**P.V.P Siddhartha Institute of Technology (Autonomous)**

**Department of Computer Science and Engineering**

**MACHINE LEARNING IV CSE SEM-II**

**MID-II Answers Academic Year: 2024-25**

**1) Illustrate any four measures to evaluate the Classification model with examples. (5M) (CO2-L3)**

### Ans: Measures to Evaluate a Classification Model:

1. **Accuracy**
   * **Definition:** Proportion of correctly predicted instances out of the total instances.
   * **Formula:** (TP + TN) / (TP + TN + FP + FN)
   * **Example:** If a model correctly predicts 90 out of 100 email messages as spam/not spam, then accuracy = 90%.
2. **Precision**
   * **Definition:** Proportion of true positive predictions out of all positive predictions made by the model.
   * **Formula:** TP / (TP + FP)
   * **Example:** In a disease detection model, if 8 out of 10 predicted positive cases are correct, precision = 80%.
3. **Recall (Sensitivity or True Positive Rate)**
   * **Definition:** Proportion of actual positives correctly identified.
   * **Formula:** TP / (TP + FN)
   * **Example:** If a model identifies 70 out of 100 actual fraud cases, recall = 70%.
4. **F1-Score**
   * **Definition:** Harmonic mean of precision and recall, useful when classes are imbalanced.
   * **Formula:** 2 × (Precision × Recall) / (Precision + Recall)
   * **Example:** If precision = 0.75 and recall = 0.60, then F1-score ≈ 0.67.

These measures help compare model performance, especially in cases of **imbalanced datasets** or **different types of errors (false positives vs false negatives).**

**2) Analyze the given data and Apply Support Vector Machine to plot hyper plane of the following data point on linearly separable data: (5M) (CO4-L4)**

**(1,1) (2,1) (1,-1) (2, -1) (4, 0) (5, 1) (5, -1) (6, 0)**

**Ans:** The SVM successfully separates the two classes with a **maximum margin hyperplane**. The support vectors determine the position of the hyperplane. This example shows a **clear linear separation** using SVM.

**Step 1: Understand and Visualize the Data:**

Given data points:

|  |  |
| --- | --- |
| **Point** | **Coordinates** |
| Class A | (1,1), (2,1), (1,-1), (2,-1) |
| Class B | (4,0), (5,1), (5,-1), (6,0) |

Let’s assign labels:

* Class A = **+1**
* Class B = **-1**

So, the dataset becomes:

| **X (Feature Vector)** | **y (Label)** |
| --- | --- |
| (1,1) | +1 |
| (2,1) | +1 |
| (1,-1) | +1 |
| (2,-1) | +1 |
| (4,0) | -1 |
| (5,1) | -1 |
| (5,-1) | -1 |
| (6,0) | -1 |

**Step 2: Plot the Points (for Understanding)**

* One on the **left side** (Class +1): centered around (1, ±1), (2, ±1)
* One on the **right side** (Class -1): centered around (4–6, 0, ±1)

These are **linearly separable**, so a **linear SVM** is suitable.

**Step 3: SVM - Find the Optimal Hyperplane**

VM seeks a hyperplane that separates the two classes with **maximum margin**.

A **2D hyperplane** is a line:

w1x1+w2x2+b=0

w\_1x\_1 + w\_2x\_2 + b = 0

w1​x1​+w2​x2​+b=0

We can calculate it using a basic idea of mid-point + perpendicular bisector (for a simple 2D, symmetric case).  
But for exact SVM hyperplane and margin, we use the SVM optimization formulation.

**3)** **Compute the final result of Hierarchical Clustering with the Complete Link by drawing a Dendrogram (5M) (CO3-L3)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **d** | **e** | **f** |
| **a** | 0 |  |  |  |  |  |
| **b** | 0.11 | 0 |  |  |  |  |
| **c** | 0.42 | 0.25 | 0 |  |  |  |
| **d** | 0.83 | 0.23 | 0.14 | 0 |  |  |
| **e** | 0.29 | 0.77 | 0.67 | 0.45 | 0 |  |
| **f** | 0.35 | 0.61 | 0.93 | 0.20 | 0.67 | 0 |

### Ans: Step 1: Interpret the Distance Matrix

We given the **lower triangular distance matrix** between 6 objects: A, B, C, D, E, F.

We first represent the **distance matrix in full form**:

|  | **A** | **B** | **C** | **D** | **E** | **F** |
| --- | --- | --- | --- | --- | --- | --- |
| A | 0 | 0.11 | 0.42 | 0.83 | 0.29 | 0.35 |
| B | 0.11 | 0 | 0.25 | 0.23 | 0.77 | 0.61 |
| C | 0.42 | 0.25 | 0 | 0.14 | 0.67 | 0.93 |
| D | 0.83 | 0.23 | 0.14 | 0 | 0.45 | 0.20 |
| E | 0.29 | 0.77 | 0.67 | 0.45 | 0 | 0.67 |
| F | 0.35 | 0.61 | 0.93 | 0.20 | 0.67 | 0 |

### Step 2: Initial Clusters

### Each object is its own cluster: {A}, {B}, {C}, {D}, {E}, {F}

### Step 3: Step-by-Step Clustering Using Complete Linkage

### **Complete Linkage**: Distance between clusters is the **maximum distance** between any pair of elements in the two clusters.

#### Step 3.1: Find the minimum distance

From matrix: **min = 0.11** between **A & B**

Merge: **Cluster1 = {A, B}**

#### Step 3.2: Update distances

Now compute distances from **Cluster1 (A,B)** to other elements using **complete link** (i.e., max distance):

| **To** | **max(d(A, x), d(B, x))** |
| --- | --- |
| C | max(0.42, 0.25) = 0.42 |
| D | max(0.83, 0.23) = 0.83 |
| E | max(0.29, 0.77) = 0.77 |
| F | max(0.35, 0.61) = 0.61 |

Other clusters: C, D, E, F (still singleton)

#### Step 3.3: Next min distance is 0.14 between C & D

Merge: **Cluster2 = {C, D}**

#### Step 3.4: Update distances

Now update distance from Cluster2 = {C, D} to others using complete link:

| **To** | **max(d(C,x), d(D,x))** |
| --- | --- |
| (A,B) | max(0.42, 0.83) = 0.83 |
| E | max(0.67, 0.45) = 0.67 |
| F | max(0.93, 0.20) = 0.93 |

#### Step 3.5: Minimum now is 0.45 between Cluster2 (C,D) and E

**Merge: Cluster3 = {C, D, E}**

#### Step 3.6: Update distances

Now compute distance from Cluster3 to others:

| **To** | **max(d(C,x), d(D,x), d(E,x))** |
| --- | --- |
| (A,B) | max(0.42, 0.83, 0.29) = 0.83 |
| F | max(0.93, 0.20, 0.67) = 0.93 |

#### Step 3.7: Next closest is 0.61 between Cluster1 (A,B) and F

**Merge: Cluster4 = {A, B, F}**

#### Sep 3.8: Final Merge

Now two clusters remain:

* Cluster3 = {C, D, E}
* Cluster4 = {A, B, F}

Distance between them is:

****

Find all:

C to A, B, F → (0.42, 0.25, 0.93)

D to A, B, F → (0.83, 0.23, 0.20)

E to A, B, F → (0.29, 0.77, 0.67)

Max of all these: **0.93**

Final Merge at **0.93**

