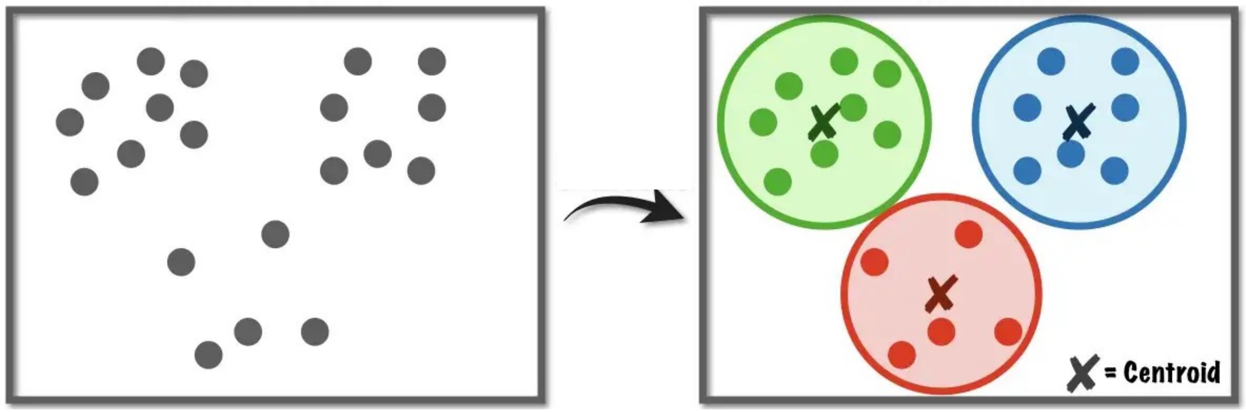
Clustering: K-Means & Nearest Neighbor

Foundation of Data Analysis

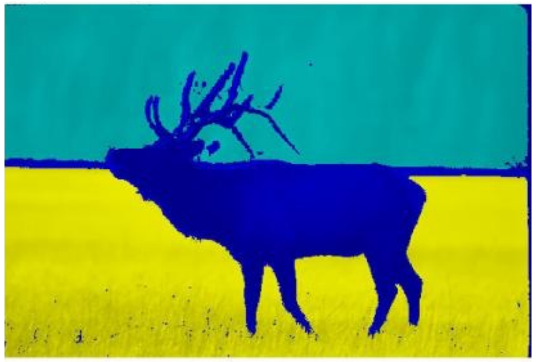
February 08, 2023

Clustering Example



Divide data into different groups

Original image

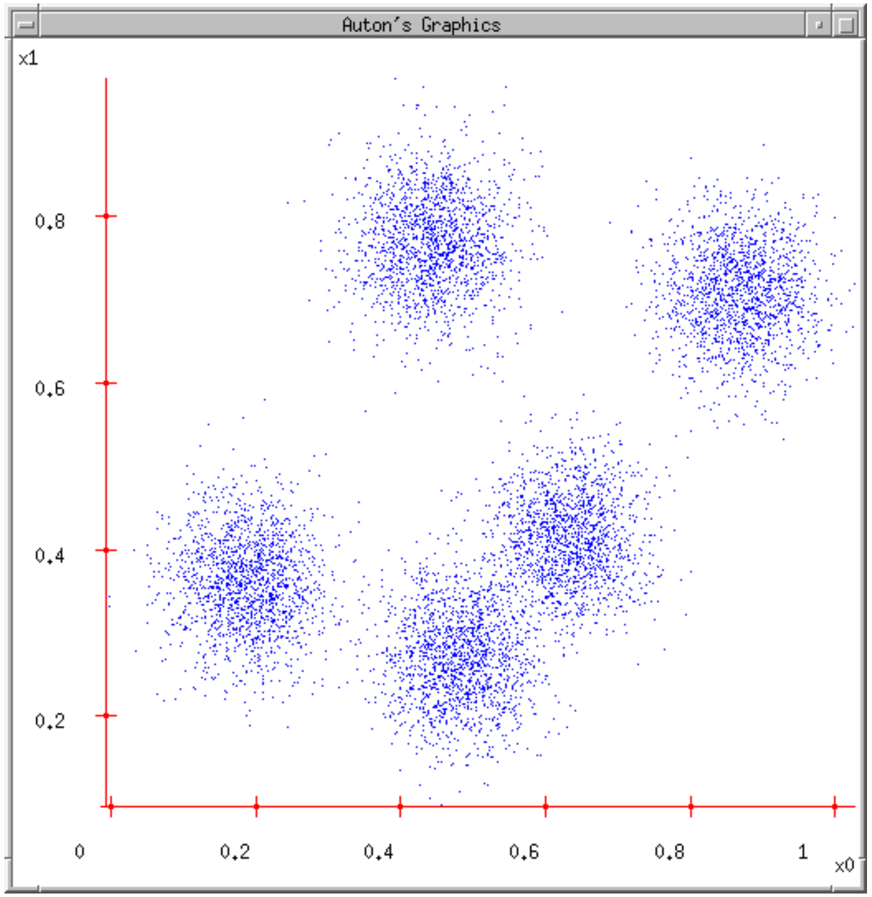

Segmented image

Types of Learning Algorithms

- **Supervised (inductive) learning**
  - Training data (observed examples) includes desired outputs
- **Unsupervised learning**
  - Training data does not include desired outputs
- **Semi-supervised learning**
  - Training data includes a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

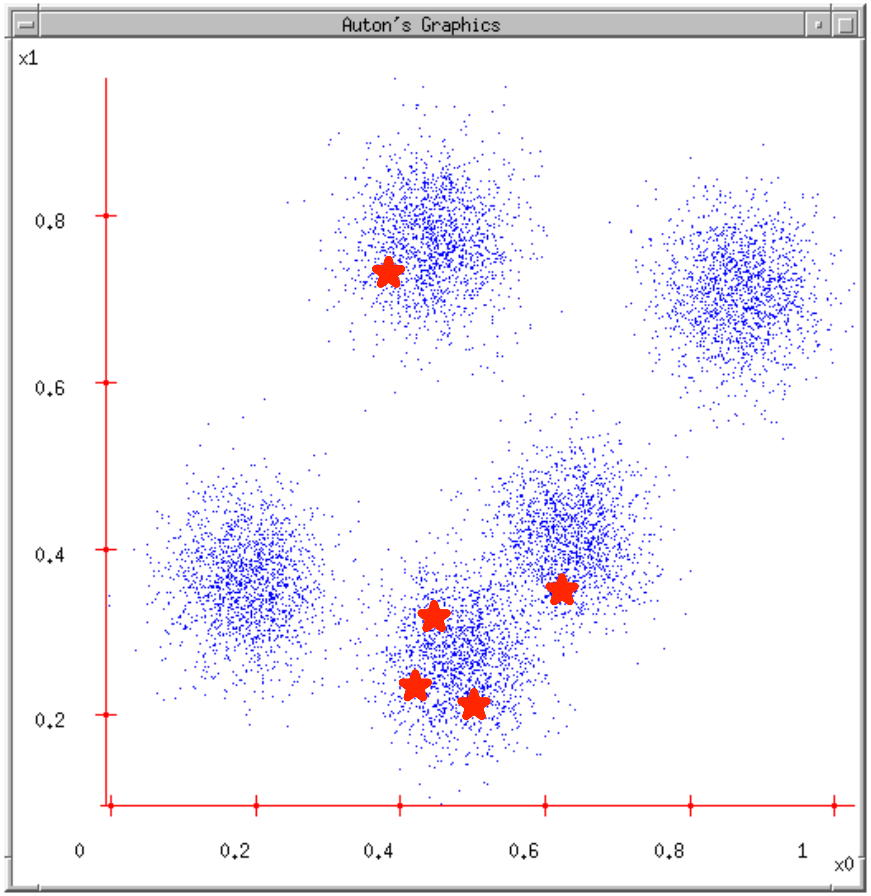
Clustering I: K-means

1. Ask user how many clusters they'd like (e.g. k=5)



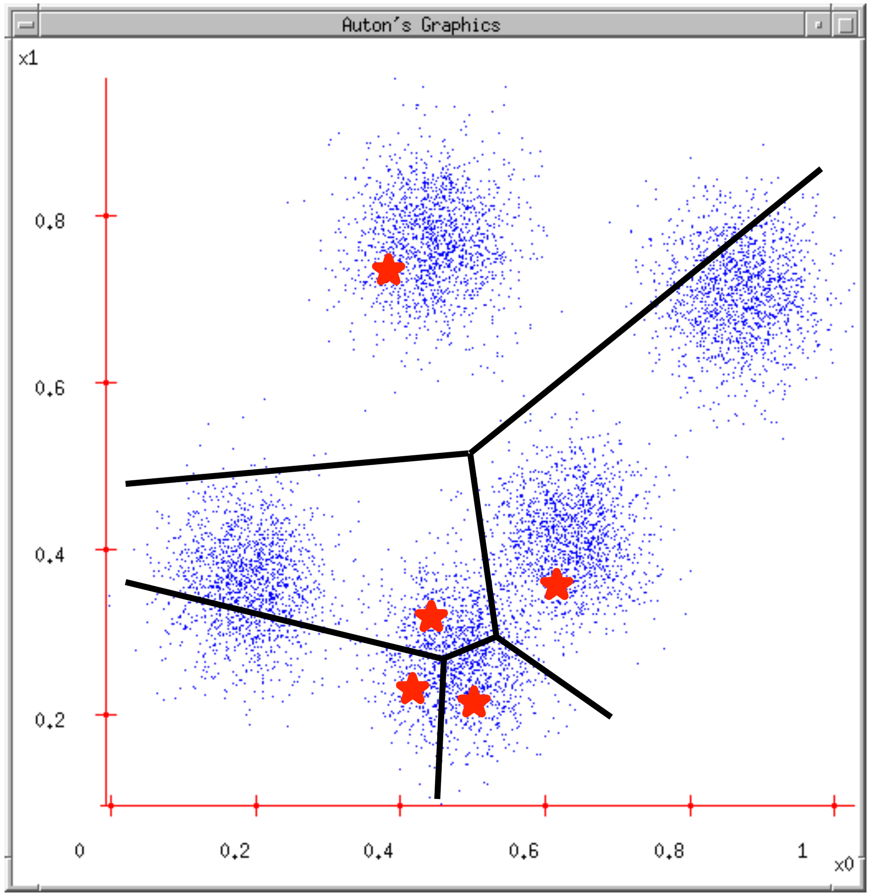
Clustering I: K-means

1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations



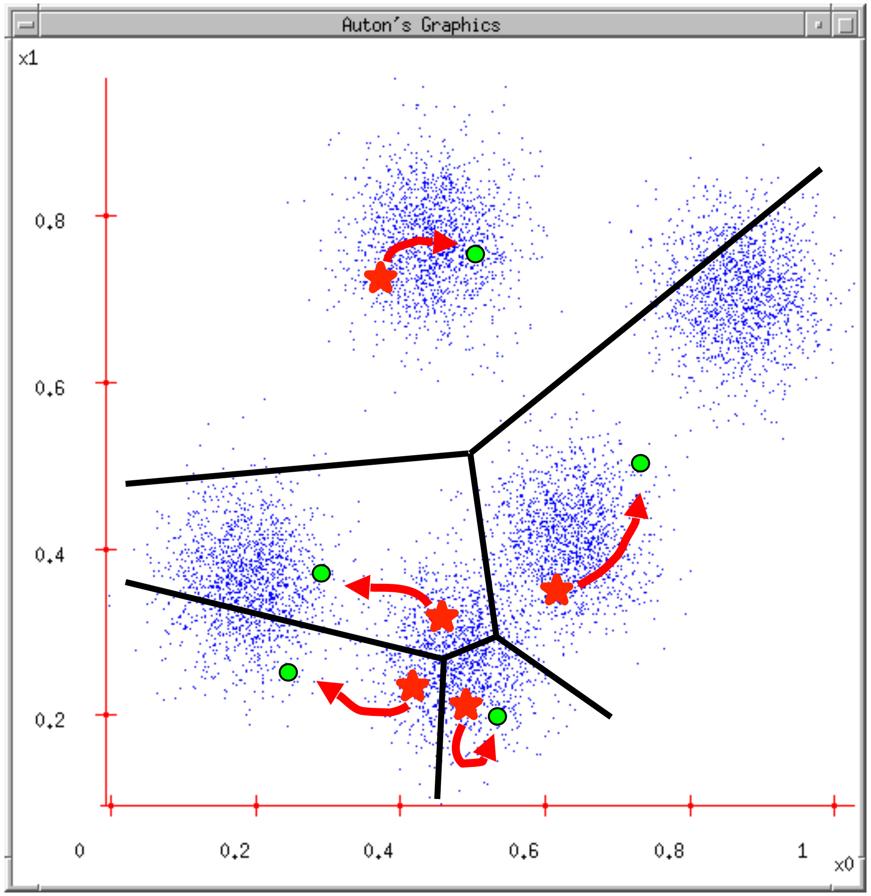
Clustering I: K-means

1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.



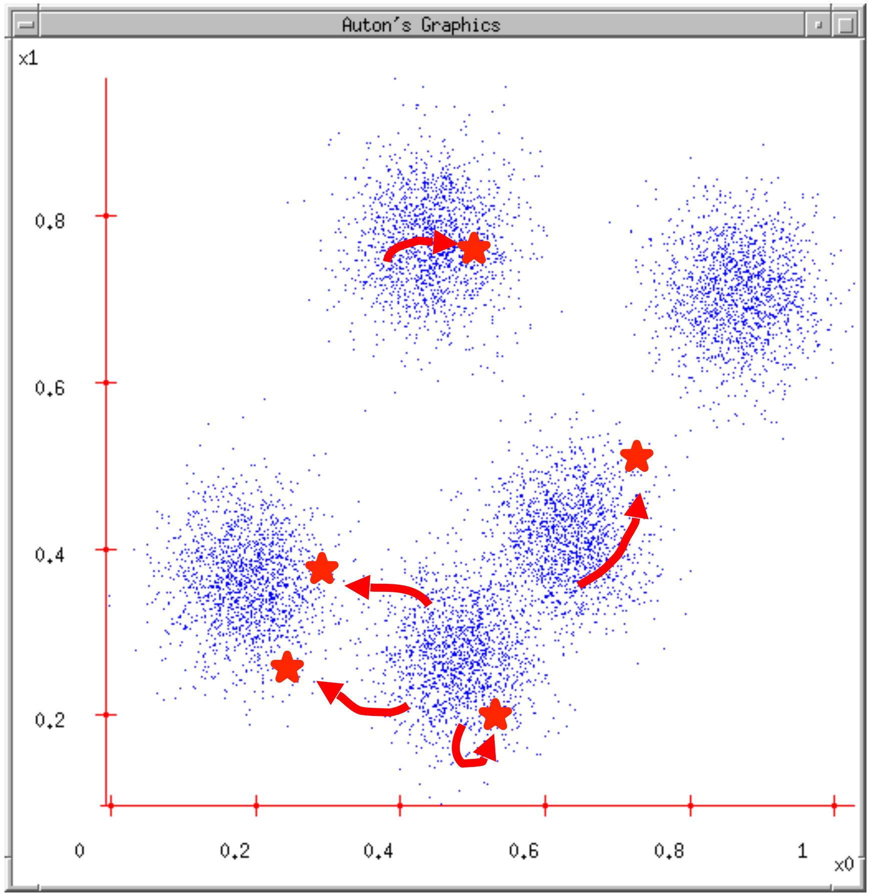
Clustering I: K-means

1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…



Clustering I: K-means

1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



Disadvantages of K-means

- Does not work efficiently with complex geometrical shaped data (mostly non-linear)

- Hard assignment for labels might lead to misgrouping

Disadvantages of K-means

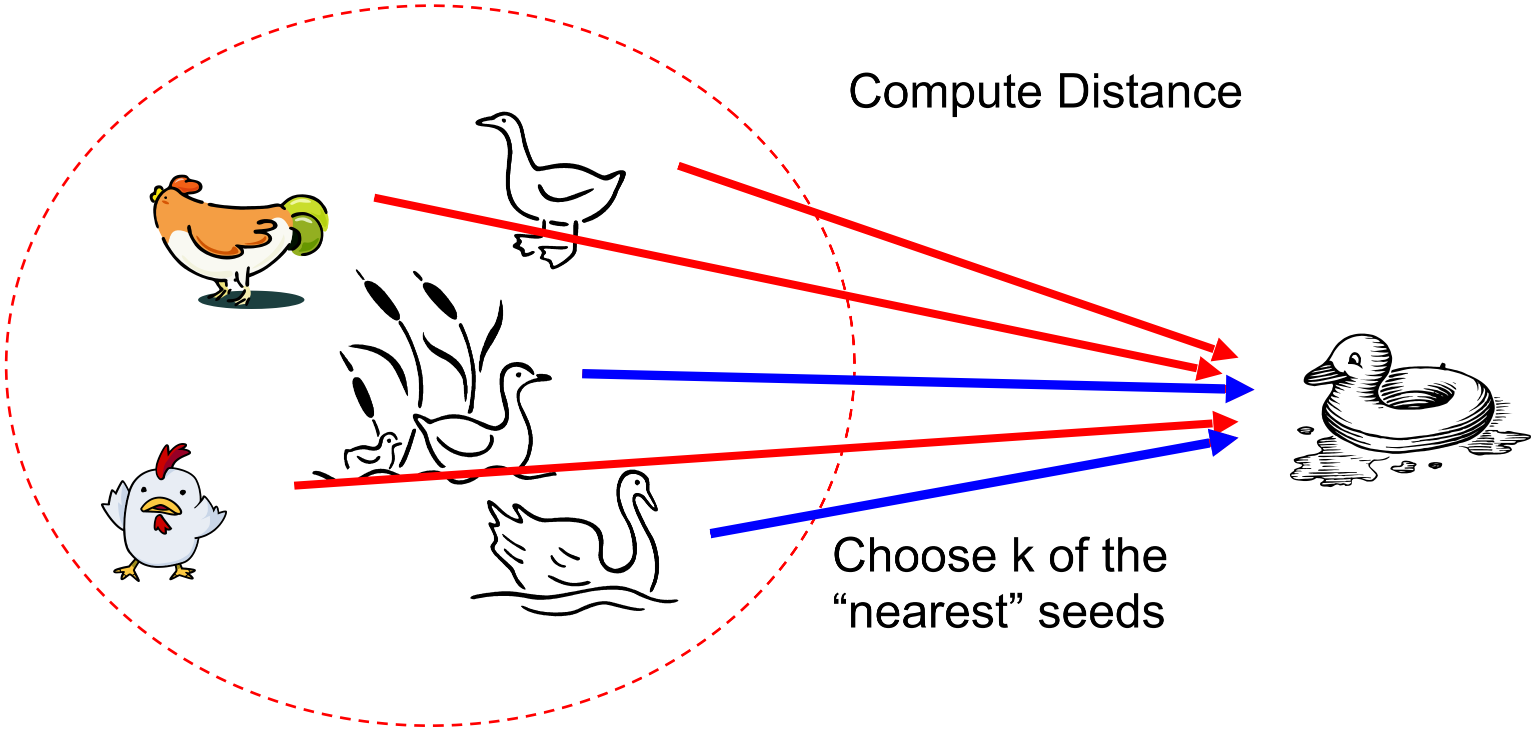
- Does not work efficiently with complex geometrical shaped data (mostly non-linear)
- Hard assignment for labels might lead to misgrouping

Clustering II: Nearest Neighbors

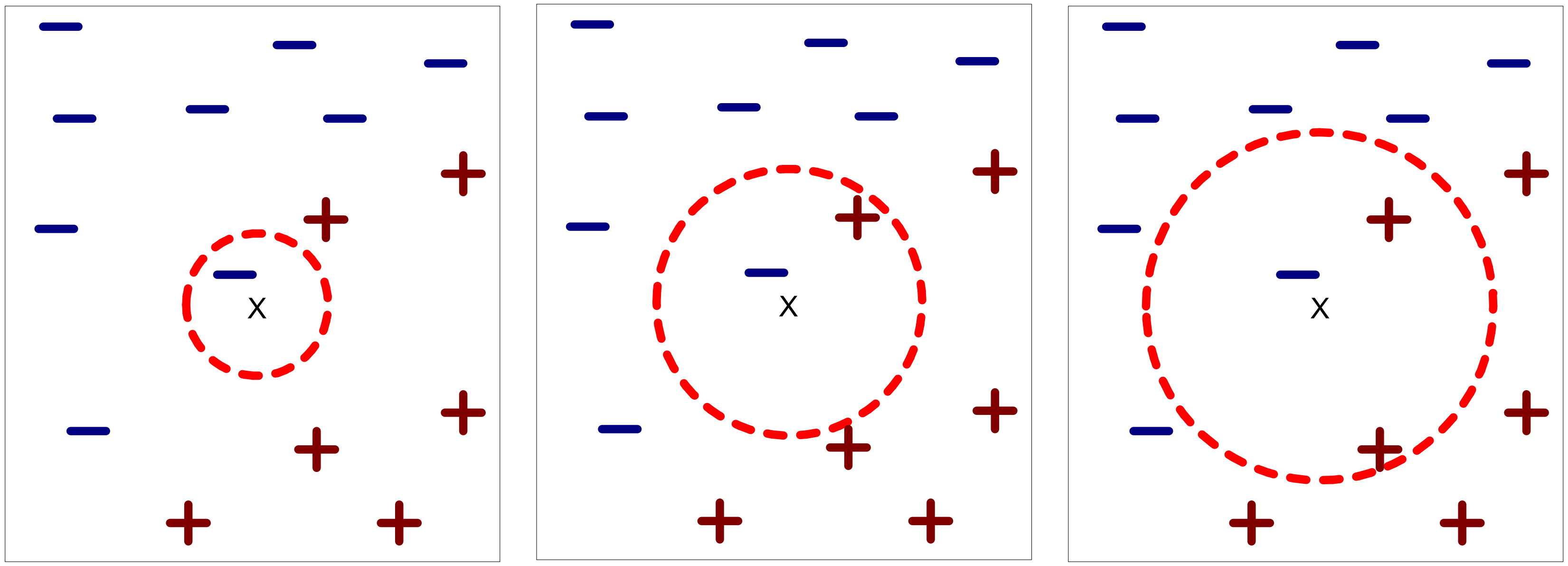
- Can be either supervised Learning or unsupervised learning

Nearest Neighbors: Unsupervised Learning

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



Nearest Neighbors

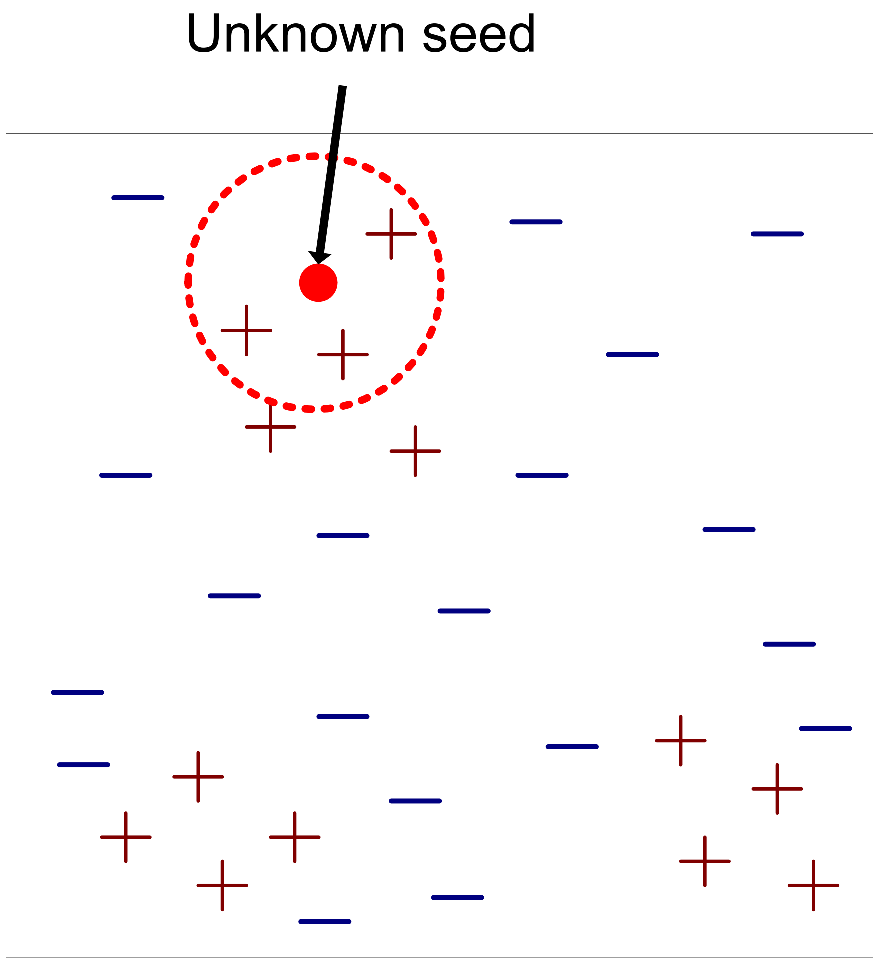


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

- K-nearest neighbors of seed x: data points that have the k smallest distance to x.

- K-Nearest Neighbor (KNN) classification - supervised learning

KNN Classifiers



- Requires three things
  - The set of stored records
  - Distance metric
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown seed:
  - Compute distance to other training seeds
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown seed (e.g., by taking majority vote)

Nearest Neighbor Classification

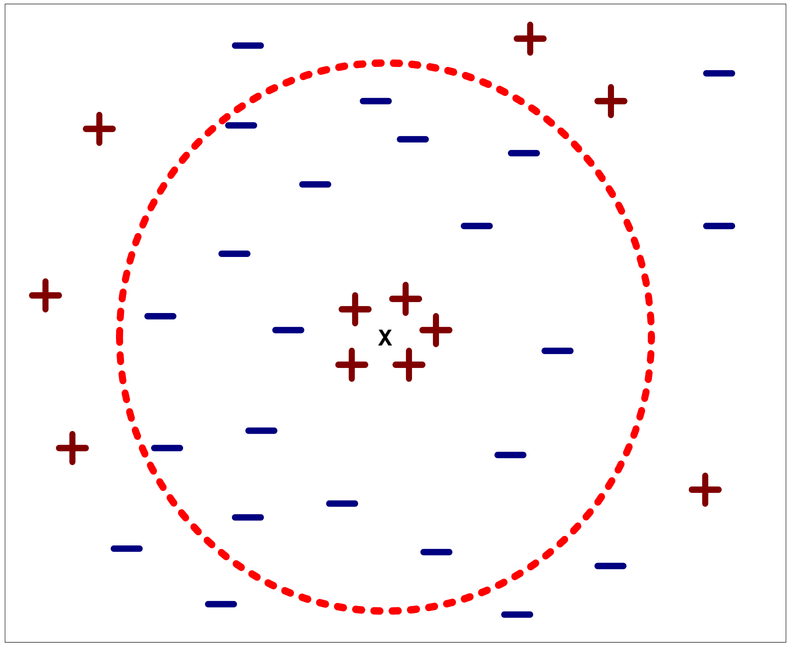
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weight the vote according to distance
    - weight factor, w = $1/d^2$

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise
  - If k is too large, neighborhood may include points from other classes



Issues of Nearest Neighbor Classification

- Scaling issues
  - Attributes (features) may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

K-NN Algorithm

- Training:
  - Save the training examples (a lazy learner; does not build training model explicitly).
- At prediction:
  - Find the k training examples $(x_1,y_1),\ldots(x_k,y_k)$ that are closest to the test example x
  - Predict the most frequent class among those y_i's.

K-NN Algorithm

- Training:
  - Save the training examples

- At prediction:
  - Find the *k* training examples $(x_1,y_1),\ldots(x_k,y_k)$ that are closest to the test example x
  - Predict the most frequent class among those y_i's.

- Improvements:
  - Weighting examples from the neighborhood
  - Measuring “closeness”
  - Finding “close” examples in a large training set quickly

Tricks with Fast k-NN

Anyway to speed up the KNN?

Tricks with Fast k-NN

K-means using r-NN

1. Pick k points $c_1=x_1,\ldots.,c_k=x_k$ as centers
2. For each x_i, find D_i=Neighborhood(x_i)
3. For each x_i, let c_i=mean(D_i)
4. Go to step 2….