.60, 0.47

f = 1, 1, 1, 1, 1

$S_n(x)$	$F_0(x)$	$\lvert S_n(x) - F_0(x) \rvert$
$\frac{1}{5}$	0.02	0.18
$\frac{2}{5}$	0.41	0.01
$\frac{3}{5}$	0.47	0.13
$\frac{4}{5}$	0.51	0.26
$\frac{5}{5}$	0.60	0.40

$D_n = \max |S_n(x) - F_0(x)| = 0.40$

$D_{\alpha=0.05, n=5} = 0.565$ Since $D_\alpha > D_n$, H_0 accepted

Note: $x \sim U(a,b)$ $(U(0,1))$ (Follows uniform distribution)

$pdf = \frac{1}{b-a}$

$cdf = \frac{x-a}{b-a} = \frac{x-0}{1-0} = x$

$\therefore F(x) = F_0(x)$

Ex: Check whether these random values are in $U(0,1)$

0.15, 0.94, 0.05, 0.51, 0.29

i	x_i	$S_n(x) = i/n$	$F(x) = x$ (for $U(0,1)$)
1	0.05	0.2	0.05
2	0.15	0.4	0.15
3	0.29	0.6	0.29
4	0.51	0.8	0.51
5	0.94	1	0.94

i	$D^+ = S_n(x) - F(x)$	$D^- = F(x) - (i-1)/n$
1	0.15	0.05
2	0.25	-0.05
3	0.31	-0.11
4	0.29	-0.09
5	0.06	0.14
	$(\max(D^+) = 0.31)$	$(\max(D^-) = 0.14)$

$D_n = \max(D^+, D^-) = 0.31$

D^+ → max positive difference (largest point where empirical CDF above theoretical CDF)

D^- → max negative difference (largest point where theoretical CDF above empirical CDF)

$D_{\substack{\alpha=0.05 \\ n=5}} = 0.565$

Since $D_\alpha > D_n$, accept H_0 (uniformly distributed)

☆ Probabilistic Seperability-Based Feature Selection

- Method used to chose features that best seperate different classes in a dataset using probability distributions.
- Features with high seperability are kept.
- If a feature has similar distribution for different classes, it is not very useful for classification.
- Estimate the probability density function (PDF) for continuous features or calculating probability for discrete features; then compute a seperability metric that quantifies difference between distributions.
- Common metrics include Bhattacharya distance and Chernoff distance.
- Bhattacharya distance measures overlap between two statistical samples / populations
- This coefficient can be used to determine the relative closeness of two samples being considered

- BC is used to compare two normalized histograms $(x_1, x_2, x_3, \ldots, x_n)$ and $(y_1, y_2, y_3, \ldots, y_n)$

Consides new vectors x and y such that:

$x = (\sqrt{x_1}, \sqrt{x_2}, \sqrt{x_3}, \ldots, \sqrt{x_n})$, $y = (\sqrt{y_1}, \sqrt{y_2}, \ldots, \sqrt{y_n})$

$$x \cdot y = |x| \cdot |y| \cdot \cos\theta$$
$$= \sqrt{x_1 y_1} + \sqrt{x_2 y_2} + \ldots \sqrt{x_n y_n}$$
$$= (\sqrt{x_1} + x_2 + \ldots + x_n)(\sqrt{y_1} + y_2 + \ldots + y_n) \cos\theta$$

Here x_i and y_i represent normalized bin values such that $\sum x_i = \sum y_i = 1$

$$\therefore \cos\theta = \sqrt{x_1 y_1} + \sqrt{x_2 y_2} + \ldots + \sqrt{x_n y_n}$$
$$= \sum \sqrt{x_i y_i} = B(x, y)$$

$\therefore$ BC measures cosine angle between x and y.

- Hellinger distance, $d_H = 1 - B(x, y)$
  Bhattacharya distance, $d_B = -\ln(B(x, y))$

- $B(x, y) \approx 1$ (overlapping) (small distance)
  ≈ 0 (different) (large distance)

Ex:

Sample	Feature Value	Class
1	Low	A
2	Low	A
3	Medium	A
4	High	B
5	High	B
6	Medium	B

Bin	P(x/A)	P(x/B)
Low	0.6	0.1
Medium	0.3	0.3
High	0.1	0.6

Note: $P(X/C) = \frac{\text{Count of value in class}}{\text{Total samples in class}}$

But since probability should not be zero, taking total = 1, $P(\text{High}/A) = 1 - (P(\text{Low}/A) + P(\text{Med}/A))$

$P(\text{Low}/A) = \frac{2}{3} = 0.6$

$P(\text{Med}/A) = \frac{1}{3} = 0.3$

$BC = \sum \sqrt{P(x/\theta)\, P(x/B)}$

Low $\rightarrow \sqrt{0.6 \times 0.1} = \sqrt{0.06}$

Medium $\rightarrow \sqrt{0.3 \times 0.3} = \sqrt{0.09}$

High $\rightarrow \sqrt{0.1 \times 0.6} = \sqrt{0.06}$

BC = 0.78 (moderate overlap)

BD = 0.2369 ($\simeq 0$, feature not that useful)

- Chernoff Distance

- In a binary classification problem, Class 1 generates data acc to P(x) distribution and Class 2 generates data acc to Q(x)

- Chernoff measures how well these two are separated.

$P(\text{error}) < e^{-nC}$, $n \rightarrow$ no. of samples

$C \rightarrow$ min classification error bound (Chernoff info)

Larger $C \rightarrow$ classes are more separable

error ↓ faster.

- For chosen parameter $s \in [0,1]$

  Discrete, $Z(s) = \sum P(x)^s Q(x)^{1-s}$ (CC)

  Continuous, $Z(s) = \int p(x)^s q(x)^{1-s} dx$

$D_c = -\ln Z(s)$ $\qquad$ $C = \max_{s \in [0,1]} D_c(s)$

- If $s = \frac{1}{2}$, $Z(\frac{1}{2}) = \sum \sqrt{P(x) Q(x)} \simeq BC$

Ex:- Let $s = 0.5$

Bin	$P(x)$	$Q(x)$
Low	0.60	0.10
Medium	0.30	0.30
High	0.10	0.60

$Z(0.5) = \sum \sqrt{P(x) Q(x)}$

Low $\rightarrow \sqrt{0.6 \times 0.1} = \sqrt{0.06}$

Medium $\rightarrow \sqrt{0.3 \times 0.3} = \sqrt{0.09}$

High $\rightarrow \sqrt{0.1 \times 0.6} = \sqrt{0.06}$

$Z(0.5) = 0.7899$

$D_c(0.5) = -\ln(0.7899) \simeq 0.2369$

Similarly try different s values and find maximum $D_c(s)$.

PATTERN CLASSIFIERS

☆ Bayesian Decision Theory

- Statistical method used for pattern classification using probability theory
- Determines best class for a data sample based on probability of each class and observed data and cost of classification errors
- We assume that decision problem is expressed in probabilistic terms and all relevant probabilities are known and can be estimated.
- Prior probability represents probability of a class before observing any data.

Ex- $P(\text{fish is salmon}) = P(\omega_1)$
$P(\text{fish is sea bass}) = P(\omega_2)$
For c classes, $\sum_{i=1}^{c} P(\omega_i) = 1$

- When priors are significantly different, choose ω_1 if $P(\omega_1) > P(\omega_2)$ else choose ω_2
- Baye's theorem is used to calculate posterior probability

$$P(h/D) = \frac{P(D/h)\,P(h)}{P(D)}$$

Where $P(h/D)$ → posterior
$P(D/h)$ → likelihood
$P(h)$ → prior probability
$P(D)$ → evidence

- Classifier assigns a sample to the class with highest posterior probability. (Maximum A Posteriori Hypothesis)

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}}\, P(h/D)$$

$$= \operatorname{argmax}\, P(D/h)P(h)$$

- Error probability measures likelihood that classifier incorrectly classifies a data sample.

$$P_e = \frac{\text{No. of misclassified instances}}{\text{Total no. of instances}}$$

- Error Rate measures how frequently classifier makes mistakes

$$\text{Error Rate} = \frac{FP + FN}{\text{Total}}$$

$$\text{Accuracy} = \frac{TP + TN}{\text{Total}}$$

- Minimum Distance Classifier assigns data point to class whose mean (centroid) is closest to the point.

(Pros) Simple to implement, computationally efficient, works well for well-seperated datasets

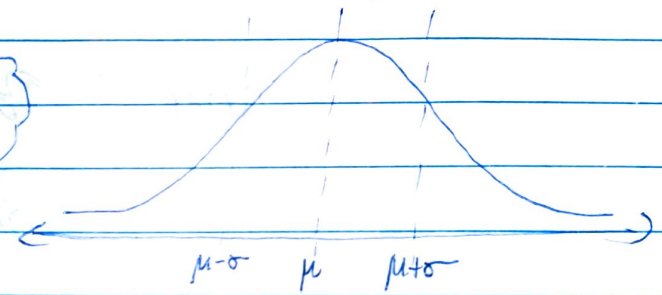
(Cons) Bad when classes overlap significantly, sensitive to feature scaling, bad with complex distributions

① Compute centroid (mean) of each class.
② For each new point x, compute distance to each centroid
③ Assign x to the closest centroid

Gaussian Distribution & Mixture Model (GMM)

- Continuous probability distribution that is symmetric and bell-shaped

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$



μ → mean (center of distribution)
σ → standard deviation (spread), σ^2 → variance

- GMM is a probabilistic model where data is generated from multiple Gaussian distributions. Each Gaussian distribution represents a component or cluster within the data

① Assign points to Gaussian clusters (assuming random Gaussians and then assigning points to them probabilistically) (Soft)
② Fitting Gaussians by computing mean (μ) (avg of points) and covariance matrix, $\Sigma = \begin{bmatrix} Var(x) & Cov(x,y) \\ Cov(x,y) & Var(y) \end{bmatrix}$ which controls shape (ellipse) and orientation.

- Variance → spread
Covariance → relationship between features.

EM Algorithm (Expectation - Maximization)

- Powerful iterative method used to find maximum likelihood estimates of parameters in statistical models, particularly when the model depends on unobserved latent variables.

- Useful for data imputation (filling missing data), unsupervised learning (clustering), HMM parameter estimation, etc.

- In real-world scenarios, we may have many features, though only a small subset are directly observable, the remaining latent variables are inferred/guessed from observed data.

1. Initial Step: Assume a set of initial values for the parameters.
2. Expectation (E): Use current observed data to estimate/guess the values of missing or incomplete data.
3. Maximization (M): Take the complete (observed + guessed) data to update parameters of our model
4. Convergence: System checks if values stopped changing significantly

Pros: Likelihood increases each step, ease of implementation, works with hidden variables

Cons: Slow convergence, can get stuck at local maxima, may require both fwd and bkwd probabilities.

Ex: Scenario: We have two coins A and B

Observations: HTTTHHTHTH → B

HHHHTHHHHH → A

HTHHHHHTHH → A

HTHTTTHHTT → B

THHHTHHHTH → A

- Problem: We do not know which coin was used for which set (missing label)

Let's start with our initial guesses for probability of obtaining heads: $\hat{\theta}_A^{(0)} = 0.60$, $\hat{\theta}_B^{(0)} = 0.50$

① HTTTHHTHTH (5H, 5T)

$P(A) = (0.6)^5 \times (0.4)^5$

$P(B) = (0.5)^5 \times (0.5)^5$

$W_A = \frac{P(A)}{P(A)+P(B)} = 0.45$ / $W_B = \frac{P(B)}{P(A)+P(B)} = 0.55$

② HHHHTHHHHH (9H, 1T)

$P(A) = (0.6)^9 \times (0.4)^1$

$P(B) = (0.5)^9 \times (0.5)^1$

$W_A = 0.80$, $W_B = 0.20$

③ HTHHHHHTHH (8H, 2T)

$P(A) = (0.6)^8 \times (0.5)^2$

$P(B) = (0.5)^8 \times (0.5)^2$

$W_A = 0.73$, $W_B = 0.27$

④ HTHTTTHHTT (4H, 6T)

$P(A) = (0.6)^4 \times (0.4)^6$

$P(B) = (0.5)^4 \times (0.5)^6$

$W_A = 0.35$, $W_B = 0.65$

⑤ THHHTHHHTH (7H, 3T)

$P(A) = (0.6)^7 \times (0.4)^3$

$P(B) = (0.5)^7 \times (0.5)^3$

$W_A = 0.65$, $W_B = 0.35$

	Coin A	Coin B
①	$(0.45 \times 5H), (0.55 \times 5T) =$	$(0.55 \times 5H), (0.55 \times 5T) =$
	$2.2H, 2.2T$	$2.8H, 2.8T$
②	$(0.8 \times 9H), (0.8 \times 1T)$	$(0.2 \times 9H), (0.2 \times 1T)$
	$7.2H, 0.8T$	$1.8H, 0.2T$
③	$5.9H, 1.5T$	$2.1H, 0.5T$
④	$1.4H, 2.1T$	$2.6H, 3.9T$
⑤	$4.5H, 1.9T$	$2.5H, 1.1T$
	$= 21.3H, 8.6T$	$= 11.7H, 8.4T$

$\therefore \hat{\theta}_A^{(1)} = \frac{21.3}{21.3+8.6} = 0.71$ / $\hat{\theta}_B^{(1)} = \frac{11.7}{11.7+8.4} = 0.58$

- After 10 iterations, $\hat{\theta}_A^{(10)} = 0.80$
  $\hat{\theta}_B^{(10)} = 0.52$

$\therefore$ Truth finally revealed. Coin A is biased towards heads (80%) and Coin B is a fair coin (52% heads) (Mathematically likely values)

Multivariate Data

- Datasets containing more than one variable / attribute per observation. (n-dimensional space)

- In univariate space, distance of point from mean $\bar{x}$ is just $(x_i - \bar{x})$.
  In multivariate space, we use Euclidean / straight-line distance
  $= \sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2 + \ldots + (n_i - \bar{n})^2}$

- However, it assumes all variables are independent. Real data often has covariance.

(PTO → Next Subject)

- Covariance is a measure of how much two variables change together

- Covariance - Variance Matrix (Symmetrical) (S) ⇒

	x	y	z
x	var(x)	cov(x,y)	cov(x,z)
y	cov(y,x)	var(y)	cov(y,z)
z	cov(z,x)	cov(z,y)	var(z)

- Mahalanobis Distance takes the data's distribution (covariance) into account.

$$D = \sqrt{(x-\bar{x})^T S^{-1} (x-\bar{x})}$$

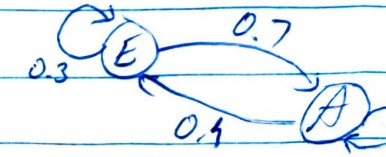
where $(x-\bar{x})$ → how far point is from the mean in each dimension

Markov Chain

- Stochastic model depicting a sequence of possible events, undergoing transitions from one state to another among finite no. of possible states.

- Prediction for next state is solely based on the current state, not sequence of events before it (memoryless)

- Probability (outward) sum from any single state = 1



- Applications: Search engines (PageRank), speech recognition, data science (modelling sequential data), MCMC (Markov Chain Monte Carlo method to solve complex problems thru normalization factors in statistics)

- Properties: (i) Irreducible (A→B in any no. of steps)
  (ii) Periodic (returning to state in finite multiples of specific time step integers)
  (iii) Transient (~~always~~ possibility of not coming back)
  (iv) Recurrent (guaranteed to eventually come back)
  (v) Absorbing (once you enter this state, you can never leave, no outgoing transitions)

☆ Non-linear Decision Boundaries

- When data is complex and overlapping in a way that no straight line can seperate them, linear boundaries fail while non-linear boundaries can take any shape, wrapping around the data points better.

- Techniques: (i) Kernel Methods (map data into higher dimensional space where linear seperation is possible)
  (Polynomial, RBF, Sigmoid)
  (ii) Decision Trees & Random Forests
  (split piecewise linear) (avg many trees → smooth, complex boundary)
  (iii) Neural Networks
  (iv) Polynomial Regression
  (v) Local Models (k-Nearest Neighbours)

☆ Linear Discriminant Analysis (LDA)

- Supervised Learning technique used for classification & dimensionality reduction. It projects data from higher dimensional space to lower dimensional space while maxing

Seperability between diff classes.

- Suppose we have datapoints belonging to two different classes in 2D plane such that not a single straight line can completely seperate them. LDA solves this by projecting the 2D datapoints onto a new 1D axis such that:

(a) Distance between centers (mean) of the two classes is maximum
(b) Variance between points in each class is minimum (tightly clustered)

- Fisher's Discriminant Analysis (FDA) is simply LDA when there are only 2 classes, related to Multiple Regression. LDA is a direct extension of FDA to any no. of classes, using matrix algebra to find best discriminants.

① Compute Class Mean (μ_i): $\mu_i = \frac{1}{N_i} \sum_{x \in w_i} x$
② Derive Covariance Matrix
③ Compute Within-Class Scatter Matrix (S_W), which is the sum of individual class scatter matrices ($S_1 + S_2$)

$$S_i = \sum (x - \mu_i)(x - \mu_i)^T$$

④ Compute Between-Class Scatter Matrix (S_B); which is the distance between means of different classes.

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

⑤ Compute Eigen values and Eigen Vectors: $S_W^{-1} S_B W = \lambda W$
(direction that maximizes seperation)

(6) Sort eigenvalues in descending order and select top k corresponding eigenvectors

(7) Take dot product of select eigen-vectors and original data to get new, reduced coordinates. (LDA)

$$(S_W^{-1} S_B - \lambda I)\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = 0$$

Ex: Sample for class ω_1: $X_1 = \{(6,2), (2,4), (2,3), (3,6), (4,4)\}^T$
Sample for class ω_2: $X_2 = \{(9,10), (6,8), (9,5), (8,7), (10,8)\}^T$

$$\mu_1 = \frac{1}{5}\sum_{x \in X_1} x = (3, 3.8)$$

$$\mu_2 = \frac{1}{5}\sum_{x \in X_2} x = (8.4, 7.6)$$

$$(X_1 - \mu_1) = \{(1, -1.8), (-1, 0.2), (-1, -0.8), (0, 2.2), (1, 0.2)\}^T$$
$$(X_2 - \mu_2) = \{(0.6, 2.4), (-2.4, 0.4), (0.6, -2.6), (-0.4, -0.6), (1.6, 0.4)\}^T$$

$$S_1 = \frac{\sum (x_1 - \mu_1)(x_1 - \mu_1)^T}{N-1} = \begin{bmatrix} 1 & -0.25 \\ -0.25 & 2.2 \end{bmatrix}$$

$$S_2 = \frac{\sum (x_2 - \mu_2)(x_2 - \mu_2)^T}{N-1} = \begin{bmatrix} 2.3 & -0.05 \\ -0.05 & 3.3 \end{bmatrix}$$

$$S_W = S_1 + S_2 = \begin{bmatrix} 3.3 & -0.3 \\ -0.3 & 5.5 \end{bmatrix}$$

$$S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T = \begin{bmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{bmatrix}$$

$$S_W^{-1} = \begin{bmatrix} 0.3045 & 0.0166 \\ 0.0166 & 0.1827 \end{bmatrix}$$

$$|S_W^{-1} S_B - \lambda I| = 0 \Rightarrow \lambda_1 = 0, \quad \lambda_2 = 12.2$$

(no separation)

~~$\lambda = 0 \Rightarrow W_1 =$~~

$$S_W^{-1} S_B = \begin{bmatrix} 9.22 & 6.49 \\ 4.23 & 2.98 \end{bmatrix}$$

$$|S_W^{-1} S_B - \lambda I| = 0$$

$$\begin{vmatrix} 9.22 - \lambda & 6.49 \\ 4.23 & 2.98 - \lambda \end{vmatrix} = 0$$

$$(9.22 - \lambda)(2.98 - \lambda) - 27.4527 = 0$$

$$27.4756 - 12.2\lambda + \lambda^2 - 27.4527 = 0$$

$$\lambda^2 - 12.2\lambda + 0.0229 = 0$$

$$\lambda = 0,\ 12.2$$

→ $\lambda \Rightarrow 0 \Rightarrow (S_W^{-1} S_B - \lambda I) \begin{pmatrix} W_{11} \\ W_{12} \end{pmatrix} = 0$

$$9.22 W_{11} + 6.49 W_{12} = 0$$

$$4.23 W_{11} + 2.98 W_{12} = 0$$

$$W_{12} = -1.42 W_{11}$$

In LDA, $\sqrt{W_{11}^2 + W_{12}^2} = 1$, $W_{11} = 0.576$

$W_{12} = -0.818$,

$$W_1 = \begin{bmatrix} 0.576 \\ -0.818 \end{bmatrix}$$

→ $\lambda = 12.2 \Rightarrow (S_W^{-1} S_B - \lambda I) \begin{pmatrix} W_{21} \\ W_{22} \end{pmatrix} = 0$

$$\begin{bmatrix} -2.98 & 6.49 \\ 4.23 & -9.22 \end{bmatrix} \begin{bmatrix} W_{21} \\ W_{22} \end{bmatrix} = 0$$

$$-2.98 W_{21} + 6.49 W_{22} = 0$$

$$4.23 W_{21} - 9.22 W_{22} = 0$$

$$W_{21} = 2.177 W_{22}$$

Normalized, $W_{22} = 0.4174$

$W_{21} = 0.9086$

$$W_2 = \begin{bmatrix} 0.9086 \\ 0.4174 \end{bmatrix} = W^* \quad \text{(chosen highest eigenvalue)}$$

X_1	4	2	2	3	4	9	6	9	8	10
X_2	2	4	3	6	4	10	8	5	7	8
LDA	4.46	3.48	3.06	5.2	5.3	12.35	8.8	10.2	10.19	12.42

Standardization (Z-score normalization)

- Rescaling features so that they have a mean ($\mu = 0$) and a standard deviation (σ) of 1.
- To center datapoints around 0, to ensure spread (variance) is within the same range, to make differently measured variables in diff units directly comparable
- Benefits algorithms using Gradient Descent to find minimum faster when on similar scale, also ensure fair penalization when preventing overfitting.
- Only calculate μ and σ on training set, and use only these on test set standardization to avoid Data Leakage.

$$z = \frac{x - \mu}{\sigma}$$

Normalization

- Scaling technique used to adjust numeric columns to a common scale ($[0, 1]$, $[-1, 1]$)
- Min-Max Scaling: $[0, 1]$

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- Robust to outliers, reversible, prevents data leakage, uniform, improves distance-based algorithms

Ensemble Learning

- Involves using a set of multiple classifiers and combining their individual decisions to classify new samples

- Solves insufficient data problems, compensates for imperfection, more expressive on representing a complex true hypothesis, provides a range of opinions.

— Voting / Averaging

- Predictions are combined from multiple models
  Regression: Average of predictions
  Classification: Sum (majority vote) of the predictions

— Bootstrap Aggregation (Bagging)

- Special case of model averaging approach to improve stability and accuracy of algorithms ~~Bag~~, effectively decreasing variance and helps reduce overfitting.

- Here we create multiple subsamples of data (bootstrapping) such that each model (same-type) trains on a different subsample.

— Random Forests

- Constructs multiple decision trees during training and ultimately merges their predictions (average).

— Boosting

- Builds a strong classifier from a ~~series~~ of weak classifiers. ~~connected~~ in (sequential learning)

(importance)

1. Assign equal weights to all datapoints
2. Train a model and identify wrongly classified points
3. Increase the weights of those wrongly classified points to make the new model focus on them.
4. Repeat until desired result reached or max models added.

Feature	Boosting	Bagging
• Combination Type	Combines predictions of different types	Combines predictions of same type
• Main Goal	Decrease Bias	Decrease Variance
• Weighting	Models weighted by performance	All models have equal weight.
• Independence	Models influenced by previous ones	Models are independent of each other.

— AdaBoost (Adaptive)

- Uses stagewise addition of weak learners to create a strong one.

- Weight assigned to each learner (α).
  α is directly proportional to error of weak learner (higher accuracy = higher weight)

— Gradient Boosting

- Builds a model from the sum of weak algorithms in stages.
- First, do not train on the data, simple return mean of target variable. Then each subsequent learner is trained to predict the residuals (errors) of previous ensemble.
- η → shrinks contribution of each tree to prevent overfitting

$$y_{pred} = y_1 + (\eta * r_1) + (\eta * r_2) + \ldots\ldots + (\eta * r_n).$$

— XGBoost

- Regularized ~~Ada~~ version of Gradient Boosting, leading to high performance, scalability for large datasets, handles missing values, provides feature importance.

(Pros)

(Cons)
- However, computationally intensive, risk of overfitting on small data, requires careful hyperparameter tuning.

— Stacking & Blending

- Stacking: Multiple base models generate intermediate predictions, fed as input to meta-model to produce final output.
- Blending: Instead of k-fold cross-validation training like in stacking, uses simple holdout validation set.

CLUSTERING

- Clustering is a type of unsupervised learning method which draws references from datasets consisting of input data without labelled responses

- Finds meaningful structure within the data, explanatory underlying processes, generative features and inherent groupings

- Clustering is the task of dividing a population or set of datapoints into groups such that:

(a) Datapoints in the same group are more similar to each other (High Intra-cluster Similarity)
(b) Datapoints are dissimilar to points in other groups (High Inter-cluster Dissimilarity)

Collection of Patterns (Raw data) → [Clustering Algorithm] → Partition

- Clustering uses distance measures to assess how similar / different two datapoints are

$$\text{Distance} \propto \frac{1}{\text{Similarity}}$$

- For a set of n tuples in Database $(D) = \{t_1, t_2, t_3, \ldots, t_n\}$
and a set of k clusters $(C) = \{C_1, C_2, \ldots, C_k\}$
Clustering pattern is defined by mapping function $f: D \to C$
such that $sim(t_{within}, t_{within}) > sim(t_{within}, t_{outside})$

(i) Euclidean Distance (straight-line distance) (displacement)

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

(in n-D space)

(ii) Manhattan Distance (city-block, absolute difference)
(can only move parallel to axes in nD space)

$$d(p,q) = |q_1-p_1| + |q_2-p_2| + \dots + |q_n-p_n|$$

(iii) Jaccard Index / Coefficient (similarity between sets)

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

(iv) Minkowski Distance
(Generalized form of Euclidean and Manhattan)

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

$p = 1 \Rightarrow$ ~~Euclidean~~ Manhattan
$p = 2 \Rightarrow$ Euclidean

(v) Cosine Similarity : Measures angle between two vectors regardless of magnitude (length)

$$\text{Similarity} = \cos(\theta) = \frac{A \cdot B}{|A||B|}$$

If $\theta = 0°$, sim = 1, vectors point in same direction
If $\theta = 90°$, sim = 0, vectors are perpendicular to each other
If $\theta = 180°$, sim = -1, vectors point in opposite direction

Note: To judge if clustering is good, intra-cluster distance (Cohesion) should be minimized (tightly packed) and inter-cluster distance (Separation) should be maximized.

☆ Dunn Index (Higher = Better)

- Metric used to evaluate quality of clustering.
- Here, Inter-Cluster Distance (δ) is the distance between closest points of two different clusters ($\min(d(x,y))$)
  Intra-Cluster Diameter (Δ) is the distance between the furthest points within the same cluster ($\max(d(x,y))$)

$$DI = \frac{\text{Minimum Inter-Cluster Distance}}{\text{Maximum Intra-Cluster Distance}}$$

☆ K-Medoids Clustering

- Unlike K-Means which uses the calculated average as the center, ~~K-Mod~~ K-Medoids uses an actual data point as the cluster center (medoid) such that total distance to all other points in that same cluster is minimum.

Pros: Simple, fast, converges in fixed no. of steps, robust to outliers

Cons: Not great at finding non-spherical / weirdly shaped clusters, also results can vary due to random initialization

☆ Hierarchial Clustering

- We build a tree of clusters that show how they are related to each other

(a) Agglomerative (Bottom-Up): Start with individual points and merge them into larger clusters.

(b) Divisive (Top-Down): Start with one giant cluster and split it into smaller sub-clusters

- Linkage = way of measuring distance between two clusters

① Single Linkage: Minimum distance between two points in different clusters.

$$L(R,S) = \min(D(i,j)) \text{ where } i \in R, j \in S$$

② Complete Linkage: Maximum distance between two points in different clusters.

$$L(R,S) = \max(D(i,j)) \text{ where } i \in R, j \in S.$$

③ Average Linkage: Arithmetic mean of distance between every possible pair of points from each cluster

$$L(R,S) = \frac{1}{n_R + n_S} \sum_{i=1}^{n_R} \sum_{j=1}^{n_S} D(i,j)$$

where n_R and n_S are no. of points in each cluster

Eg: (Agglomerative Clustering) (Single linkage)

Input: Proximity Matrix :

	A	B	C	D	E
A	0	9	3	6	11
B	9	0	7	5	10
C	3	7	0	9	2
D	6	5	9	0	8
E	11	10	2	8	0

① Find minimum non-zero value and merge into one
$(C \leftrightarrow E) = \boxed{2}$

	A	B	CE	D
A	0	9	3	6
B	9	0	7	5
CE	3	7	0	8
D	6	5	8	0

$L(CE, A) = \min(d(C,A), d(E,A)) = \min(3, 11)$
$= 3 \Rightarrow 3$

$L(CE, B) = \min(d(C,B), d(E,B)) = \min(7, 10)$
$= 7$

$L(CE, D) = \min(d(C,D), d(E,D)) = \min(9, 8)$
$= 8$

② Minimum $\Rightarrow (CE \leftrightarrow A) = \boxed{3}$, Merge ACE

	ACE	B	D
ACE	0	7	6
B	7	0	5
D	6	5	0

$L(ACE, B) = \min(d(A,B), d(C,B), d(E,B))$
$= \min(9, 7, 10) = 7$

$L(ACE, D) = \min(d(AD), d(C,D), d(E,D))$
$= \min(6, 9, 8) = 6$

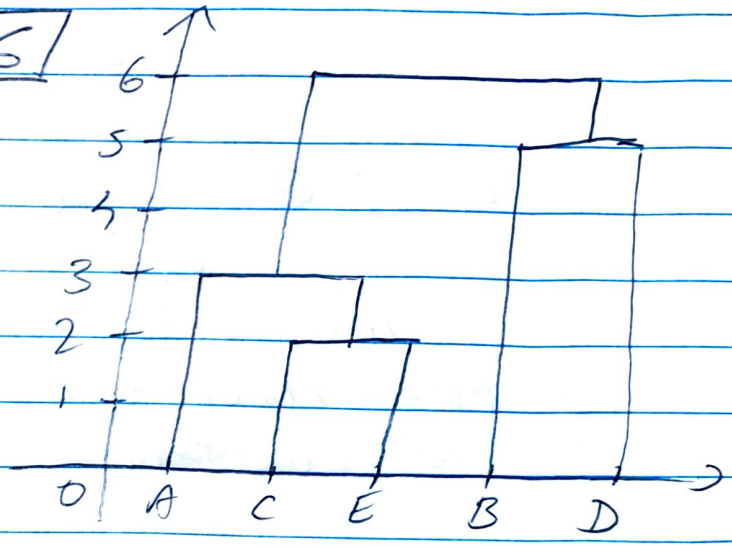
③ Merge BD = 5

	ACE	BD
ACE	0	6
BD	6	0

$L(ACE, BD) = \min(d(A, BD), d(CE, BD))$
$= \min(d(ACE, B), d(ACE, D))$
$= 6$

④ Merge ABCDE = 6

⑤ Plot Dendogram



Ex: (Complete Linkage)
(Same Proximity Matrix)

① Find minimum non-zero value and merge
CE = 2

	A	B	CE	D
A	0	9	11	6
B	9	0	10	5
CE	11	10	0	9
D	6	5	9	0

$L(CE, A) = \max(d(C,A), d(E,A)) = \max(3, 11) = 11$
$L(CE, B) = \max(d(C,B), d(E,B)) = \max(7, 10) = 10$
$L(CE, D) = \max(d(C,D), d(E,D)) = \max(9, 8) = 9$

② Merge BD = 5

$L(BD, A) = \max(d(B,A), d(D,A))$
$= \max(9, 6) = 9$

$L(CE, A) = \max(d(C,A), d(E,A))$
$= \max(3, 11) = 11$

	A	BD	CE
A	0	9	11
BD	9	0	10
CE	11	10	0

$L(BD, CE) = \max(d(CE, B), d(CE, D)) = \max(10, 9) = 10$

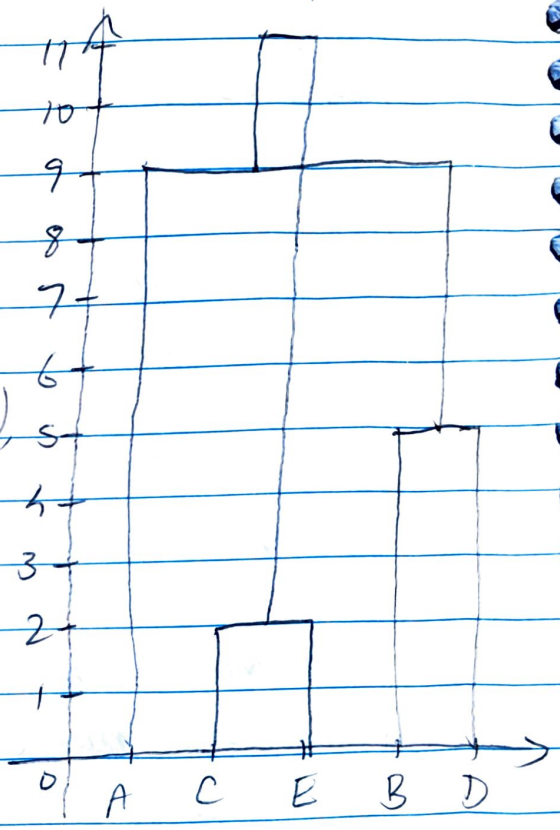
③ Merge ABD = 9

	ABD	CE
ABD	0	11
CE	11	0

$L(ABD, CE) = \max(d(CE, A), d(CE, B), d(CE, D)) = \max(11, 10, 9) = 11$

④ Merge ABCDE = 11

⑤ Plot Dendogram



Ex: (Average Linkage)

Points	x	y
A	1	4
B	1	2
C	0	4
D	6	1
E	7	0

⇒

	A	B	C	D	E
A	0	2	1	5.83	7.21
B	2	0	2.23	5.1	6.32
C	1	2.23	0	6.71	8.06
D	5.83	5.1	6.71	0	1.41
E	7.21	6.32	8.06	1.41	0

① Minimum = CA = 1

$L(C,A) = \text{avg}(d(C,A), d(E,A))$

=

	A	B	C	D
A	0	2		5.83
B	2	0		5.1
CE			0	
D	5.83	5.1		0

	AC	B	D	E
AC	0	2.12	6.27	7.685
B	2.12	0	5.1	6.32
D	6.27	5.1	0	1.41
E	7.685	6.32	1.41	0

$L(AC,B) = \text{avg}(d(A,B), d(C,B))$

$= \frac{2 + 2.24}{2 \times 1}$ $(n_{AC} = 2, n_B = 1)$

$= 2.12$

$L(AC,D) = \text{avg}(d(A,D), d(C,D))$

$= \frac{5.83 + 6.71}{2 \times 1} = 6.27$

① Min: DE = 1.41

$L(AC,E) = \text{avg}(d(A,E), d(C,E))$

$= \frac{7.21 + 8.06}{2 \times 1} = 7.685$

	AC	B	DE
AC	0	2.12	6.95
B	2.12	0	5.71
DE	6.95	5.71	0

$L(AC,DE) = \text{avg}(d(A,D), d(A,E), d(C,D), d(C,E))$

$= \frac{5.83 + 7.21 + 6.71 + 8.06}{2 \times 2}$

③ Min: ABC = 2.12

$= 6.95$

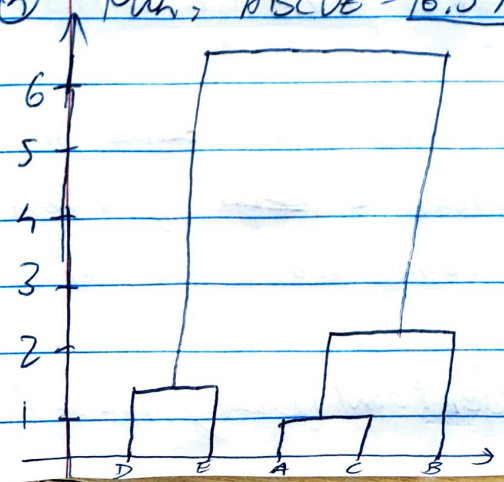
	ABC	DE
ABC	0	6.54
DE	6.54	0

$L(B,DE) = \frac{5.1 + 6.32}{2 \times 1} = 5.71$

⑤ Min: ABCDE = 6.54

$L(ABC,DE) = \frac{5.83 + 7.21 + 5.1 + 6.32 + 6.71 + 8.06}{3 \times 2}$

$= 6.54$



⑤ Plot Dendogram

☆ Minimum Spanning Tree (MST)

- Graph-based method used to construct clusters hierarchy

① Start with a point and the closest neighbouring point
② Continue until all points are connected in a tree with no loops
③ To clusters, remove the largest edge from the MST graph one-by-one

☆ DBSCAN Clustering

- Density-Based Spatial Clustering of Applications with Noise

- Finds groups based on how many datapoints are packed together in a specific area. No need to know no. of clusters (k) in advance ~~beforehand~~ like in K-Means.

- Also can find clusters of weird-ahh shapes too and highly robust to outliers and noise.

- DBSCAN classifies every point into:

(a) Core Point: Has more than MinPts points within its Σ radius
(b) Border Point: Has less than MinPts points ~~within~~ as neighbours but within Σ ~~range~~ radius of a Core point.
(c) Noise (Outlier): Neither Core Point nor a Border Point.

- Core Point (Party Host), Border Point (Guest), Noise (Loner)

- Σ defines the distance around a datapoint.
If distance between points $< \Sigma$, they are neighbours.
Too small → data can be largely classified as noise
Too large → clusters will merge together, one giant group.

- Min Pts is minimum no. of points required within Σ radius to consider that area dense. (including itself)
Min Pts $\geq D + 1$ (D → no. of dimensions)
Min Pts ≥ 3 (minimum)

- For each Core Point that isn't in a cluster, create a new cluster.

- If A is neighbours of B and B is neighbours of C then A and C are densely connected.

- Any remaining points not assigned to a cluster are marked as Noise.

Ex: Min Pts = 4
$\Sigma = 1.9$

Core Points
(P_2, P_5, P_{11})

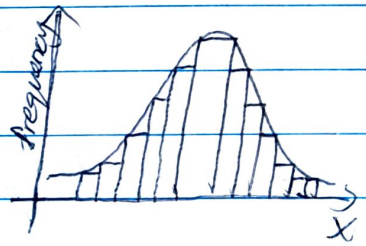
(Assigning Border points to those belonging to Core Points)

$P_1 \rightarrow P_2$, P_{10} (Noise → Border)
$P_2 \rightarrow P_1$, P_3, P_{11} (Core)
$P_3 \rightarrow P_3$, P_5 (Noise → Border)
$P_5 \rightarrow P_3$, P_6, P_7, P_8 (Core)
$P_6 \rightarrow P_5$, P_7 (Noise → Border)
$P_7 \rightarrow P_5$, P_6 (Noise → Border)
$P_8 \rightarrow P_5$ (Noise → Border)
$P_9 \rightarrow P_{12}$ (Noise)
$P_{10} \rightarrow P_1$, P_{11} (Noise → Border)
$P_{11} \rightarrow P_2$, P_{10}, P_{12} (Core)
$P_{12} \rightarrow P_9$, P_{11} (Noise → Border)

☆ Visualization of Datasets

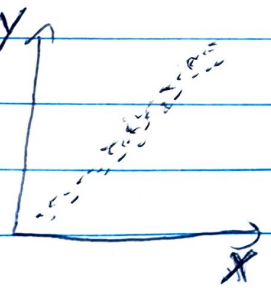
- To interpret complex data, identify trends, correlations and outliers effectively, understand underlying patterns, relationships and overall structure of data.

① **Histogram**: Representing (binned) grouped frequency distribution for continuous classes



- **Base**: Represents class intervals
  Area: Proportional to frequency of variables in that class
  Height: Proportional to frequency (for similar classes) or frequency density (for diff class widths)

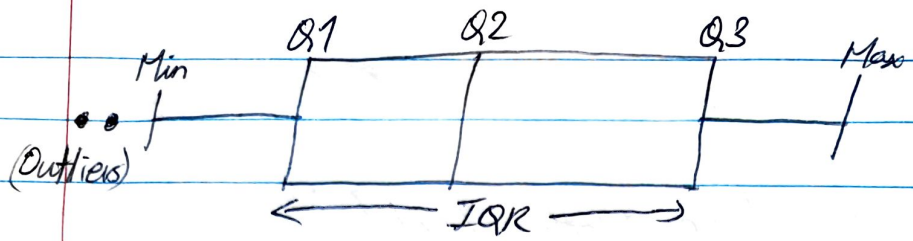
② **Scatter Plot**: Used to observe and show relationships between two variables



- Horizontal axis → Independent variable
  Vertical axis → Dependent variable

③ **Box Plots**: Displays distribution of numerical variable through a 5 number summary.

(i) Minimum (lowest value)
(ii) First Quartile (Q1) (25th percentile)
(iii) Median (Q2) (middle value) (50th percentile)
(iv) Third Quartile (Q3) (75th percentile)
(v) Maximum (highest value)



- Interquartile Range (IQR): $IQR = Q3 - Q1$
- Low ~~Quartile~~ Outliers: Any value $< Q1 - (1.5 \times IQR)$
  High Outliers: Any value $> Q3 + (1.5 \times IQR)$

(4) Bar Charts: Used to display and compare discrete categories.

(5) Heatmap: Depicts value for main variable across two axis variables using a grid of colored squares. Color intensity indicates value magnitude in that cell.

(6) Pair Plot: Plots pairwise relationships between all variables in a dataset at once.

(7) Line Charts: Illustrate trends over time by connecting data points with straight lines

(8) Area Charts: Similar to line charts but space below line is filled to emphasize the magnitude of values over time

(9) Maps: Show ~~of~~ spatial variation ~~using~~ in geospatial data.

(10) Multidimensional data can be represented using parallel coordinates (each dimension represented by vertical line, connections between lines show individual points), Radial Charts and T-SNE and PCA (techniques to reduce high-dimensional data into 2D/3D scatter plots)

(11) Word Clouds: Word size reflects frequency in a text

(12) Network Data: Use Nodes (entities) and Edges (relationships) to show interconnections

(13) Hierarchical Data - Dendograms (cluster arrangements)
Tree Maps (nested rectangles → branches of a hierarchy)

☆ Storage Capacity of Memories

- Model's ability to remember patterns (neural networks)
More weights/biases allow for more complex memory but require more data. Deeper layers increase what the network can model.

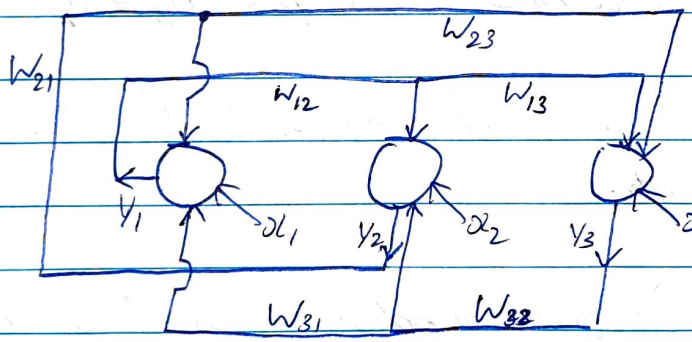
- Meanwhile, Memory networks (LSTM) are designed to hold info over time, using memory units of a specific size.
~~Sequence length is ability~~
- Ability to remember beginning of sentence when processing the end is crucial for memory networks.

☆ Associative Memory (Content Addressable Memory)

- Unlike normal RAM (accessed by address), here it is accessed by content, even if you give it a partial pattern, it returns the whole stored pattern (auto-associative)

- Hetero-associative memory associates one pattern with a different one.

☆ Hopfield Network

- Recurrent Neural Network used for associative memory that stores patterns like memories so that it can recover the original stored pattern from noisy inputs.
- Associative Memory works such that one pattern triggers another related pattern like human memory associations
- Hopfield Networks are fully connected RNNs where output feeds back into the network unlike feedforward NN.
- 

$W_{ij} = W_{ji}$ (Symmetric)

$W_{ii} = 0$ (no self connections)
- Each neuron recieves input / sends output to all others
- I/O ~~Output~~ Types : Binary $\{0, 1\}$
  Bipolar $\{-1, 1\}$ (preferred)
- Energy, $E = -\frac{1}{2} \sum_i \sum_j w_{ij} x_i x_j$ (for bipolar network)

  should be minimized to reach stable state.
- Training a Hopfield Network uses the idea of Hebbian learning rule (neurons firing together become strongly connected)

(Discrete)

Ex: Given: Stored pattern $[1\ 1\ 1\ 0]$

Convert to bipolar: $[1\ 1\ 1\ -1]$

① Compute Weight Matrix

For binary patterns: $w_{ij} = \sum_{p=1}^{P} [2s_i(p)-1][2s_j(p)-1]$

For bipolar patterns: $w_{ij} = \sum_{p=1}^{P} [s_i(p)\, s_j(p)] \quad \forall i \neq j$

$$W = XX^T = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{bmatrix}$$

② Remove Self Connections,

$$W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

③ For given noisy input, $N_0^{(0)} = [-1\ -1\ 1\ -1]$

(i) $y_1 = \sum w_{1j} N_j^{(0)} = -1(0) - 1(1) + 1(1) - 1(-1)$

$= 1$

$y_1 = \text{sgn}(1) = 1$ ~~no change~~ (update)

$N^{(1)} = [1\ -1\ 1\ -1]$

(ii) $y_2 = 1(1) - 1(0) + 1(1) - 1(-1) = 3$

$y_2 = \text{sgn}(3) = 1$ (update)

$N^{(2)} = [1\ 1\ 1\ -1]$

(iii) $y_3 = 1(1) + 1(1) + 1(0) + (-1)(-1) = 3$

$y_3 = \text{sgn}(3) = 1$ (no update)

$N^{(3)} = [1 \quad 1 \quad 1 \quad -1]$

(iv) $y_4 = 1(-1) + 1(-1) + 1(-1) + (-1)(0) = -3$

$y_4 = \text{sgn}(-3) = -1$ (no change)

$\therefore N^{(4)} = [1 \quad 1 \quad 1 \quad -1] = X$ (stored)

- The issue with discrete Hopfield network is that sudden jumps can happen in output due to small changes (not smooth)
- Continuous Hopfield solves this by allowing continuous values, $V_i = g(u_i)$ where g → activation function (Sigmoid $[0, 1]$)
- Neuron states evolve continuously over time (smoothly. (more stable + smooth convergence + better optimization)

(Pros)
- Error Correction (recovers noisy patterns)
- Associative Memory (store + recall memories)
- Scheduling / Optimization problem uses this

(Cons)
- Limited Storage Capacity
- Spurious states (fake unwanted memories can appear)
- Slow Convergence

☆ Inter-Class Distance-Based Criterion Functions

- Suppose we have 2 classes A and B. We want features that separate the classes far apart and keeps samples inside each class close together.

(Large) - Interclass distance is usually measured between class means. If means are apart, classes are easier to separate and better classification.

AAABBB (Bad) AAA BBB (Good)

(Small) - Intracluster Variance is spread inside each class. Small variance implies that points are tightly packed and compact class, while Large variance implies messy overlapping classes.

AAA (Good) A A A (Bad)

— - Fisher's Criterion (Higher = better)

$$J = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$$

$\mu_i \rightarrow$ class mean
$\sigma_i \rightarrow$ class variance

— Mahalanobis Distance

- Normal Euclidean distance ignores feature correlations and scaling differences. Mahalanobis fixes that.

$$D_M = \sqrt{(x_1 - x_2)^T \Sigma^{-1} (x_1 - x_2)}$$

Dictionary Learning

- Represent data using a small set of basis vectors and sparse coefficient instead of storing full data.

- Many signals/images contain repeated patterns and redundant information, hence we can compress them efficiently using
  (i) Dictionary Matrix (D) (basis vectors / building blocks)
  (ii) Sparse Coefficient Matrix (R)

such that $X \approx DR$, $X \rightarrow$ original data

$$X_{J \times k} = D_{J \times n} R_{n \times k}$$

- Dictionary learning tries to learn the best dictionary (D) and find sparse coefficients R such that reconstruction error is minimum:

$$\min \|X - DR\|_2^2 \quad \text{(square of L2 norm)}$$

- K-SVD (Single Valued Decomposition) is the algorithm used to update dictionary atoms efficiently.

- Applications: Image Compression, Denoising, Signal Recovery

★ Fuzzy Pattern Recognition

- In real-world, we cannot stick to the logic that an object/scenario belongs to one specific class/category. Overlap does occurs.

- Instead of classifying based on whether it belongs or not, we use degree of belonging, using membership functions

Ex: Fuzzy set = "Tall people" (A)

Height (x)	Membership	$= \mu_A(x)$
150 cm	0	
170 cm	0.5	$A = \{\mu_A(x)/x\}$
190 cm	1	

— Fuzzy Inference System

① Convert crisp values into fuzzy values (Fuzzification)

Temperature = 35°C ⟹ Hot = 0.7, Cold = 0.4

② Rule Processing (apply fuzzy rules)

If temperature is HOT and humidity is HIGH
THEN fan speed is FAST

③ Defuzzification (convert fuzzy result back into actual value)

FAN SPEED = 78 RPM instead of FAST/MEDIUM

- Fuzzy Set Operations include:

(a) Fuzzy Union: $\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$

(b) Fuzzy Intersection: $\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$

(c) Fuzzy Complement: $\mu_{A'}(x) = 1 - \mu_A(x)$

(d) Algebraic Sum: $\mu_{A+B} = \mu_A + \mu_B$

(e) Algebraic Product: $\mu_{AB} = \mu_A \times \mu_B$

(f) Absolute Difference: $\mu_{A-B} = |\mu_A - \mu_B|$

Pros: Handles ambiguity (partial truth), human-like reasoning and flexible

Cons: Complex Design, Subjective to expert definition, high computational cost.

Applications: Medical Diagnosis, Financial Risk, Image Processing, Control Systems.

-x-