Supervised Learning

k-Nearest Neighbor Classifier

- *k*-NN classification rule is to assign to a test sample the majority category label of its *k* nearest training samples.
- In practice, *k* is usually chosen to be odd, so as to avoid ties
- The *k* = 1 rule is generally called the nearest-neighbor classification rule.

k-Nearest Neighbor Classifier

- The model is **not trained** beforehand, it runs at a time of execution to find the query output. There's no point in training the model earlier as an input of the model is training data, **hyperparameter (k), and a query point (xq).**
- **Steps of K-NN classifier:**
  - **Distance Calculation:** KNN calculates the distance between a new data point and all other data points in the dataset.
  - **Finding Nearest Neighbors:** It then identifies the 'k' nearest neighbors to the new data point.
  - **Prediction:** For classification, the new data point is assigned to the class that is most common among its 'k' nearest neighbors. For regression, the predicted value is the average of the values of its 'k' nearest neighbors.

Nearest-Neighbor Classifiers: Issues

- The value of *k*, the number of nearest neighbors to retrieve
- Choice of Distance Metric to compute distance between records
- Computational complexity
  - Size of training set
  - Dimension of data

Euclidean $\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$

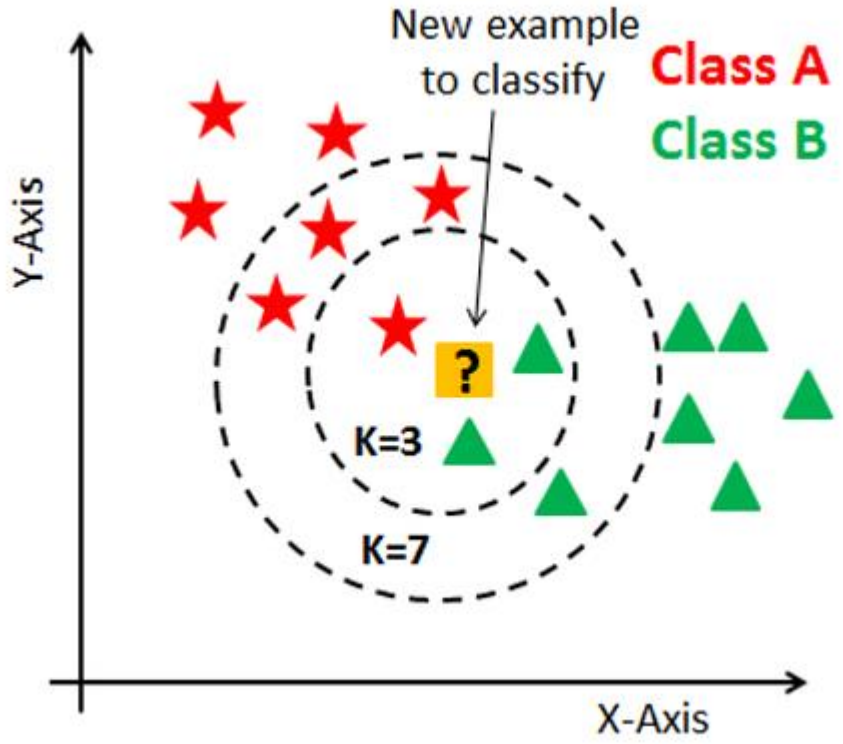
Manhattan $\sum_{i=1}^{k}|x_i - y_i|$

Minkowski $\left(\sum_{i=1}^{k}(|x_i - y_i|)^q\right)^{1/q}$

Value of K

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes

Rule of thumb:
K = sqrt(N)
N: number of training points



k-Nearest Neighbor Applications

KNN is used in various **applications**, including:

- Image Recognition: Identifying objects in images.
- Recommendation Systems: Suggesting products or content based on user preferences.
- Fraud Detection: Identifying fraudulent transactions.

Advantages:

- Simplicity: KNN is a straightforward algorithm to understand and implement.
- Versatility: It can be used for both classification and regression tasks.
- No Training Phase: It doesn't require a separate training phase, making it efficient for certain datasets.

Nearest Neighbour : Computational Complexity

- Expensive
  - To determine the nearest neighbour of a query point *q*, must compute the distance to all *N* training examples
    - \+ Pre-sort training examples into fast data structures (kd-trees)
    - \+ Compute only an approximate distance (LSH)
    - \+ Remove redundant data (condensing)
- Storage Requirements
  - Must store all training data **P**
    - \+ Remove redundant data (condensing)
    - \- Pre-sorting often increases the storage requirements
- High Dimensional Data
  - "Curse of Dimensionality"
    - Required amount of training data increases exponentially with dimension
    - Computational cost also increases dramatically
    - Partitioning techniques degrade to linear search in high dimension