Summary Report: K-Nearest Neighbors Classification

This report outlines the findings from a K-Nearest Neighbors (KNN) classification analysis conducted on a simulated dataset. The dataset was created using the *make_blobs* function included 150 data points that were divided into three separate classes. The process included training the KNN model, predicting outcomes using a test dataset, and creating visual representations of the results to evaluate how accurate and effective the model was.

The goal is to find out how well the KNN classifier performs when it comes to classifying these data points correctly.

Dataset Formation and Processing

The dataset was generated using *make_blobs*, with the cluster centers set at (2,4), (6,6), and (1,9). The data was divided into two parts: a training set and a testing set, with 80% of the data used for training and 20% for testing. This way, we could evaluate how well the model performs on unseen data. We also set a *random_state* during the split to make sure we could get the same results if we did it again.

Dataset Details

- Number of Samples: 150
- Number of Features: 2
- Number of Classes: 3
- Class Centers: [2, 4], [6, 6], [1, 9]
- Data Split: 80% for training (120 samples) and 20% for testing (30 samples).

Model Formation

The K-Nearest Neighbors (KNN) classifier was set up with five neighbors (n_neighbors = 5). It used the Minkowski distance formula with p=2, which is the same as the Euclidean distance, to determine how similar the points are to each other. The model was trained on the given training dataset, to learn patterns. After the model was trained, it was used to predict the class labels for the test dataset. To evaluate how well it performed, the predicted labels were matched against the actual labels from the test set. The accuracy score was then calculated as the main metric, to evaluate the model's performance.

Results

The KNN model performed well and achieved an accuracy score of 1.0 (100%) which means all the 30 test samples were classified correctly without any mistake. The model works very well on the unseen data in this scenario.

Two scatter plots were generated to visualize the results. The first plot displayed the entire dataset, with each data point represented in different colors according to its actual class label. This gave a good overview of how the data was distributed and highlighted the clear differences between the classes. The second plot focused specifically on the test data, with points colored based on their predicted class labels. By comparing these two plots, KNN classified the data points correctly and no datapoints were misclassified.

Conclusion

In conclusion, the KNN algorithm showed outstanding results on the simulated dataset with the specified parameters. The accuracy score of 100% highlights how well the algorithm can learn and predict on unseen data, especially when the classes are clearly separated. The scatter plots also supported these results, giving a better view of how successfully the model classified the test data.