Instance-based Learning

K-Nearest Neighbour Learning

Instance-based Learning

To classify a new instance, its relationship to the previously stored examples is examined in order to assign a target function value for the new instance.

Some commonly used instance-based methods are:

- Nearest neighbor method
- Locally weighted regression method
- Case-based reasoning method

Instance-based methods are "Lazy"

Instance based learning methods are sometimes referred to as "lazy" learning methods because they delay processing until a new instance must be classified.

The advantage of this approach is that instead of estimating the target function once for the entire instance space these methods can estimate it locally and differently for each new instance to be classified.

Characteristics of Instance-based Learning Methods

Instance-based methods can construct a different approximation to the target function for each distinct query instance that must be classified.

These methods can use more complex, symbolic representations for instances.

Disadvantage of these approaches is that the cost of classifying new instances can be high.

Another limitation of these methods, specially KNN, is that they typically consider all attributes of the instances while attempting to retrieve similar training examples from memory.

k-Nearest Neighbour Learning

k-Nearest Neighbour Learning

It is the most basic instance-based learning method.

kNN is a non-parametric supervised learning technique.

kNN captures information of all training cases and classifies new cases based on a similarity.

The nearest neighbors (i.e. similarity) of an instance are defined in terms of the standard Euclidean distance.

The target function may be discrete-valued or real-valued.
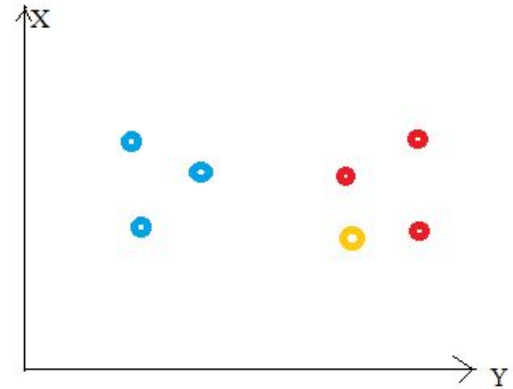
Euclidean Distance

Euclidean Distance

In mathematics, the Euclidean distance between two points in Euclidean space is the length of a line segment between the two points.

It can be calculated from the Cartesian coordinates of the points using the Pythagorean theorem, therefore occasionally being called the Pythagorean distance.

Distance between two points (p_1,q_1) and (p_2,q_2) can be calculated as shown in the figure.

$$d(p, q)^2 = (q_1 - p_1)^2 + (q_2 - p_2)^2$$
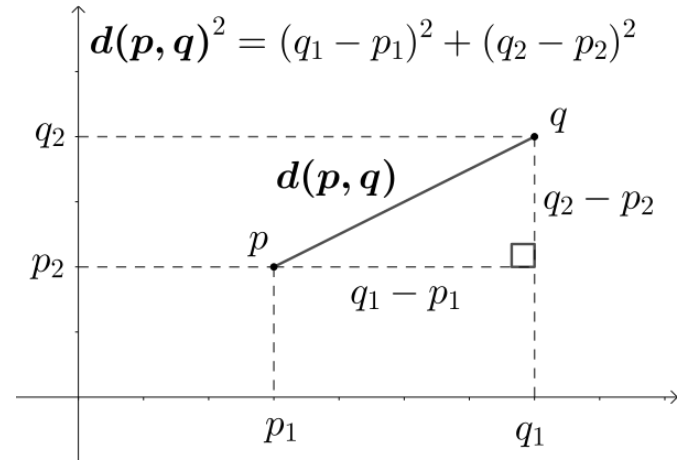
Euclidean Distance

In three dimensions, for points given by their Cartesian coordinates, the distance is

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}.$$

In general, for points given by Cartesian coordinates in n-dimensional Euclidean space, the distance is

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

Example: Euclidean Distance

Find out the euclidean distance between the points (1,2,-3) and (4,-1,2).

$$= \sqrt{(1-4)^2 + (2-(-1))^2 + ((-3)-2)^2}$$

$$= \sqrt{9 + 9 + 25} = \sqrt{43}$$

$$= 6.56$$

How kNN works?

Suppose we have height, weight and T-shirt size of some customers and we need to predict the T-shirt size of a new customer given only height=161cm and weight=61kg information we have. Predict the size of T-shirt for the new customer.

Few rows of the data including height, weight and T-shirt size information is shown below -

Height (in cms)	Weight (in kgs)	T-shirt Size
158	58	M
163	60	M
165	62	L
170	64	L

Step 1: Calculate Similarity based on distance function

Calculate the euclidean distance between the new test example and all the training examples.

Rank the training examples in terms of their distance from new test example. The smallest distance value will be considered as rank 1 and considered as nearest neighbour.

	A	B	C	D	E
				fx =SQRT((A21-A6)^2+(B21-B6)^2)	
1	Height (in cms)	Weight (in kgs)	T Shirt Size	Distance	
2	158	58	M	4.2	
3	158	59	M	3.6	
4	158	63	M	3.6	
5	160	59	M	2.2	3
6	160	60	M	1.4	1
7	163	60	M	2.2	3
8	163	61	M	2.0	2
9	160	64	L	3.2	5
10	163	64	L	3.6	
11	165	61	L	4.0	
12	165	62	L	4.1	
13	165	65	L	5.7	
14	168	62	L	7.1	
15	168	63	L	7.3	
16	168	66	L	8.6	
17	170	63	L	9.2	
18	170	64	L	9.5	
19	170	68	L	11.4	
20					
21	161	61			

Step 2: Find K-Nearest Neighbors

Select the top K nearest neighbours of the test example based on their distance calculated in previous step.

Assign the most common value among the k nearest training examples.

In case of real valued target function, the mean of target values of k nearest examples is output as target value for test example.

Visualizing KNN (for k=5)

In the graph below, binary dependent variable (T-shirt size) is displayed in blue and orange color. 'Medium T-shirt size' is in blue color and 'Large T-shirt size' in orange color.
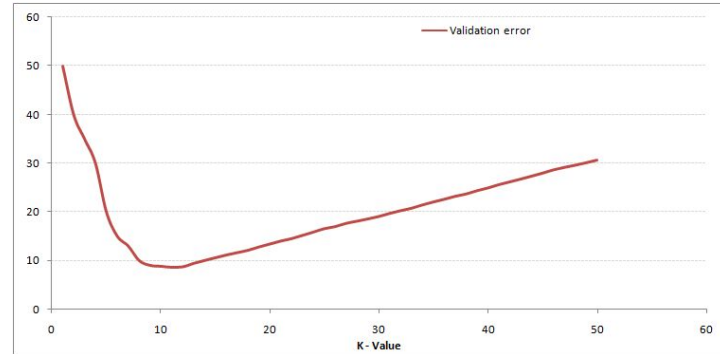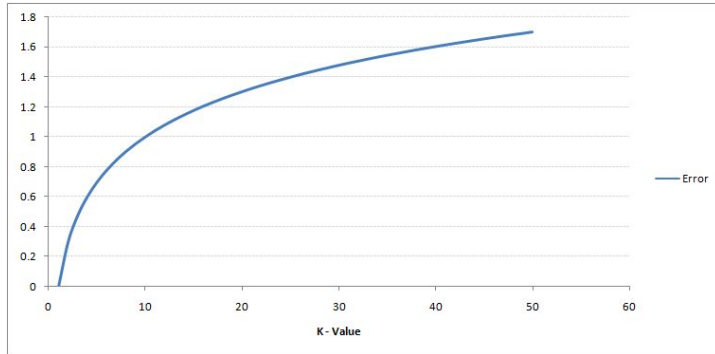
New customer information is exhibited in yellow circle.

Four blue highlighted data points and one orange highlighted data point are close to yellow circle. So, the prediction for the new case is blue highlighted data point which is **Medium** T-shirt size.
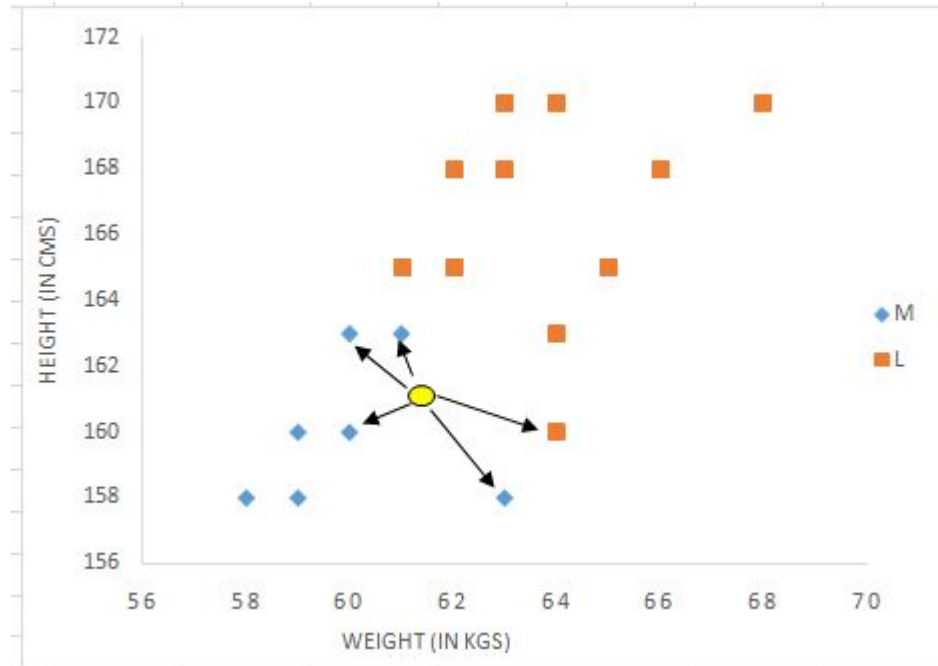
How to choose 'k'?

Using cross validation charts, we can find out the optimum value of k.

For a very low value of k (suppose k=1), the model overfits on the training data, which leads to a high error rate on the validation set. On the other hand, for a high value of k, the model performs poorly on both train and validation set. If you observe closely, the validation error curve reaches a minima at a value of k = 9. This value of k is the optimum value of the model (it will vary for different datasets).

IMAGE SOURCE: https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/

Assumptions of KNN

1. Standardization

    When independent variables in training data are measured in different units, it is important to standardize variables before calculating distance. For example, if one variable is based on height in cms, and the other is based on weight in kgs then height will influence more on the distance calculation. In order to make them comparable we need to standardize them.

2. Outlier

    Low k-value is sensitive to outliers and a higher K-value is more resilient to outliers as it considers more voters to decide prediction.

Pros of KNN

1. Easy to understand

2. No assumptions about data

3. Can be applied to both classification and regression

4. Works easily on multi-class problems

Cons of KNN

1. Memory Intensive / Computationally expensive

2. Sensitive to scale of data

3. Not work well on rare event (skewed) target variable

4. Struggle when high number of independent variables

References

- Chapter 8, Tom M. Mitchell, Machine Learning
- https://en.wikipedia.org/wiki/Euclidean_distance#:~:text=In%20mathematics%2C%20the%20Euclidean%20distance,being%20called%20the%20Pythagorean%20distance.
- https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/
- https://www.listendata.com/2017/12/k-nearest-neighbor-step-by-step-tutorial.html
-