## Introduction

The K-Nearest Neighbor algorithm, in its most general form, is an intuitive, simple, and non-parametric method for classification and regression. It finds the k closest training examples for a given test instance and makes predictions based on their labels. This algorithm is highly intuitive, relying on the idea that similar data points are often close in feature space.

In this assignment, a k-NN classifier will be implemented, performance will be evaluated using LOOCV, and effectiveness in prediction will be analyzed. Additional goals include encoding categorical features and ensuring robust error handling.

---

## Methodology

### Data Preparation

1. The dataset play_tennis.csv was imported and briefly reviewed.
2. Categorical features were one-hot encoded using a custom function to ensure compatibility with the k-NN algorithm, which relies on numeric operations.
3. Features (X) and output labels (y) were separated for further processing.

### Implementation Details

1. **Distance Metrics**: Two metrics were implemented:
    - Euclidean Distance: Computes the straight-line distance in feature space.
    - Manhattan Distance: Calculates the sum of absolute differences between feature values.
2. **k-NN Algorithm**: For a given test sample:
    - Distances to all training samples were computed.
    - The nearest k neighbors were identified.
    - The predicted label was determined based on the majority class of the nearest neighbors.
3. **Leave-One-Out Cross-Validation (LOOCV)**:
    - Each data point served as the test instance exactly once, while the remaining thirteen points formed the training set.
    - The overall accuracy and confusion matrix were calculated.
4. **Model Persistence**: The model, including hyperparameters (k and distance metric), was saved to a JSON file for reproducibility.

### Challenges and Handling

- **One Hot Encoding**: When applying one hot encoding, a new column is created for each unique value on each feature and the table is refilled with 1s and 0s according to their actual values.
- **Performance**: Iterative computations were optimized with pandas and numpy to manage distances efficiently.

---

## Results

### Performance Metrics

The classifier was evaluated on the encoded dataset using LOOCV. Below are the key performance metrics:

```
Accuracy: 0.50
Precision (Yes): 0.67
Recall (Yes): 0.44
F1 Score (Yes): 0.53
```

k=3 euclidean

### Confusion Matrix

```
Confusion Matrix:
          Predicted: Yes  Predicted: No
Actual: Yes    4               5
Actual: No     2               3
```

k=3 euclidean

---

## Discussion

### Analysis of Results

- The k-NN algorithm has shown good performance by achieving high accuracy and balanced precision/recall values using the loocv technique.
- Most misclassifications occurred in cases where the nearest neighbors were evenly split between classes, leading to ties.

### Reasons for Misclassifications

- Data imbalances in certain categories could bias predictions.

- Overlapping feature spaces for different classes made accurate separation difficult.

**Limitations**

- Computational inefficiency: As the dataset grows, calculating distances becomes resource-intensive.
- Sensitivity to noisy or irrelevant features: Poorly chosen features can degrade performance.

---

**Conclusion**

This project provided a more practical understanding of the k-NN algorithm, including its implementation and evaluation. Key takeaways include the importance of proper data preprocessing and the impact of hyperparameter selection on model performance. The LOOCV approach offered a robust method for evaluating generalization. While the k-NN algorithm is simple and effective, it again points to the need for optimization and careful feature selection in real-world applications.

---

**References**


1. Introduction to Machine Learning by Ethem Alpaydin
2. Pattern Recognition and Machine Learning by Christopher Bishop
3. Python Documentation: https://docs.python.org/