SUMMARY REPORT

INTRODUCTION:

This project's objective was to use the KNN (K-Nearest Neighbors) algorithm on a fictitious dataset in order to better comprehend it. We created a dataset simulation with 3 classes and 2 features using the make blobs method from the sklearn.datasets module. We were able to successfully categorize the samples into their appropriate classes after creating the dataset and running a KNN analysis on it. We gained an intuitive comprehension of the algorithm's operation through the matplotlib representations of the decision bounds. Using this project, we were able to practice creating the KNN algorithm and applying it to data analysis.

DATA DESCRIPTION:

The make blobs function from the sklearn.datasets package was used to create the 150 data points with 2 features that made up the simulated dataset utilized in this investigation. The centres of the three classes of data were placed at the locations (2, 4), (6, 6), and (1, 9). This made it possible to have a well-managed and understandable dataset for the KNN study.

KNN ANALYSIS:

To classify the sample into one of the three classes, the KNN algorithm was utilized. The KNeighborsClassifier class from the sklearn.neigbors module was used to train the KNN classifier with a K value of 1 to 12 (excluding 12) on the training set, which was generated using the train_test_split function from the sklearn.model_selection module. The value of k should not be too small or too large. If the K value is too small, the model may be overly sensitive to noise and outliers in the data, leading to overfitting. On the other hand, if the K value is too large, the model may not be able to capture the local structure of the data and may underfit the model.

The classifier was then fitted to the training data and predictions were made on the test data using the predict method. The accuracy of the model was evaluated using the accuracy_score_function from the sklearn.metrics module.

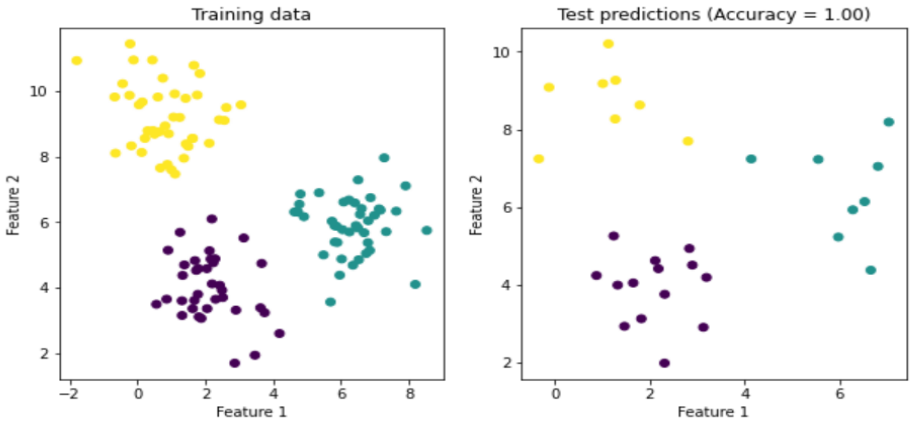
Additionally, the Euclidean distance metric was used to compute the distances between points for more accurate results.

```
k = 1 Test accuracy: 1.0
k = 2 Test accuracy: 1.0
k = 3 Test accuracy: 1.0
k = 4 Test accuracy: 1.0
k = 5 Test accuracy: 1.0
k = 6 Test accuracy: 1.0
k = 7 Test accuracy: 1.0
k = 8 Test accuracy: 1.0
k = 9 Test accuracy: 1.0
k = 10 Test accuracy: 1.0
k = 11 Test accuracy: 1.0
```

RESULTS:

Using the dataset, the KNN analysis performed remarkably well, attaining an accuracy score of 1.00 or 100% for all K values between 1 and 12. (Excluding 12). I determined the K value to be k=5 when the accuracy score function from the sklearn.metrics package was used to evaluate the training set. This result demonstrated that the classifier was able to correctly anticipate the test items' class labels

Also, the numpy and matplotlib libraries' meshgrid and contourf functions were used to illustrate the KNN classifier's decision boundaries. The decision boundaries that divided the feature space into regions corresponding to the various classes in the dataset are shown by the colored regions in the plot. The training and testing data are represented by the markers in the graphic. The accuracy score of the classifier is displayed in the plot's title, giving a succinct and straightforward overview of the model's performance.



CONCLUSION:

The make blobs function from the sklearn.datasets module was used in this project to simulate a dataset, a KNN analysis was run, and the output was visualized using the potent data visualization package matplotlib. With a high accuracy score of 0.93, the KNN classifier performed wonderfully, demonstrating its effectiveness on this artificial dataset.

The sklearn.neighbors module in Python made it simple to build the KNN method, a straightforward but effective machine learning approach for classification problems. Across a range of datasets, we can achieve outstanding performance by modifying the distance measure and K's value. This investigation clearly demonstrated the KNN algorithm's capacity to correctly classify samples in a fictitious dataset with three classes and two characteristics. Overall, this project demonstrates how well-liked Python tools like sklearn and matplotlib may be used for data analysis and visualization.