# K-Nearest Neighbors (KNN) Algorithm

Md Abdullah-Al-Kafi
Lecturer,Department of CSE
Daffodil International University

# Introduction to KNN

- **K-Nearest Neighbors (KNN)**: A simple, non-parametric, and lazy learning algorithm for classification and regression tasks.
- **Lazy learning**: No explicit model is built; instead, it stores training data and makes predictions during testing.
- **Non-parametric**: KNN makes no assumptions about data distribution.

# How KNN Works

1. Choose the number of neighbors $K$.
2. Calculate the distance between the test point and all training points (usually Euclidean distance).
3. Identify the $K$ closest neighbors.
4. Classification: Assign the most common class among the neighbors.
5. Regression: Take the average of the neighbors' values.

# Distance Metrics in KNN

- **Euclidean Distance**:

$$d(p, q) = \sqrt{\sum (p_i - q_i)^2}$$

- **Manhattan Distance**:

$$d(p, q) = \sum |p_i - q_i|$$

- **Minkowski Distance**: Generalized form of both Euclidean and Manhattan distances.
- **Cosine Similarity**: Measures the cosine of the angle between two vectors (used for text data).

# Choosing $K$ in KNN

- Small $K$: Sensitive to noise (overfitting).
- Large $K$: Smoothens the decision boundary (risk of underfitting).
- Cross-validation can be used to find the optimal $K$.

# Weighted KNN

- Neighbors closer to the test point are sometimes weighted more heavily than farther ones.
- Useful when the distances between points vary significantly.

# Advantages of KNN

- Simple to understand and implement.
- No training phase.
- Effective for small datasets and well-separated classes.

# Disadvantages of KNN

- Computationally expensive during testing.
- Performance degrades with high-dimensional data.
- Sensitive to irrelevant or redundant features.

# Applications of KNN

- **Recommendation Systems**: KNN is used in collaborative filtering for recommendations.
- **Image Recognition**: Finds similar images based on pixel values.
- **Anomaly Detection**: Identifies rare events in time series or financial data.
- **Text Classification**: Can classify text using similarity measures like cosine distance.

# KNN for Classification and Regression

- **Classification**: Majority class of $K$ neighbors is the predicted label.
- **Regression**: Average of $K$ neighbors' values is the predicted value.

# Improvements and Variations

- **KD-Trees/Ball Trees**: Speed up nearest-neighbor searches.
- **Condensed and Edited KNN**: Reduces training samples without sacrificing accuracy.
- **Distance-Weighted KNN**: Weighs neighbors by their distance to the test point.

# Practical Considerations

- **Data Scaling**: Important due to the distance-based nature of KNN.
- **Handling Missing Values**: Impute missing values using KNN imputation.
- **Computational Complexity**: Time complexity during prediction is $O(n \times d)$, where $n$ is the number of points and $d$ is the number of features.

# KNN in Python (Scikit-learn)

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(X_train, y_train)
predictions = knn.predict(X_test)
```