Day 9 Training Report

3 July 2025

k-Nearest Neighbors (k-NN) Algorithm — Working, Choosing ‘k’, Hands-on Implementation

On **Day 9**, the focus was on **k-Nearest Neighbors (k-NN)**, a simple yet powerful **supervised learning algorithm** used for both **classification and regression**. Students learned the **working principle**, **how to choose the optimal ‘k’**, and implemented k-NN on sample datasets.

---

1. Introduction to k-NN

k-NN is a **lazy learning algorithm**, meaning it **does not learn a model explicitly** during training. Instead, it **stores all the training data** and makes predictions only when given a new input.

- **Classification:** Assigns the class most common among the k nearest neighbors.
- **Regression:** Predicts the average value of k nearest neighbors.

Key Concepts:

- **Distance Metric:** Determines “closeness” (commonly Euclidean distance):

d=∑i=1n(xi−yi)2d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}d=i=1∑n(xi−yi)2

- **Hyperparameter ‘k’:** Number of neighbors considered for prediction.
- **Voting Mechanism:** Majority vote (for classification) or averaging (for regression).

---

2. Choosing the Optimal ‘k’

- Small k → Can lead to **overfitting** (sensitive to noise).
- Large k → Can lead to **underfitting** (smooths out distinctions).
- **Common Practice:** Use **cross-validation** to select the best k.
- Odd numbers are preferred for binary classification to avoid ties.

---

3. Hands-on Implementation (Classification Example)

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
```

```
# Load dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create k-NN classifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)

# Predict
y_pred = knn.predict(X_test)

# Evaluate accuracy
print("Accuracy:", accuracy_score(y_test, y_pred))
```

Key Observations:

- k-NN works best with **scaled/normalized data**.
- Choice of k directly impacts **prediction accuracy**.
- Easy to implement and interpret.