Blue text on a black background

Description automatically generated

**Department name: Computer Science**

**Course title: MCS5803 – Algorithm Design & Analysis**

**Presentation title: K- Nearest Neighbors Algorithm**

**Student Name: Kenil Patel**

**: Abhishek Patil**

**: Vansh Savani**

**Date: 12-06-2023**

**GitHub:**

[**https://github.com/Vanshsavani/Algorithm-presentation.git**](https://github.com/Vanshsavani/Algorithm-presentation.git)

**K-Nearest Neighbor (KNN) Algorithm**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The K-Nearest Neighbors (KNN) algorithm is a versatile and intuitive machine learning approach utilized for addressing classification and regression challenges. Operating on the principle of similarity, KNN predicts the label or value of a new data point by considering its K closest neighbors in the training dataset. In this overview, we delve into the fundamentals of the supervised learning algorithm, emphasizing its user-friendly nature.

**What is the K-Nearest Neighbors Algorithm?**

K-Nearest Neighbors (KNN) stands as one of the fundamental classification algorithms in machine learning. It finds wide application in pattern recognition, data mining, and intrusion detection due to its non-parametric nature, which avoids assumptions about data distribution. Given prior training data that classifies coordinates into attribute-defined groups, KNN allocates new, unclassified data points to groups by analyzing proximity.

A screenshot of a computer

Description automatically generated

**Intuition Behind KNN Algorithm**

Visualizing data points on a graph reveals clusters or groups. When classifying an uncharted point, its proximity to a cluster influences the assigned classification. For instance, a point close to a 'Red' cluster is more likely to be classified as 'Red.' This intuitive approach is evident in determining classifications for points like (2.5, 7) as 'Green' and (5.5, 4.5) as 'Red.'

**Why Use KNN?**

KNN's appeal lies in its simplicity, ease of implementation, and non-parametric nature, accommodating both numerical and categorical data. It proves less sensitive to outliers compared to other algorithms, making it versatile for various datasets in classification and regression tasks.

**Distance Metrics Used in KNN Algorithm**

To identify the nearest points or groups for a query point, distance metrics such as Euclidean, Manhattan, and Minkowski distances are commonly employed. These metrics quantify the dissimilarity or similarity between data points, crucial for KNN's decision-making process.

**Choosing the Value of K**

The choice of the parameter K, representing the number of neighbors in the algorithm, is vital. It should align with the characteristics of the input data, with odd values recommended to prevent ties in classification. Cross-validation aids in selecting an optimal K value based on the dataset.

**How KNN Works**

The KNN algorithm operates on the principle of similarity, predicting labels or values for new data points based on the K nearest neighbors in the training dataset. Calculating distances using metrics like Euclidean distance, the algorithm identifies the closest neighbors and makes predictions based on majority voting (classification) or averaging (regression) of the neighbors' labels or values.

**Applications of KNN Algorithm**

KNN finds applications in data preprocessing (e.g., KNN Imputer), pattern recognition, and recommendation engines. It adapts well to various tasks, making it a valuable tool in machine learning.

**WORKINGS OF KNN ALGORITHM**

**A diagram of a diagram of a training

Description automatically generated with medium confidence**

**Advantages:**

Easy implementation with low algorithmic complexity.

Adaptive to new data points, adjusting itself for future predictions.

Requires minimal hyperparameters, primarily K and the choice of distance metric.

**Disadvantages:**

Inefficient scaling, demanding considerable computing power and storage.

Susceptible to the curse of dimensionality, leading to challenges in high-dimensional data.

Prone to overfitting, necessitating feature selection and dimensionality reduction techniques.

**Time complexity of KNN algorithms**

the time complexity is O(nd). However, we can reduce the time complexity of KNN algorithms to O(log(n)).