K Nearest Neighbor

Created By	Kazi Junaid Mahmud

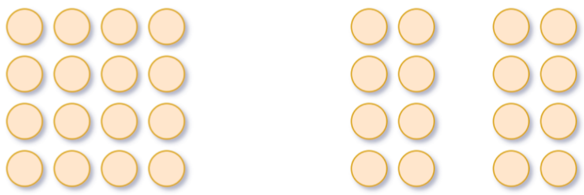
$$\textit{Assignment } 3$$

$$K\ \ Nearest\ \ Neighbor$$

Gestalt Principle of Proximity

This principle says that, elements that are closer together are perceived to be more related than elements that are farther apart, helping us to understand and organize information faster and more efficiently.

Just because the distance of the objects are less, they appear more similar than the rest.



Proximity and Similarity

Basically the closer two points are, the more similar they will be than to other data. We use this principle in building recommendation system for movies, texts, songs and more.

Machine Learning Algorithm

KNN relies on the concept of proximity and similarity

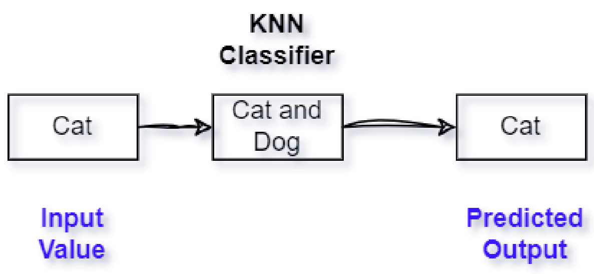
KNN Overview

- Supervised machine learning algorithm
- Can be used both as a regressor or a classifier.

Use case:

1. Agriculture: to perform climate forecasting, estimating soil water parameters, or predicting crop yields.
2. Finance: To predict bankruptcies, understanding and managing financial risk.
3. Healthcare: To identify cancer risk factors, predict heart attacks, or analyze gene expression data.
4. Internet: Using clickstream data from websites, to provide automatic recommendations to users on additional content.

Understand KNN Intuitively

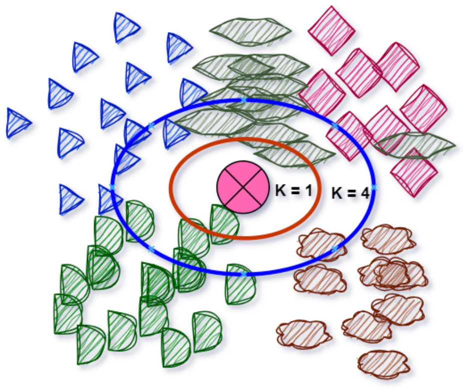


Intuitively, to understand the characteristics of person can be understood from, the group of people he hangs out with. By evaluating the characteristics of the group, we can make a judgement.

Also, for calculating the housing price, we can

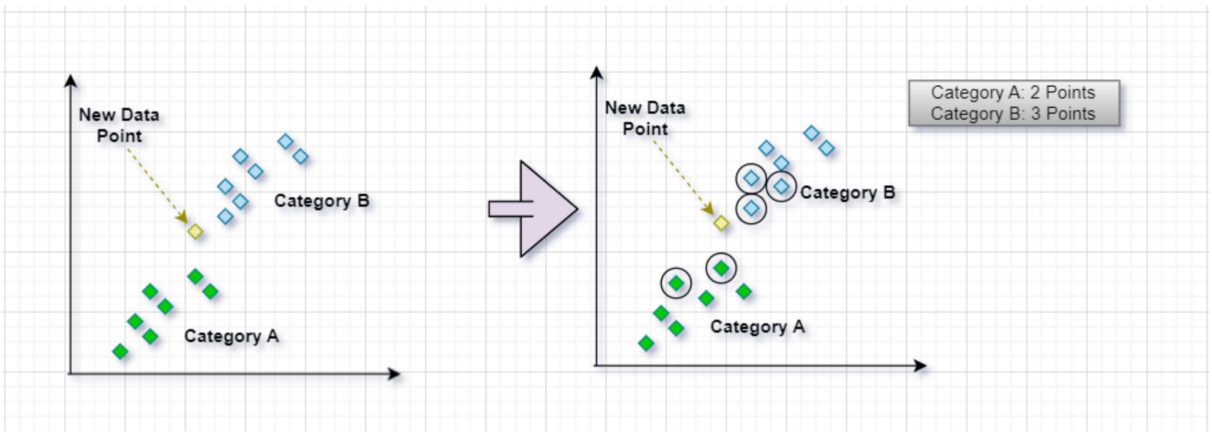
1. Locate the area
2. Identify the prices of surrounding houses
3. Make a rough calculation

- As we know, Houses with similar characteristics in the same area tend to have the same price range.

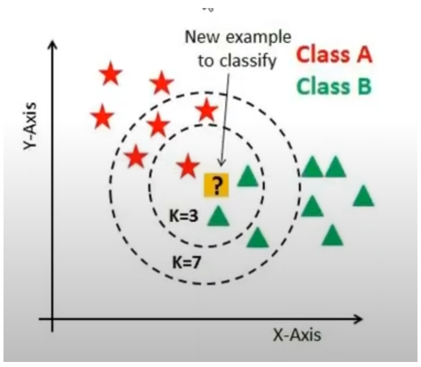


We are trying to find the similarity of the test data with the existing data points, by keeping the test data **in the center.**

- Based on number of neighbors, the class of the test data may change.
- We will draw the circles in such a way that only the specified number of neighbors fall inside the circle.



In the above image, since the test data has more category B data points (3 points) close to it than Category A (2 points), we can classify the new data point as **Category B.**



In the above picture, If we consider

- 3 neighbors, then the class is **Class B**
- 7 Neighbors, then the class is **Class A**

So the **prediction** changes if the **number of neighbors** change, which does not contain the authenticity/fidelity of the model.

How to find the nearest neighbor

We will use the Euclidean Distance/Pythagorean Distance.

$$Euclidean\ Distance \ = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Will the neighbor be selected group wise or individual wise?

- Can not do group wise by finding **mean** of all the points of a category. Because KNN does not work like that. And also it is a rough estimation which is not practical.

$\Rightarrow$ **When the training data is loaded, the calculation is done only in the inference phase. Not during training phase.**

	X1	X2	X3	Class
Training Sample 1	5	4	3	1
Training Sample 2	1	2	3	2
Training Sample 3	1	2	3	2
Test Sample	4	4	2	?

	X1	**X2**	**X3**	**Class**	**Distance**
Training Sample 1	5	4	3	1	1.4
Training Sample 2	1	2	3	2	3.6
Training Sample 3	1	2	3	2	3.7
Test Sample	4	4	2	?	

⇒ So the **Euclidean Distance** between the test sample and each training sample is calculated.

	X1	**X2**	**X3**	**Class**	**Distance**	Rank Minimum Distance
Training Sample 1	5	4	3	1	1.4	1
Training Sample 2	1	2	3	2	3.6	2
Training Sample 3	1	2	3	2	3.7	3
Test Sample	4	4	2	?		

⇒ Now we find k number of closest neighbors based on ranking.

⇒ Vote based on the maximum number of neighbors.

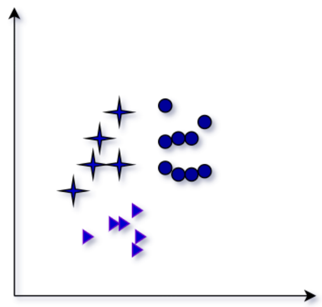
⇒ How should we determine the class of test sample?

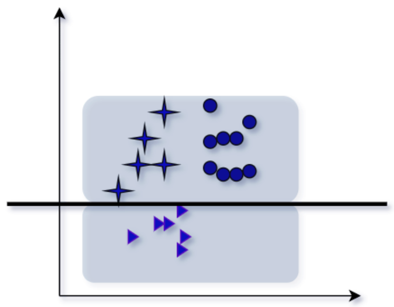
- We will consider the first instance of the rank and consider that class for the test sample.
- We can not do that using distance as distance is arbitrary and some distances can be same.
- So we are not checking distance and assigning class based on rank.

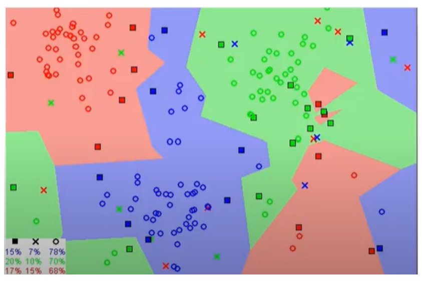
Key Aspects of KNN

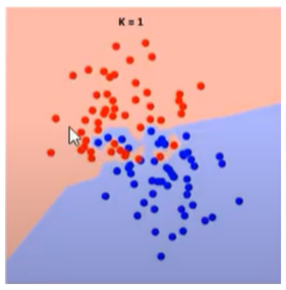
Non-Linear Model

- Higher dimension than a plane for training data.
- The regressor or classifier will be a hyperplane.









Non-parametric model

- KNN is a non-parametric model. This means that this model
- There are no parameters to learn.
- Fixed number of parameters irrespective of the data size.

Lazy Model

- KNN is one of the laziest learning methods.
- This implies storing all training data and waits until having the test data produced, without having to create a **learning model.**

Hyperparameters

Only 1 hyperparameter.

⇒ General rule for K value?

- Should be an odd value.

$$Euclidean\ Distance\ =\ \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

This function is considered as a a metric if it satisfies a certain number of properties that include the following:

- **Non-negativity:** The distance between x and y is always a **value ≥ 0**

$$d(x,y)\ \geq\ 0$$

- **Identity of indiscernibles:** The distance between x and y is equal to 0 if and only if x is equal to y.

$$d(x,y)=0\ \ iff\ \ x=y$$

- **Symmetry:** The distance between x and y is equal to the distance between y and x.

$$d(x,y)\ =\ d(y,x)$$

- **Triangle Inequality:** Considering the presence of a third point z, the distance between x and y is always less than or equal to the sum of the distance between x and z and the distance between y and z.

$$d(x,y)\leq d(x,z)+d(z,y)$$

⇒ Distance is a type of similarity measure. How?

- When the distance is in the range [0, 1], the calculation of a corresponding similarity measure s(x, y) is as follows:

$$s(x, y) = 1 - d(x, y)$$

Distance Measures

- **Lp distance/ Minkowski Distance Measures**

$$D_{Mink}(x, y) = \sqrt[p]{(}\sum_{i=1}^{n} |x_i - y_i|^p)$$

If p = 1**, Manhattan Distance**

MD: The MD, also known as L1 norm, Taxicab Norm, Rectillinear distance or City Block Distance or Manhattan Distance.

$$MD(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

If p = 2, **Euclidean Distance**

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

If p = infinity, **Chebyshev Distance**

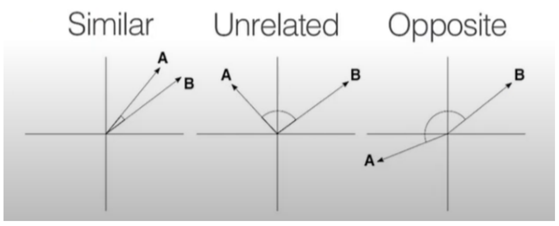
$$CD(x, y) = \overset{max}{i} \ |x_i - y_i|$$

- **Inner Product Distance Measures**

CosD: The CosD, also called **angular distance,** is derived from the cosine similarity that measures the angle between two vectors, where CosD is obtained by **subtracting the cosine similarity from 1.**

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

Similarity is calculated by the cosine of the angle between two vectors. Vectors which are most similar will have a value of 0 degrees between them (the value of cos = 0 is 1), while vectors are most dissimilar will have a value of -1. The smaller the angle, the higher the similarity.

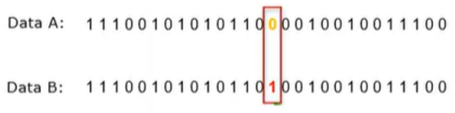


Other distance measures

Hamming Distance

- Measures the number of mismatches between two vectors.
- Mostly used for nominal data, string, and bit-wise analyses, and also can be useful for no numerical data.

$$HamD(x,y) = \sum_{i=1}^{n} 1_{x_i \neq yi}$$

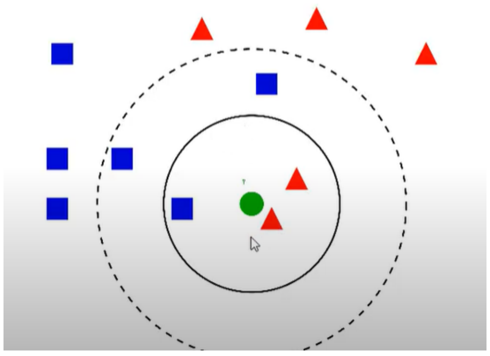

Data A: 1 1 1 0 0 1 0 1 0 1 0 1 1 0 0 0 0 1 0 0 1 0 0 1 1 1 0 0
Data B: 1 1 1 0 0 1 0 1 0 1 0 1 1 0 1 0 0 1 0 0 1 0 0 1 1 1 0 0

Advantages of KNN

- The algorithm is simple and easy to implement.
- There's no need to build a model, tune several parameters, or make additional assumptions.
- Versatile algorithm. Can be used for classification, regression.

Disadvantages of KNN

- Slow for large dataset.
- Choosing the best K-neighbors problem.
- Choosing the best distance/similarity measure is an important problem.
- Performance of the KNN classifier is dependent on the distance/similarity measure used.



- Time complexity.
- High computational time cost as we need to compute the distance between each test sample and all training samples.
- For each test example we need O(nm) time complexity (number of operations).
- n is the number of examples in the training data and m is the number of features for each example.

KNN pseudo-code

```
Input: Training samples D, test sample d, K
Output: Class label of test sample
    1. Compute the total distance between d and every sample in D.
    2. Choose the K samples in D that are nearest to d; denote the set by p ( D)
    3. Assign d the class that is the most frequent class (or the mejority class)
```

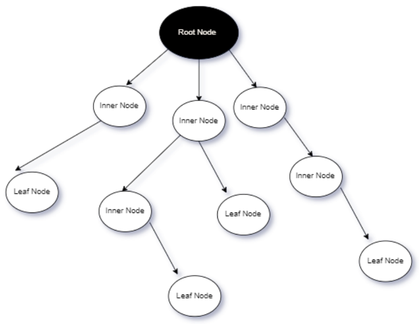
Training Phase

- The training samples and the class labels of these samples are stores.
- No missing data allowed.
- No non-numeric data allowed.

Testing Phase

1. Load the data.
2. Initialize K to your chosen number of neighbors.
3. For each example in the data
    a. Calculate the distance between the query example and the current example from the data.
    b. Add the distance and the index of the example to an ordered collection.
    c. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances.
    d. Pick the first K entries from the sorted collection.
    e. Get the labels of the selected K entries.
    f. If regression, return the mean of the K labels.
    g. If classification, return the mode of the K labels.

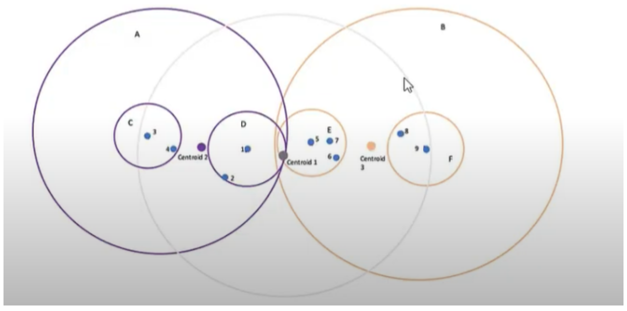
k-Dimensional Tree (kd tree)



kd tree

A k-dimensional tree, or k-d tree, is a data structure commonly used in machine learning for organizing and manipulating point data in a k-dimensional space. It helps to solve various problems more efficiently, such as nearest neighbour search, which is vital in many machine learning algorithms.

Ball Tree



Ball Tree

A ball tree is another useful data structure in machine learning, especially designed to accelerate the process of nearest neighbor search for points in a multi-dimensional space. Like k-d trees, ball trees are particularly efficient for organizing data in ways that make search operations faster and more scalable, particularly when dealing with complex metric spaces or high-dimensional data.