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

KNN is a **supervised learning** algorithm used for classification and regression.  
It works by:

* Calculating the **distance** between the input sample and all samples in the training dataset.
* Selecting the **K nearest neighbors** (smallest distances).
* For **classification**, it assigns the label most common among those K neighbors.
* For **regression**, it averages the values of the K nearest neighbors.

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

Choosing the right K is critical and often done via:

* **Cross-validation**: Try different K values and choose the one with the best performance.
* **Odd K values**: For binary classification, odd K prevents ties.
* **Rules of thumb**: Start with √N (where N is number of training samples), and tune from there.
* Too small K → noisy, overfitting.
* Too large K → smooth, but may underfit.

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

KNN uses **distance metrics** (like Euclidean distance), which are **sensitive to feature scales**.  
If one feature has a much larger scale than others, it will **dominate** the distance calculation.

**Normalization (e.g., Min-Max Scaling or Standardization)** ensures all features contribute equally to the distance.

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

* **Training time**: O(1) (no training needed, it’s a lazy learner).
* **Prediction time**:
  + **O(N × D)** per query, where:
    - N = number of training samples
    - D = number of features
  + Can be optimized with data structures (KD-Tree, Ball Tree) for lower complexity in low dimensions.

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

**Pros:**

* Simple and intuitive
* No training phase (fast training)
* Naturally handles multi-class problems
* Non-parametric (no assumption about data distribution)

**Cons:**

* Slow at prediction time for large datasets
* Sensitive to irrelevant or scaled features
* Requires a good choice of K and distance metric
* Memory-intensive (stores entire training data)

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

Yes, KNN is **sensitive to noise**, especially with small K values.  
A single mislabeled data point can mislead the classification of nearby points.  
Larger K reduces this effect but can lead to underfitting.

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

KNN handles multi-class classification **naturally**:

* During prediction, it counts the number of neighbors from **each class** among the K nearest.
* The class with the **most votes** is assigned as the prediction.

No special adaptation is needed.

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

Distance metrics determine **which neighbors are considered "nearest."**  
Common ones:

* **Euclidean distance** – for continuous data
* **Manhattan distance** – for grid-like data
* **Minkowski distance** – generalization of Euclidean/Manhattan
* **Cosine similarity** – for high-dimensional data (e.g., text)

**Choosing the right metric** affects accuracy, especially for mixed-type or sparse data.