k-Nearest Neighbour classifier

KNN Classifier

K Nearest neighbour

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new data points and training data and put the new data point into the category that is most similar to the available categories.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

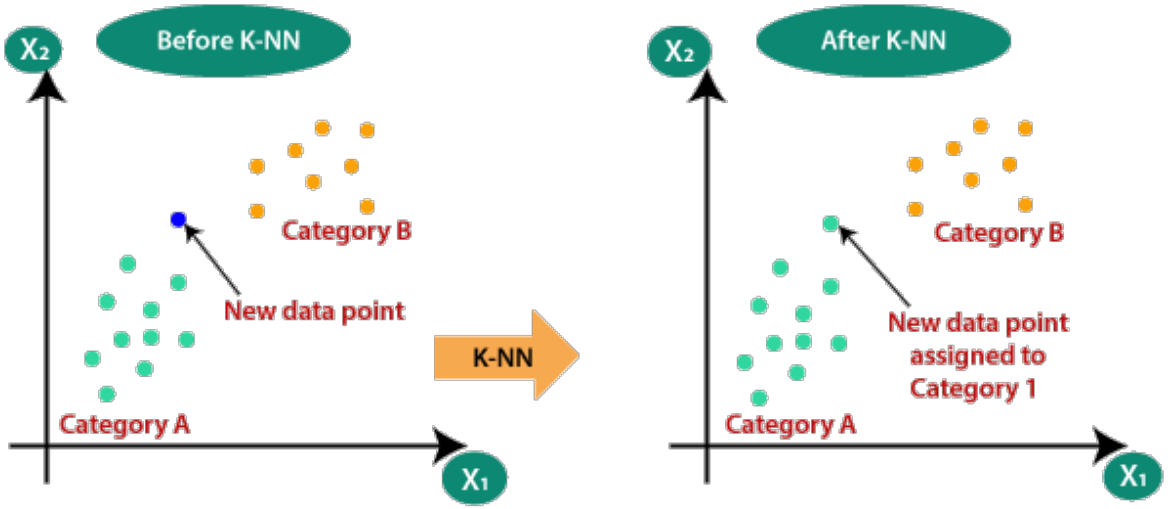
The algorithm's learning is:

1. **Instance-based learning:** Here we **do not learn weights from training data** to predict output (as in model-based algorithms) but use entire training instances to predict output for unseen data.
2. **Lazy Learning:** Model is **not learned using training data** prior and the learning process is **postponed to a time when prediction is requested** on the new instance.
3. **Non -Parametric:** In KNN, there is no predefined form of the mapping function.

KNN Classifier

Need of KNN

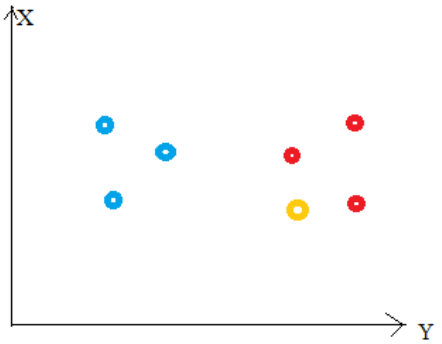
- Suppose there are two target categories, i.e., Category A and Category B, and we have a new data point x1, **so this data point will lie in which of these categories?**
- To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.



Working of KNN Classifier

Need of KNN

- Suppose there are two target categories, i.e., Category A and Category B, and we have a new data point x1, **so this data point will lie in which of these categories?**
- To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.



- Here, nearest neighbors are those data points that have minimum distance in feature space from our new data point.
- K is the number of such data points we consider in our implementation of the algorithm. Therefore, **distance metric and K value are two important considerations** while using the KNN algorithm.
- For predicting class value for a new data point, it considers all the data points in the training dataset.
- **For classification:** A class label assigned to the majority of K Nearest Neighbors from the training dataset is considered as a predicted class for the new data point.
- **For regression:** Mean or median of continuous values assigned to K Nearest Neighbors from training dataset is a predicted continuous value for our new data point.

Distance metrics in KNN Classifier

In order to determine which data points are closest to a given query point, the distance between the query point and the other data points will need to be calculated. These distance metrics help to form decision boundaries, which partitions query points into different regions.

The following distance metrics are widely used in KNN algorithm

- Euclidean distance
- Manhattan distance
- Minkowski distance
- Hamming distance

Distance metrics in KNN Classifier

Euclidean distance (p=2)

This is the most commonly used distance measure, and it is **limited to real-valued vectors**.

Using the below formula, it measures the distance between the query point and the other point being measured.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

Manhattan distance (p=1)

This is also another popular distance metric, which measures the absolute value between two points.

$$d(x,y) = \left(\sum_{i=1}^{m} |x_i - y_i|\right)$$

Distance metrics in KNN Classifier

Minkowski distance

This distance measure is the generalized form of Euclidean and Manhattan distance metrics. The parameter, p, in the formula below, allows for the creation of other distance metrics. **Euclidean distance** is represented by this formula when **p is equal to two**, and **Manhattan distance** is denoted with **p equal to one**.

$$\text{Minkowski Distance} = \left(\sum_{i=1}^{n} |x_i - y_i|\right)^{1/p}$$

Hamming distance

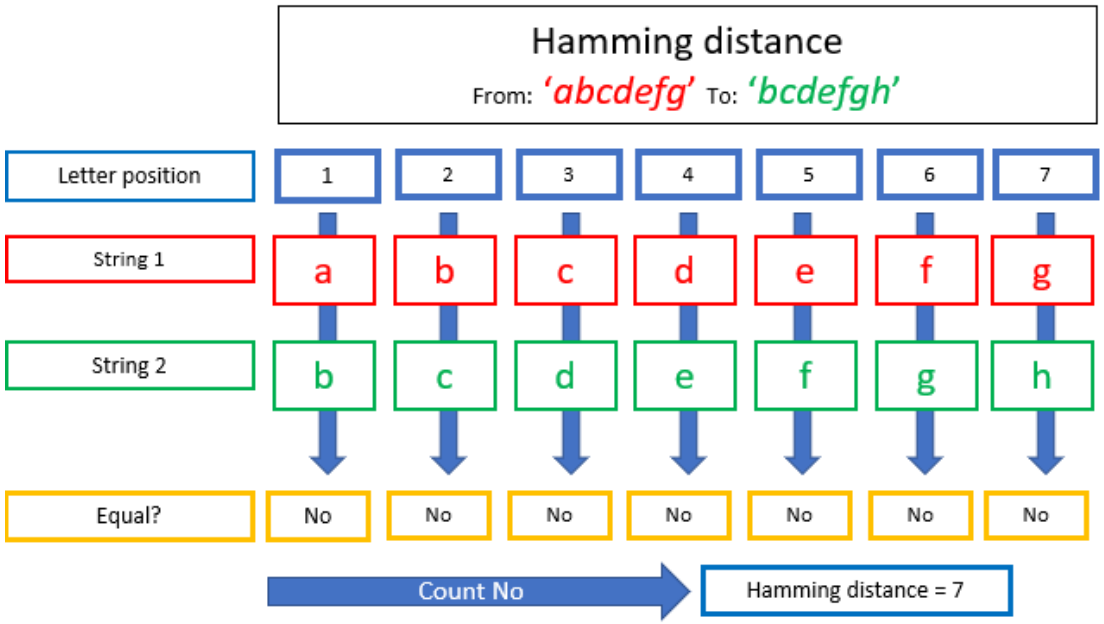
This technique is used typically used with **Boolean or string vectors**, identifying the points where the vectors do not match.

$$\text{Hamming Distance} = D_H = \left(\sum_{i=1}^{k} |x_i - y_i|\right)$$

$$x=y \quad D=0$$
$$x\neq y \quad D\neq 1$$

Hamming Distance Example

Hamming distance



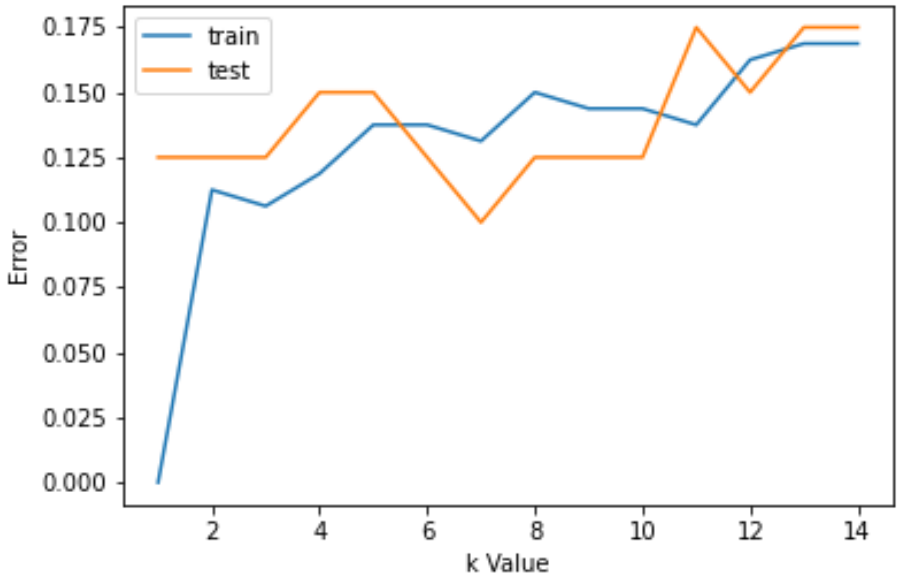
How to decide K?

Selection of K in KNN

There are two ways to decide the optimal value of K

1) For choosing the K value, we use error curves and K value with optimal train and test error is chosen as K value for prediction purposes.

2) Hyperparameter tuning to choose the value that gives the best performance.



Parameters in KNN

n_neighbors : Number of neighbors to use

weights : **'uniform'** : uniform weights. All points in each neighborhood are weighted equally.

'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.

p **:** Power parameter for the Minkowski metric. When p = 1, this is equivalent to using manhattan_distance (l1), and euclidean_distance (l2) for p = 2.

n_jobs : The number of parallel jobs to run for neighbors search. None means 1.

metric **:** Distance metric to be used