K-Nearest Neighbours

K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

When do we use KNN algorithm

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output
2. Calculation time
3. Predictive Power

Let us take a few examples to place KNN in the scale:

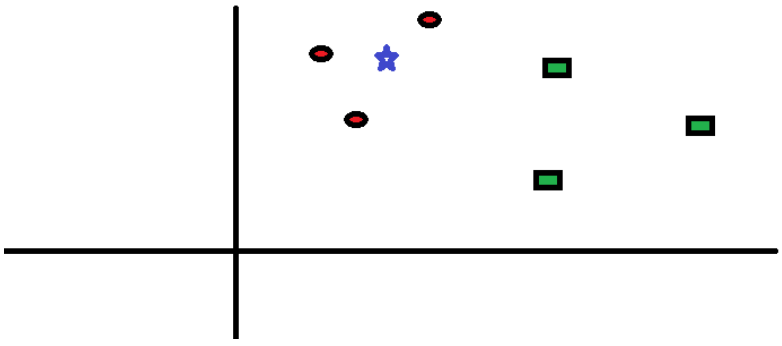
	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.

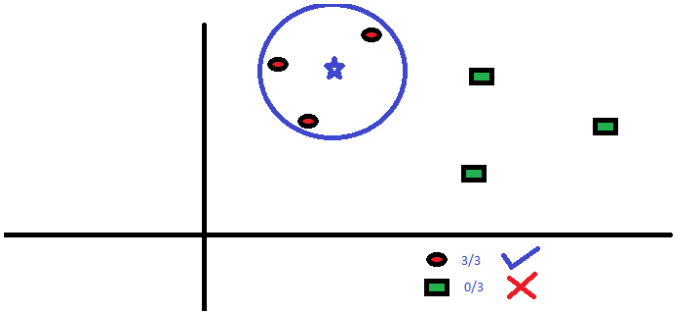
It is widely disposable in real-life scenarios since it is non-parametric, meaning; it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data).

How does the KNN algorithm work

Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS):



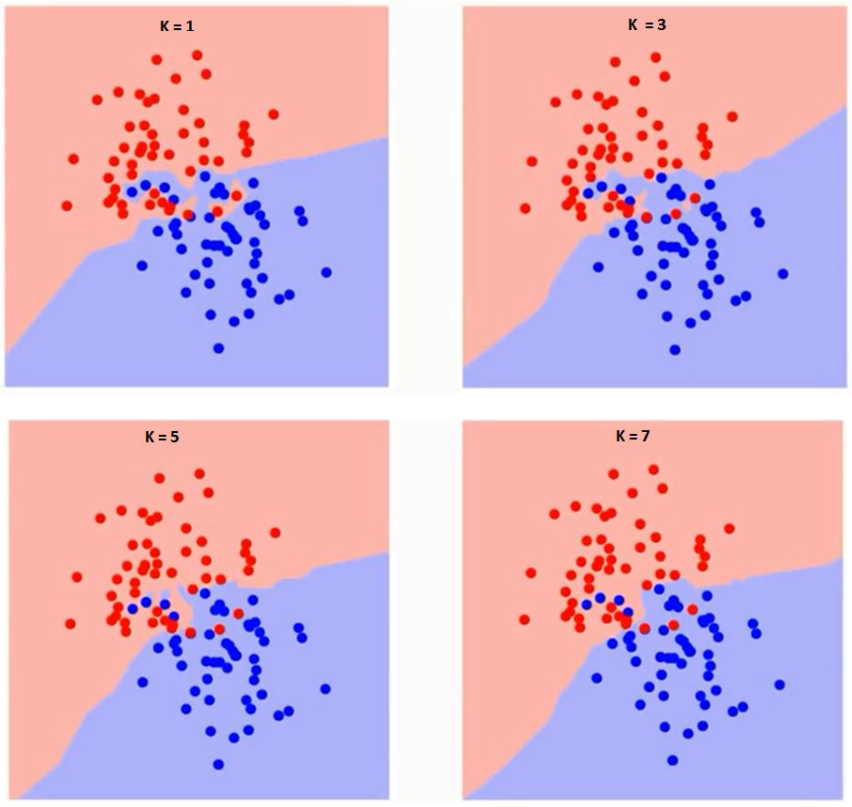
You intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The "K" is KNN algorithm is the nearest neighbour we wish to take the vote from. Let's say K = 3. Hence, we will now make a circle with BS as the centre just as big as to enclose only three data points on the plane. Refer to the following diagram for more details:



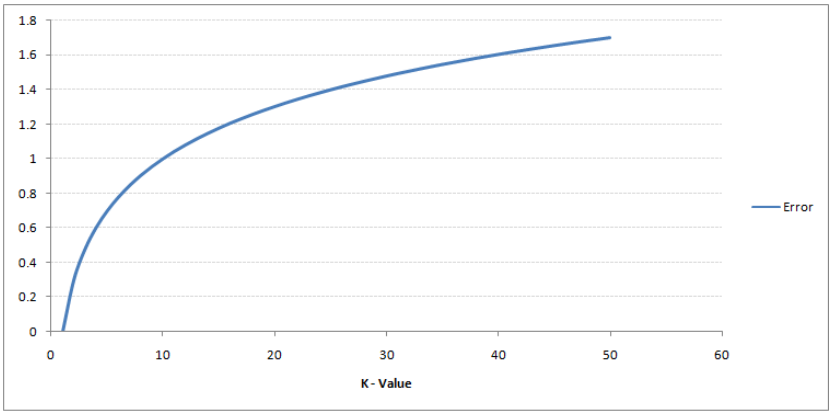
The three closest points to BS is all RC. Hence, with a good confidence level, we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbour went to RC. The choice of the parameter K is very crucial in this algorithm. Next, we will try to understand what are the factors to be considered to conclude the best K.

How do we choose the factor K

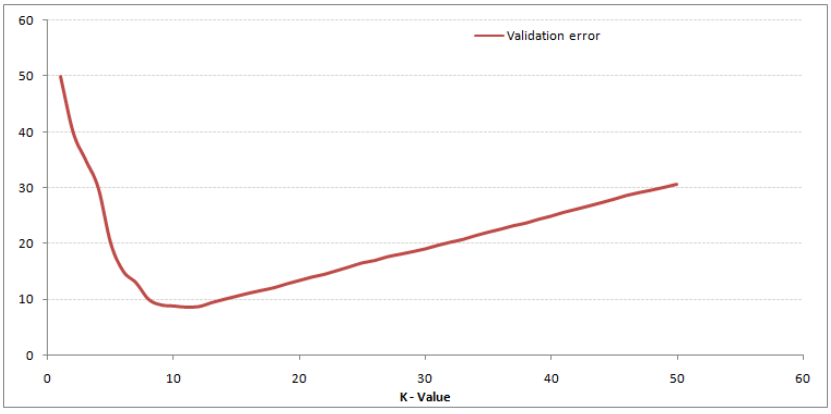
First let us try to understand what exactly K influences in the algorithm. If we see the last example, given that all the 6 training observation remain constant, with a given K value we can make boundaries of each class. These boundaries will segregate RC from GS. In the same way, let's try to see the effect of value "K" on the class boundaries. The following are the different boundaries separating the two classes with different values of K.



If you watch carefully, you can see that the boundary becomes smoother with increasing value of K. With K increasing to infinity it finally becomes all blue or all red depending on the total majority. The training error rate and the validation error rate are two parameters we need to access different K-value. Following is the curve for the training error rate with a varying value of K:



As you can see, the error rate at K=1 is always zero for the training sample. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with K=1. If validation error curve would have been similar, our choice of K would have been 1. Following is the validation error curve with varying value of K:



This makes the story clearer. At K=1, we were over fitting the boundaries. Hence, error rate initially decreases and reaches a minimal. After the minima point, it then increases with increasing K. To get the optimal value of K, you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K. This value of K should be used for all predictions.

Algorithm

Let m be the number of training data samples. Let p be an unknown point.

1. *Store the training samples in an array of data points arr[]. This means each element of this array represents a tuple (x, y).*
2. *for i=0 to m:*
3. *Calculate Euclidean distance d(arr[i], p).*
4. *Make set S of K smallest distances obtained. Each of these distances corresponds to an already classified data point.*
5. *Return the majority label among S.*

K can be kept as an odd number so that we can calculate a clear majority in the case where only two groups are possible. With increasing K, we get smoother, more defined boundaries across different classifications. Also, the accuracy of the above classifier increases as we increase the number of data points in the training set.

Referance:

https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

https://www.geeksforgeeks.org/k-nearest-neighbours/