UNIT-V ADVANCED LEARNING

1) K-means clustering:

K-means is a partitional clustering algorithm that divides a dataset into k-clusters based on similarity.

Steps:

1) Choose number of clusters k.
2) Randomly initialize k centroids
3) Assign each data point to the nearest centroid (using distance, usually Euclidean)
4) Recompute centroids as the mean of points in each cluster.
5) Repeat steps 3-4 until centroids do not change significantly.

Objective: Minimize $J = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$

where: C_i = cluster i

μ_i = centroid of cluster i

Applications:

- Customer segmentation
- Image compression.

2) Principal Component Analysis (PCA)

Principal component Analysis (PCA) is a dimensionality reduction technique used to transform high-dimensional data into fewer dimensions while preserving maximum variance.

Steps:

1) Standardizing the data.
2) Compute covariance matrix.
3) Find eigenvalues and eigenvectors.
4) Select the top k eigenvectors (principal components)
5) Transform data into new reduced feature space.

Example:

Dataset with 2 features:

x_1	x_2
2	0
0	2
3	1
1	3

Step 1: Standardizing (subtract mean = 1.5)

Step 2: Covariance Matrix

$$\Sigma = \begin{bmatrix} 1.67 & -1.0 \\ -1.0 & 1.67 \end{bmatrix}$$

First principal component aligns along direction [1, -1].

The data can be represented along one axis (v1, -v2) capturing most of the variance.

Advantages:

- Removes noise and redundancy.
- Reduces computation
- Improves visualization.

Applications:

- Face and handwriting recognition
- Image compression
- Exploratory data analysis.

3. GAUSSIAN MIXTURE MODELS (GMM)

Introduction:

A Gaussian Mixture Model(GMM) is a probabilistic model for representing normally distributed subpopulation within an overall population. Unlike k-means, it allows clusters of different shapes.

Model definition,

$$p(x) = \sum_{i=1}^{k} \pi_i N(x | \mu_i, \Sigma_i)$$

- π_i - mixing coefficient (weights)
- μ_i - mean vector
- Σ_i - covariance matrix

Learning Parameters:

Expectation- Maximization (EM) Algorithm:

1. Initialize means (μ), covariance (Σ) and weights (π)
2. E-step : compute probability each point belongs to each Gaussian.
3. M-step : Update μ, Σ, π based on those probabilities.
4. Repeat : until log-likelihood converges.

Example:

Consider 1D data : [1.0, 1.2, 1.4, 5.0, 5.2, 5.4]

We assume 2 Gaussians.

- Initial means $\mu_1 = 1.0$ $\mu_2 = 5.0$
- EM will adjust $\mu_1 = 1.2$, $\mu_2 = 5.2$ and computes σ, π
- Result : two overlapping normal curves fit the two clusters better than K-Means circles.

Advantages:

- Handles overlapping clusters.
- Provides probability of membership.

Applications:

- Speaker recognition
- Object detection
- Anomaly detection.

4. Q-Learning Algorithm [Reinforcement Learning]

Introduction:

Reinforcement Learning [RL] is learning through interaction. An agent learns to make a sequence of decisions by getting rewards from the environment. Q-learning is a model-free off policy RL algorithm that learns the optimal action-value function.

Components:

- State : environment's situation
- Action : what the agent can do
- Reward : numerical feedback
- Policy : mapping from states to actions.
- Q-value : expected future reward for (S, a).

Q-Learning Update Equation:

$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$

Example:

Grid world:

Agent starts at bottom-left and must reach top-right goal.

Actions: [Up, Down, Left, Right]

Reward: +10 for goal, -1 per step

Initially $Q(s,a) = 0$

When the agent moves and gets reward r, the Q-table updates using the formula.

After many episodes, the agent learns optimal actions (shortest path to goal).

Advantages:

- Learns without environment model.
- Works for stochastic tasks.

Applications:

- Game AI (chess, GO)
- Robotic navigation.
- Traffic signal optimization.

PROBLEMS BASED ON K-MEANS CLUSTERING:

cluster these points into K=2 clusters:

(2,3), (3,3), (6,6), (8,7).

step 1: Initialize centroids

$C_1 = (2,3)$, $C_2 = (6,6)$

step 2: Assign Points

Point	d(C1)	d(C2)	Assigned
(2,3)	0	5	C1
(3,3)	1	4.24	C1
(6,6)	4.24	0	C2
(8,7)	6.4	2.24	C2

clusters : C1 : [(2,3), (3,3)]

C2 : [(6,6), (8,7)]

Step 3: Recompute centroids:

$C_1 = (2.5, 3.0)$

$C_2 = (7.0, 6.5)$

step 4: Reassign Points → same clusters

$C_1 = [(2,3), (3,3)]$

$C_2 = [(6,6), (8,7)]$