Machine Learning

Assignment - 2

Sithishwar . B. M.

2023115062

Unit-5 : Advanced Learning

1) K-Means Clustering :

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

Steps :

1) Choose no of Clusters k.
2) Randomly initialize k centroids.
3) Assign each data points to the nearest centroid (using distance usually Euclidean).
4) Recompute centroids as the mean of points in each cluster.
5) Repeat steps 3-4 untill 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

Application :

* Customer Segmentation
* Image compression..

2023115062

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) Standardize the data
2) Compute covariance Matrix.
3) Find eigen values and eigen vectors
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: Standardize (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 $(x_1, -x_2)$ capturing most of the variance.

Advantages:

* Remove noise & redundancy
* Reduces computation.
* Improves visualisation.

Application:

* 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:

$$P(x) = \sum_{i=1}^{k} \pi_i N(x \mid \mu_i; \Sigma_i)$$

* π_i - mixing coefficient (weights).
* μ_i - mean vector
* Σ_i - Covariance matrix

Learning Parameter

Expectation Maximization (EM) Algorithm:

1. Initialize means (μ), Covariance (Σ) and weights (π_i)
2. E-step: Compute probability each point belongs to each Gaussian.
3. M-step: Update μ, Σ, π based on these 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 compute 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
* Anamoly detection

4. Q - Learning Algorithm [Reinforcement Learning]

Introduction:

Reinforcement Learning [RL] is learning through interation. An agent learns to make a sequence of divisions by receiving 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 action (shortest path to goal).

Advantages :

* Learns without environment model.
* Works for stochastic tasks.
* ~~Traffic signal~~

Application :

* Game AI (Chess, Go)
* Robotics 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 Centroid

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

Step 2 : Assign points.

Point	$d(C_1)$	$d(C_2)$	Assigned
(2,3)	0	5	C_1
(3,3)	1	4.24	C_1
(6,6)	4.24	0	C_2
(8,7)	6.4	2.24	C_2

Clusters : $C_1 = [(2,3), (3,3)]$
$C_2 = [(6,6), (8,7)]$

Step 3: Recompute Centroids :

$C_1 = (2.5, 3.0)$

$C_2 = (7.0, 6.5)$

Step 4 : Reassign points → Same clusters

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

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