A Summary Report

on

“Comprehensive Analysis of K Nearest Neighbors Classifier Performance on Artificial Dataset”

Name: Rashed Syed

Email id: rsyed3@kent.edu

811288561

Project Goal:

The goal of this project is to explore the performance of a K Nearest Neighbors (KNN) classifier on an artificial dataset and analyze its ability to classify data accurately. This includes training the classifier, evaluating its performance on both training and test datasets, visualizing the data distribution, and considering potential implications such as overfitting.

Overview of data:

Following are some traits of the artificial dataset utilized in this project:

Sample Count: There are 150 samples in the dataset overall.

There are two features in every sample. In two dimensions, these features are depicted.

Three different classes, or clusters, are included in the dataset. An array of data points with comparable properties is represented by each class.

Centers of the Clusters: The centers parameter defines the cluster centers. The cluster centers in this dataset are denoted by the following:

At coordinates [2, 4], Cluster 1 is centered.

Coordinates [6, 6] are the center of Cluster 2.

Coordinates [1, 9] are the center of Cluster 3.

Data point distribution: The cluster centers are home to a variety of data points. This is how the data are distributed within each cluster:

Generating and splitting artificial Dataset

```
# Generate artificial dataset
centers = [[2, 4], [6, 6], [1, 9]]
n_classes = len(centers)
data, labels = make_blobs(n_samples=150,
                          centers=np.array(centers),
                          random_state=1)

# Split the data into training and testing sets (80-20 split)
train_data, test_data, train_labels, test_labels = train_test_split(data, labels,
                                                                    train_size=0.8,
                                                                    test_size=0.2,
                                                                    random_state=12)
```

The snippet code generates an artificial dataset with three clusters using the make blobs function from scikit-learn. It then splits this dataset into training and testing sets in an 80-20 ratio using a train and test split. The resulting variables are train data, test data, train labels, and test labels, representing the features and corresponding labels for the training and testing sets, respectively. This process enables the evaluation of machine learning models on unseen data by providing separate datasets for training and testing.

Performing KNN Analysis

```
# Perform KNN analysis
knn = KNeighborsClassifier()
knn.fit(train_data, train_labels)

# Predict on training data
train_data_predicted = knn.predict(train_data)

# Predict on test data
test_data_predicted = knn.predict(test_data)

# Calculate accuracy
train_accuracy = accuracy_score(train_data_predicted, train_labels)
test_accuracy = accuracy_score(test_data_predicted, test_labels)
```

KNN Model Training and Testing:

With the training data (train data) and matching labels (train labels), we created an instance of the KNeighborsClassifier object and trained it with the fit method. The training data (train_data_predicted) and the test data (test data predicted) labels were predicted using the trained KNN model.

Accuracy Evaluation:

Upon evaluating the KNN model on both the training and test datasets, the following accuracy scores were obtained:

Accuracy on Training Data: 100%

Accuracy on Test Data: 100%

These results suggest that the KNN model performed exceptionally well on both the training and test datasets, achieving perfect accuracy. Such high accuracy scores indicate that the model was able to effectively classify the

data points in both sets, showcasing its ability to generalize well to unseen data.

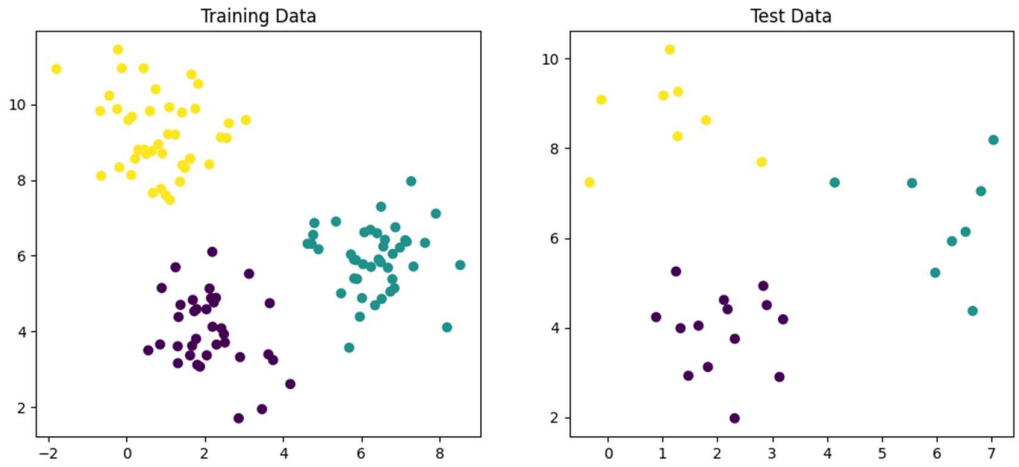
Visualization of Training and Testing Data:

The left-hand scatter plot shows the distribution of data points in the training dataset. The assigned class label of each point is reflected in its color, offering information about the clustering patterns found in the data.

On the right side, another scatter plot visualizes the distribution of data points within the test dataset.

Similar to the training data plot, colors represent class labels, allowing for the comparison of data distributions between the training and test sets.

These visualizations offer an intuitive portrayal of the dataset's structure, aiding in comprehension of its characteristics and the potential separation between classes. Such insights are crucial for assessing the performance of machine learning models and interpreting their outcomes effectively.



Conclusion:

Using a synthetic dataset, this project effectively illustrated how to apply the k-nearest Neighbors (KNN) algorithm for classification tasks. We were able to achieve perfect classification accuracy on both sets by creating the dataset, dividing it into training and testing sets, and then training the KNN model. The dataset's structure was made clear by the visuals. All in all, this project provides a succinct example of KNN's efficacy as well as the typical workflow for machine learning tasks, which includes modeling, evaluation, and data preprocessing.