Evaluation Metrics for Supervised ML Models

Hands-on: Classification Metrics

Learning Objectives

By the end of this lesson, students will be able to:

- Compute key classification metrics such as **accuracy, precision, recall, and F1-score**.
- Interpret these metrics to assess the performance of a classification model.
- Apply Python’s `sklearn.metrics` module to evaluate a machine learning model.

Understanding Classification Metrics

Classification models predict **categorical outcomes** (e.g., “spam” vs. “not spam”). Different metrics help us understand how well a model performs:

Metric	Formula	Best Use Case
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	Works well when classes are balanced.
Precision	$\frac{TP}{TP+FP}$	Important when false positives are costly (e.g., fraud detection).
Recall	$\frac{TP}{TP+FN}$	Important when missing positives is costly (e.g., disease detection).
F1-score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$	Best when both precision and recall are important.

Where:

- **TP (True Positive)** = Correct positive predictions.
- **TN (True Negative)** = Correct negative predictions.
- **FP (False Positive)** = Incorrectly predicted as positive.
- **FN (False Negative)** = Incorrectly predicted as negative.

Hands-On: Computing Classification Metrics in Python

We will use **scikit-learn** to evaluate a simple classification model.

Step 1: Import Libraries

Copy

```
import numpy as np
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_sco
```

Step 2: Create Sample Predictions

Copy

```
# True labels (actual outcomes)
y_true = np.array([1, 0, 1, 1, 0, 1, 0, 0, 1, 0])
```

```
# Model's predicted labels
y_pred = np.array([1, 0, 1, 0, 0, 1, 1, 0, 1, 0])
```

Step 3: Calculate Key Metrics

Copy

```
accuracy = accuracy_score(y_true, y_pred)
precision = precision_score(y_true, y_pred)
recall = recall_score(y_true, y_pred)
f1 = f1_score(y_true, y_pred)

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"F1-score: {f1:.2f}")
```

Step 4: Get a Full Report

Copy

```
print("\nClassification Report:\n")
print(classification_report(y_true, y_pred))
```

Try It Yourself!

Modify `y_true` and `y_pred` to see how different predictions affect the metrics.

- What happens when **all predictions are correct**?
- What happens when the model **predicts only one class**?

Key Takeaways

- **Accuracy** is not always reliable when dealing with imbalanced datasets.

- **Precision** and **recall** offer deeper insights into model behavior.
- **F1-score** balances precision and recall when both are equally important.

Next Steps: Now that we’ve covered classification, let’s move to evaluating regression models.

< Previous

Attributions

Next >