Classification: K-Nearest Neighbours (K-NN)

Prof. (Dr.) Honey Sharma

KNN Classification

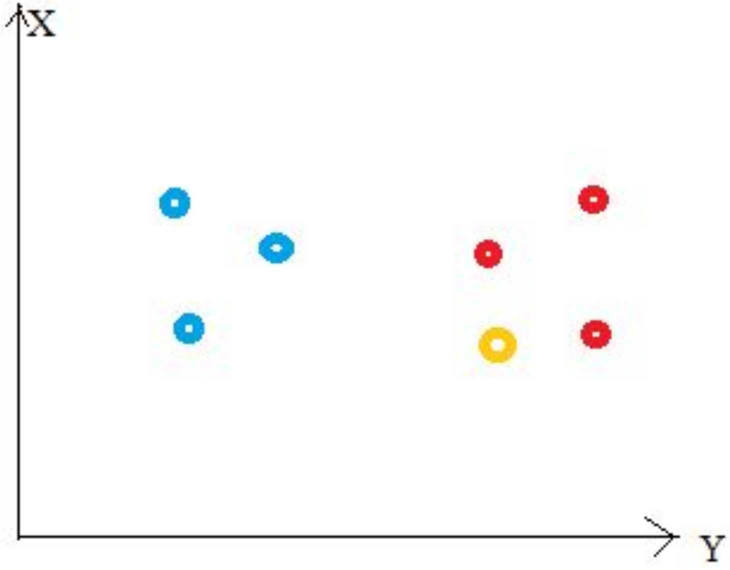
The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

“Birds of a feather flock together.”

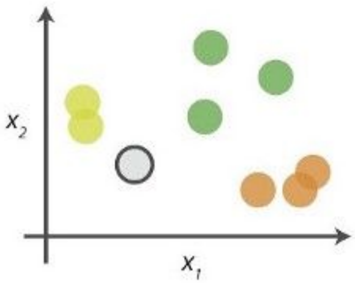
The KNN algorithm hinges on assumption similar data points are close to each other. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics — calculating the distance between points on a graph.

It manipulates the training data and classifies the new test data based on distance metrics. It finds the k-nearest neighbors to the test data, and then classification is performed by the majority of class labels.

Consider the following figure. Let us say we have plotted data points from our training set on a two-dimensional feature space. As shown, we have a total of 6 data points (3 red and 3 blue). Red data points belong to ‘class1’ and blue data points belong to ‘class2’. And yellow data point in a feature space represents the new point for which a class is to be predicted. Obviously, we say it belongs to ‘class1’ (red points)

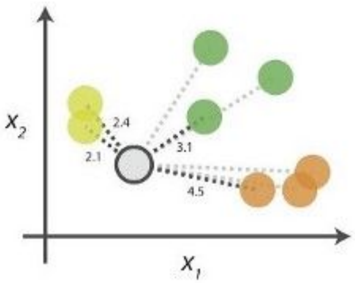


0. Look at the data



Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances



Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point	*Distance*	
⚪··🟡	2.1	→ 1st NN
⚪··🟡	2.4	→ 2nd NN
⚪··🟢	3.1	→ 3rd NN
⚪··🟠	4.5	→ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

Class	*# of votes*
🟡	2
🟢	1
🟠	1

➡

Class 🟡 wins the vote!

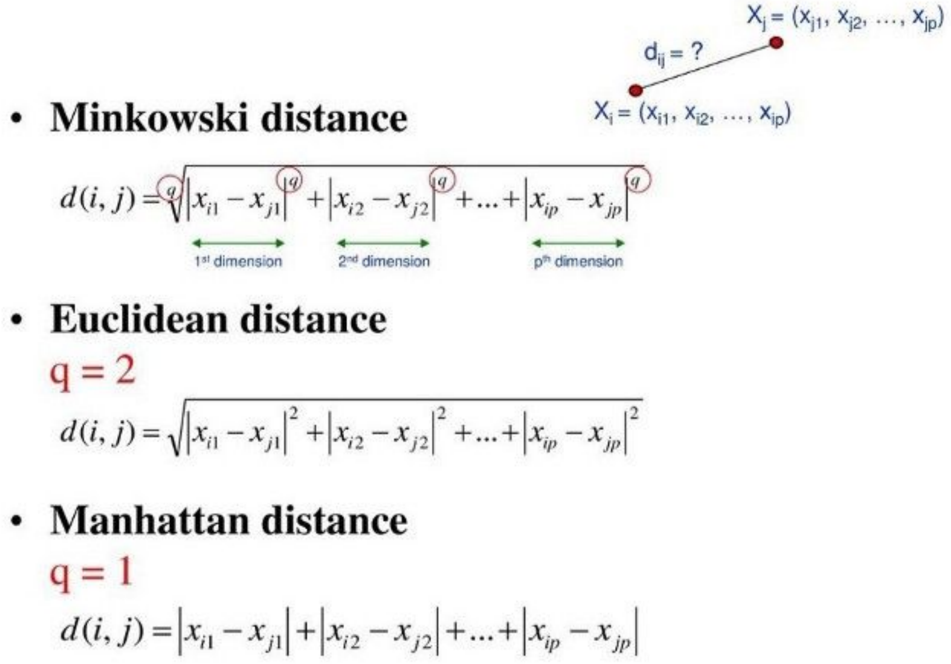
Point ⚪ is therefore predicted to be of class 🟡.

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Distance Metrics

The distance metric is the effective hyper-parameter through which we measure the distance between data feature values and new test inputs.

Usually, we use the Euclidean approach, which is the most widely used distance measure to calculate the distance between test samples and trained data values.


$X_j = (x_{j1}, x_{j2}, \ldots, x_{jp})$
$d_{ij} = ?$
$X_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$

- **Minkowski distance**

$$d(i,j)=\sqrt[q]{\underbrace{|x_{i1}-x_{j1}|^{q}}_{\text{1st dimension}}+\underbrace{|x_{i2}-x_{j2}|^{q}}_{\text{2nd dimension}}+\ldots+\underbrace{|x_{ip}-x_{jp}|^{q}}_{\text{pth dimension}}}$$

- **Euclidean distance**

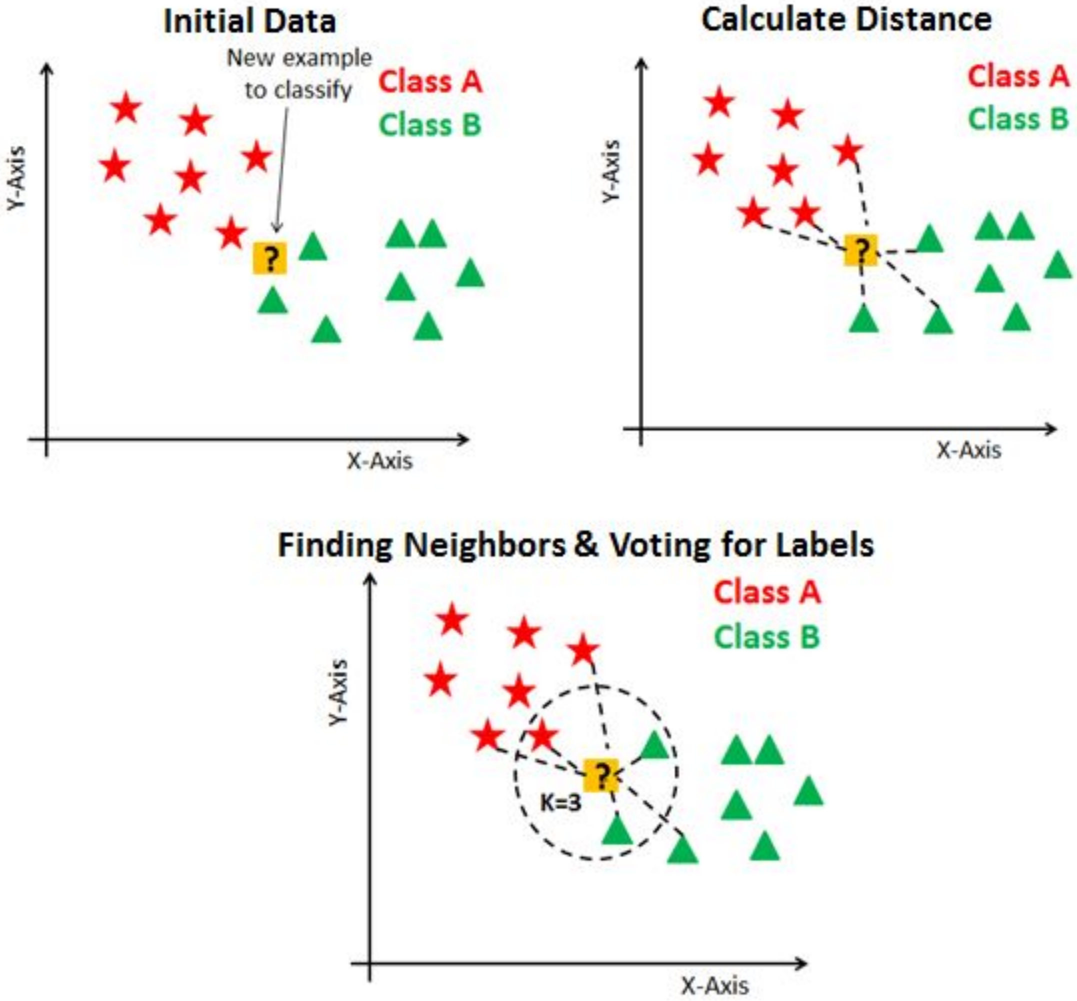
$q = 2$

$$d(i,j)=\sqrt{|x_{i1}-x_{j1}|^{2}+|x_{i2}-x_{j2}|^{2}+\ldots+|x_{ip}-x_{jp}|^{2}}$$

- **Manhattan distance**

$q = 1$

$$d(i,j)=|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+\ldots+|x_{ip}-x_{jp}|$$


Initial Data
New example to classify
Class A
Class B
?
Y-Axis
X-Axis
Calculate Distance
Class A
Class B
?
Y-Axis
X-Axis
Finding Neighbors & Voting for Labels
Class A
Class B
?
K=3
Y-Axis
X-Axis

How to choose a K value?

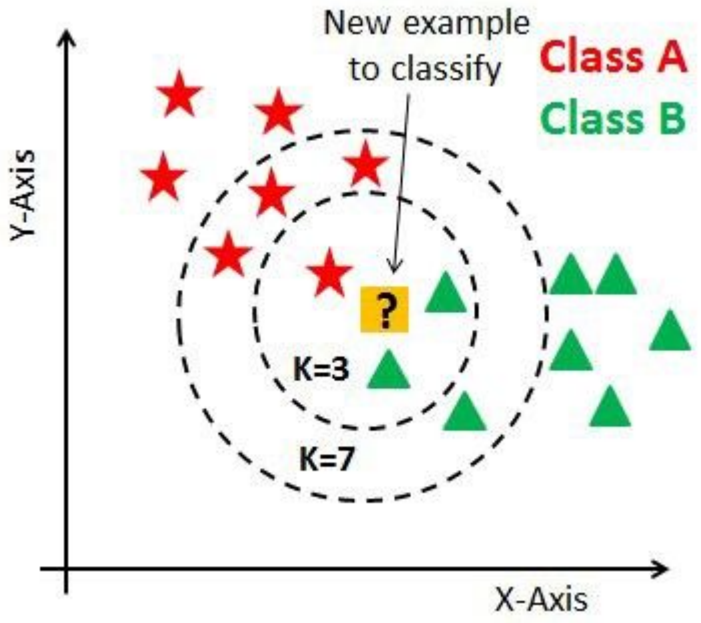
Selecting the optimal K value to achieve the maximum accuracy of the model is always challenging for a data scientist.

K value indicates the count of the nearest neighbors. We have to compute distances between test points and trained labels points.

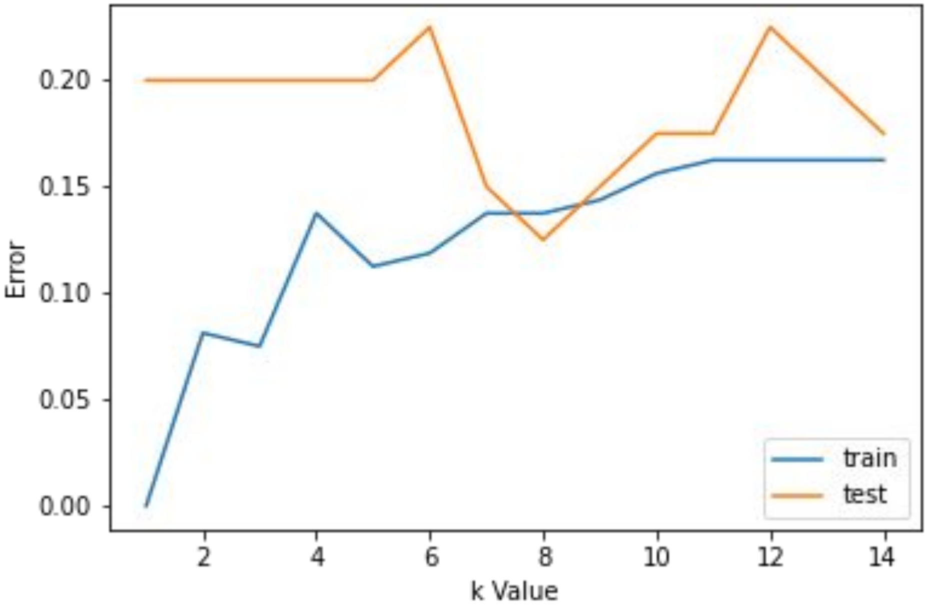
Updating distance metrics with every iteration is computationally expensive, and that’s why KNN is a lazy learning algorithm.

As you can verify from the above image, if we proceed with K=3, then we predict that test input belongs to class B, and if we continue with K=7, then we predict that test input belongs to class A.

That’s how you can imagine that the K value has a powerful effect on KNN performance.



- There are no pre-defined statistical methods to find the most favorable value of K.
- Initialize a random K value and start computing.
- Choosing a small value of K leads to unstable decision boundaries.
- The substantial K value is better for classification as it leads to smoothening the decision boundaries.
- Derive a plot between error rate and K denoting values in a defined range. Then choose the K value as having a minimum error rate.


Error
0.00
0.05
0.10
0.15
0.20
2
4
6
8
10
12
14
k Value
train
test