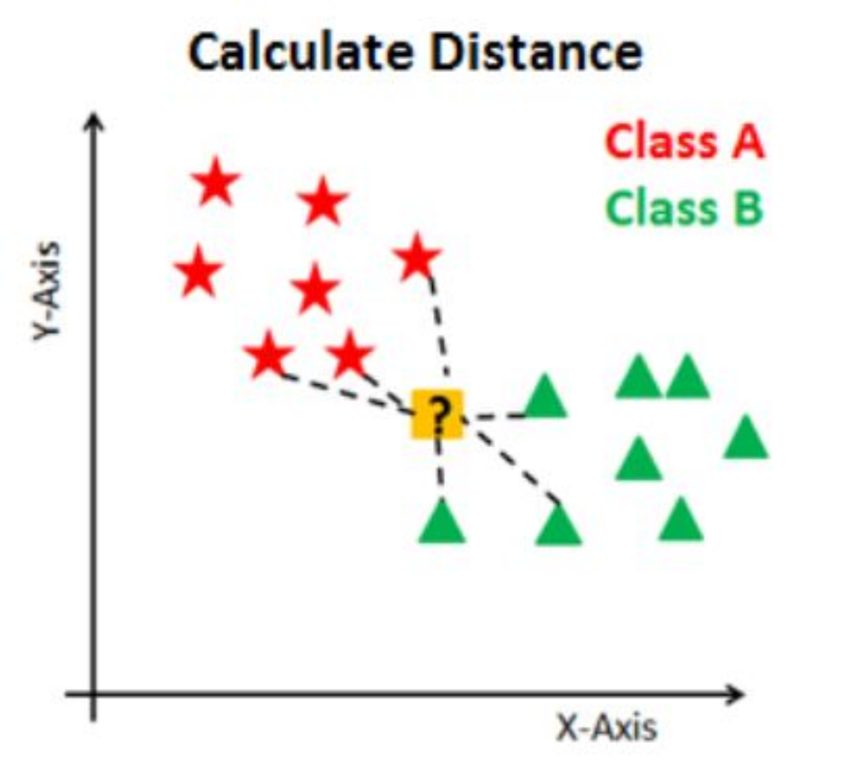
K- Nearest Neighbors (K-NN)

Group 5
Stephen, Meghan, Sarvani, Albert

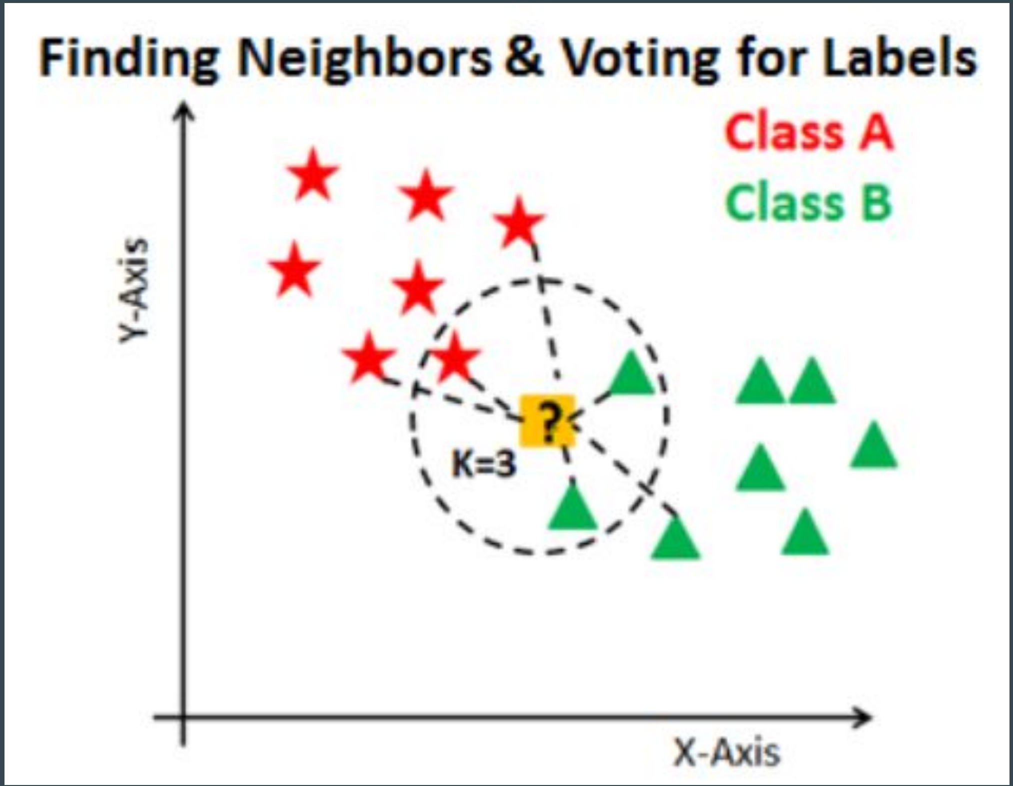
How Does The Algorithm Work?

- Makes the assumption that similar things exist in close proximity
- Uses **feature similarity** to predict the values of new data points
- **Non-parametric**: it means that it does not make any assumption of the data distribution.
- KNN can be used for regression and classification. However, it is mainly used for classification.

Step 1


Calculate Distance
Class A
Class B
Y-Axis
?
X-Axis

Step 2


Finding Neighbors & Voting for Labels
Class A
Class B
Y-Axis
?
K=3
X-Axis

Advantages/Disadvantages of The Algorithm:

Advantages

- Simple and Effective
- Robust to data that contains a lot of noise (where decision boundary is irregular)
- Can perform both classification and regression
- As $n \rightarrow \infty$, k-NN becomes increasingly accurate
- Unbiased in nature: makes no prior assumption of the underlying data

Disadvantages

- Curse of dimensionality, doesn't work well when there are many variables
- Costly computation
- Cannot Handle Missing Value without Imputation
- As $n \rightarrow \infty$, k-NN becomes increasingly slow
- As dimensionality increases, points tend to never be close together breaking down the assumptions of k-NN.

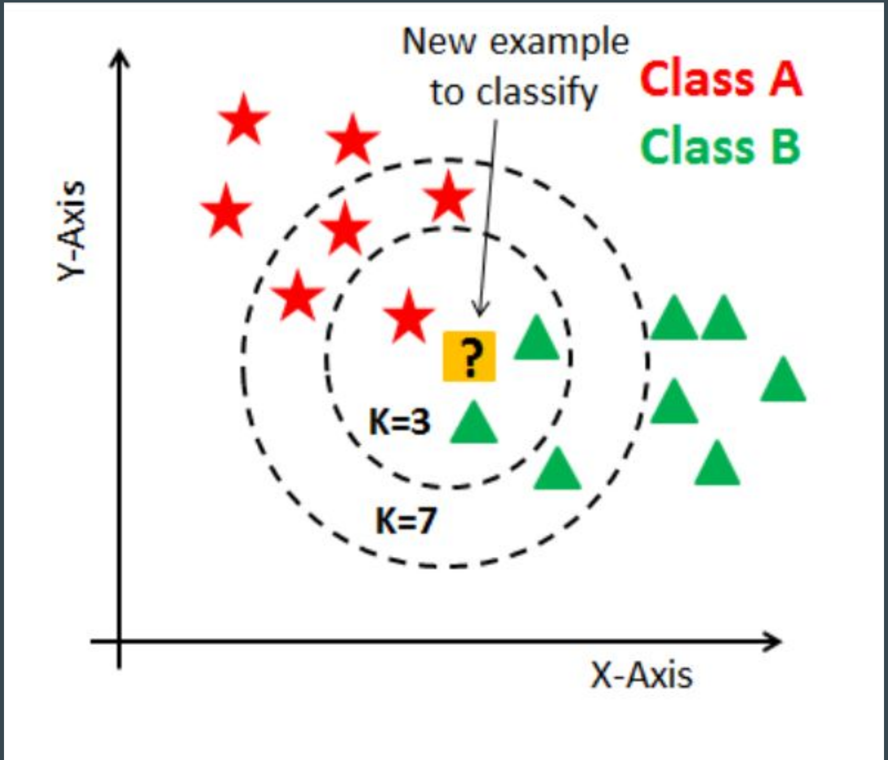
Data Processing Steps of The Algorithm:

- To deal with the problem of the curse of dimensionality, must perform principal component analysis or use feature selection
- Scaling/Normalizing Variables (very important since this is distance based algorithm. Otherwise we will bias towards variables with larger values).

What Hyperparameters Can Be Tuned? What Does Each One Represent?

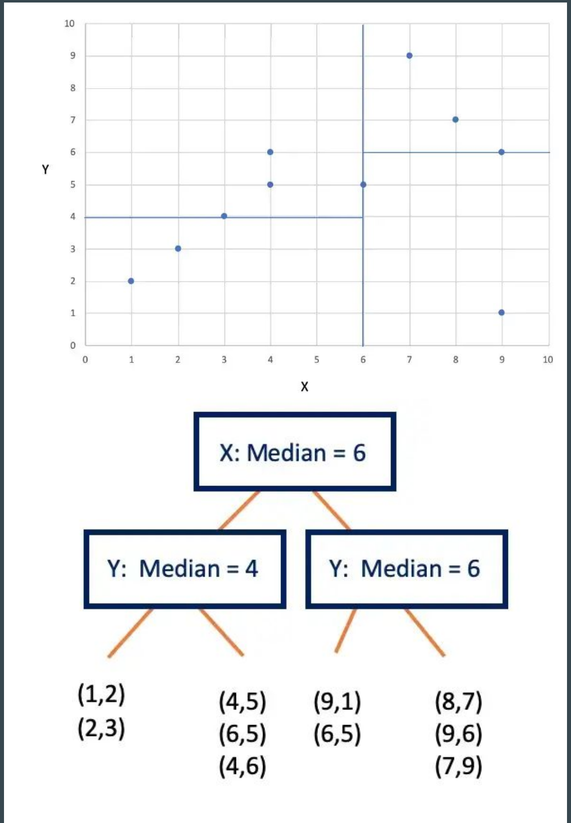
n_neighbors = int

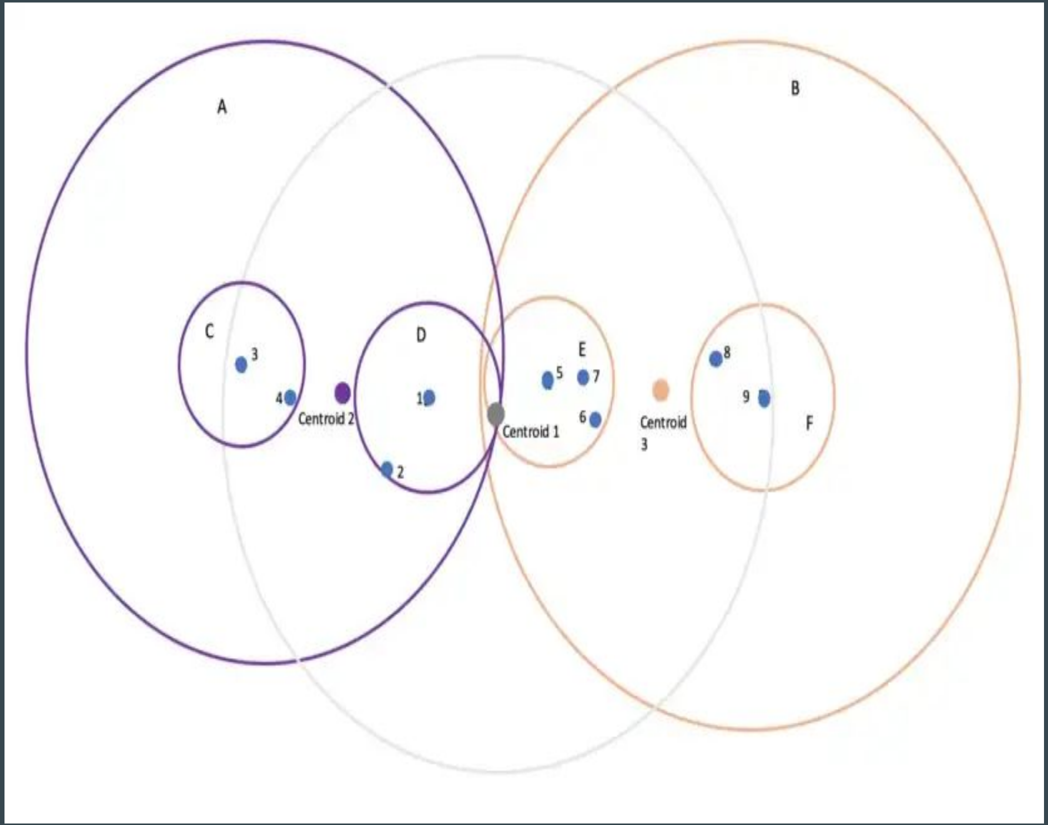
- This parameter sets how large k is.
- Determines how many neighbors are included in the vote
- “k” value is easier to measure when it is an positive odd integer.



- algorithm = {auto, kd_tree, ball_tree, brute}
    - Auto: attempts to find the ‘best’ algorithm based on the values passed in
    - Kd_tree: a binary search tree where data in each node represents a K-dimensional point in space.
    - Ball_tree: each ball represents a cluster around a centroid containing all data. The data is further clustered into increasingly distant clusters from the centroid.
    - Brute: calculate the distance from the new point to every point in the training data, then take the specified K nearest hyperparameter and perform a vote.

KD and Ball Tree Diagrams





weights = uniform, distance

- ‘uniform’ : uniform weights. All distances of ‘nearest neighbors’ are weighted equally.
- ‘distance’ : weight points by the inverse of their distance. In this case, closer neighbors of a test point will have a greater influence than neighbors which are further away.

Minkowski Distance

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}.$$

P = 1 → Manhattan Distance

P = 2 → Euclidean Distance

P = ∞ → Chebyshev Distance (Maximum Distance)

Manhattan distance is usually preferred over the more common Euclidean distance when there is high dimensionality in the data.

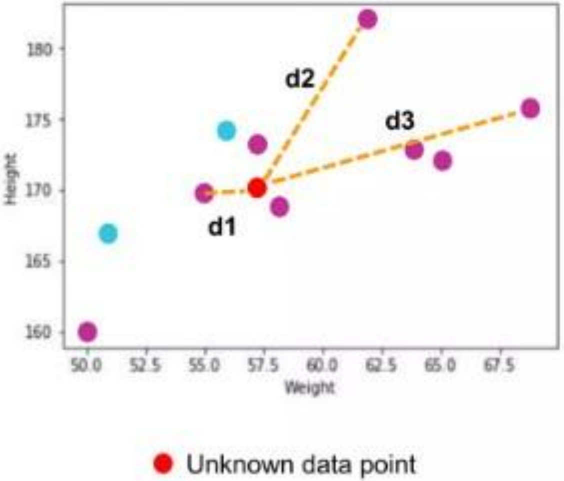
p = 1 → Manhattan Distance

$$\text{dist}(d) = |x - a| + |y - b|$$

Manhattan Distance is longer than Euclidean, and can have more than one path

$p = 2 \longrightarrow$ Euclidean Distance

$$\text{dist}(d) = \sqrt{(x - a)^2 + (y - b)^2}$$



$\text{dist}(\mathbf{d1}) = \sqrt{(170-167)^2 + (57-51)^2} \sim= 6.7$

$\text{dist}(\mathbf{d2}) = \sqrt{(170-182)^2 + (57-62)^2} \sim= 13$

$\text{dist}(\mathbf{d3}) = \sqrt{(170-176)^2 + (57-69)^2} \sim= 13.4$

Similarly, we will calculate Euclidean distance of unknown data point from all the points in the dataset

Work Cited

https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/

https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

What is the k-nearest neighbors algorithm? | IBM

https://towardsdatascience.com/tree-algorithms-explained-ball-tree-algorithm-vs-kd-tree-vs-brute-force-9746debcd940