K-Nearest Neighbor Classifier

Introduction

KNN is a simple and easy to implement **supervised** and **non-parametric** machine learning algorithm that is categorized as **a lazy learner** and is used for profusely used for **classification** and sometimes for **regression**. Let us investigate the above highlighted machine-learning terms before moving ahead.

Supervised Machine Learning

This is a category of Machine Learning that uses labeled datasets as its input to feed into the model for training and later predicts or classifies the outcome based on the training dataset. Such kind of algorithms are used for classification and regression problem. Accordingly, k-NN is classified as one of the supervised learning algorithms due to its working mechanism identical to that of this learning principle. The working convention of the k-NN algorithm is discussed further.

Non-parametric learning

This is the kind of machine learning methodology in which there exists no pre-defined parameter or mapping function for predicting the outcome. K-NN is one of such learning models as it predicts based on similarity measures and not any pre-defined functions.

Lazy Learner

K-NN is categorized as a lazy learner because it does not determine the outcome based on a certainly trained model but instead simply memorizes the overall dataset for decided the result.

Industrial Implementation of k-NN algorithm

K-NN is one of the widely used algorithms today majorly for classification purpose around various sectors such as agriculture, Finance, Medical, Facial Recognition and recommendation systems. In agriculture K-NN has been used to for forecasting the climate and estimating the soil and water parameters. Likewise, it also can be used in the finance sector, for stock prediction and analysis. It can be used as a classifier to identity the possibilities of a health condition based on various related features. And finally, one of the most used use cases of this algorithm is the recommendation systems present on various online shopping spaces. In fact, these e-businesses have been able to gather some additional revenue with the integration of this machine learning algorithm.

Working Convention:

Principle:

To build an elementary understanding of the algorithm let us contemplate the statement, "*Your identity is determined by the kind you are dominantly surrounded to.*" As stated, K-NN works based on a similarity measure that brings out the outcome depending on its most prominent neighboring class.

Mechanism:

To efficiently figure out the strategy of the algorithm. Let us visualize the following graphs.

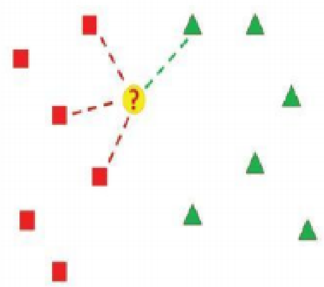
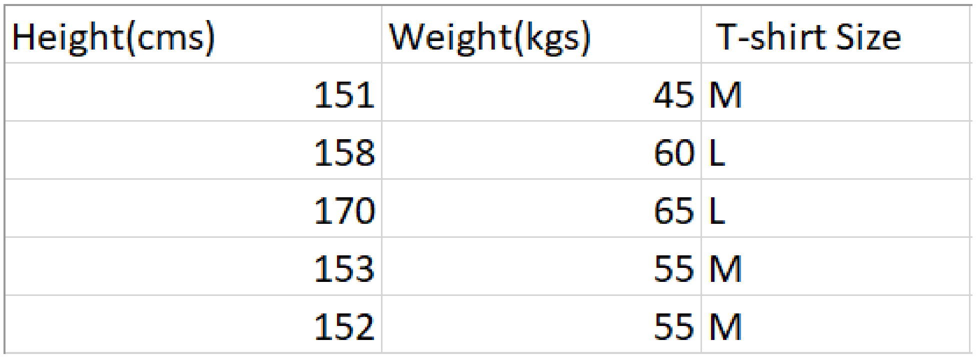


Figure 1: Sample Representation of K-NN classifier.

In the above diagram, the question mark refers to the new data point that requires classification whereas the red squares and green triangles refer to the labeled dataset we primarily have. Now, as per the principal, k-NN algorithm determines the k most-nearest neighbors to the new data point. Like in above, we have fed the value of k to be 4 which led to the identification of four nearest neighbors. These nearest neighbors are determined based through the proximity evaluation i.e., through the registration of top k data points nearest in the distance.

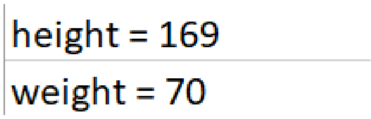
Determining the k Nearest Neighbors



Height(cms)	Weight(kgs)	T-shirt Size
151	45	M
158	60	L
170	65	L
153	55	M
152	55	M

Figure 2: Sample Dataset

Let us use the above created sample data set consisting of a classified collection of t-shirt sizes based on the height and weight features. Our task now is to predict the class (either M or L in based on the current condition) of the new set of data.



height = 169

weight = 70

Figure 3: Sample Dataset

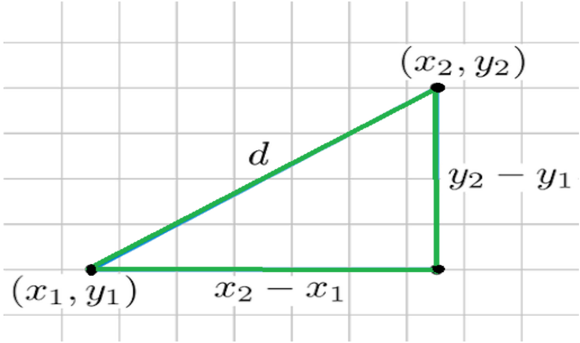
More specifically, let us work to find out whether which category the t-shirt size falls into when the height measures 169 and weight 70.

Step 1: The first task is to determine the value of k. As mentioned earlier, *K* refers to the number of nearest neighbors we are to determine for the new data point.

Here,

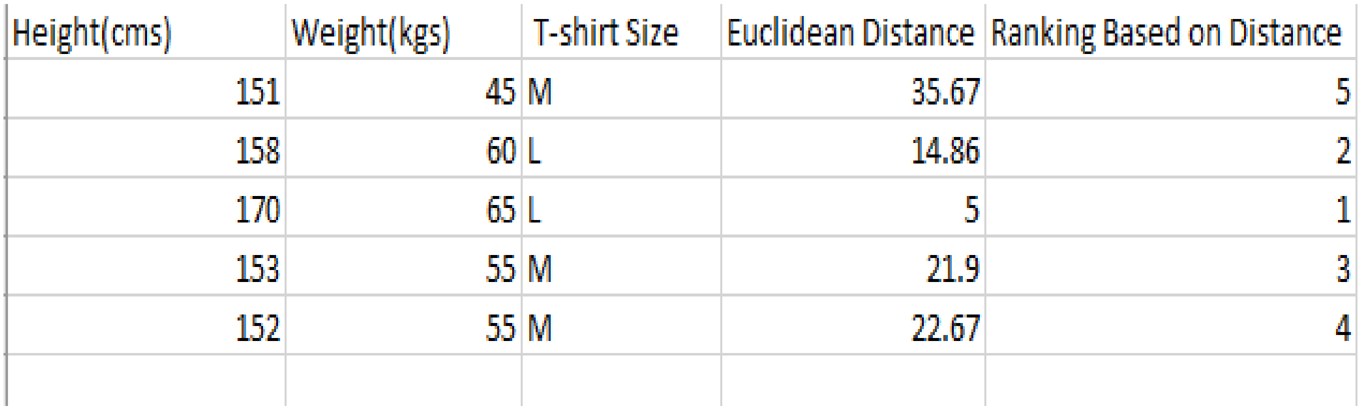
$$k = 3$$

We first calculate the Euclidean distance between the data point and labeled data. The formula to calculation the distance is given as



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

Figure 4: Formula for computing Euclidean Distance

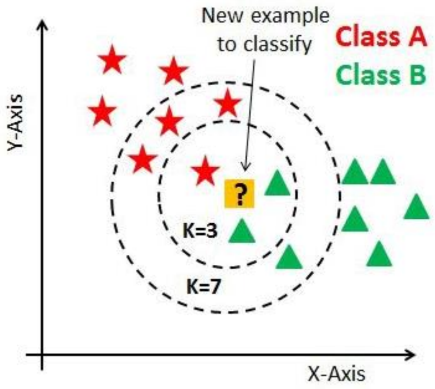


Height(cms)	Weight(kgs)	T-shirt Size	Euclidean Distance	Ranking Based on Distance
151	45	M	35.67	5
158	60	L	14.86	2
170	65	L	5	1
153	55	M	21.9	3
152	55	M	22.67	4

Figure 5: Finding the k-nearest neighbors.

Above we have calculated the distance of the new data point with all the existing instances and ranked based on the nearest first basis. Now as k = 3 we can select the third, second and third instance as the top three nearest neighbors, and L as the most dominant class, the data point falls under the "L" category.

Importance of the K- value in k-NN



The beside diagram portrays the variation in outcome brought by the value of k during the implementation of k-nearest neighbor. The data point gets classified as a Class B when k gets initialized as 3 whereas b when k = 7.

Figure 6: Via datacamp.com

This condition here elaborates the influence of k while implementation K-NN.

Selecting the Optimal Value for K

The truth be told, there are no instant statistical methods to find the optimal k. The most prevalent way is to randomly initialize your k and start computing. But few of the things to be considered while submitting the k-value is to not assign it based on the size of the dataset. Less value is likely to lead to unstable decisions. The second tip is to go for an uneven value to prevent the oddity brought by ties. Lastly, another optimal way would be to pick a k-value with the lowest error rate that would also help in acceleration of the efficiency.

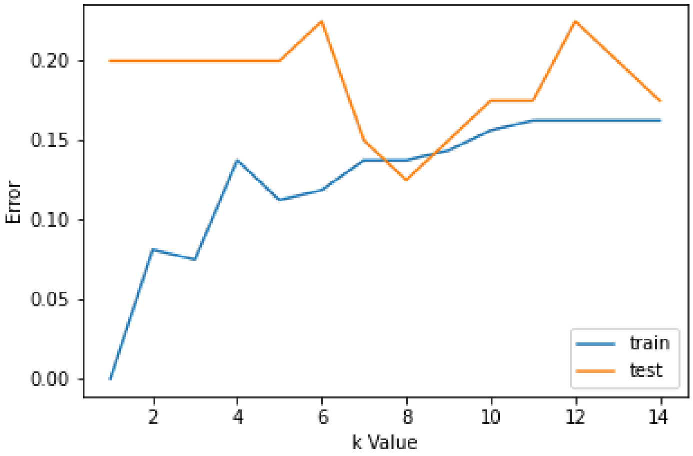


Figure 7: k Value vs Error value plot.

As we can see that the above graph gives out least error rate when k = 8 the optimal value for the plotted dataset is 8.

Pseudocode for k-NN Algorithm

```
kNN (dataset, sample) {

traverse through each dataset instance and calculate the distance between the   sample
and each instance.

Find the k nearest neighbors.

Classify the sample as the majority class present among the nearest neighbors.

}
```

Pros

- Simple to implement.
- Requires no training model.
- Can be used for regression as well as classification.
- Adapts based on the evolving dataset.

Cons

- Requires Missing Value Treatment.
- Necessity of homogenous features.
- Sensitive to Outlier dataset.
- A victim to the curse of dimensionality.

Figure 8: Via miro.medium.com