**SULTAN**

**56189**

**BS/DS-5_1**

**DW & BI**

**LAB WORK**

## Evaluation Metrics for Classification, Clustering, and Reinforcement Learning

---

## 1. Classification Evaluation Metrics

Evaluation metrics in classification quantify how well a predictive model assigns classes to instances. These metrics assess different dimensions such as correctness, error rate, class balance handling, and ranking ability.

---

### 1.1 Accuracy

#### Definition

Accuracy measures the proportion of correctly predicted instances out of all instances.

#### Formula

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

#### Interpretation

- Best used when classes are **balanced**.
- Misleading when dataset is **imbalanced**.

#### Python Implementation

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_true, y_pred)
```

---

### 1.2 Precision

#### Definition

Precision measures the proportion of predicted positives that are actually positive.

#### Formula

$$\text{Precision} = \frac{TP}{TP + FP}$$

**Interpretation**

- Useful when **false positives are costly** (e.g., spam detection).

**Python**

```
from sklearn.metrics import precision_score
precision = precision_score(y_true, y_pred)
```

---

## 1.3 Recall (Sensitivity / True Positive Rate)

**Definition**

Recall indicates how many actual positive instances the classifier correctly identified.

**Formula**

$$\text{Recall} = \frac{TP}{TP + FN}$$

**Interpretation**

- Important when missing a positive instance is costly (e.g., disease detection).

**Python**

```
from sklearn.metrics import recall_score
recall = recall_score(y_true, y_pred)
```

---

## 1.4 F1-Score

**Definition**

F1 score is the **harmonic mean** of Precision and Recall.

**Formula**

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

**Interpretation**

- Balances Precision and Recall.
- Useful for **imbalanced datasets**.

**Python**

```
from sklearn.metrics import f1_score
```

```
f1 = f1_score(y_true, y_pred)
```

---

## 1.5 Matthews Correlation Coefficient (MCC)

**Definition**

MCC measures the correlation between predicted and actual classes. It is considered one of the most reliable metrics for imbalanced classification.

**Formula**

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

**Interpretation**

- +1 → Perfect prediction
- 0 → Random prediction
- -1 → Completely wrong prediction

**Python**

```
from sklearn.metrics import matthews_corrcoef
mcc = matthews_corrcoef(y_true, y_pred)
```

---

## 1.6 Precision–Recall Curve (PRC)

**Definition**

A curve showing the tradeoff between Precision and Recall across thresholds.

**Key Points**

- Useful for **highly imbalanced** datasets.
- Area under PR curve helps evaluate ranking ability.

**Python**

```
from sklearn.metrics import precision_recall_curve
prec, rec, thr = precision_recall_curve(y_true, y_score)
```

---

## 1.7 ROC Curve & AUC

**Definition**

ROC curve plots True Positive Rate (TPR) vs. False Positive Rate (FPR).

#### Formula

$$TPR = \frac{TP}{TP + FN}$$
$$FPR = \frac{FP}{FP + TN}$$

**AUC (Area Under Curve)**
Evaluates the performance across all classification thresholds.

#### Interpretation

- AUC = 1 → Perfect classifier
- AUC = 0.5 → Random
- AUC < 0.5 → Worse than random

#### Python

```
from sklearn.metrics import roc_auc_score
auc = roc_auc_score(y_true, y_score)
```

---

## 2. Clustering Evaluation Metrics

Clustering metrics are divided into:

- **Internal Metrics** → Do not require ground truth labels
- **External Metrics** → Require true labels for comparison

---

### 2.1 Internal Clustering Metrics

These evaluate the quality of clusters based on the structure of the dataset itself.

---

### 2.1.1 Silhouette Score

#### Definition

Measures how well each point fits within its cluster.

**Formula**

$$S = \frac{b - a}{\max(a, b)}$$

Where:

- **a** = average intra-cluster distance
- **b** = nearest-cluster distance

**Interpretation**

- +1 → Well-clustered
- 0 → Overlapping
- Negative → Wrong cluster

**Python**

```
from sklearn.metrics import silhouette_score
sil = silhouette_score(X, labels)
```

---

### 2.1.2 Davies–Bouldin Index (DBI)

**Definition**

Measures average similarity between clusters (lower is better).

**Formula**

$$DBI = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)$$

**Python**

```
from sklearn.metrics import davies_bouldin_score
dbi = davies_bouldin_score(X, labels)
```

---

### 2.1.3 Dunn Index

**Definition**

Measures separation and compactness.

**Formula**

$$D = \frac{\text{Minimum inter-cluster distance}}{\text{Maximum intra-cluster distance}}$$

**Interpretation**

Higher Dunn index → Better clustering

(No built-in sklearn function)

---

## 2.2 External Clustering Metrics

These compare cluster labels with true labels.

---

### 2.2.1 Rand Index

**Definition**

Measures similarity between predicted and true labels.

**Formula**

$$RI = \frac{TP + TN}{TP + TN + FP + FN}$$

---

### 2.2.2 Adjusted Rand Index (ARI)

**Definition**

Corrects Rand Index for chance.

**Python**

```
from sklearn.metrics import adjusted_rand_score
ari = adjusted_rand_score(y_true, labels)
```

---

### 2.2.3 Homogeneity Score

**Definition**

Measures whether each cluster contains only members of a single class.

**Python**

```
from sklearn.metrics import homogeneity_score
homo = homogeneity_score(y_true, labels)
```

---

### 2.2.4 Completeness Score

**Definition**

Measures whether all members of a class are assigned to the same cluster.

**Python**

```
from sklearn.metrics import completeness_score
comp = completeness_score(y_true, labels)
```

---

### 2.2.5 V-Measure

**Definition**

Harmonic mean of Homogeneity and Completeness.

**Python**

```
from sklearn.metrics import v_measure_score
v = v_measure_score(y_true, labels)
```

---

### 2.2.6 RMSE (Root Mean Square Error)

**Definition**

Measures reconstruction error in centroid-based clustering (e.g., K-Means).

**Formula**

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (x_i - c_{label(i)})^2}$$

---

## 3. Reinforcement Learning Evaluation Metrics

Reinforcement Learning metrics evaluate policy quality, reward efficiency, and learning stability.

### 3.1 Cumulative Reward / Return

**Definition**

Total reward accumulated by the agent.

**Discounted Return Formula**

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

### 3.2 Value Function Error

**Definition**

Measures how far estimated value function is from true value.

**Formula**

$$MSE = \frac{1}{n}\sum(V(s) - V^{\pi}(s))^2$$

### 3.3 Policy Loss

**Definition**

Measures how well the policy improves over iterations.

**General Formula (Policy Gradient)**

$$L(\theta) = -\mathbb{E}_{\pi_\theta}[\log \pi_\theta(a \mid s)A(s,a)]$$

Lower loss → Better policy

### 3.4 Additional Important RL Metrics

**Success Rate**

Percentage of times agent achieves the goal.

**Episode Length**

Number of steps per episode.

**Stability (Variance of Reward)**

Measures consistency.

**GITHUB:**
https://github.com/sultanali543/Evaluation-Metrics-for-Classification-Clustering-and-Reinforcement-Learning