|  |  |  |
| --- | --- | --- |
| No. | Student Name | Student ID |
| 1 | Vu Tuan Anh | 17110004 |
| 2 | Pham Vu Thien | 17110075 |

**k-Nearest Neighbors**

1. **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:

* Ease to interpret output
* Calculation time
* Predictive power

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

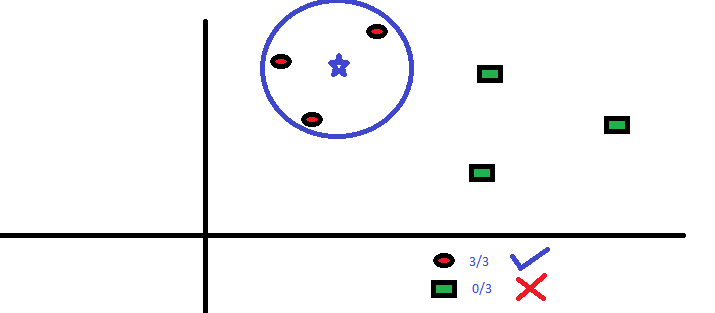
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Logistic Regression | CART | Random Forest | KNN |
| Ease to interpret output | 2 | 3 | 1 | 3 |
| Calculation time | 3 | 2 | 1 | 3 |
| Predictive power | 2 | 2 | 3 | 2 |

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

1. **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 neighbor we wish to take the vote from. Let’s say K = 3. Hence, we will now make a circle with BS as the center just as big as to enclose only three datapoints on the plane. Refer to the following diagram for more details:



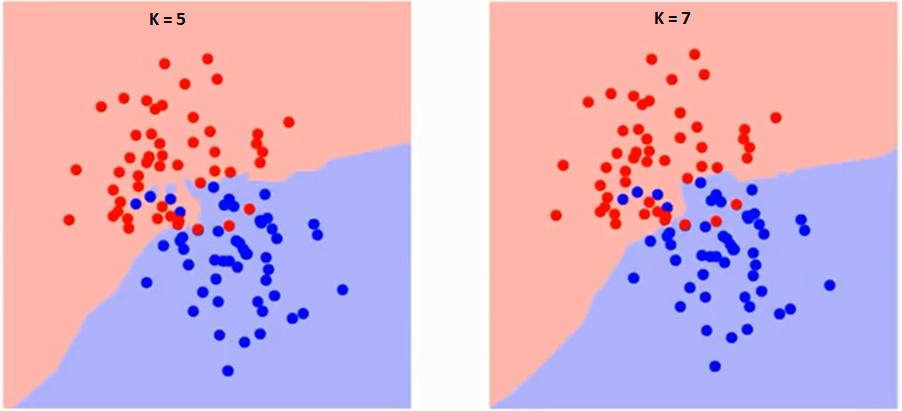
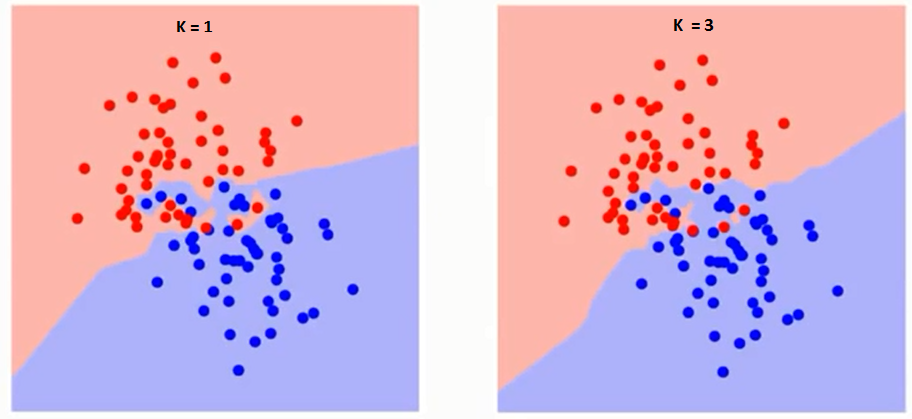
**Figure 1.** KNN algorithm applied

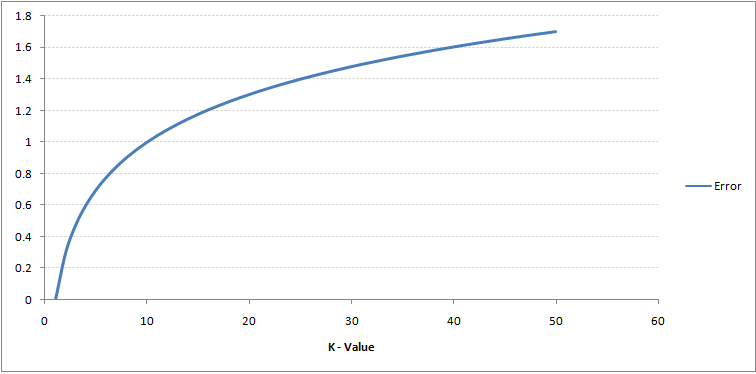
The three closest points to BS are all RCs. Hence, with a good confidence level, we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm. Next, we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

1. **How do we choose the factor K?**

First let us try to understand what exactly does K influence in the algorithm. If we see the last example, given that all the 6 training observations 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.

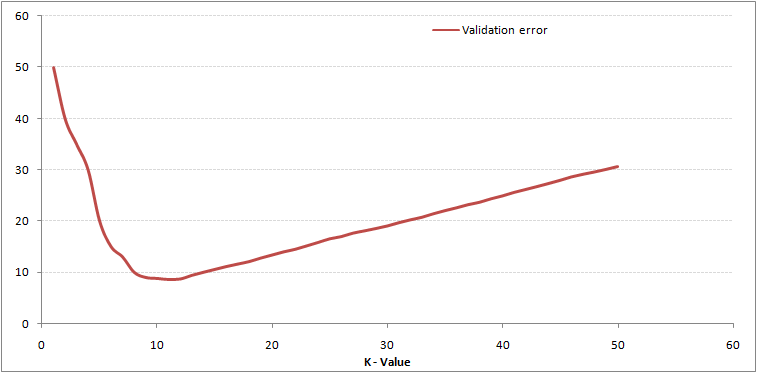
**Figure 2**. The differences between vary k



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:

**Figure 3.** Relationship between the rate of error and the value 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:



**Figure 4.** The relationship between the rate of validation error and the value k

This makes the story more clearly. At K=1, we were overfitting the boundaries. Hence, error rate initially decreases and reaches a lowest point. 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.

1. **Breaking it Down – Pseudo Code of KNN**

We can implement a KNN model by following the below steps:

* Load the data
* Initialize the value of k
* For getting the predicted class, iterate from 1 to total number of training data points
  + Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it’s the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
  + Sort the calculated distances in ascending order based on distance values
  + Get top k rows from the sorted array
  + Get the most frequent class of these rows
  + Return the predicted class

**Reference**

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