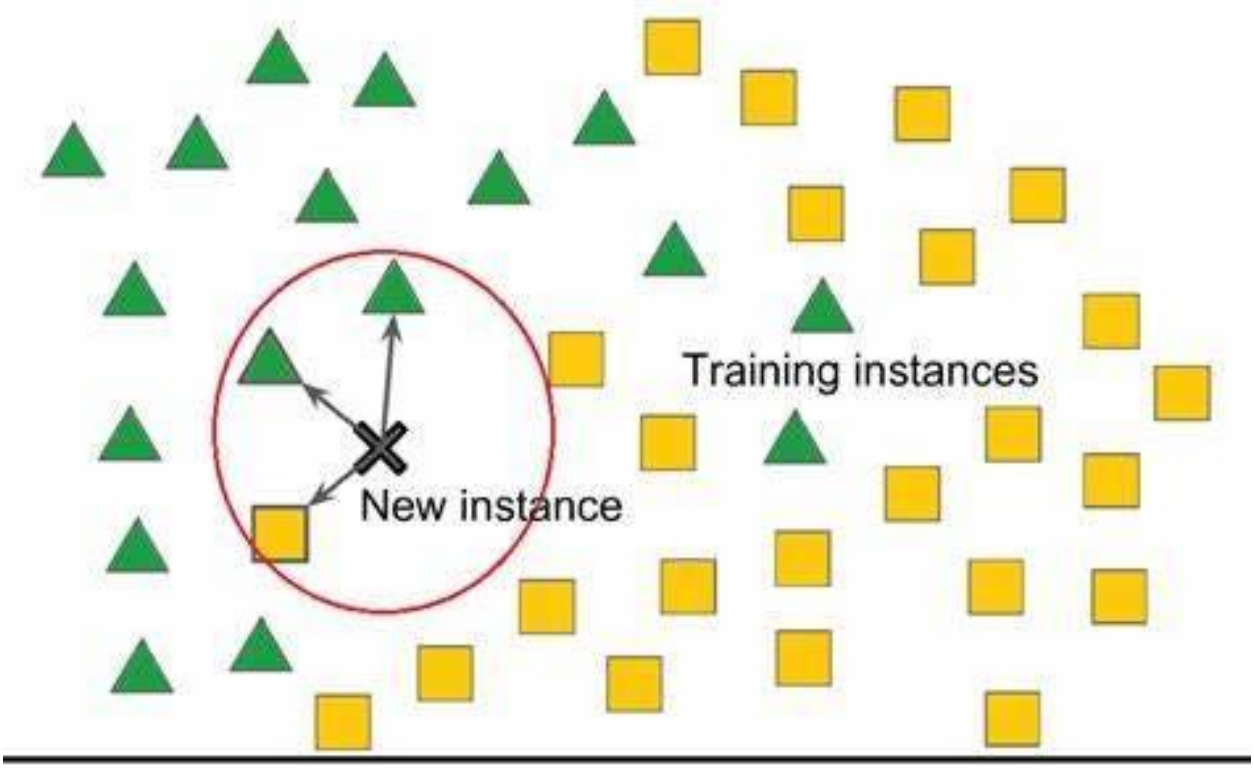
K-Nearest Neighbors (KNN) Algorithm



Introduction:

In the vast landscape of machine learning algorithms, two primary categories stand out: **supervised** and **unsupervised** learning. In supervised learning, algorithms learn from labeled data, making predictions or decisions based on input-output pairs. On the other hand, unsupervised learning algorithms deal with unlabeled data, identifying patterns and relationships without predefined outcomes.

K-Nearest Neighbors (KNN) belongs to the family of supervised learning algorithms.

It’s a simple yet powerful method for classification and regression tasks. Its intuitive approach makes it a popular choice for beginners and professionals alike. In this article, we’ll delve into what KNN is, when to use it, its advantages, disadvantages, and provide a simple example to illustrate its workings.

What is the K-Nearest Neighbors Algorithm? K-Nearest Neighbors (KNN) is a non-parametric, lazy learning algorithm used for classification and regression tasks.

Non-parametric means it doesn’t make any assumptions about the underlying data distribution. Instead, it memorizes the entire training dataset and makes predictions based on similarity measures.

When is KNN Used?

KNN is often used in scenarios where the decision boundaries are irregular ordifficult to define mathematically. It’s suitable for both classification and regression tasks, making it versatile across various domains such ashealthcare, finance, and recommendation systems.

Advantages of KNN:

1. Simplicity: KNN is easy to understand and implement, making it a go-to choice for beginners.

2. No Training Phase: Since KNN is a lazy learner, it doesn’t require a training phase. The model memorizes the training data, making predictions directly during inference.

3. Non-parametric: KNN doesn’t make any assumptions about the underlying data distribution, making it flexible and robust to different types of datasets.

Disadvantages of KNN:

1.Computational Complexity: As the size of the dataset grows, the computation required for finding the nearest neighbors increases, making KNN computationally expensive, especially for large datasets.

2. Sensitivity to Noise and Outliers: KNN is sensitive to noisy data and outliers since it relies on proximity-based measures. Outliers can significantly affect the decision K-Nearest Neighbors (KNN) Algorithm

3. Need for Feature Scaling: KNN calculates distances between data points, so it’s essential to scale the features to ensure that no single feature dominates the distance calculations.

Basic Flow of KNN Algorithm:

- Calculate the distance between the query instance and all the training samples.
- Sort the distances and determine the k-nearest neighbors based on the nearest distance.
- Among these k neighbors, count the number of data points in each category.
- Assign the new data point to the category where the majority of its neighbors belong.

Picking a Value for K:

A common approach to determine the value of k is to use the square root of the number of data points divided by 2.

The formula is: **k = sqrt(N)/2**

Simple Example:

Let's say we have a dataset with two features, 'X1' and 'X2', and two classes, 'A' and 'B'. We want to classify a new data point based on its nearest neighbors using KNN with k=3.

Given a new data point, we calculate its Euclidean distance to all other data points in the training set. Then, we select the k nearest neighbors based on these distances. If the majority of the neighbors belong to class 'A', we classify the new point as 'A', and vice versa.

For instance, if the distances to the three nearest neighbors are 2, 3, and 4, and their corresponding classes are 'A', 'B', and 'A', respectively, the majority class is 'A'. Thus, we classify the new data point as 'A'.

Conclusion:

KNN is a simple yet effective algorithm for classification and regression tasks. Its flexibility, ease of implementation, and ability to handle complex decision

boundaries make it a valuable tool in the machine learning toolkit. However, it's essential to consider its computational complexity and sensitivity to noise when applying it to real-world datasets.

For a more detailed explanation of KNN, you can read the article by Sean Atukorala on Medium: [K-Nearest Algorithm in Machine Learning Explained](https://medium.com/@SeanAT19/k-nearest-algorithm-in-machine-learning-explained-d56ac396958b).

Group Members

MAHNDSE23.2F=007-J.G.S.GAYANIKA

MAHNDSE23.2F=008-W.G.A.W.G.JAYARATHNA

MAHNDSE23.2F=009-W.A.K.N.DARSHANA

MAHNDSE23.2F=010-G.I.COLOMBAGE

MAHNDSE23.2F=022-C.P.NARASINGHA

MAHNDSE23.2F=050-N.S.K.GIMHNAN