A. **What are the key hyperparameters in KNN?**

Ans:

The K-Nearest Neighbors (KNN) algorithm has several key hyperparameters that can significantly influence its performance. Understanding and tuning these hyperparameters is essential to optimize the model's accuracy, computational efficiency, and ability to generalize.

Below is the hyperparameters in KNN.

1. Number of Neighbors (k):

The number of nearest neighbors considered when making predictions.

* Smaller k (e.g., k=1): The model becomes more sensitive to noise and overfits the training data.
* Larger k: The model smoothes predictions by averaging over more neighbors, which can improve generalization but may underfit the data.

2. Distance Metric:

Determines how the "distance" between points is calculated. Common distance metrics include:

* Euclidean Distance: 
* Manhattan Distance: 
* **Minkowski Distance** (generalization of Euclidean and Manhattan):  When p=2, it's Euclidean; when p=1, it's Manhattan.
* **Hamming Distance**: Used for categorical variables.

3. Weights:

Determines the weight assigned to the neighbors' votes.

Options include:

* **Uniform**: All neighbors are weighted equally.
* **Distance**: Closer neighbors have more influence.
* The choice of weights can affect the bias-variance trade-off and how sensitive the predictions are to the neighbors' distances.

4. Algorithm (for Computing Nearest Neighbors **(**algorithm**)**:

Specifies the algorithm used to compute the nearest neighbors:

* **Brute Force**: Computes distances between all points, suitable for small datasets.
* **KD-Tree**: Efficient for low-dimensional data but not ideal for high-dimensional datasets.
* **Ball-Tree**: Similar to KD-Tree but better handles high-dimensional data.

Affects computation time and scalability.

5. Leaf Size (for KD-Tree or Ball-Tree):

Applicable for ball\_tree or kd\_tree algorithms, specifying the number of points at which to switch to brute-force search.

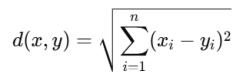
It affects the speed of constructing and querying the tree, with trade-offs between memory usage and computational efficiency.

**B. What distance metrics can be used in KNN?**

In K-Nearest Neighbors (KNN), the choice of distance metric plays a crucial role in determining which neighbors are considered "close" and therefore affects the classification or regression results.

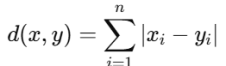
Commonly used distance metrics in KNN:

1. **Euclidean Distance:**

**Formula:** 

**Measures the straight-line distance between two points in Euclidean space. It is the most commonly used metric.**

1. **Manhattan Distance (L1 Distance):**

**Formula:** 

Measures the distance between two points along axes at right angles

1. **Minkowski Distance:**

**Formula:** 

**Generalizes both Euclidean and Manhattan distances. When p=1, it is equivalent to Manhattan distance; when p=2, it is equivalent to Euclidean distance.**

1. **Chebyshev Distance (L∞ Distance):**

**Formula:** 

**Measures the maximum absolute difference along any coordinate dimension. It is useful when you want to detect outliers based on the largest single deviation.**

1. **Hamming Distance:**

**Formula:** 

**Measures the number of positions at which the corresponding elements are different. It is typically used for categorical data.**

1. **Cosine Distance:**

**Formula:**  **where cos(θ) is the cosine of the angle between x and y.**

**Measures the cosine of the angle between two vectors. It is often used in high-dimensional spaces where the magnitude of vectors matters less than their direction.**

1. **Mahalanobis Distance:**

**Formula:**  **where SSS is the covariance matrix of the dataset.**

**Takes into account the correlations of the data set and is scale-invariant. It is useful in multivariate analysis**.