Clustering: K-Means, Nearest Neighbors

Foundation of Data Analysis

01/30/2020

Clustering Example

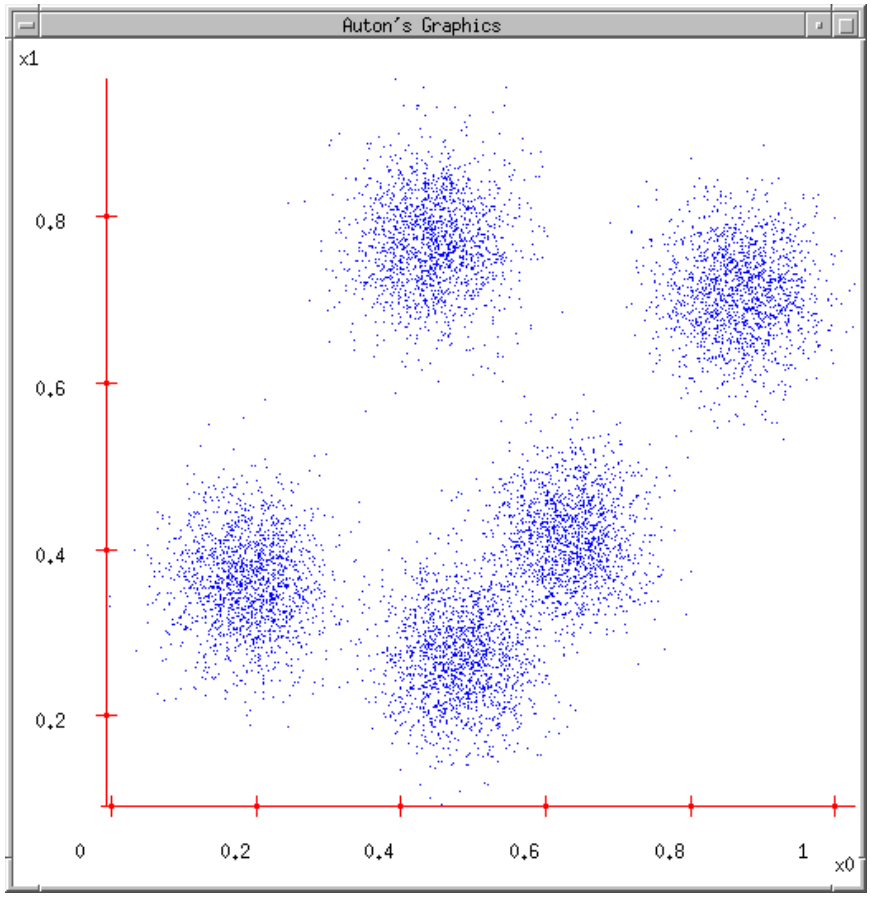
Original image

Segmented image

Divide data into different groups

Clustering: K-means

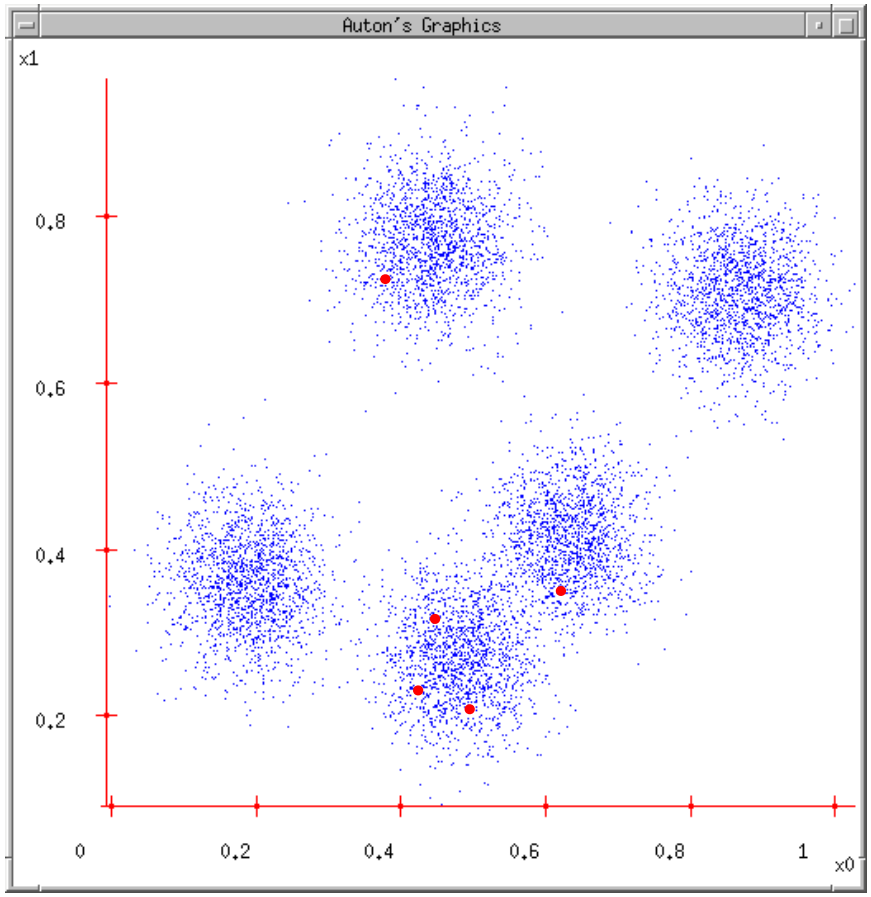
1. Ask user how many clusters they'd like (e.g. k=5)



Sides credit to Andrew W. Moore, CMU

Clustering: K-means

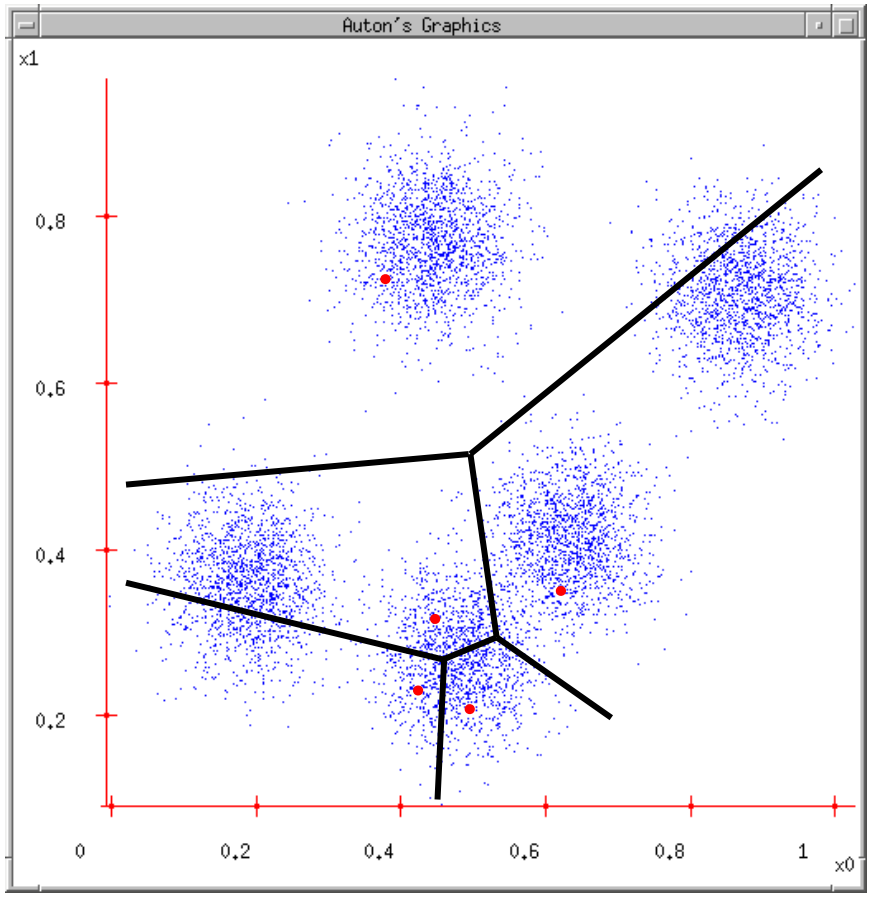
1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations



Sides credit to Andrew W. Moore, CMU

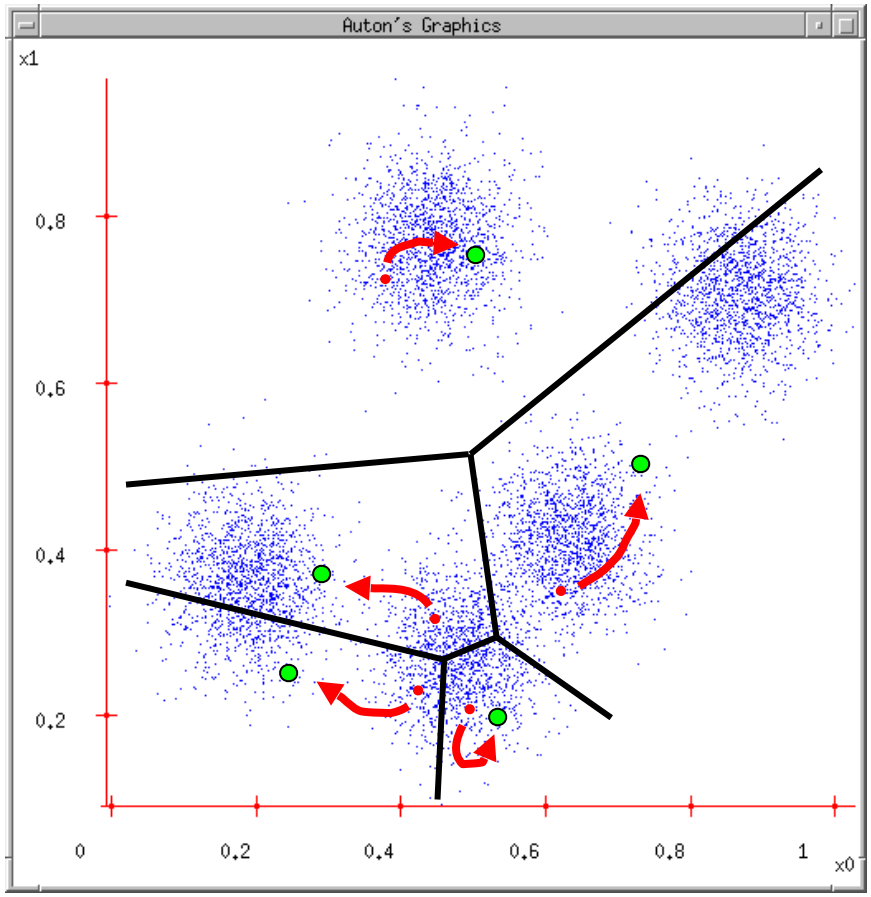
Clustering: K-means

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: K-means

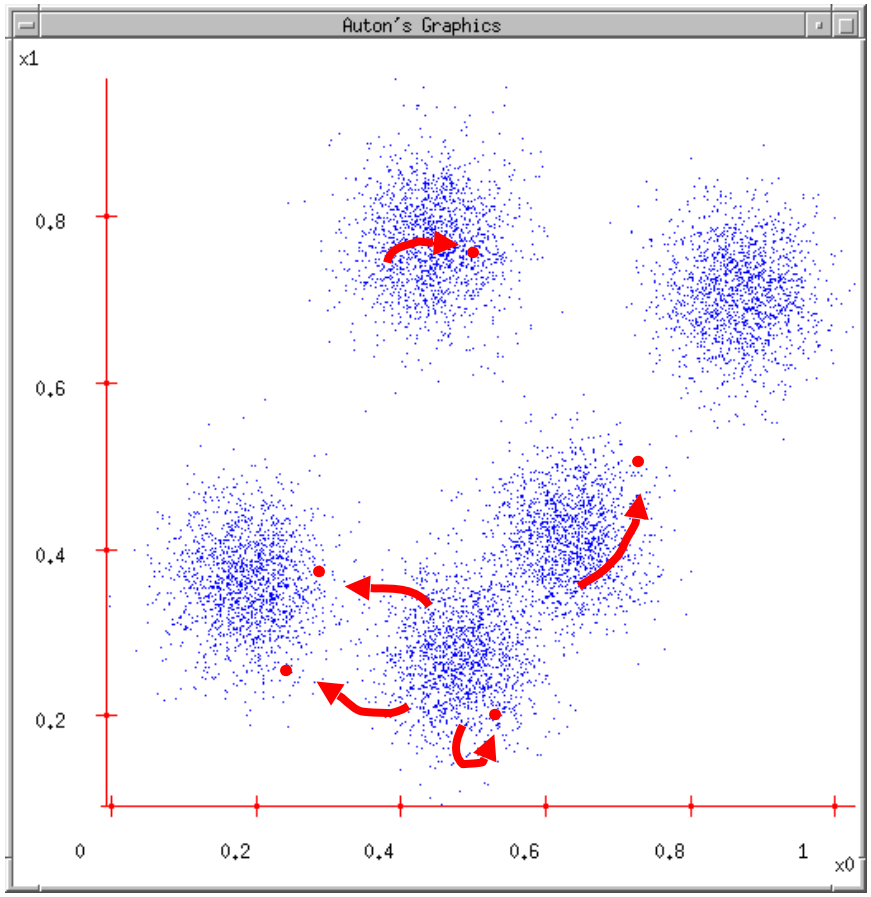
1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center re-finds the centroid of the points it owns…



Sides credit to Andrew W. Moore, CMU

Clustering: K-means

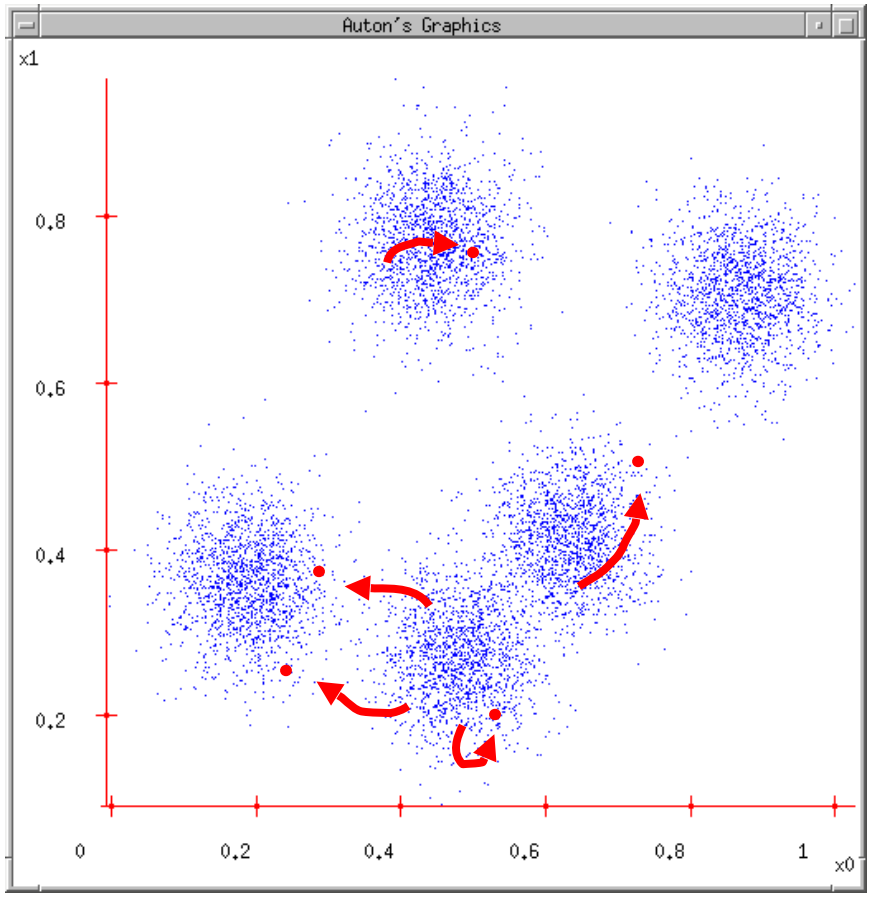
1. Ask user how many clusters they'd like (e.g. k=5)
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center re-finds the centroid of the points it owns…
5. …and jumps there



Sides credit to Andrew W. Moore, CMU

Clustering: 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 re-finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat steps 3-5 until terminated!



Sides credit to Andrew W. Moore, CMU

Disadvantages of K-means

Disadvantages of K-means

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

Disadvantages of K-means

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

- Hard assignment for labels might lead to misgrouping

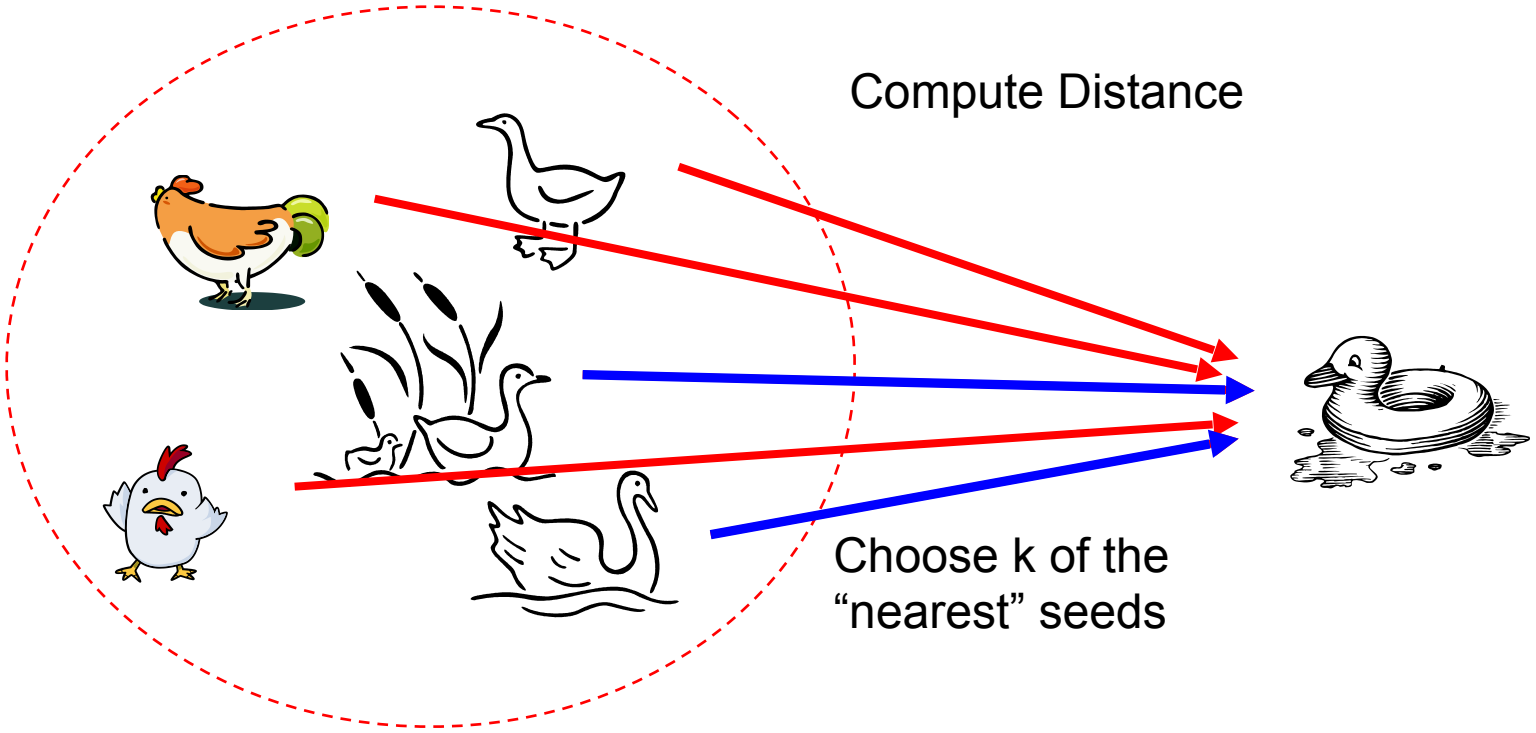
Disadvantages of K-means

- Does not work efficiently with complex structured data (mostly non-linear)
- Hard assignment for labels might lead to misgrouping
- Random guess for initialization might be a hassle

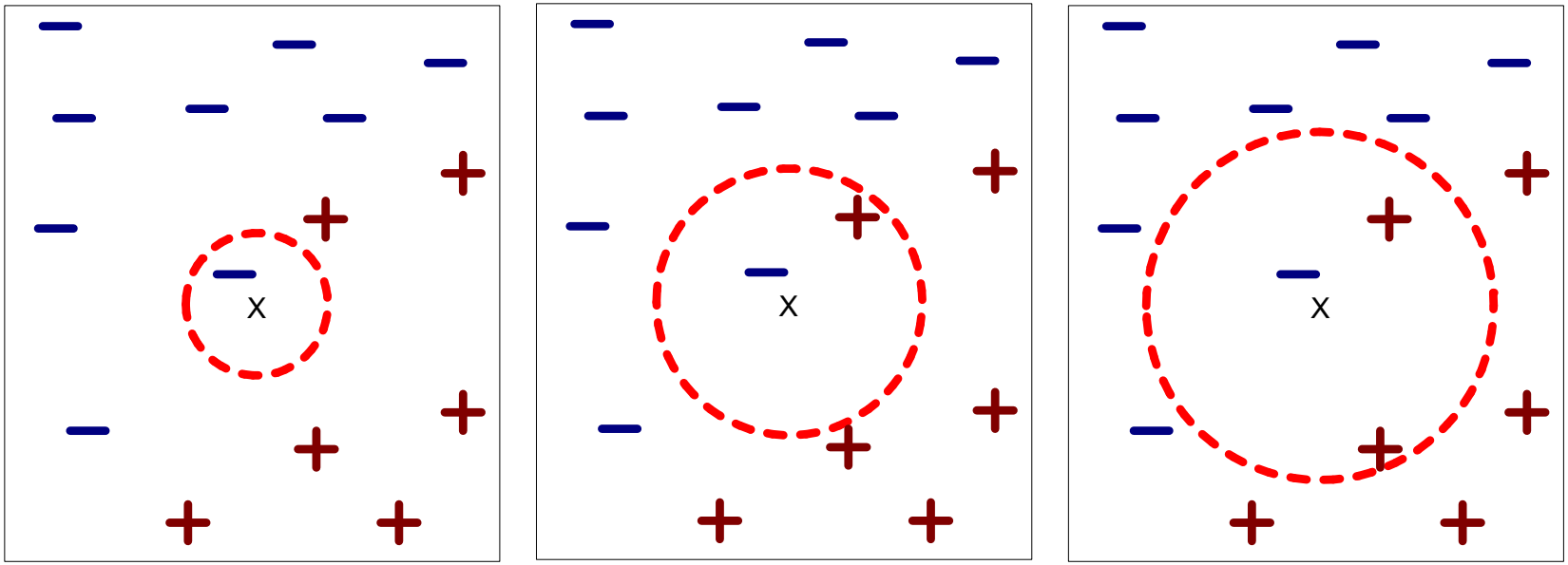
- Nearest Neighbors: (Un)supervised Learning (non-parametric model)

Nearest Neighbors: Unsupervised Learning

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



Nearest Neighbors

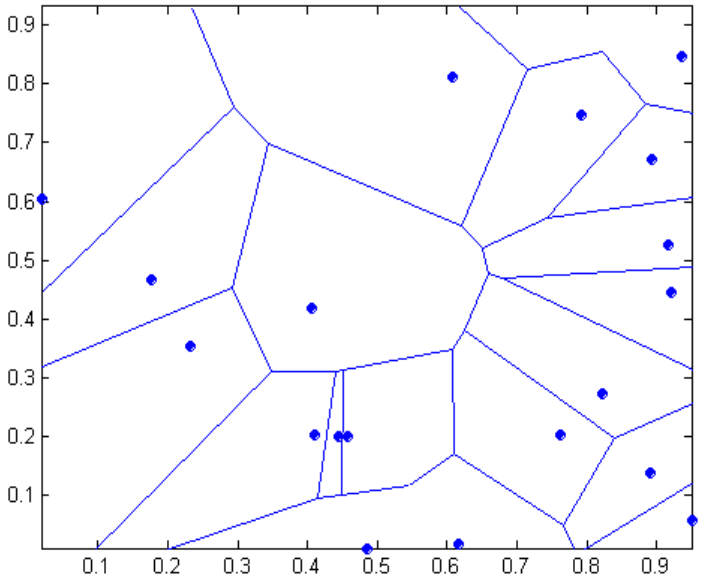


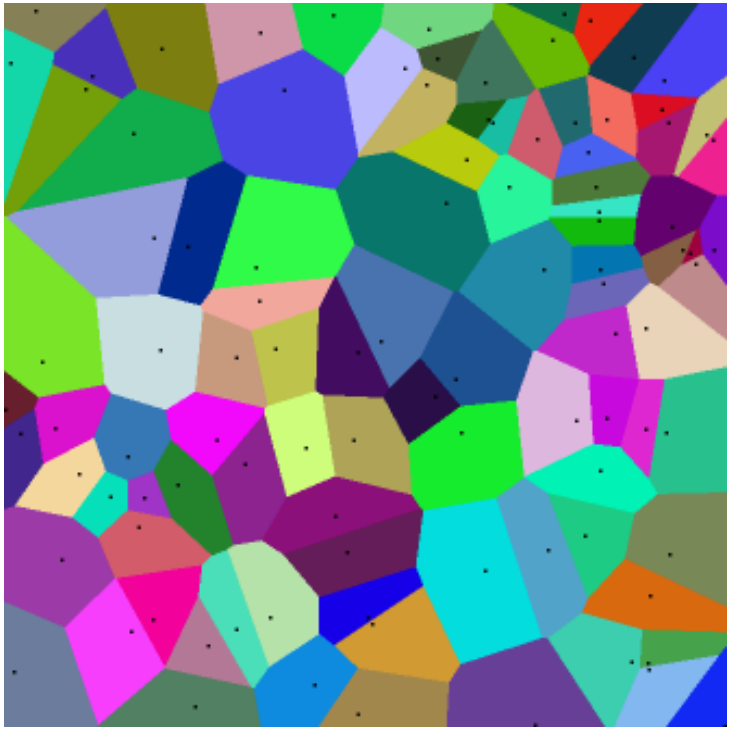
(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

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

Nearest Neighbor

Voronoi Diagram

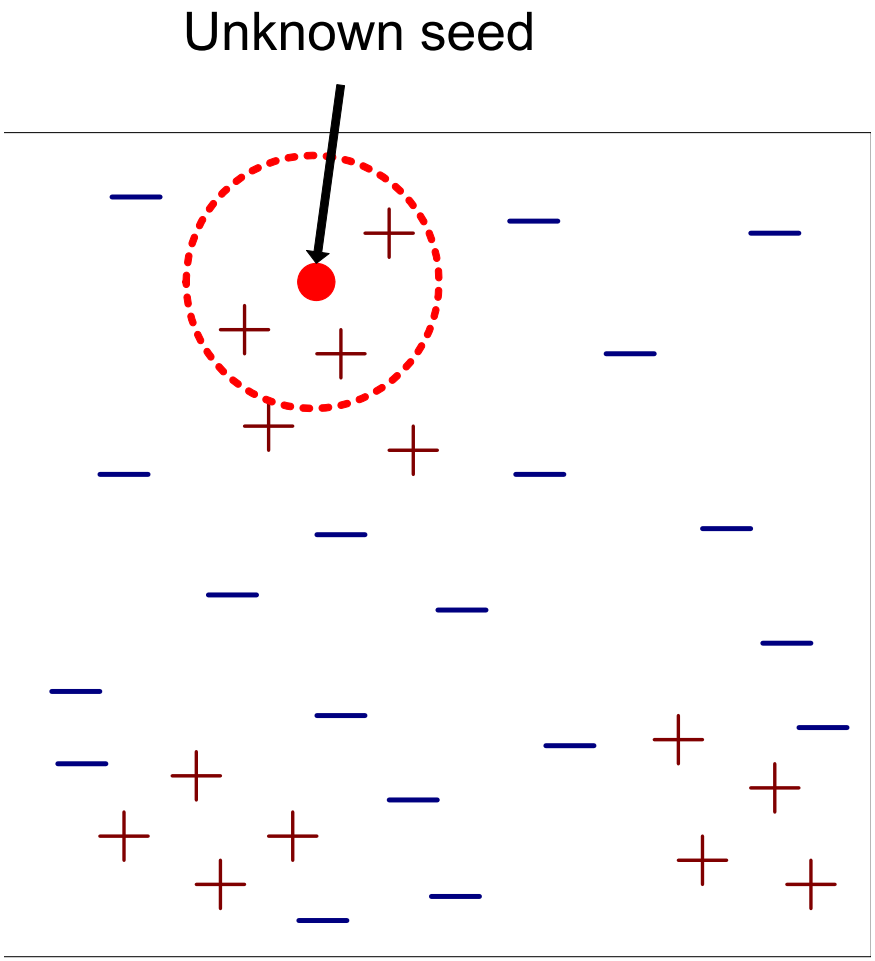




- Partitions space into regions
- boundary: points at the same distance from two different training examples

- 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, $w = 1/d^2$

Combining Distance to Neighbors

Standard KNN:

$$\hat{y} = \arg\max_y C(y, Neighbors(x))$$

$$C(y, D') \equiv |\{(x', y') \in D' : y' = y\}|$$

Distance-weighted KNN:

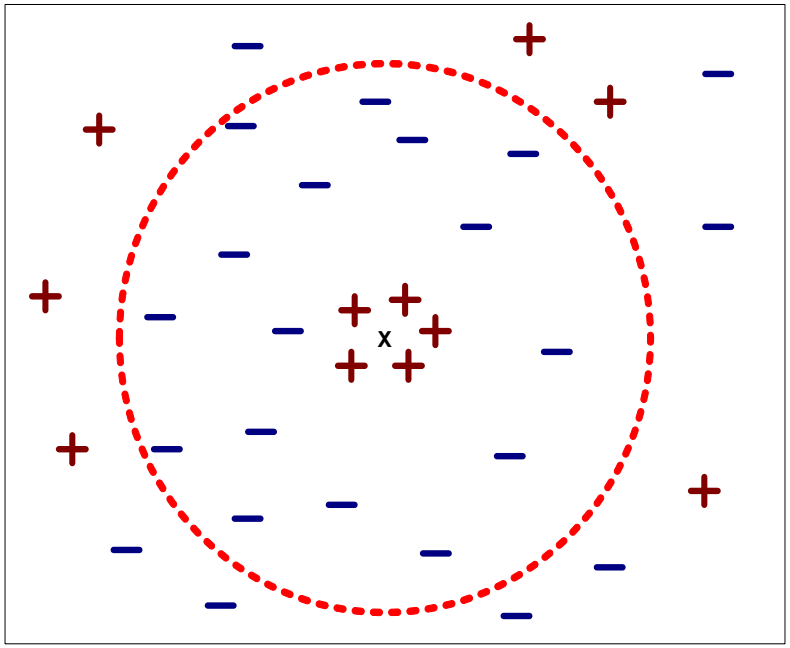
$$\hat{y} = \arg\max_y C(y, Neighbors(x))$$

$$C(y, D') \equiv \sum_{\{(x', y') \in D' : y' = y\}} (SIM(x, x'))$$

$$C(y, D') \equiv 1 - \prod_{\{(x', y') \in D' : y' = y\}} (1 - SIM(x, x'))$$

Nearest Neighbor Classification…

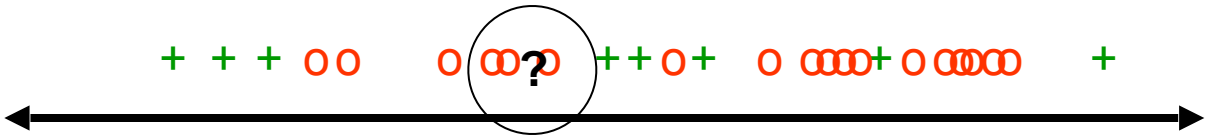
- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



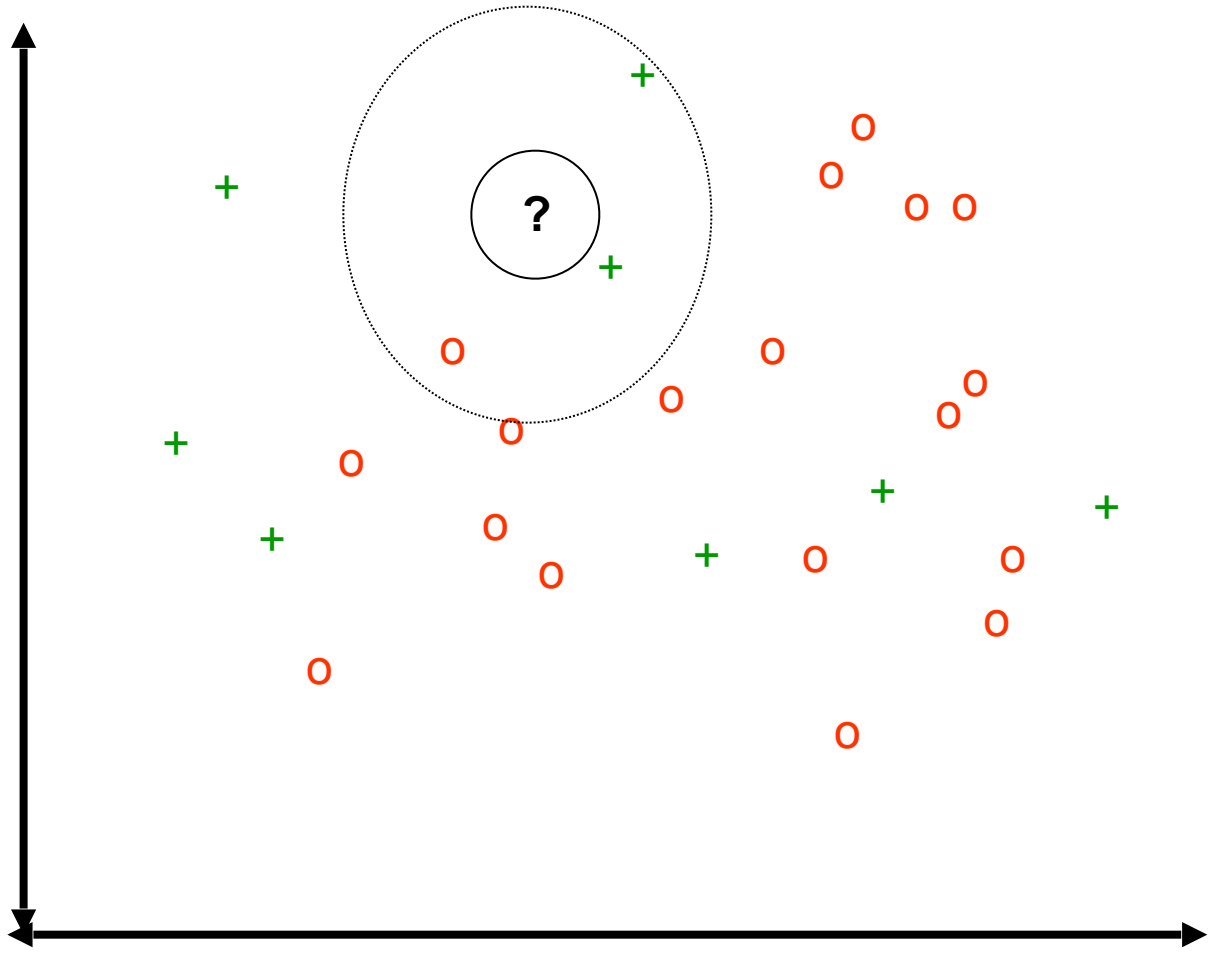
Issues of Nearest Neighbor Classification

- Scaling issues
  - Attributes 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

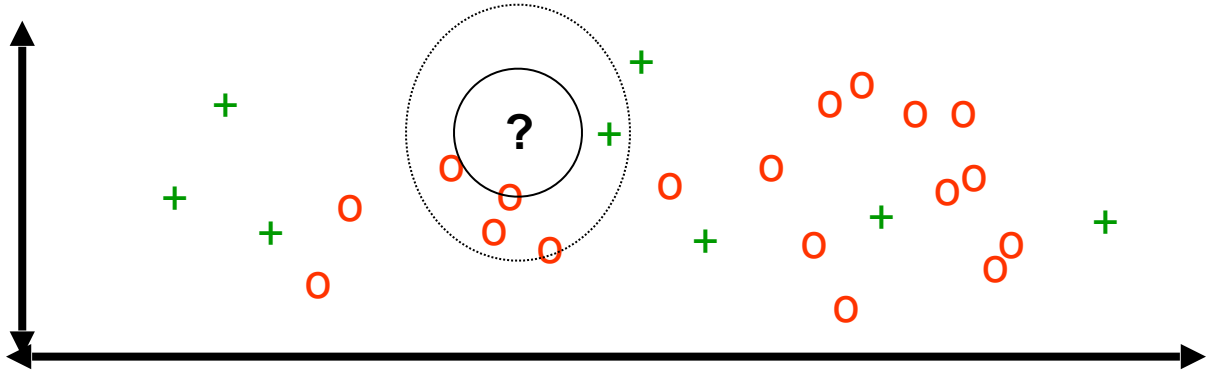
K-NN and Irrelevant Features



K-NN and Irrelevant Features



K-NN and Irrelevant Features



Issues of Nearest Neighbor Classification

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

1 1 1 1 1 1 1 1 1 1 1 0		1 0 0 0 0 0 0 0 0 0 0 0
	vs	
0 1 1 1 1 1 1 1 1 1 1 1		0 0 0 0 0 0 0 0 0 0 0 1
d = 1.4142		d = 1.4142

Solution: Normalize the vectors to unit length.

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.

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

Discussion on the k-NN Algorithm

- k-NN classifiers are lazy learners
  - Does not build models explicitly
  - Classifying unknown seeds are relatively expensive

Discussion on the k-NN Algorithm

- k-NN classifiers are lazy learners
  - Does not build models explicitly
  - Classifying unknown seeds are relatively expensive
- The k-NN algorithm for continuous-valued target functions
  - Calculate the mean values of the k nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the seed x_q
    - giving greater weight to closer neighbors
- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes.
  - To overcome it, scaling or eliminating the least relevant attributes

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….