**Interview Questions:**

**1. How does the KNN algorithm work?**

**K-Nearest Neighbours (KNN)** is an **instance-based learning** algorithm — it memorizes the training data instead of learning an explicit model.

**Steps:**

1. Choose the number of neighbours **K**.
2. For a new data point, calculate the **distance** (usually Euclidean) to all training points.
3. Select the **K nearest neighbours**.
4. **Classify** the new point based on the **majority class** (for classification) or **average value** (for regression).

**Example:**If K=3 and among the 3 nearest neighbours, 2 belong to class A and 1 to class B → the new point is classified as **A**.

**2. How do you choose the right K?**

Choosing **K** is crucial for performance.

**Guidelines:**

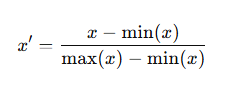
* **Small K** → model is sensitive to noise (overfitting).
* **Large K** → smoother decision boundary (may underfit).
* Usually, **odd K** values are chosen to avoid ties.
* Use **Cross-validation** to find the best K.

**Example:**Test different K values (like 3, 5, 7, 9) and select the one giving **highest accuracy** on validation data.

**3. Why is normalization important in KNN?**

KNN depends on **distance calculations**.  
If features have different scales (e.g., height in cm, income in ₹), the larger-scale feature dominates the distance.

**Normalization/Standardization** ensures all features contribute equally.

**Example:**  
Use **Min-Max Scaling**:  
 

**4. What is the time complexity of KNN?**

KNN has **no training phase** — only prediction is costly.

* **Training time:** O (1)
* **Testing time:** O (N × D)
  + N = number of training samples
  + D = number of features

Because it must calculate the distance to every training sample during prediction.

**5. What are the pros and cons of KNN?**

**Pros:**

* Simple and easy to implement.
* Works well with small datasets.
* No training phase (lazy learner).

**Cons:**

* Slow for large datasets (high prediction cost).
* Requires proper scaling.
* Sensitive to irrelevant features and noise.
* Memory-intensive (stores all data).

**6. Is KNN sensitive to noise?**

Yes, **KNN is sensitive to noise** because it relies directly on the data points.  
If noisy or mislabelled data exist near a test point, it can **mislead predictions**.

**Solution:**

* Use larger K values.
* Use **distance-weighted KNN** (closer neighbours get higher weight).
* Remove outliers or clean data before training.

**7. How does KNN handle multi-class problems?**

KNN naturally supports **multi-class classification**.

* It counts the neighbours belonging to each class.
* The class with **most votes** is assigned to the new sample.

**Example:**If K=5 and neighbours = [A, B, A, C, A] → output = **Class A** (most votes).

**8. What’s the role of distance metrics in KNN?**

Distance metrics decide **how "closeness"** between data points is measured.  
Common metrics:

* **Euclidean Distance:**  
* **Manhattan Distance:** |x\_1 - y\_1| + |x\_2 - y\_2|
* **Minkowski Distance:** general form of both.
* **Cosine Similarity:** for text or high-dimensional data.

Choosing the right metric improves KNN accuracy depending on data type.