KNN

K-Nearest Neighbors (KNN) is a simple, versatile, and widely used machine learning algorithm for both classification and regression tasks.

Distance Metrics

Common distance measures include:

- Euclidean distance (most common)
- Manhattan distance
- Minkowski distance
- Hamming distance (for categorical variables)

1. Distance Calculation:

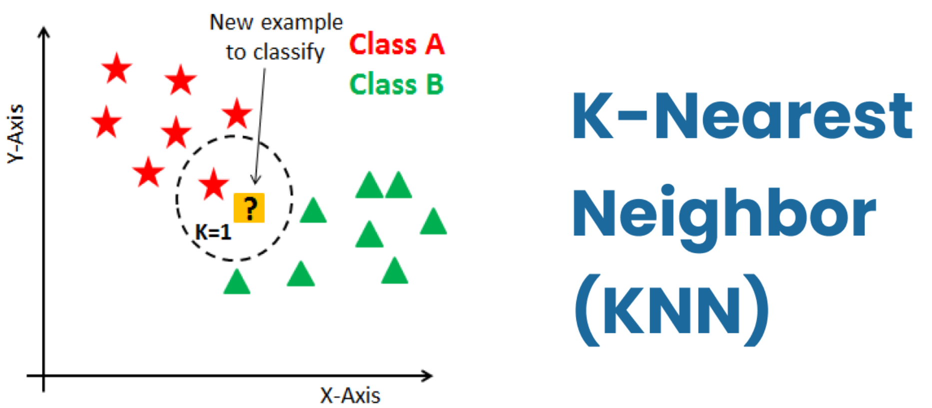
The core of KNN is calculating distances between data points. The most common distance metric is Euclidean distance, but others can be used.

Euclidean distance between two points p and q in n-dimensional space:

$$d(p, q) = \sqrt{}[(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2]$$

Or more concisely:

$$d(p, q) = \sqrt{}[\Sigma_i (p_i - q_i)^2]$$



KNN for Regression:

Process: a. For a new data point, find the K nearest neighbors in the training set. b. Calculate the average (or weighted average) of the target values of these K neighbors. c. Use this average as the prediction for the new data point.

Steps in regression

Let $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ be the training set, where x are feature vectors and y are target values.

For a new point x:

a. Calculate distances to all training points: $d(x, x_i)$ for i = 1 to n

b. Select K nearest neighbors. Let S be the set of these K neighbors.

c. Prediction is the average of the K neighbors' target values:

$$y_{pred} = (1/K) * \Sigma(y_i)$$

in S Weighted version:

$$y_{pred} = \Sigma(w_i * y_i)/\Sigma(w_i)$$

where $w_i = 1 / d(x, x_i)^2$

KNN for Classification:

The process is similar, but instead of averaging, we use majority voting.

Let C be the set of classes, and $c(x_i)$ be the class of point x_i.

Steps in classification

a. Calculate distances as in regression.

b. Select K nearest neighbors (set S).

c. For each class j in C, count occurrences in S: $count_j = \Sigma I(c(x_i) = j)$, for all x_i in S where I is the indicator function (1 if true, 0 if false)

d. Predict the class with the highest count: y_pred = argmax_j($count_j$)

Advantages:

- Simple to understand and implement
- No assumptions about data distribution (non-parametric)
- Can be used for both classification and regression
- Works well with multi-class problems

Disadvantages:

- Computationally expensive for large datasets
- Sensitive to irrelevant features and the scale of the data
- Requires feature scaling
- Does not work well with high-dimensional data (curse of dimensionality)