Similarity & Clustering Methods

Alessandro Leite

November 22th, 2019

Outline

1. **Introduction & Motivation**

2. **Nearest Neighbor Methods**
   - Decision boundary
   - Algorithm
   - Similarity distances
   - Characteristics
   - Summary

3. **Clustering Methods**
   - Algorithm
   - K-Means clustering
   - Algorithm
   - Summary

Introduction

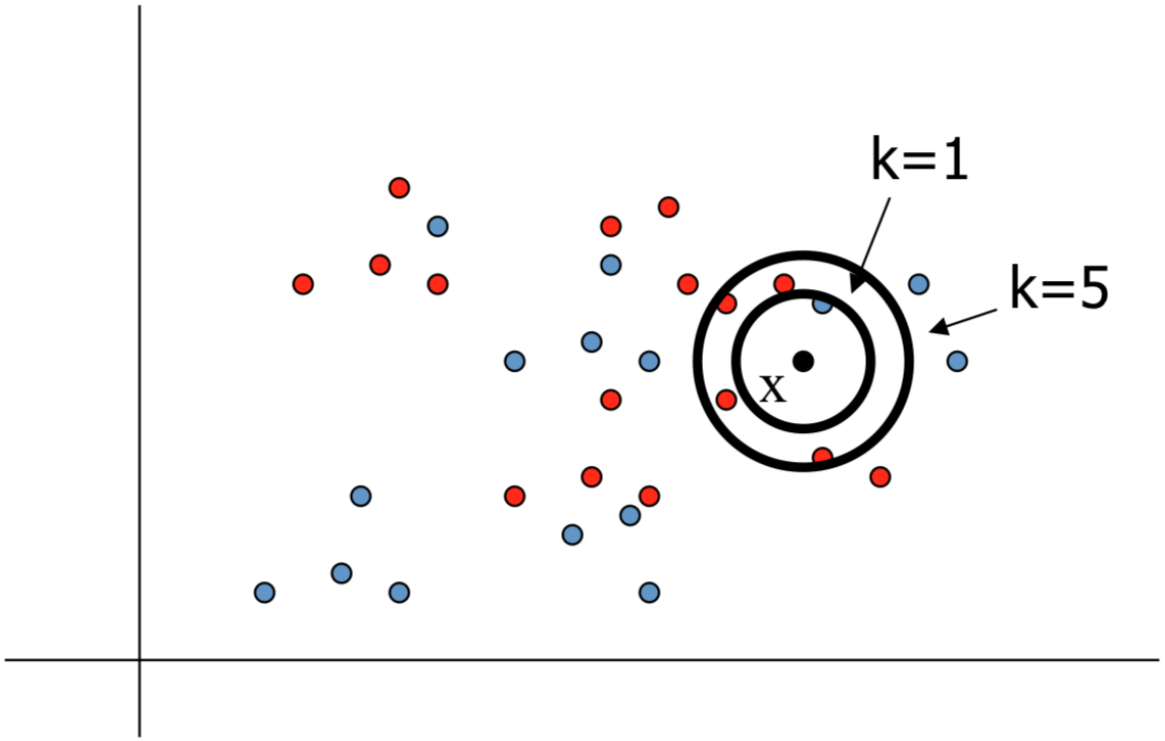
- When faced with a classification problem, it may be useful to start by exploring simple methods
- It enables us to have a baseline against which more complex models can be compared
- **Nearest neighbor** is a popular method that can performs surprisingly well
- Nearest neighbor method comprises a useful starting point since they readily encode basic smoothness intuitions and are easy to implement
- It follows the strategy: do what your neighbor does

Do what your neighbor does

- Given a data set $\mathcal{D} = \{X^n, c^n\}$ and a novel data point $\boldsymbol{x}$, the goal is to return the correct class $c(\boldsymbol{x})$
- For each new data point $\boldsymbol{x}$, find the nearest input in the training set and use the class of it
- How do we identify a neighbor?
- For the vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ representing two different data points, we measure "nearness" by using a dissimilarity function $d(\boldsymbol{x}, \boldsymbol{x}')$

Nearest Neighbor Methods

- To classify a new input vector x', compute the k-closest training data points to x' and assign the most frequently occurring class it



Computing the dissimilarity distance

- An example of dissimilarity function includes the Euclidean distance

$$d(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^T (\boldsymbol{x} - \boldsymbol{x}')$$

- More conveniently written as

$$d(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} - \boldsymbol{x}')^2$$

- The decision is determined by a perpendicular bisectors of the closest training points with the different training labels
- The partition of the input space into regions equally classified is called **Voronoi tesselation**

Decision boundaries

- Nearest neighbor methods do not explicitly compute the decision boundaries
- The decision boundaries comprises a subset of the Voronoi diagram for the training data

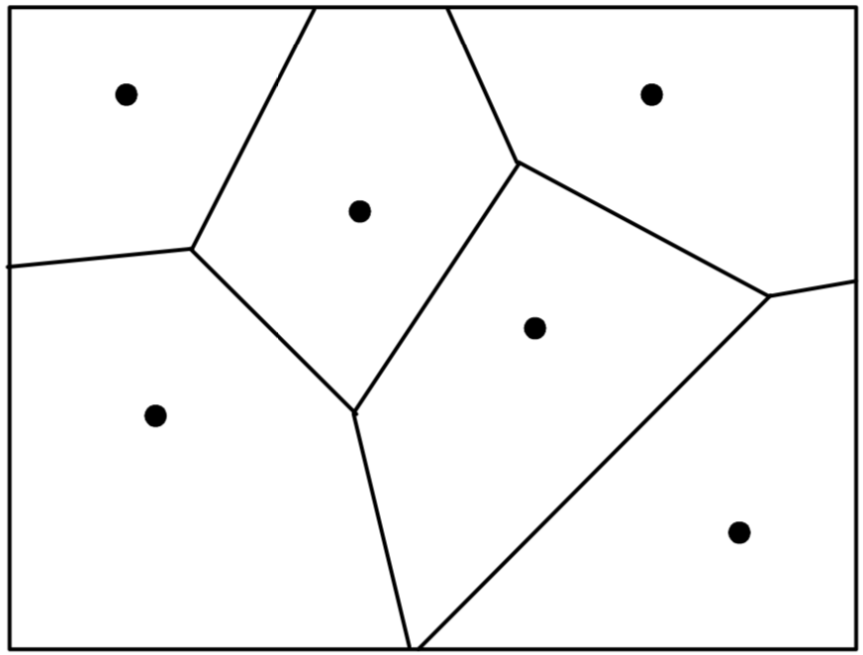
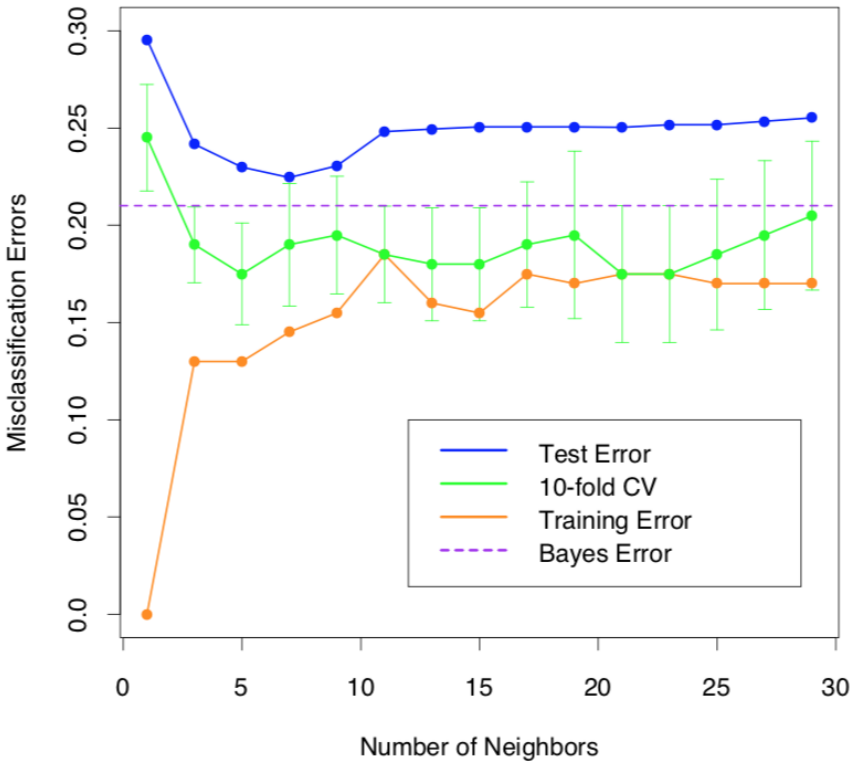
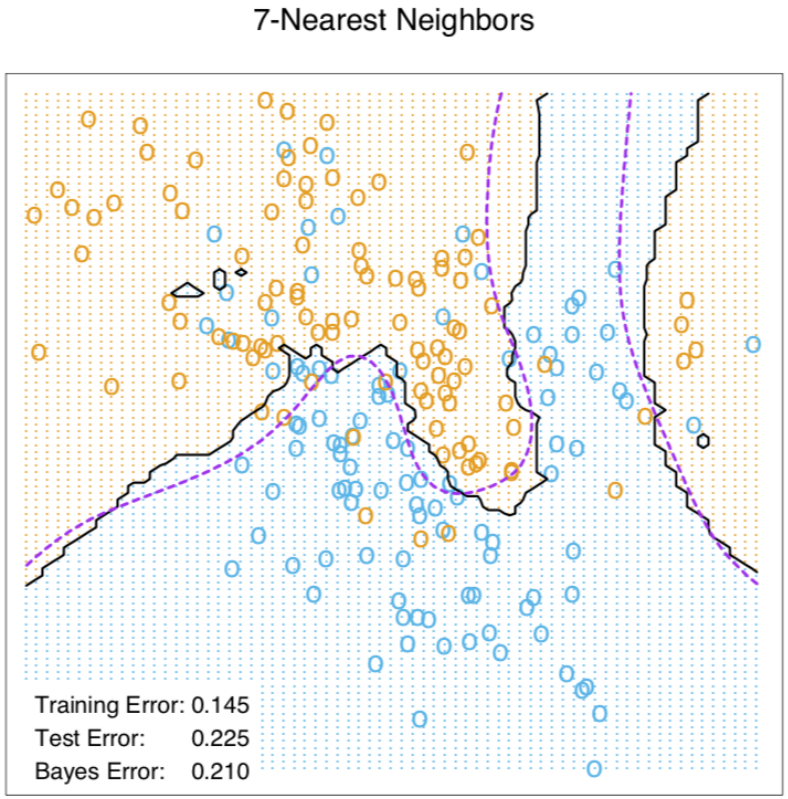


Figure 1: 1-NN decision surface

- The complexity of the decision boundaries depends on the number of examples

Example results for kNN





Source: Hastie & Tibshirani (2016), Chapter 13

Nearest neighbor algorithm

Input:
a novel vector $\boldsymbol{x}$
training set $\mathcal{D} = \{(\boldsymbol{x}^n, c^n)), n = 1 \ldots, N\}$

1. Compute the dissimilarity of the new point x to each of the training data points, $d^n = d(x, x^n), n = 1, \ldots N$

2. Find the training point $x^{n^\star}$ which is nearest to x:

$$n^\star = \underset{n}{\operatorname{argmin}}\, d(x, x^n)$$

3. Assign the class label $c(x) = c^{n^*}$

4. In the case that there are two or more nearest neighbors with different class labels, the most numerous class is chosen. If there is no one single most numerous class, we use the k-nearest neighbors

Choosing the number of k neighbors

- We choose more than one neighbor to average out the noise of the data
- Therefore, a large value of k increase the computing time (i.e., it is computationally intensive)
- k can be set by **cross-validation**
- A heuristic comprises in setting $k \approx \sqrt{n}$

Computing distances between examples

- **Euclidean distance**

$$d\left(\boldsymbol{x}^1, \boldsymbol{x}^2\right) = \left\|\boldsymbol{x}^1 - \boldsymbol{x}^2\right\|_2 = \sqrt{\sum_{j=1}^{p} \left(x_j^1 - x_j^2\right)^2}$$

- **Manhatan distance**

$$d\left(\boldsymbol{x}^1, \boldsymbol{x}^2\right) = \left\|\boldsymbol{x}^1 - \boldsymbol{x}^2\right\|_1 = \sum_{j=1}^{p} \left|x_j^1 - x_j^2\right|$$

- **Minkowski distance**

$$d\left(\boldsymbol{x}^1, \boldsymbol{x}^2\right) = \left\|\boldsymbol{x}^1 - \boldsymbol{x}^2\right\|_q = \left(\sum_{j=1}^{p} \left|x_j^1 - x_j^2\right|^q\right)^{\frac{1}{q}}$$

- **Pearson's correlation**

$$\rho(\boldsymbol{x}, \boldsymbol{z}) = \frac{\sum_{j=1}^{p} (x_j - \bar{x})(z_j - \bar{z})}{\sqrt{\sum_{j=1}^{p} (x_j - \bar{x})^2} \sqrt{\sum_{j=1}^{p} (z_j - \bar{z})^2}}$$

$$\bar{x} = \frac{1}{p} \sum_{j=1}^{p} x_j$$

- Assuming the data are centered

$$\rho(\boldsymbol{x}, \boldsymbol{z}) = \frac{\sum_{j=1}^{p} x_j z_j}{\sqrt{\sum_{j=1}^{p} x_j^2} \sqrt{\sum_{j=1}^{p} z_j^2}}$$

Nearest neighbor methods belong to a subcategory of non-parametric models

- In **parametric** models, we estimate the parameters from the training dataset to learn a function that can classify new data points without requiring the original dataset
- **Non-parametric** models cannot be characterized by a fixed set of parameters
  - the complexity of the decision function grows with the number of data points
  - The decision function is expressed directly in terms of the training set

Nearest neighbor are an instance-based learning methods

- Nearest neighbor methods belong to a subcategory of non-parametric models know as **instance-based learning**
  - it learns by memorizing the training dataset
  - the cost of learning is zero
- The learning strategy is similar to the **case-based reasoning**
  - Doctors treats a patient based on how patients with similar symptoms were treated
  - Judges rule court cases based on legal precedent

What are the characteristics of nearest neighbor methods?

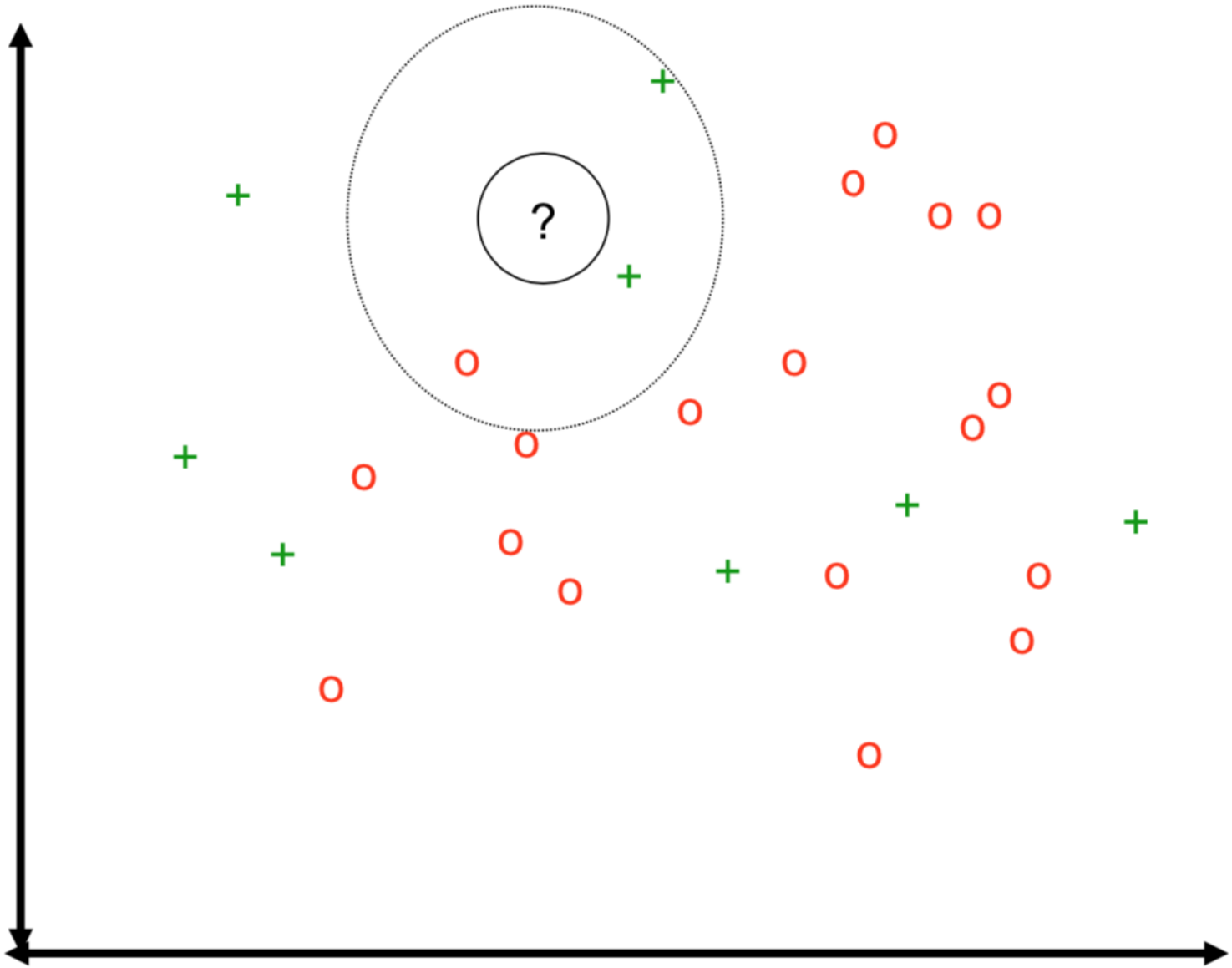
- It **memorizes** all the **training set**
- It is a typical example of **lazy learner**: it doesn't learn a discriminative function from the training data
- When a new data point x is presented, it looks for k training examples that are closest to it and then, labels it accordingly
- In the **classification** scenario, it adopts the **majority vote strategy**, which means that it tries to predict the **class** of the most frequent label among the k neighbors
- For regression, the prediction is based on the **average** of the labels of the k neighbors

What are the advantages and drawbacks of kNN?

- **Advantages**:
  - **Training phase is fast**: just store the training examples
  - **Keeps the training data**: useful there is something using to do with it later on
  - **Almost robust to noisy data**: averaging the k votes
  - **Can learn complex function**
- **Drawbacks**:
  - **Memory and storage requirements** is a big issue when dealing with large amounts of data
    - this requires efficient data structures such as KD-trees[1]
  - **Prediction can be slow**: the complexity of labeling a new data point is $\mathcal{O}(pn + n \log k)$
  - It can be fooled by **irrelevant features**
  - Nearest neighbor methods are susceptible to overfit due to the **curse of dimensionality**
    - feature space becomes increasingly sparse on high-dimension training sets

[1] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. "An Algorithm for Finding Best Matches in Logarithmic Expected Time". In: *ACM Transactions on Mathematical Software* 3.3 (1977), pp. 209–226.

Nearest neighbor methods and irrelevant features



Nearest neighbor methods: summary

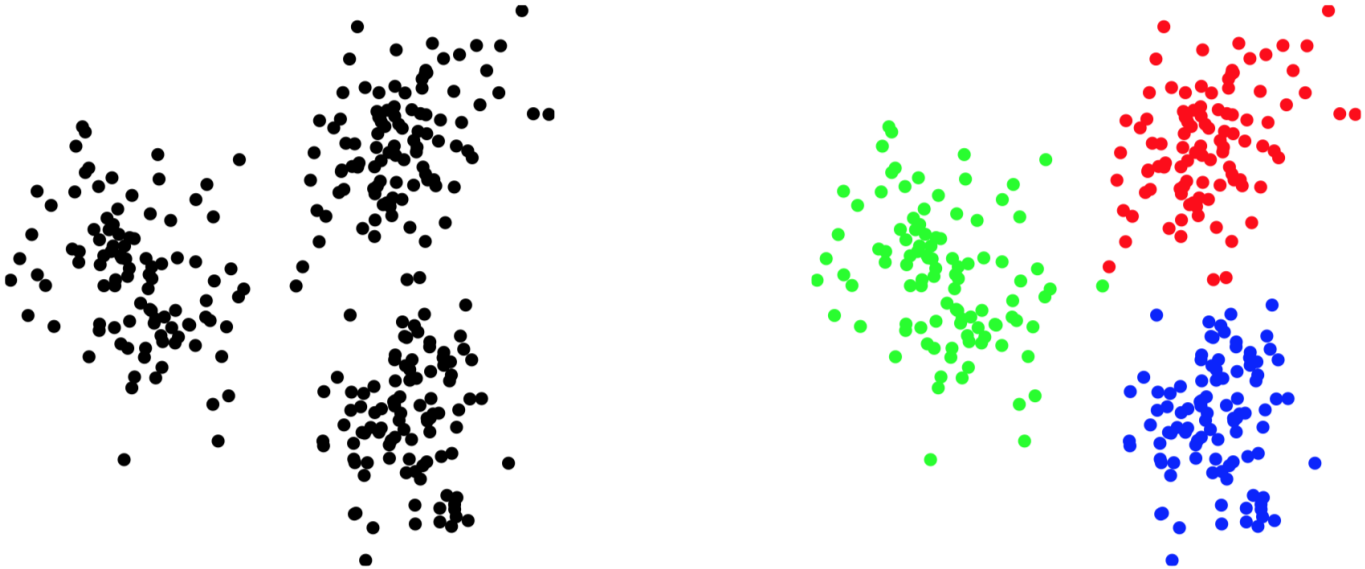
- **Can be considered when**:
  - The input space maps to points in $\mathbb{R}^n$
  - There are less than 20 attributes per sample
  - There are lot of training data
- **Advantages**:
  - Training phase is very fast; i.e., cost of training is zero
  - Can learn complex target functions
  - Do not lose any information
- **Disadvantages**:
  - Slow at query time
  - Depends on good distance/similarity function between examples
  - Can be easily fooled by irrelevant attributes
    - Susceptible to overfit due to the **curse of dimensionality**

Introduction

- Clustering is an unsupervised learning approach
  - Requires data, but not labels
  - Can detect pattern in
    - topic discovery and document classification
    - Customer marketing segmentation
    - Regions of images
    - financial sectors
  - Useful when we do not known what we are looking for

What does comprise the clustering approach?

- Given N n-vectors $x_1, \ldots, x_n$
- Partition the vectors into k groups
- In a way that vectors in the same group are close to each other



What is the objective of clustering methods?

- $G_j \subset \{1, \ldots, N\}$ is group j, for $j = 1, \ldots, k$
- c_i is a group that x_i is in $i \in G_{c_i}$
- with n-vectors $z_1, \ldots, z_k$ of group representatives
- Clustering objective is:

$$J^{\text{clust}} = \frac{1}{N} \sum_{i=1}^{N} \|x_i - z_{c_i}\|^2$$

- Mean square distance from vectors to associated representative
- J^{clust} small means good clustering
- Goal comprises in choosing clustering c_i and representatives z_j to minimize J^{clust}

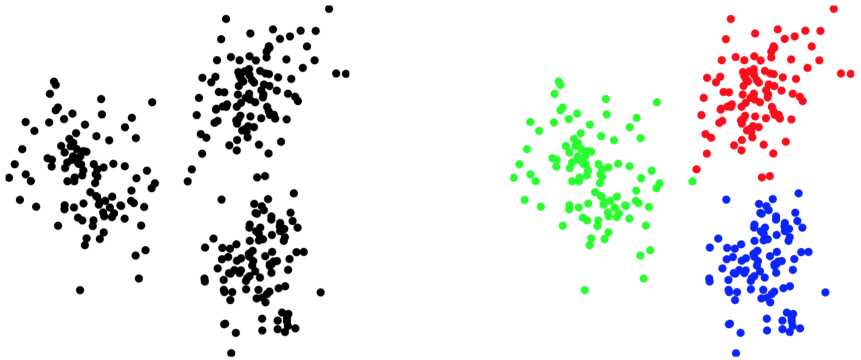
Clustering example: image segmentation

- The goal is to break up the image into meaningful similar regions

Clustering algorithms

- Partitions algorithms
  - K-means
  - Mixture of Gaussian
  - Spectral Clustering
- Hierarchical algorithms
  - Bottom up – agglomerative
  - Top down – divisive

K-means algorithm



- An iterative clustering algorithm
- Initially, picks up k random points as cluster centers
- Iteratively
  1. Assign data points to the closest cluster center
  2. Change the cluster center to the average of its assigned points
- Stop when there is no assignment changing

K-means algorithm: properties

- Guaranteed to converge in a finite number of iterations
- Running time per iteration:
  1. Assign data points to closest cluster center: $\mathcal{O}(KN)$
  2. Change the cluster center to the average of its assigned points: $\mathcal{O}(n)$

K-means algorithm: convergence guarantees

- **Objective**

$$\min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

1. Fix μ and optimize C:

$$\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2 = \min_{c} \sum_{i}^{n} |x_i - \mu_{x_i}|^2$$

2. Fix C and optimize μ:

$$\min_{\mu} \sum_{i=1}^{k} \sum_{x \in C_i} |x - \mu_i|^2$$

   - Take partial derivative of μ_i and set to zero, we have:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

3. K-means algorithm implements an alternative optimization approach, where in each step contributes to decrease the objective, leading to guaranteed convergence.

k-means algorithm

The goal is to predict k centroids and a label $c^{(i)}$ for each data point. The k-means clustering algorithm is as follows:

Input:

a training set $x^{(1)}, \ldots, x^{(n)}$

k number of clusters to group the data

1. Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$

2. **repeat**

   For every i, set:

$$c^{(i)} = \underset{j}{\operatorname{argmin}} ||x^{(i)} - \mu_j||^2$$

   For each j, set:

$$\mu_j = \frac{\sum_{i=1}^{N} 1\{c^(i) = j\} x^{(i)}}{\sum_{i=1}^{N} 1\{C^{(i)} = j\}}$$

until $\mu_1, \mu_2, \ldots, \mu_k$ *stop changing*

In resume, clustering are very useful methods

- K-means is method for assigning individual data points to a collection of clusters
- Data points are assigned to a cluster based on its distance from the center of the cluster
- It is useful when:
  - Looking for structure in data sets
  - We have unclassified data and we suspect that the data fall into several categories

References

- Hal Daume III. *A Course in Machine Learning*. 2nd. Self-published, 2017. URL: http://ciml.info/dl/v0_99/ciml-v0_99-all.pdf

  kNN: sessions 3.2 and 3.2

- Andrew Moore. *An introductory tutorial on kd-trees*. Tech. rep. Technical Report No. 209, Computer Laboratory, University of Cambridge. Pittsburgh, PA: Carnegie Mellon University, 1991. URL: https://bit.ly/2GeVatO

- Nearest neighbor search with kd-trees (*bit.ly/2lhcguv*)

- Voronoi tessellation (*bit.ly/2GlGqdV*)