F1 Score in Machine Learning

Last Updated : 23 Jul, 2025

F1 Score is a performance metric used in machine learning to evaluate how well a classification model performs on a dataset especially when the classes are imbalanced meaning one class appears much more frequently than another. It is the harmonic mean of precision and recall which combine both metrics into a single value that balances their importance.

Before understanding F1 Score let's understand two key terms:

1. Precision:

It refers to the proportion of correct positive predictions (True Positives) out of all the positive predictions made by the model (True Positives + False Positives). It is a measure of the accuracy of the positive predictions. The formula for Precision is:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

For example, if there are 10 positive cases and 5 negative cases. The model can identify 5 positive cases. But out of these 5 identified cases only 4 are positive and 1 is negative. Thus precision becomes 80% (4/5).

2. Recall:

It is also known as Sensitivity or True Positive Rate where we measures the proportion of actual positive instances that were correctly identified by the model. It is the ratio of True Positives to the total actual positives (True Positives + False Negatives). The formula for Recall is:

Let's use the previous example. Although the model's **precision** is quite high at 80% the recall will be significantly lower. Given 10 actual positive cases the model only identified 4 positive cases correctly. Therefore the recall can be calculated as 40% (4/10).

> *To better understand Precision and Recall we can use a **Confusion Matrix** which summarizes the performance of a classifier in four essential terms:*
>
> - ***True Positives (TP)**: Correctly predicted positive instances.*
> - ***False Positives (FP)**: Incorrectly predicted positive instances.*
> - ***True Negatives (TN)**: Correctly predicted negative instances.*
> - ***False Negatives (FN)**: Incorrectly predicted negative instances.*

F1 Score by combining Precision and Recall

Now the F1 Score combines precision and recall using the harmonic mean:

$$F_1 \text{ Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

This formula ensures that both precision and recall must be high for the F1 score to be high. If either one drops significantly the F1 score will also drop.

Why We Use Harmonic Mean Instead of Simple Average?

Harmonic mean is preferred over the arithmetic mean because it better handles rates like precision and recall. It balances both metrics equally ensuring that both need to be high for a good F1 score. The harmonic mean helps combine precision and recall when their denominators differ by averaging their reciprocals and then transforming the result back. This approach is especially useful in imbalanced datasets where a low value in either precision or recall can significantly lower the F1 score.

Calculating F1 Score

1. Binary Classification

In binary classification where there are only two classes (positive and negative) the F1 score can be computed from the **confusion matrix** that helps calculate metrics such as precision, recall and the F1 score.

Let's take an example of a dataset with 100 total cases. Out of these 90 are positive and 10 are negative cases. The model predicted 85 positive cases out of which 80 are actual positive and 5 are from actual negative cases. The confusion matrix would look like:

Example	Actual		Total	
Model Prediction	80	5	85	Precision = (80/85) = 0.94
	10	5	15	
Total	90	10	100	
	Recall = (80/90) = 0.88		Accuracy = (80+5)/100 = 85%	F1 = 0.91

Let us see how does F1 score help when there is a class imbalance:

Example 1: Consider the below case where there are only 9 cases of true positives out of a dataset of 100.

	Actual		Total
Model Prediction	1	1	2

	Actual		Total
	8	90	98
Total	9	91	100

Precision 0.50
Recall 0.11
Accuracy 0.91
F1 0.18

In this case if we give importance to accuracy over model will predict

Example 2: However one must also consider the opposite case where the positives outweigh the negative cases. In such a case our model will try to predict everything as positive.

	Actual		Total
Model Prediction	90	8	98
	1	1	2
Total	91	9	100

Precision 0.92
Recall 0.99
Accuracy 0.91
F1 0.95

2. Multiclass Classification

In a multi-class classification problem where there are more than two classes we calculate the F1 score per class rather than providing a single overall F1 score for the entire model. This approach is often referred to as the one-vs-rest (OvR) or one-vs-all (OvA) strategy.

- For each class in the multi-class problem a binary classification problem is created.
- We treat one class as the positive class and the rest of the classes as the negative class.
- Then we proceed to calculate the F1 score as outlined above.
- For a specific class the true positives (TP) are the instances correctly classified as that class, false positives (FP) are instances incorrectly classified as that class and false negatives (FN) are instances of that class incorrectly classified as other classes.

This means that you train a separate binary classifier for each class considering instances of that class as positive and instances of all other classes as negative.

Once we have calculated the F1 score for each class we might want to aggregate these scores to get an overall performance measure for your model. Common approaches include calculating a micro-average, macro-average or weighted average of the individual F1 scores.

Implementing F1 Score in Python

We can easily calculate the F1 score in Python using the f1_score function from the **sklearn.metrics** module. This function supports both binary and multi-class classification.

Here's an explanation of the function and its parameters:

- **f1_score** function takes two required parameters: y_true and y_pred along with an optional parameter average.
- **y_true**: This parameter represents the true labels for the instances,

- **average:** This parameter defines the type of averaging performed on the data. It is a optional parameter.

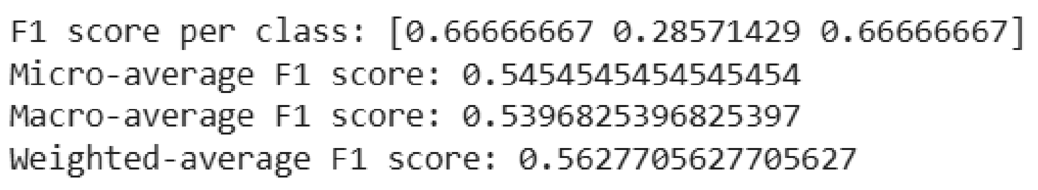
```python
from sklearn.metrics import f1_score

y_true = [0, 1, 2, 2, 2, 2, 1, 0, 2, 1, 0]
y_pred = [0, 0, 2, 2, 1, 2, 1, 0, 1, 2, 1]

f1_per_class = f1_score(y_true, y_pred, average=None)
f1_micro = f1_score(y_true, y_pred, average='micro')
f1_macro = f1_score(y_true, y_pred, average='macro')
f1_weighted = f1_score(y_true, y_pred, average='weighted')

print("F1 score per class:", f1_per_class)
print("Micro-average F1 score:", f1_micro)
print("Macro-average F1 score:", f1_macro)
print("Weighted-average F1 score:", f1_weighted)
```

Output:



```
F1 score per class: [0.66666667 0.28571429 0.66666667]
Micro-average F1 score: 0.5454545454545454
Macro-average F1 score: 0.5396825396825397
Weighted-average F1 score: 0.5627705627705627
```

Implementation of F1 Score

- **Micro-average**: Calculates metrics globally by counting the total true positives, false negatives and false positives.
- **Macro-average**: Averages the F1 score for each class without considering class imbalance.
- **Weighted-average**: Considers class imbalance by weighting the F1 scores by the number of true instances for each class.

Therefore F1 score provides a balanced evaluation of a model's performance especially when dealing with imbalanced datasets.

Comment More info