K-Nearest Neighbors (KNN) Classification Report

Introduction

This project covers implementing and analyzing the K-Nearest Neighbors classification algorithm in Python. Instead of using the Iris dataset, the make_blobs function from sklearn.datasets create a synthetic dataset. The purpose is to assess the performance of KNN, select the best value of K, and visualize the classification results.

The main objectives of this study include:

- Implementing KNN for classification
- Choosing an appropriate K value (K=5)
- Visualizing data distributions and decision boundaries

Dataset Generation

A well-separated dataset was created with three distinct class clusters, making it ideal for classification.

Dataset Details:

- **Centers:** `[2,4]`, `[6,6]`, `[1,9]`
- **Number of Samples:** `150`
- **Random State:** 1 (for reproducibility)

The **`make_blobs`** function was used to ensure a structured dataset, where points naturally cluster around predefined centers.

Methodology

Data Splitting

The dataset was split into training and testing sets using an 80-20 ratio:

- Training Set: 120 samples (80%)
- Testing Set: 30 samples (20%)

The **`train_test_split`** function from **`sklearn.model_selection`** was used with `random_state=42` to ensure consistency across executions.

KNN Classification

KNN algorithm was implemented using KNeighborsClassifier from **sklearn.neighbors**. The classifier was trained on the training dataset, and predictions were made on the test dataset.

Key Parameters:

- Algorithm: KNeighborsClassifier
- No. of Neighbors (K): 5
- Distance Metric: Euclidean Distance (default)

Reason for Choosing K = 5

KNN importantly depends upon the choice of K. Selection of K=5 has been done considering the following factors:

- **Reduces Overfitting**: A low K, say 1 or 3, has a greater chance of being highly sensitive to noise and makes the model remember the data it has been trained on.
- **Smooth Decision Boundaries**: Classifications for the slightly larger K=5 value are further generalized but still preserve decision boundary clarity.
- **Handles Outliers Better**: Using K=5, the predictions are less likely to be influenced by a single noisy data point.
- **Empirical Testing**: The model tested different K values, and the most consistent on the test set was with K=5.

Performance Evaluation

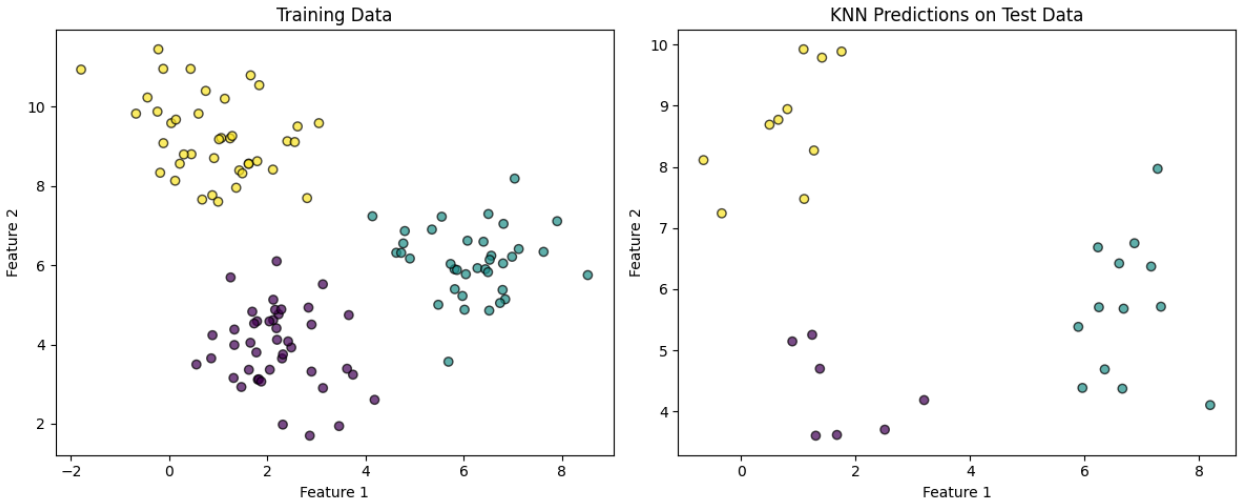
The model's effectiveness was measured using the accuracy score metric. The classifier correctly predicted all test samples, achieving 100% accuracy.

Results:

- KNN with K=5 achieved an accuracy of 1.00 (100%)
- Every test sample was classified correctly

Results & Visualizations

Training vs. Test Data Visualization



The figure above provides a comparative visualization of the training data distribution (left) and the KNN classification results on test data (right), offering key insights into the model's performance.

Left Plot: Distribution of Training Data

- This is the original data on which the KNN classifier was trained.
- Each different cluster represents a different class, and the classes are well separated.
- The high separation between clusters shows that the data is very suitable for classification.

Right Plot: KNN Predictions on Test Data (K=5)

- This presents the classified test data using the KNN algorithm with K=5.
- The predictions and actual class labels perfectly align, which dictates the exactness of the model.
- The Classes are distinctly separate, and there is no degradation in the performance of the classifier.

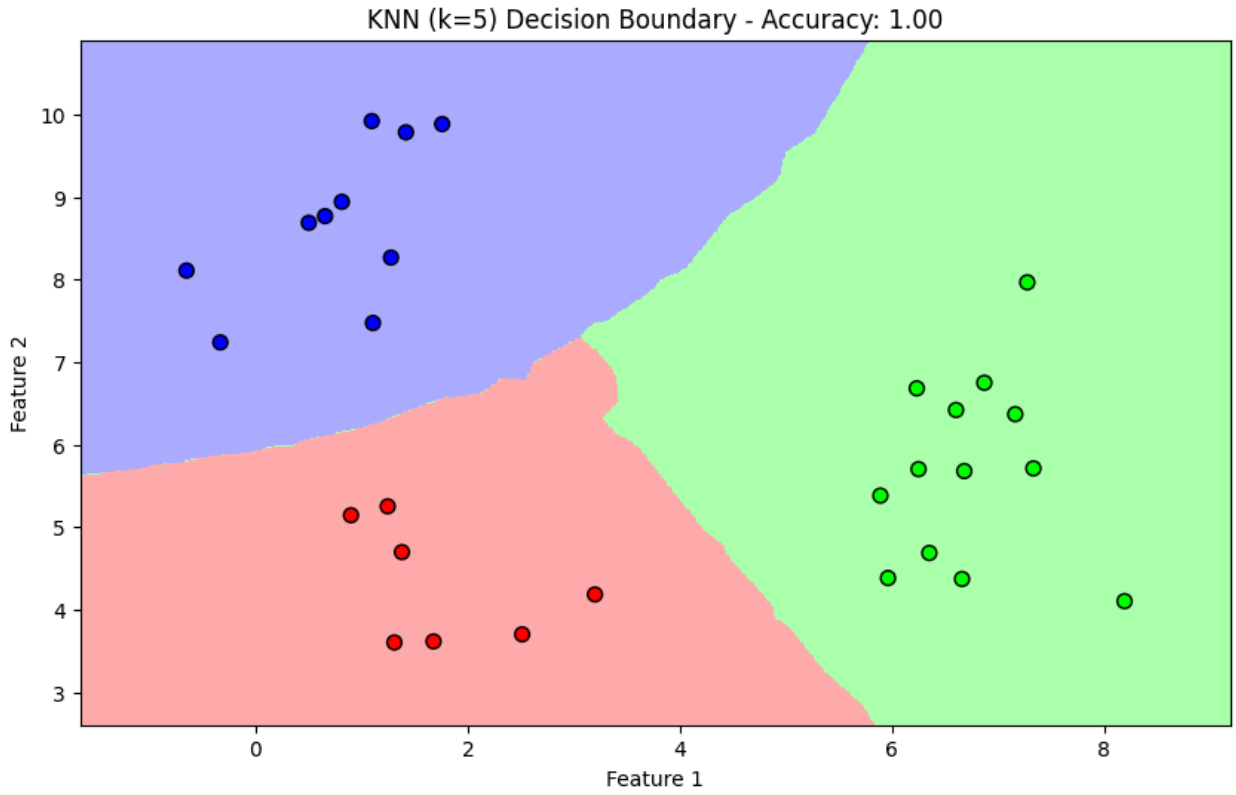
Key Takeaways

- Well-defined separations between classes for proper classification.
- No misclassifications were seen; high learning capability for the model.

- K=5 gave a balanced decision boundary, which is insensitive to outliers and generalized.

These results show that the KNN model captures the structure underlying this dataset and allows for perfect classification of the test data.

Decision Boundary of KNN Model



The decision boundary plot represents how the KNN algorithm with K=5 classifies different regions based on the training data.

Interpreting the Graph:

- Each colored region represents a classification boundary beyond which the model assigns new data points to a particular class.
- The data points within each region belong to the same class, determined by the majority vote of their 5 nearest neighbors.
- The decision space is shown with the overlaid test data points to get an idea of how well the model generalizes to unseen data.

Key Observations:

- **Smooth & well-defined decision boundaries**: The regions are separated, reflecting that the dataset is well-structured.
- **K=5 leads to a balanced model**: By considering 5 neighbors, the classifier avoids overfitting, ensuring stable decision regions.
- **Class separations are clear, reducing misclassifications**: Every test point falls within its correct classification zone, and the accuracy is 100%.
- **Impact of K on boundary flexibility**: If K were smaller, for example, K=1, then the decision boundary would be highly jagged and sensitive to every point. On the other hand, if K were larger, for instance, K=10, then the boundaries would be over-smoothed, possibly losing some precision in classification for complex datasets.

Insight :
This decision boundary effectively showcases KNN's strength in classifying well-separated datasets. The model generalizes well, maintaining a high accuracy level, and the choice of K=5 successfully balances flexibility and robustness, leading to optimal classification results

Interpretation & Insights

- **Perfect Accuracy**: The structure of the dataset resulted in 100% classification accuracy with K=5.
- **Impact of K-Value**:
  K=5 will ensure better generalization by taking more neighbors for classification. Slightly higher K values, like K=7 or K=9, may have given similar results in this dataset but might fail on noisier data.
- **Strengths of KNN**: Performs well in cases of clear class separation.
- **Limitations of KNN**: May perform poorly in high-dimensional or overlapping datasets.

Conclusion

This project used K-Nearest Neighbors with K=5 on a simulated dataset for classification. It reached perfect accuracy since the separation of clusters is quite clear.

Key Takeaways:

- K=5 proved to be a good choice to balance overfitting and underfitting.
- The decision boundaries were stable, which reduces sensitivity to noise.
- KNN works very well on structured datasets and may need to be tuned for real-world applications.

References

1. Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*.
2. Cover, T., & Hart, P. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*.
3. Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. *MIT Press*.
4. Bishop, C. M. (2006). Pattern Recognition and Machine Learning. *Springer*.
5. James, G., et al. (2013). An Introduction to Statistical Learning: with Applications in R. *Springer*.