Experiment 6

Implementation of K – nearest neighbors algorithm from scratch

Name: - Anshika Pathak

Roll No.: - 23/CS/066

LAB REPORT

This laboratory experiment focuses on the implementation and analysis of the K-Nearest Neighbors (KNN) classification algorithm using the Iris and Wine dataset. The experiment encompasses the development of a custom KNN classifier without relying on existing machine learning libraries, allowing for a thorough understanding of the algorithm's core mechanisms.

The experiment consists of several key components:

1. Implementation of data preprocessing and feature extraction
2. Development of the Euclidean distance calculation method
3. Creation of visualization tools for feature analysis
4. Evaluation of model performance across different k values
5. Assessment of the algorithm's adaptability using the Wine dataset

The following sections present the methodology, results, and analysis of the implementation, providing insights into the effectiveness of the KNN classification approach.

A. Exploratory Data Analysis:

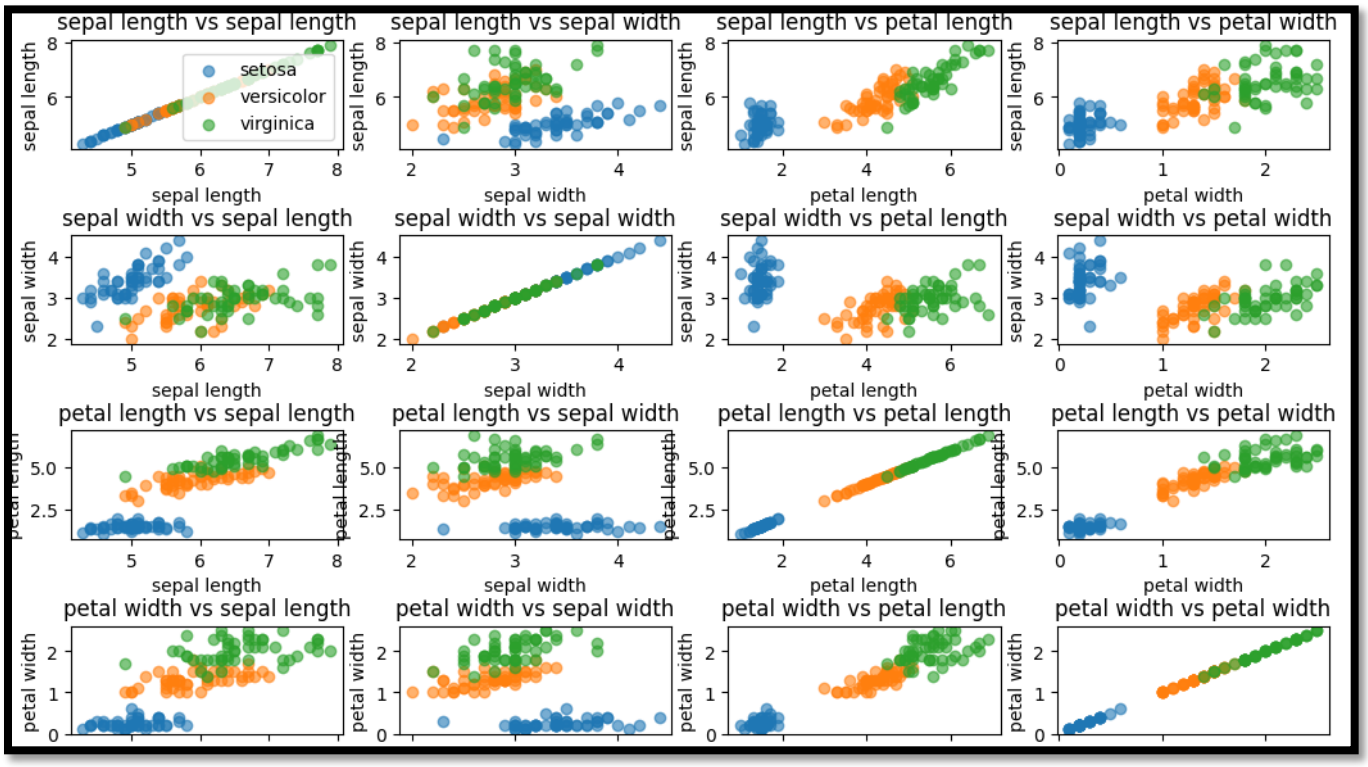
a. Feature Analysis

From the scatter plot matrix visualization, the following observations can be made:

i. Best Feature Pair for Class Separation: Petal length vs Petal width shows the clearest separation between classes. This pair exhibits distinct clustering of the three Iris species with minimal overlap.

ii. Class Separability:

1. The Setosa class is the most easily distinguishable, forming a separate cluster in most feature combinations.
2. Versicolor and Virginica show some overlap, particularly in sepal measurements.
3. The separation is most prominent in petal measurements, suggesting these features are more informative for classification.



B. Model Performance:

a. Accuracy achieved on the Iris dataset (with k=3).

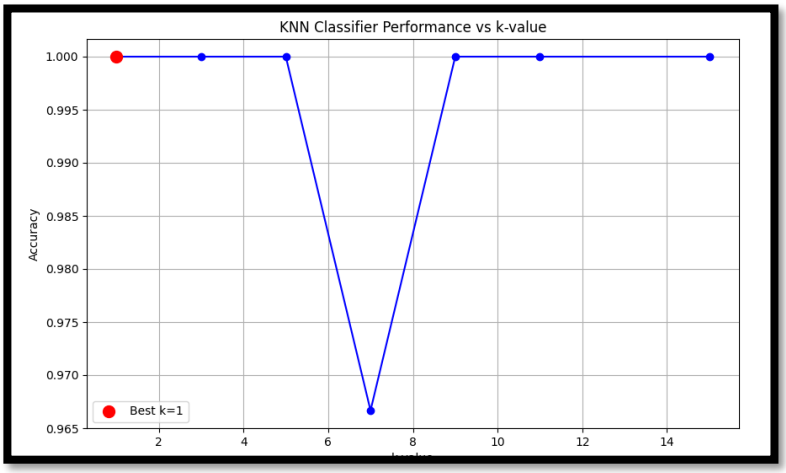
Classification Accuracy: 100%

```
KNN Classification for iris dataset (k=3)
Classification Accuracy: 1.0000
```

b. Hyperparameter Analysis

The "Accuracy vs. k-value" plot reveals several important insights:

- **i.** Best Performing k: k=1 achieved the highest accuracy (100%)
- **ii.** Performance Pattern:
  1. Small k values (1-5): Consistently high performance (100%)
  2. Medium k values (7): Slight drop in performance (96.67%)
  3. Large k values (9-15): Return to high performance (100%)



c. **Why k Values Matter:**

   i. **Small k values (k=1 or k=3):** They Can capture fine decision boundaries but may be sensitive to noise in the data

   ii. **Large k values (k=15):** They are more robust to noise but may oversimplify decision boundaries. In this case, performed well due to well-separated classes.

d. **Wine Dataset Results:**
   When applying the best performing model found on the Iris dataset (best k = 1) to the Wine dataset:

```
Evaluating on Wine dataset using best k
Wine Dataset Accuracy (k=1): 0.7714
```

   i. Wine Dataset Accuracy (evaluated with k=1): 77.14% (0.7714)

   ii. The Wine dataset shows lower accuracy compared to Iris under k=1, indicating the two datasets differ in difficulty and class structure.

   iii. It is recommended to perform hyperparameter tuning (varying k) and apply feature scaling before concluding the model's performance on Wine.

C. Conclusions:

The implementation and evaluation of the K-Nearest Neighbors classifier provided clear insights into both the strengths and limitations of this instance-based learning method. Through careful examination of the Iris dataset, it became evident that specific feature's petal length and petal width, contribute disproportionately to class separability, which explains the high accuracy observed for several k values. The hyperparameter study further demonstrated that model performance can depend strongly on k, in this experiment, very small values (e.g., k=1) achieved the best accuracy on Iris, while the Wine dataset required a different approach to reach optimal performance. Overall, the exercise reinforced the importance of exploratory data analysis and targeted feature selection when applying distance-based classifiers.

The development process highlighted several practical challenges. Implementing distance calculations and neighbor selection in a clear and efficient manner required careful indexing and use of NumPy operations to avoid unnecessary overhead. Additionally, the sensitivity of KNN to feature scales and noisy samples became apparent when transferring the model between datasets: without feature standardization and dataset-specific tuning, accuracy can degrade significantly. These observations motivate further steps such as applying scaling, using cross-validation for robust hyperparameter selection, and experimenting with alternative distance metrics to improve reliability across diverse datasets.