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

The K-Nearest Neighbours (KNN) algorithm has several important hyperparameters that can be tuned to improve model performance. The key hyperparameters are:

**a. n\_neighbors (K)**

* **Definition**: This hyperparameter defines the number of nearest neighbors to consider when classifying a data point.
* **Effect**: A smaller value of K may lead to overfitting, while a larger value of K may smooth out the decision boundary and lead to underfitting.
* **Example**:

knn = KNeighborsClassifier(n\_neighbors=3) # Use 3 nearest neighbors

**b. weights**

* **Definition**: This determines how the neighbors' votes are weighted.
* **Options**:
  + **uniform**: All neighbors are given equal weight.
  + **distance**: Closer neighbors have more influence on the prediction.
* **Example**:

knn = KNeighborsClassifier(weights='distance') # Closer neighbors have more weight

**c. algorithm**

* **Definition**: This specifies the algorithm used to compute the nearest neighbors. It affects the speed of the computation, especially for large datasets.
* **Options**:
  + **auto**: The algorithm is selected based on the data.
  + **ball\_tree**: Uses a tree structure to search for neighbors.
  + **kd\_tree**: Uses a KD-tree structure for search.
  + **brute**: Uses brute-force computation (slower for large datasets).
* **Example**:

knn = KNeighborsClassifier(algorithm='ball\_tree') # Use Ball Tree for neighbor search

**d. leaf\_size**

* **Definition**: This controls the size of the leaf in the tree-based algorithms (ball\_tree and kd\_tree). Smaller values improve accuracy but may slow down the computation.
* **Example**:

knn = KNeighborsClassifier(algorithm='ball\_tree', leaf\_size=30) # Set leaf size

**e. p (Power parameter)**

* **Definition**: This hyperparameter is used when calculating the distance between data points.
  + **p=1**: Manhattan distance (L1 norm).
  + **p=2**: Euclidean distance (L2 norm).
* **Example**:

knn = KNeighborsClassifier(p=1) # Use Manhattan distance

**f. metric**

* **Definition**: This defines the distance metric used to compute the distance between points. Common distance metrics include Euclidean, Manhattan, and others.
* **Example**:

knn = KNeighborsClassifier(metric='euclidean') # Use Euclidean distance

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

KNN uses distance metrics to measure how close a data point is to its neighbors. Below are the common distance metrics that can be used in KNN:

**a. Euclidean Distance (L2 Norm)**

* **Definition**: The straight-line (or "as-the-crow-flies") distance between two points in Euclidean space. It's the most commonly used metric.
* **Formula**: d(p1,p2) = sqrt{(x1-x2)^2 + (y1-y2)^2}
* **Example**:

knn = KNeighborsClassifier(metric='euclidean') # Default metric

* **When to Use**: It's generally used when the features are continuous and you are working in a Euclidean space.

**b. Manhattan Distance (L1 Norm)**

* **Definition**: The sum of the absolute differences between two points. This is like the distance traveled in a grid-like system, moving along grid lines rather than diagonally.
* **Formula**: d(p1,p2)=∣x1−x2∣+∣y1−y2∣
* **Example**:

knn = KNeighborsClassifier(metric='manhattan') # Use Manhattan distance

* **When to Use**: This metric is used when the features are on a grid, such as in city-blocks, or when you want to capture more local relationships between points.

**c. Minkowski Distance**

* **Definition**: A generalization of both Euclidean and Manhattan distances. It's parameterized by a power parameter p.
* **Formula**: d(p1,p2)=(∑(i=1,n){∣xi−yi∣^p})^1/p
  + **p=1**: Equivalent to Manhattan distance.
  + **p=2**: Equivalent to Euclidean distance.
* **Example**:

knn = KNeighborsClassifier(metric='minkowski', p=3) # Use Minkowski with p=3

* **When to Use**: This is useful if you want a flexible metric that can work with various powers to tweak the "tightness" of the distance measure.

**d. Chebyshev Distance**

* **Definition**: The maximum absolute difference between two points. It’s similar to Manhattan but with a focus on the maximum difference.
* **Formula**: d(p1,p2)=max∣x1−x2∣,∣y1−y2∣
* **Example**:

knn = KNeighborsClassifier(metric='chebyshev') # Use Chebyshev distance

* **When to Use**: This metric is useful when you care about the largest single difference across dimensions, rather than the overall distance.

**e. Cosine Similarity**

* **Definition**: A measure of similarity rather than distance. It computes the cosine of the angle between two vectors.
* **Formula**: Cosine Similarity(A,B)=A⋅B / ∥A∥.∥B∥
* **Example**:

knn = KNeighborsClassifier(metric='cosine') # Use Cosine Similarity

* **When to Use**: Typically used in text mining and information retrieval when the magnitude of vectors is less important than their direction or angle.