# K-Nearest Neighbors (K-NN) Algorithm: Implementation and Analysis

## The implementation includes:

- Distance calculation using Euclidean and Manhattan metrics.
- A fit method that stores the training data.
- A predict method that finds the k k-nearest neighbors and determines the class label.
- A score method to compute classification accuracy.

## Data Preparation and Preprocessing:

- The dataset consists of:
  - Training Inputs: X_train (features)
  - Training Labels: y_train (target class)
  - Test Inputs: X_test
  - Test Labels: y_test
- Data Normalization:
  - To ensure consistency across feature scales, the StandardScaler from Scikit-learn was used

## Results and Analysis

- The model had its lowest accuracy of 72.93% when k = 1 and k = 2. A smaller k makes the model more sensitive to noise and outliers, which can lead to incorrect classifications.
- As k increases to 3 and 4, the accuracy improves gradually, reaching 77%. This suggests that a slightly larger k helps the model make better decisions by reducing the effect of random noise.
- When k is 5 or higher, the accuracy continues to improve. At k = 5, the model reaches 78.88% accuracy, and from this point, accuracy remains stable without large fluctuations. This means the model is finding a good balance between learning from data and avoiding overfitting.
- The highest accuracy of 79.51% is observed when k = 26. This suggests that considering more neighbors helps improve predictions. However, if k becomes too large, the model may start losing important details. Large k values also require computing distances for many neighbors, increasing computation time
- The best value for k is k = 5, where the model achieves 78.88% accuracy. This value provides a good balance between accuracy and stability with increasing computational time.