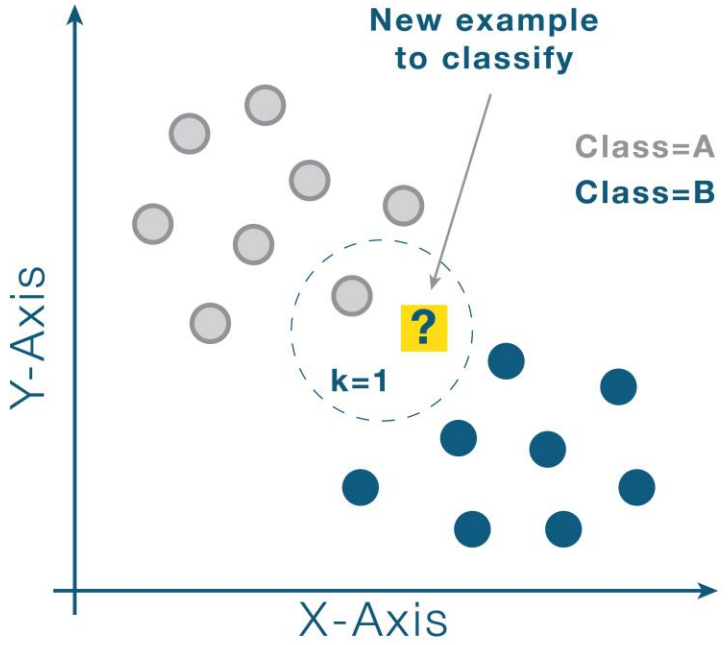
CSC 462: Machine Learning

3.5 k-Nearest Neighbors

Dr. Sultan Alfarhood

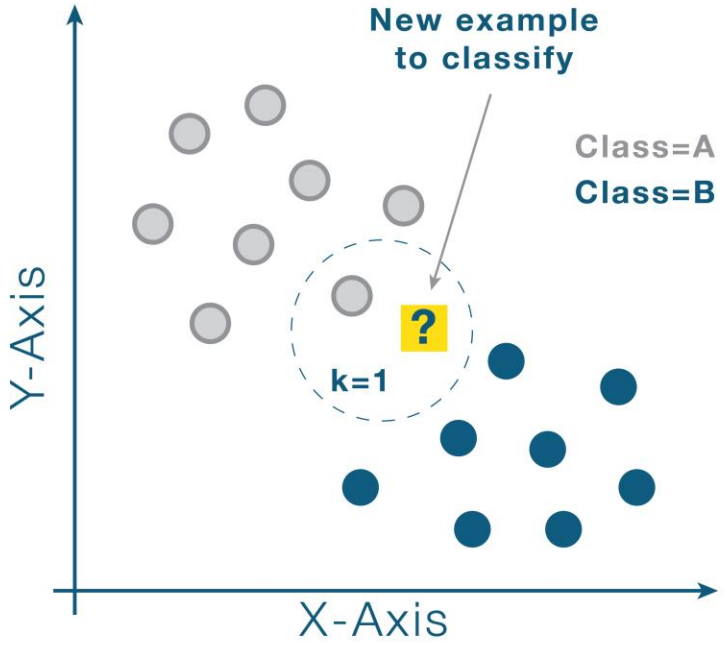
k-Nearest Neighbors (kNN)

- In kNN, k is the number of nearest neighbors
  - The number of neighbors is the key factor
- Its simplest case is when k=1 (called the nearest neighbor algorithm)



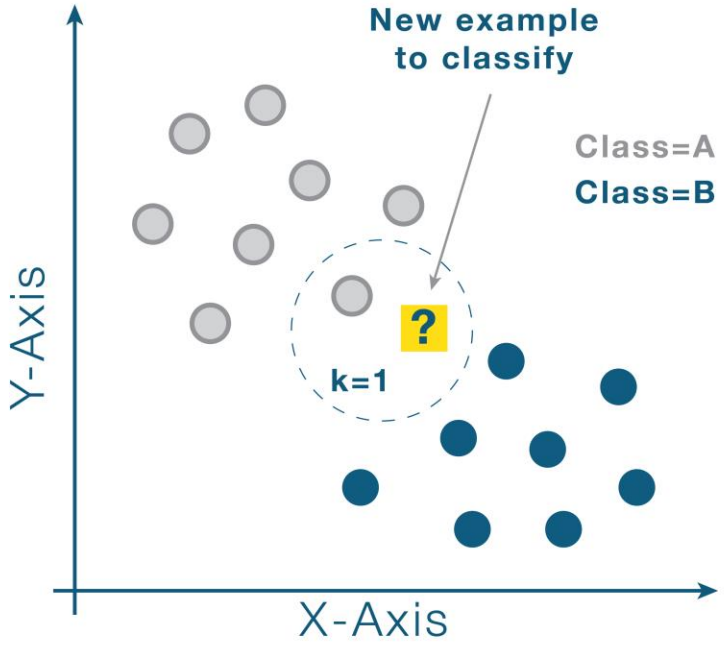
k-Nearest Neighbors (kNN)

- k-Nearest Neighbors (kNN) is a non-parametric learning algorithm
  - Contrary to other learning algorithms that allow discarding the training data after the model is built, **kNN keeps all training examples in memory**
- kNN can be used for both classification and regression

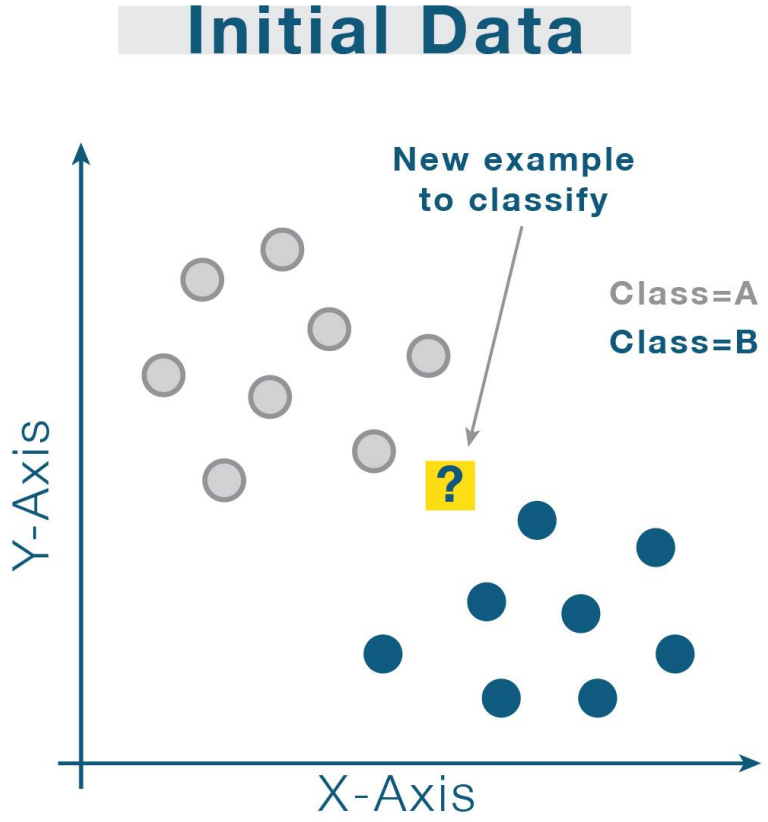


k-Nearest Neighbors (kNN)

- A new point called X for which the label must be predicted
  - The algorithm **finds the k nearest** points to X
  - It then classifies X by **majority vote** of its k neighbors, or calculates the average target (in case of regression)
- kNN has the following basic steps:
  1. Calculate the distance
  2. Find the k nearest neighbors
  3. Vote for the labels
     - Or calculates the average target in case of regression

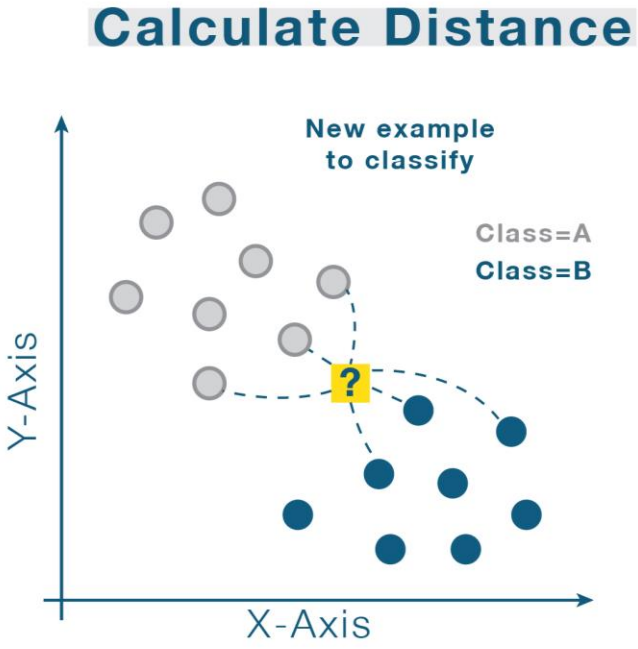


k-Nearest Neighbors (kNN)



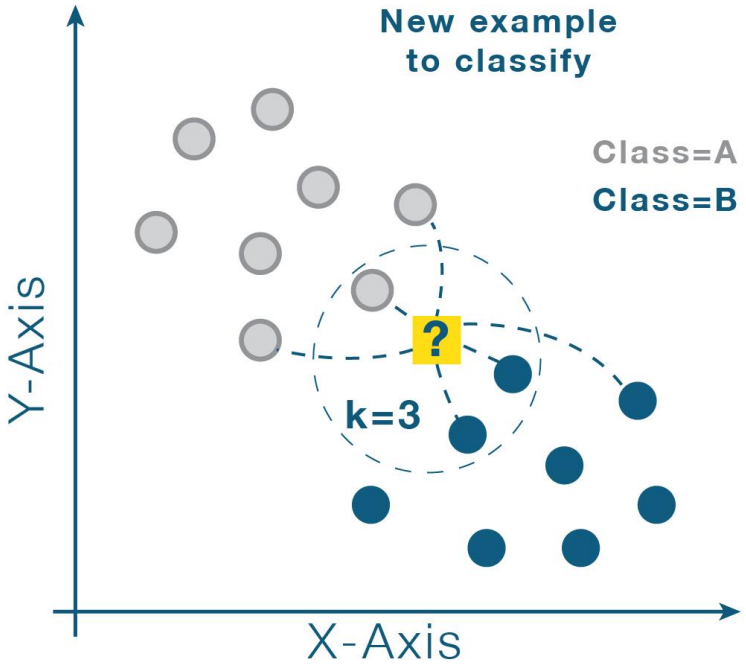
k-Nearest Neighbors (kNN)

- Once a new, previously unseen example x comes in the kNN algorithm finds k training examples closest to x
- **Euclidean distance** is frequently used in practice
  - $d(x_i, x_k) = \sqrt{\sum_{j=1}^{D} \left(x_i^{(j)} - x_k^{(j)}\right)^2}$
- Another popular choice of the distance function is the negative **cosine similarity**
  - $s(x_i, x_k) = \frac{\sum_{j=1}^{D} x_i^{(j)} x_k^{(j)}}{\sqrt{\sum_{j=1}^{D} \left(x_i^{(j)}\right)^2} \sqrt{\sum_{j=1}^{D} \left(x_k^{(j)}\right)^2}}$

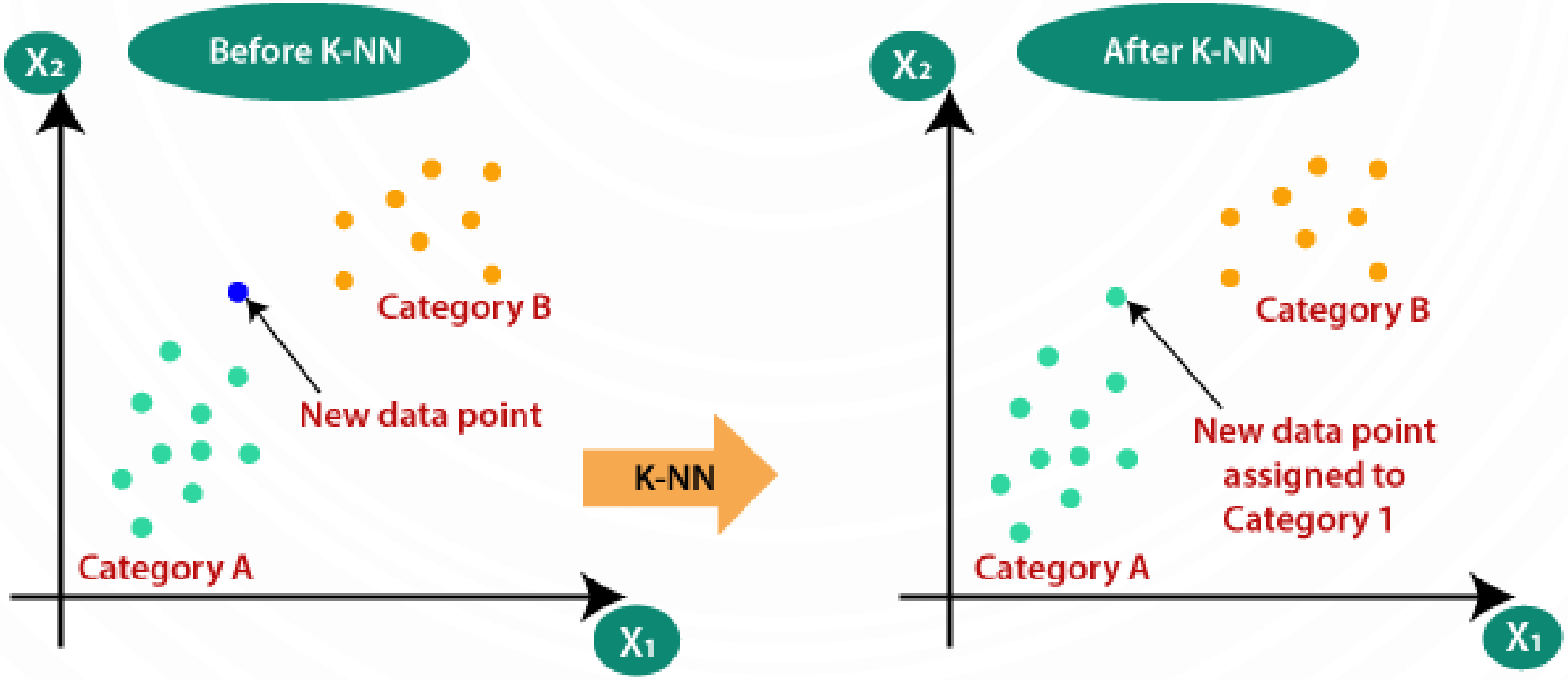


k-Nearest Neighbors (kNN)

- Once a new, previously unseen example x comes in the kNN algorithm finds k training examples closest to x and returns the majority label (in case of classification) or the average label (in case of regression)

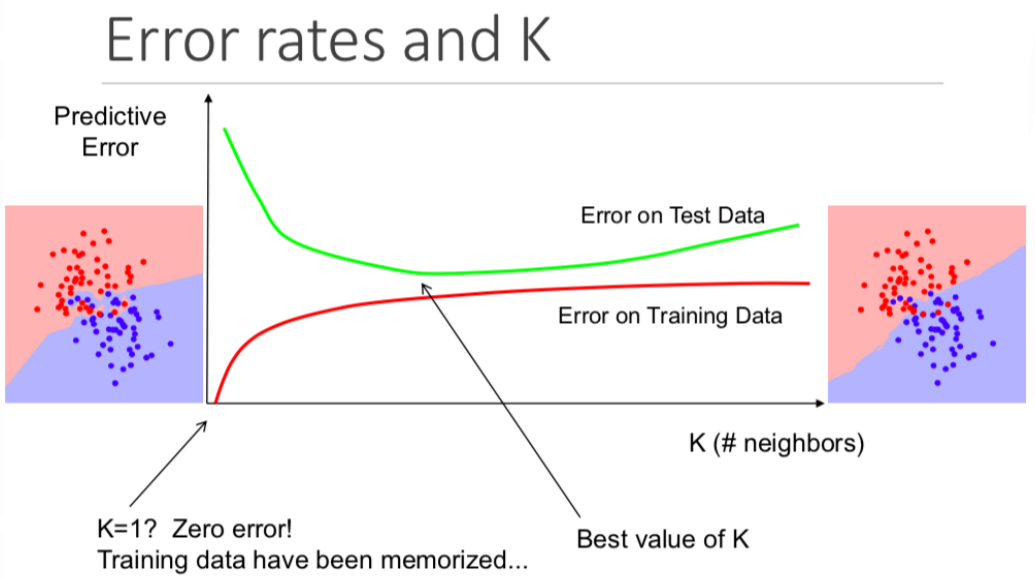


kNN Example


Before K-NN
X2
Category B
New data point
Category A
X1
K-NN
After K-NN
X2
Category B
New data point
assigned to
Category 1
Category A
X1

Choosing the right value for K

- Try the kNN algorithm several times with different values of K and choose the K that reduces the number of errors while maintaining the algorithm's ability to accurately make predictions
  - As the value of K decreases to 1, predictions become less stable
  - As the value of K increases, predictions become more stable due to majority voting / averaging. Eventually, we begin to witness an increasing number of errors. It is at this point we know we have pushed the value of K too far
  - In cases where we are taking a majority vote (e.g. picking the mode in a classification problem) among labels, we usually make K an odd number to have a tiebreaker.



k-Nearest Neighbors (kNN)

- KNN is an algorithm that is considered both **non-parametric** and an example of **lazy learning**.
  - **Non-parametric** means that it makes no assumptions. The model is made up entirely from the data given to it rather than assuming its structure is normal.
  - **Lazy learning** means that the algorithm makes no generalizations. This means that there is little training involved when using this method. Because of this, all of the training data is also used in testing when using kNN.
    - Also called instance-based learner since it uses all instances in the dataset as the model.

kNN Advantages & Disadvantages

- **Advantages**
  - The algorithm is simple and easy to implement
  - It is robust to the noisy training data
  - The algorithm is versatile (can be used for classification and regression)
- **Disadvantages**
  - Must find an optimal k value (number of nearest neighbors)
  - The algorithm gets significantly slower as the number of examples increase
    - The computation cost is high because of calculating the distance between the data points for all the training samples
  - Accuracy depends on the quality of the data