How can I calculate the balanced accuracy in Python using sklearn?

Authored by

stats writer

May 13, 2024

The balanced accuracy is a metric used to evaluate the performance of a classification model by taking into account the proportion of correctly classified instances for each class. In order to calculate the balanced accuracy in Python, the sklearn library offers a function called "balanced_accuracy_score". This function takes in the predicted labels and the true labels as inputs and returns the balanced accuracy score. By using this function, one can easily assess the overall performance of a classification model and make informed decisions on its effectiveness.

Calculate Balanced Accuracy in Python Using sklearn

Balanced accuracy is a metric we can use to assess the performance of a .

It is calculated as:

Balanced accuracy = (Sensitivity + Specificity) / 2

where:

Sensitivity: The "true positive rate" - the percentage of positive cases the model is able to detect.Specificity: The "true negative rate" - the percentage of negative cases the model is able to detect.

This metric is particularly useful when the two classes are imbalanced - that is, one class appears much more than the other.

For example, suppose a sports analyst uses a to predict

whether or not 400 different college basketball players get drafted into the NBA.

The following confusion matrix summarizes the predictions made by the model:

		Predicted	
		Drafted = Yes	**Drafted = No**
Actual	**Drafted = Yes**	**15** (True Positive)	**5** (False Negative)
	Drafted = No	**5** (False positive)	**375** (True Negative)

To calculate the balanced accuracy of the model, we'll first calculate the sensitivity and specificity:

Sensitivity: The "true positive rate" = 15 / (15 + 5) = 0.75Specificity: The "true negative rate" = 375 / (375 + 5) = 0.9868

We can then calculate the balanced accuracy as:

Balanced accuracy = (Sensitivity + Specificity) / 2Balanced accuracy = (0.75 + 9868) / 2Balanced accuracy = 0.8684

The balanced accuracy for the model turns out to be

0.8684.

The following example shows how to calculate the balanced accuracy for this exact scenario using the balanced_accuracy_score() function from the sklearn library in Python.

Example: Calculating Balanced Accuracy in Python

The following code shows how to define an array of predicted classes and an array of actual classes, then calculate the balanced accuracy of a model in Python:

```
import numpy as np
from sklearn.metricsimport balanced_accuracy_score

#define array of actual classes
actual = np.repeat(, repeats=)

#define array of predicted classes
pred = np.repeat(, repeats=)

#calculate balanced accuracy score
balanced_accuracy_score(actual, pred)
```

0.868421052631579

The balanced accuracy is 0.8684. This matches the value that we calculated earlier by hand.

Note: You can find the complete documentation for the balanced_accuracy_score() function .