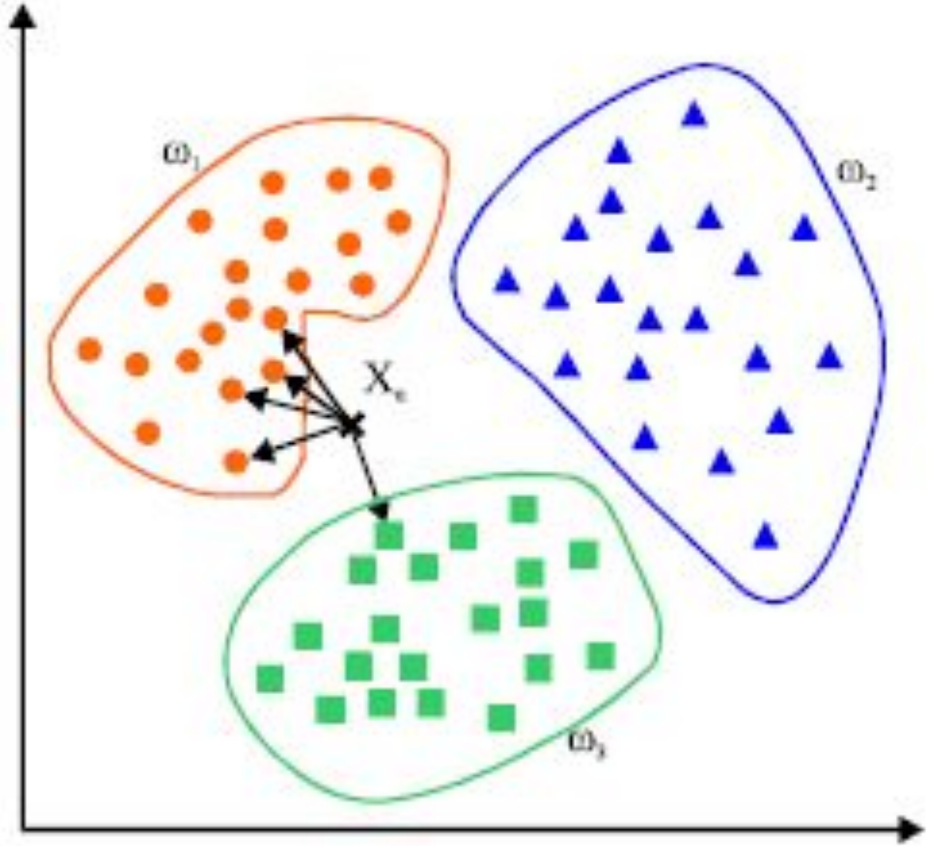
# K Nearest Neighbour

Byte Academy

# The Big Idea



# KNearest Neighbor

KNN is a typical example of a lazy learner. It is called lazy because it doesn't learn a discriminative function from the training data but memorizes the training dataset instead.

# Steps in KNN

1. Choose the number of k and a distance metric.
2. Find the k nearest neighbors of the sample that we want to classify.
3. Assign the class label by majority vote.

# Distance Metric

Euclidean p=1/2

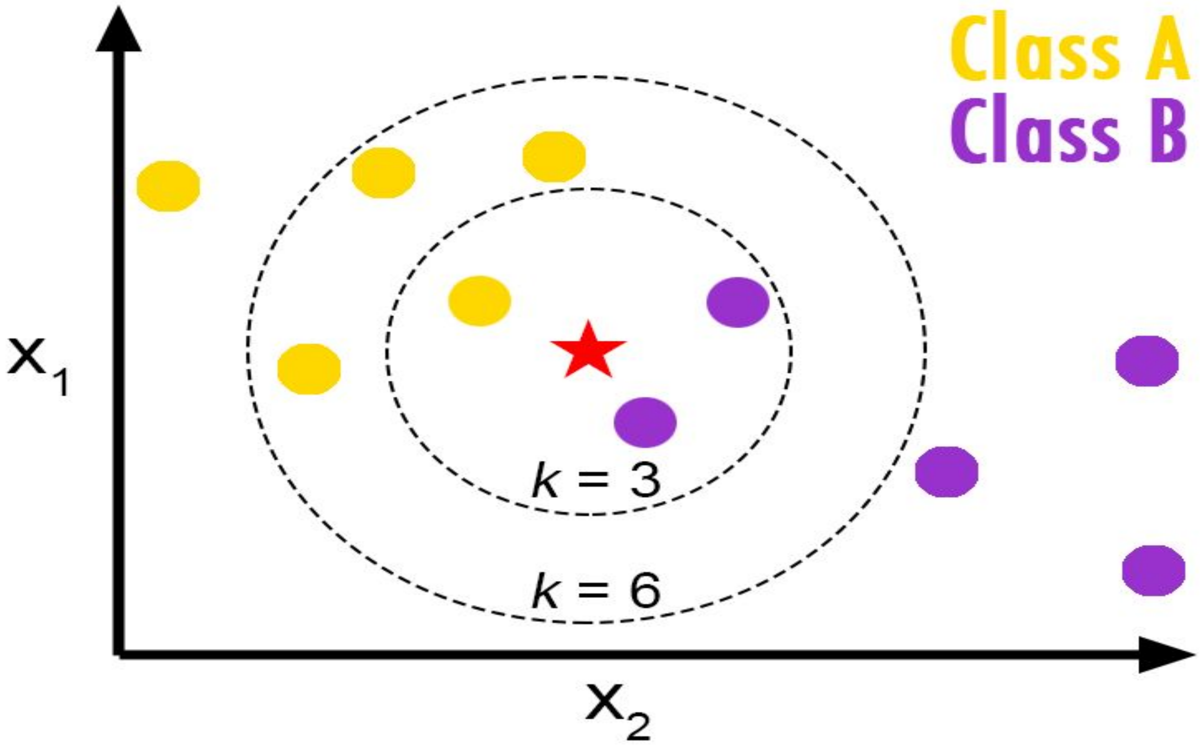
Manhattan P=1

$$D(\mathbf{x}_i, \mathbf{x}_j) = \left( \sum_{l=1}^{d} \left| x_{il} - x_{jl} \right|^{1/p} \right)^{p}.$$

# Flow of KNN

1) Calculate euclidean distance between a test point and all other training points. E.g X1 = {3,3} ,X2 = {6,1}, X3={7,9} ,X4={3,6}
2) Data points sorted by distance (lowest first)
3) 
4) In a classification, we do ***majority voting***, the most frequent class between the K closest points is the class assigned to our test data point. This works in both binary and multiclass contexts
5) In a regression setting, we use the average Y value of K nearest training points. So y_pred_i = 1/K $\sum y_{training\_i}$.

# KNN: Classification



# Let's look at the pros and cons

Pros:

- Very quick and efficient algo (**low training time)**
- Surprisingly effective and easy to interpret

Cons:

- Each feature has the same weight, relative importance of features not learned.
- Needs to be scaled to work well

| ID | Height | Age | Weight |
|---|---|---|---|
| 1 | 5 | 45 | 77 |
| 2 | 5.11 | 26 | 47 |
| 3 | 5.6 | 30 | 55 |
| 4 | 5.9 | 34 | 59 |
| 5 | 4.8 | 40 | 72 |
| 6 | 5.8 | 36 | 60 |
| 7 | 5.3 | 19 | 40 |
| 8 | 5.8 | 28 | 60 |
| 9 | 5.5 | 23 | 45 |
| 10 | 5.6 | 32 | 58 |
| 11 | 5.5 | 38 | ? |

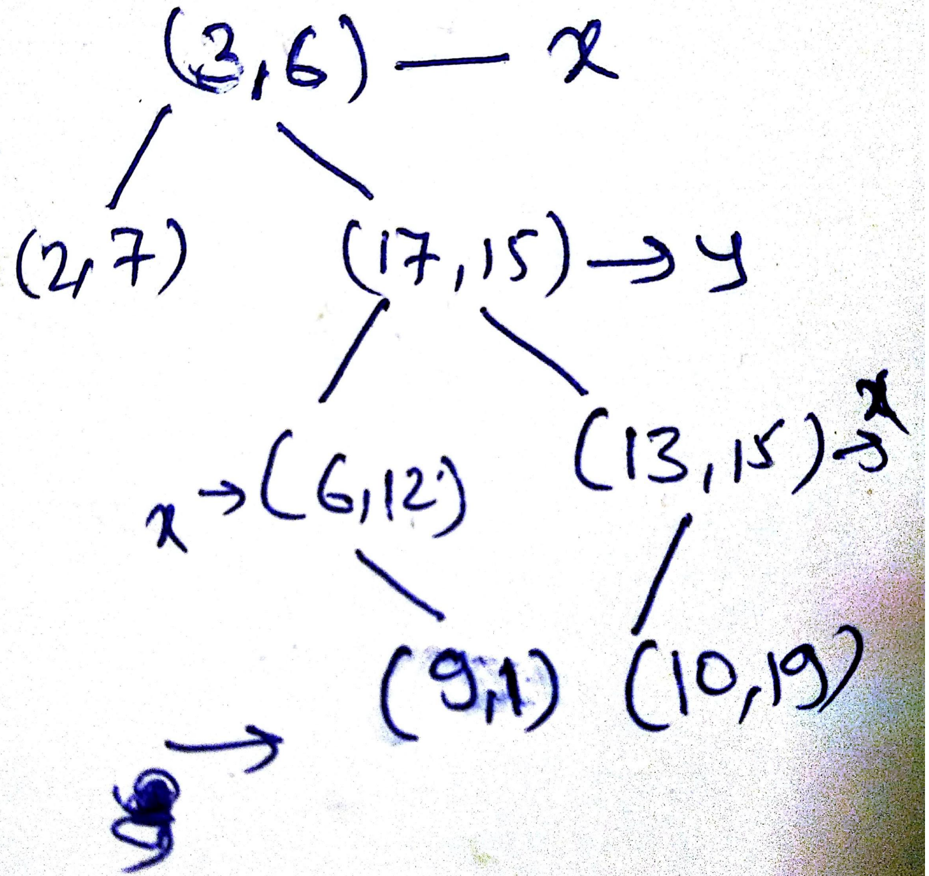
# KNN for data retrieval

- KNN used for data retrieval.
- KNN  search has O(N) time complexity for each query where N= Number of data points. For KNN with K neighbor search, the time complexity will be O(N*K) .just imagine what happen millions of datapoint and thousand of query.
- Any efficient solution? KD Tree
- **KD Tree** is one such algorithm which uses a mixture of Decision trees and KNN to calculate the nearest neighbour(s).

# KD Tree: K-dimensional tree

Binary x,y… compare

(3, 6), (17, 15), (13, 15), (6, 12), (9, 1), (2, 7), (10, 19)


(3,6) — x
(2,7)
(17,15) → y
x → (6,12)
(13,15) → x
(9,1)
(10,19)

# Assignment

Implement euclidean and manhattan distance  function

Calculate k value for knn