Title: Day 13 - K-Nearest Neighbors (K-NN)

Introduction:

K-Nearest Neighbors (K-NN) is a versatile machine learning algorithm used for classification tasks. It operates on the principle of proximity, where it assigns class labels to data points based on the majority class of their nearest neighbours. In this document, we will explore how K-NN can be applied to classify fake bills using Python and Scikit-Learn.

Tasks and Operations Performed:

1. Data Loading: The first step is to load the dataset from the 'FAKEBILL_third_day.csv' file using the Pandas library. This dataset likely contains various features and a target variable, which is 'is_genuine.' The dataset is loaded into a Pandas DataFrame.
2. Data Splitting: The dataset is divided into two parts: the features (X) and the target variable (y). The 'height_right' column is dropped from the feature set, as it seems to be the target variable. The 'is_genuine' column is set as the target variable. Then, the data is further split into training and testing sets using the train_test_split function from Scikit-Learn.
3. K-NN Classifier Creation: A K-NN classifier is created using the KNeighborsClassifier from Scikit-Learn. In this case, K=2 neighbors are chosen as the parameter. This value of K can be adjusted based on the dataset and domain knowledge.
4. Model Training: The K-NN model is trained using the training data. The training process essentially involves storing the training data in memory so that the model can later make predictions based on it.
5. Prediction: The model is used to make predictions on the testing dataset (X_test) using the predict method. This step classifies each data point as genuine or fake based on the K-NN algorithm.

6. Model Evaluation: To measure the model's performance, the accuracy score is calculated using the accuracy_score function from Scikit-Learn. This score indicates the proportion of correctly classified instances in the testing data.

Benefits of K-NN:

- Simplicity: K-NN is easy to understand and implement, making it a suitable choice for beginners in machine learning.
- No Assumptions: K-NN makes minimal assumptions about the underlying data distribution, which can be advantageous when the true data distribution is unknown.
- Adaptability: K-NN can be applied to both classification and regression tasks, and it can be used with various distance metrics, providing flexibility.

Conclusion:

In this task, we applied the K-Nearest Neighbors (K-NN) algorithm to classify fake bills. The code demonstrates data loading, splitting, model creation, training, prediction, and evaluation. K-NN is a useful and straightforward method for classification tasks, and it can yield good results, especially when the optimal value of K and feature scaling are carefully chosen. However, it's important to consider its limitations, such as sensitivity to noisy data and computational requirements when working with large datasets. This example serves as a starting point for implementing K-NN in real-world applications.