K-Nearest Neighbours

**Interview Questions:**

1. What are the key hyperparameters in KNN?

The **K-Nearest Neighbors (KNN)** algorithm has a few key hyperparameters that significantly impact its performance:

* **k (Number of Neighbors):**
  + Determines how many neighbors are considered when making a prediction.
  + **Small k:** The model is sensitive to noise and may overfit.
  + **Large k:** The model becomes smoother and more generalized but may underfit, especially with very large k.
  + **Common Practice:** Start with k = √n (where n is the number of samples) and adjust based on performance.
* **weights:**
  + Defines how much influence each neighbor has on the prediction.
    - **uniform:** All neighbors have equal weight.
    - **distance:** Closer neighbors have more influence (weighted by the inverse of their distance).
  + **When to Use:** Use distance when you believe closer neighbors should have more say in the prediction.
* **metric (Distance Metric):**
  + Determines how the distance between data points is calculated (explained in the next question).
* **algorithm:**
  + Specifies the algorithm to compute nearest neighbors. Options include:
    - **auto** (lets the system decide),
    - **ball\_tree**,
    - **kd\_tree**,
    - **brute** (brute-force search).
  + **For large datasets,** ball\_tree or kd\_tree can be faster than brute-force.
* **leaf\_size:**
  + Relevant for ball\_tree or kd\_tree algorithms.
  + Controls the speed of construction and query time. Smaller leaf sizes result in faster queries but slower tree construction.
* **p:**
  + Used when specifying the Minkowski distance.
    - **p=1** is equivalent to Manhattan distance.
    - **p=2** is equivalent to Euclidean distance.

2. What distance metrics can be used in KNN?

KNN relies on measuring the similarity (or dissimilarity) between data points using various **distance metrics**. Here are the most common ones:

* **Euclidean Distance (L2 Norm):**

d(x,y)=∑i=1n(xi−yi)2d(x, y) = \sqrt{\sum\_{i=1}^{n} (x\_i - y\_i)^2}d(x,y)=i=1∑n​(xi​−yi​)2​

* + **Use Case:** Best for continuous numerical data.
  + **Intuition:** Measures the straight-line distance between points.
* **Manhattan Distance (L1 Norm):**

d(x,y)=∑i=1n∣xi−yi∣d(x, y) = \sum\_{i=1}^{n} |x\_i - y\_i|d(x,y)=i=1∑n​∣xi​−yi​∣

* + **Use Case:** Works well when dealing with grid-like data (e.g., city block distances).
  + **Intuition:** Measures the distance if you can only move along axes (like navigating city streets).
* **Minkowski Distance:**
  + Generalization of both Euclidean and Manhattan distance:

d(x,y)=(∑i=1n∣xi−yi∣p)1/pd(x, y) = \left(\sum\_{i=1}^{n} |x\_i - y\_i|^p \right)^{1/p}d(x,y)=(i=1∑n​∣xi​−yi​∣p)1/p

* + - **p=1** → Manhattan Distance
    - **p=2** → Euclidean Distance
    - **Custom p values** can be used depending on the problem.
* **Chebyshev Distance:**

d(x,y)=max⁡(∣xi−yi∣)d(x, y) = \max(|x\_i - y\_i|)d(x,y)=max(∣xi​−yi​∣)

* + **Use Case:** Useful when you care about the maximum difference in any dimension.
* **Cosine Similarity:**
  + Measures the cosine of the angle between two vectors.
  + **Use Case:** Often used with text data or high-dimensional spaces to measure similarity rather than distance.
* **Hamming Distance:**
  + Measures the number of positions at which the corresponding elements are different.
  + **Use Case:** Suitable for categorical data or binary vectors.
* **Mahalanobis Distance:**
  + Accounts for correlations between variables and the variance within the data.
  + **Use Case:** Effective when dealing with multivariate data with different scales.