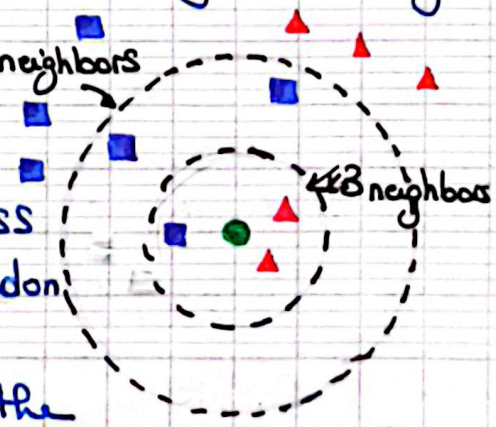
K-Nearest Neighbors (K-NN)

. A simple non-parametric, classification algorithm
. It classifies a new data instances based on the K most similar training examples
. Similarity means that examples that are close (neighbors) are likely to have a similar output label

. The K-NN algorithm assumes that similar things exist in close proximity. In other words, similar things are "close" to each other

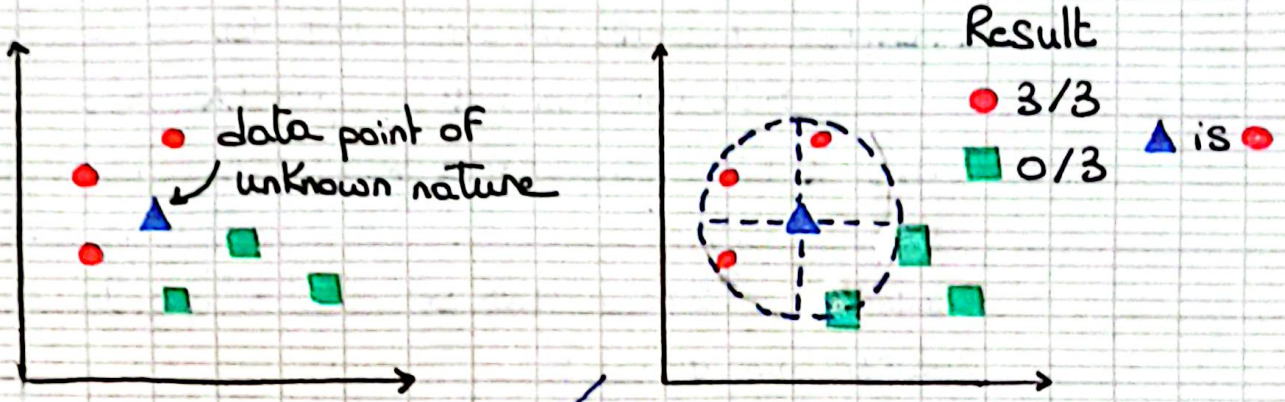
. How does the K-NN capture the similarity between those examples?
It basically captures the idea of similarity (sometimes called distance, proximity, or closeness) using some mathematical metrics - e.g. Calculating the distance between points on a graph

. K-NN does not require training/learning
. Basic Algorithm
- we have initially labeled data points (▲■)
- we need to predict the class of a new data point (●) basedon its K nearest neighbor
- The K nearest neighors are the data points that are enclosed inside of the circle
- The prediction is made by a majority vote



. K-NN . Example

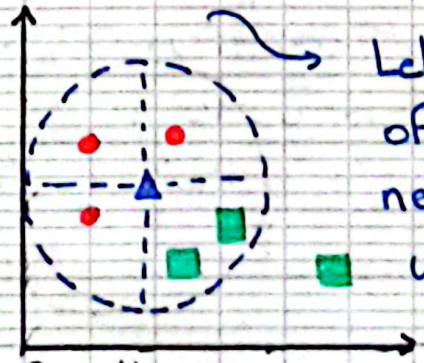
- Below is a spread of red circles (RC) and green squares (GS)
  - → We wish to find the class of the blue triangle (BT)
  - → BT can be either RC or GS



↳ . Assum K=3, which means that we will predict the nature of of BT based on its 3 nearest neighbors.

. We'll make a circle with BT as the center just as big to enclose only 3 data points

. As we can see, the 3 closed points to BT are all RC, hence we can say that based on the majority vote from the neighbors BT should belong to RC class

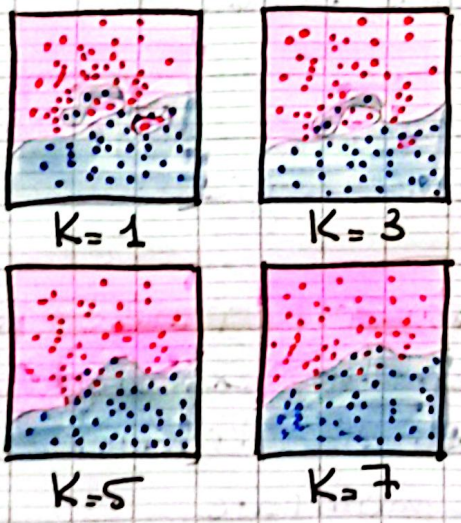


→ Let's say we need to change the value of K and make our decision on the 5 nearest neighbor of BT. In this case our circle will involve 5 data points. And as we can see, among those 5, 3 are RC and 2 are GS. By majority vote we can conclude that BT should be classified as RC

- The choice of the parameter K is very crucial in this algorithm

- What exactly does K influence in the algorithm? In our example, given that all the 6 training observations remain constant, with a given K value we can make boundaries for each class. These boundaries will separate red circles from the green squares.

- Here's a K-NN classification for different increasing values:



- When we take K=1, we have a training error of 0 because each point is closest to itself.

- As we increase the value of K, the boundary between the classes will become more smooth

- This will be better for the model to generalize in the future, but this is up to a certain limit because after this limit, our predictions won't be accurate at all. And it's at this point that we know that we have pushed the value of K too far