Evaluate the classifier

First, we predict the labels for the testing data using the trained classifier:

y\_pred = clf.predict(X\_test)

Next step is to compute accuracy, precision, recall and F1 score:

* First, we need to import from sklearn.metrics:

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

* Accuracy measures the proportion of correctly classified instances out of the total number of instances. The formula to compute accuracy is as follows:

Accuracy = (Number of correctly classified instances) / (Total number of instances)

Code to compute accuracy:

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy)

* Precision is the ratio of true positive predictions to the total number of positive predictions. It measures the classifier's ability to correctly identify positive instances.

Code to compute accuracy:

precision = precision\_score(y\_test, y\_pred)

print("Precision:", precision)

* Recall (also known as sensitivity or true positive rate) is the ratio of true positive predictions to the total number of actual positive instances. It measures the classifier's ability to correctly capture positive instances.

Code to compute accuracy:

recall = recall\_score(y\_test, y\_pred)

print("Recall:", recall)

* The F1 score is a metric commonly used in binary classification tasks to evaluate the performance of a classifier. It is the harmonic mean of precision and recall. The formula to compute the F1 score is as follows:

F1 score = 2 \* (precision \* recall) / (precision + recall)

Code to compute F1 score:

f1 = f1\_score(y\_test, y\_pred)

print("F1 score:", f1)

The result:

1. Accuracy: 1.0
2. Precision: 1.0
3. Recall: 1.0
4. F1 score: 1.0

With an accuracy, precision, recall, and F1 score of 1.0, it indicates that the classifier is performing perfectly on the testing data. Achieving such high scores can also be an indication of overfitting. Overfitting occurs when the model learns the training data too well and fails to generalize to unseen data. It is most likely due to the current model is quite simple.