K Nearest Neighbors (KNN)

What is KNN?

- KNN is a simple
- supervised machine learning (ML) algorithm
- that can be used for classification or regression tasks
- frequently used in missing value imputation.

What is KNN?

- It is based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points. By choosing *K*, the user can select the number of nearby observations to use in the algorithm.

How Does KNN Algorithm Work

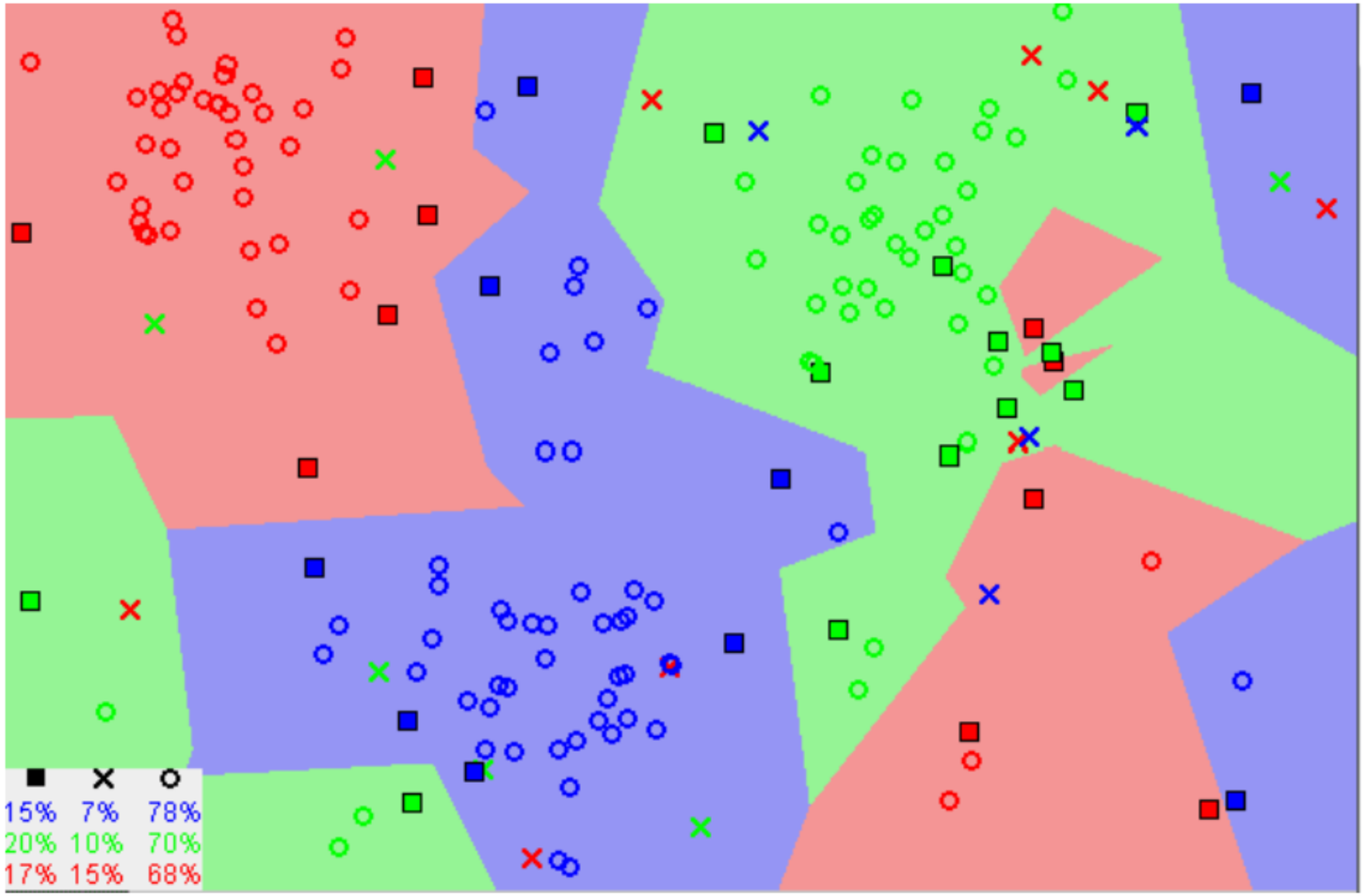
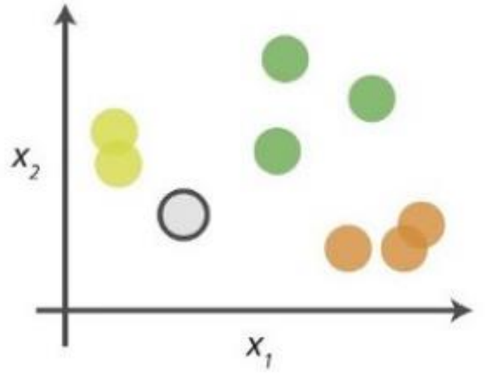


Image showing how similar data points typically exist close to each other

How Does KNN Algorithm Work

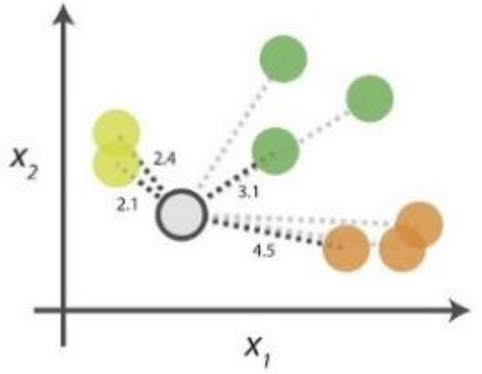
1. Load the data
2. Initialize K to your chosen number of neighbors
3. For each example in the data
   - Calculate the distance between the query example and the current example from the data.
   - Add the distance and the index of the example to an ordered collection
4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances
5. Pick the first K entries from the sorted collection
6. Get the labels of the selected K entries
7. If regression, return the mean of the K labels
8. If classification, return the mode of the K labels

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

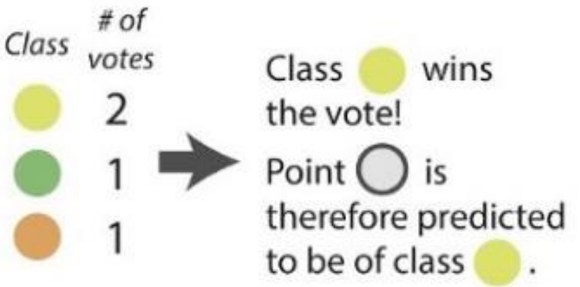


Start by calculating the distances between the grey point and all other points.

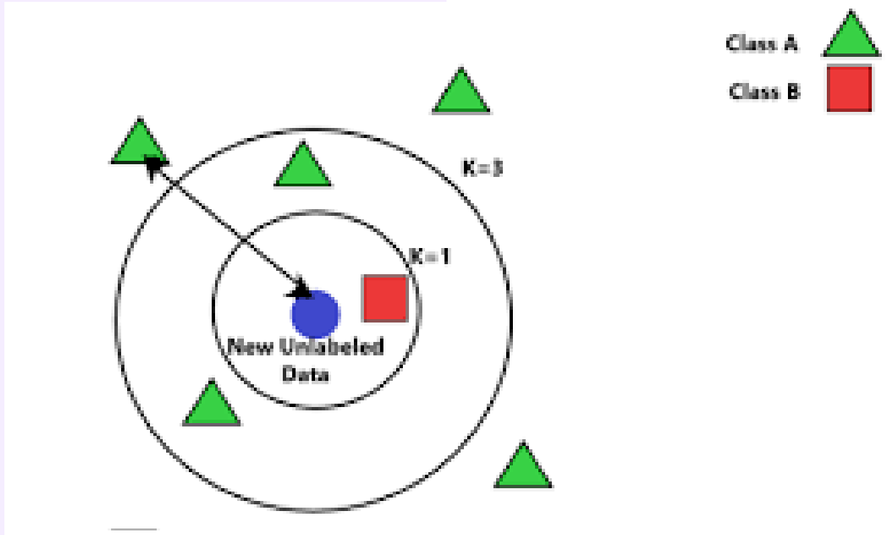
2. Find neighbours

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



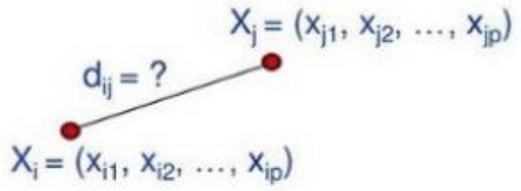
Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.



Choosing the right value for K

- There are no pre-defined statistical methods to find the most favorable value of K.
- Initialize a random K value and start computing.
- Taking a **low k** will increase the influence of noise and the results are going to be less generalizable. On the other hand, taking a **high k** will tend to blur local effects which are exactly what we are looking for.
- It is also recommended to take an **odd k** for binary classes to avoid ties.
- The optimal K value usually found is the **square root of N,** where N is the total number of samples.

How to calculate the distance?



- **Minkowski distance**

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

1st dimension 2nd dimension 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

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

Euclidean distance

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^{n} \left(x_i - y_i\right)^2}$$