Introduction

The make blobs function from sklearn.datasets is used in this project to simulate a dataset. We next categorize the data points and evaluate the algorithm's performance using the K-Nearest Neighbors (KNN) technique.

Dataset

Using make blobs, we generated a dataset with 150 data points. The three classes in the sample have centers at (2,4), (6,6), and (1,9), respectively. We used the train test split method from sklearn. model selection to divide the dataset into 80% for training and 20% for testing.

K-Nearest Neighbors

We categorize the data points using the K-Nearest Neighbors (KNN) technique. To utilize the five nearest neighbors in the categorization, we set n neighbors=5. With the training set of data, we built the KNN model, and we used the prediction function of the KNeighbors Classifier object to generate predictions about the testing set.

Results

Using the accuracy score function from sklearn.metrics, we determined the KNN model's accuracy score. The accuracy score was determined to be 0.9667, indicating that 96.67% of the test data points could have been classified properly using the KNN method.

Also, we plotted the KNN model's decision border and data points using Matplotlib to visualize the outcomes. The figure demonstrates that the three classes of the dataset were successfully separated using the KNN model.

Conclusion

In this research, we simulated a dataset made up of three classes and evaluated the K-Nearest Neighbors algorithm's performance. With an accuracy score of 0.9667, the KNN method was found to be quite accurate in classifying the data points. The figure demonstrates that the three classes of the dataset were successfully separated using the KNN model. Overall, the KNN algorithm is a useful tool for categorizing data points, especially when the dataset has been carefully divided into distinct clusters.