Clustering

Stephen Boyd
(with thanks to Karanveer Mohan)

EE103
Stanford University

September 29, 2015

Outline

Clustering

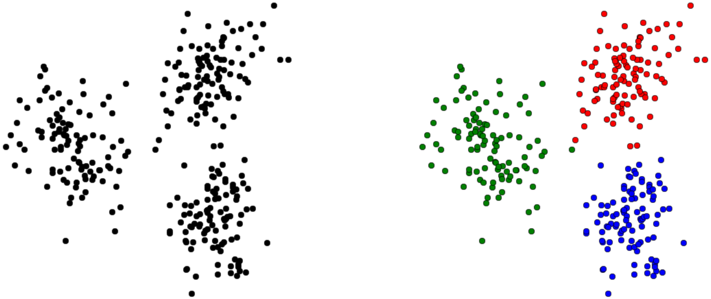
Algorithm

Examples

Applications

Clustering

- given N n-vectors $x_1, \ldots, x_N$
- goal: partition (divide, cluster) into k groups
- want vectors in the same group to be close to one another



Example settings

- topic discovery and document classification
  - x_i is word count histogram for document i
- patient clusering
  - x_i are patient attributes, test results, symptoms
- customer market segmentation
  - x_i is purchase history and other attributes of customer i
- color compression of images
  - x_i are RGB pixel values

Clustering objective

- $G_j \subset \{1, \ldots, N\}$ is group j, $j = 1, \ldots, k$
- c_i is group that x_i is in: $x_i \in G_{c_i}$
- group *representatives*: n-vectors $z_1, \ldots, z_k$

- clustering objective is

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

  mean square distance from vectors to associated representative
- J small means good clustering
- goal: choose clustering c_i and representatives z_j to minimize J

Outline

Clustering

Algorithm

Examples

Applications

Partitioning the vectors given the representatives

- suppose representatives $z_1, \ldots, z_k$ are given
- how do we assign the vectors to groups, *i.e.*, choose $c_1, \ldots, c_N$?

- c_i only appears in term $\|x_i - z_{c_i}\|^2$ in J
- to minimize over c_i, choose c_i so $\|x_i - z_{c_i}\|^2 = \min_j \|x_i - z_j\|^2$
- *i.e., assign each vector to its nearest representative*

Choosing representatives given the partition

- given the partition $G_1, \ldots, G_k$, how do we choose representatives $z_1, \ldots, z_k$ to minimize J?
- J splits into a sum of k sums, one for each z_j:

$$J = J_1 + \cdots + J_k, \qquad J_j = (1/N) \sum_{i \in G_j} \|x_i - z_j\|^2$$

- so we choose z_j to minimize mean square distance to the points in its partition
- this is the mean (or average or centroid) of the points in the partition:

$$z_j = (1/|G_j|) \sum_{i \in G_j} x_i$$

k-means algorithm

- alternate between updating the partition, then the representatives
- a famous algorithm called *k-means*
- objective J decreases in each step

```
given $x_1, \dots, x_N \in \mathbf{R}^n$ and $z_1, \dots, z_k \in \mathbf{R}^n$.
repeat
    Update partition. assign $i$ to $G_j$, $j = \text{argmin}_{j'} \|x_i - z_{j'}\|^2$
    Update centroids. $z_j = \frac{1}{|P_j|} \sum_{i \in P_j} x_i$
until $z_1, \dots, z_k$ stop changing
```

Convergence of k-means algorithm

- J goes down in each step, until the z_j's stop changing
- but (in general) the k-means algorithm *does not find the partition that minimizes J*

- k-means is a *heuristic*: it is not guaranteed to find the smallest possible value of J
- the final partition (and its value of J) can depend on the initial representatives
- common approach:
  - run k-means 10 times, with different (often random) initial representatives
  - take as final partition the one with the smallest value of J

Outline

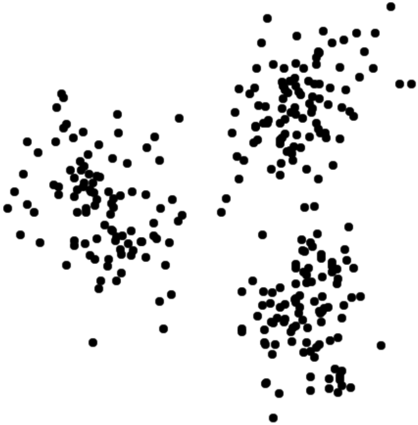
Clustering

Algorithm

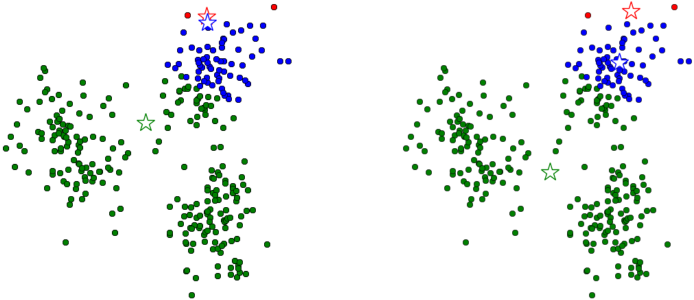
Examples

Applications

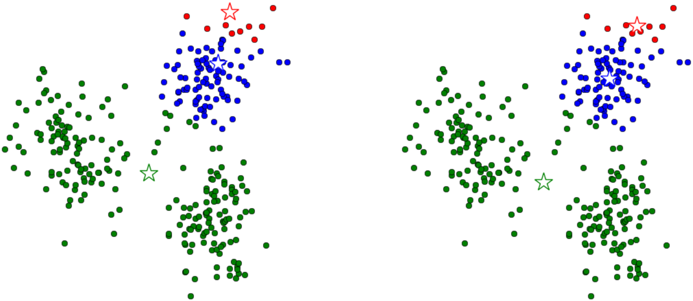
Data



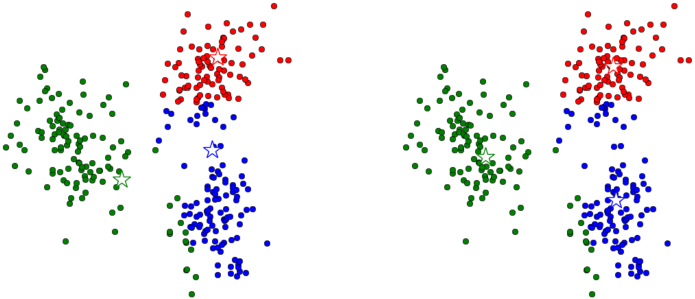
Iteration 1



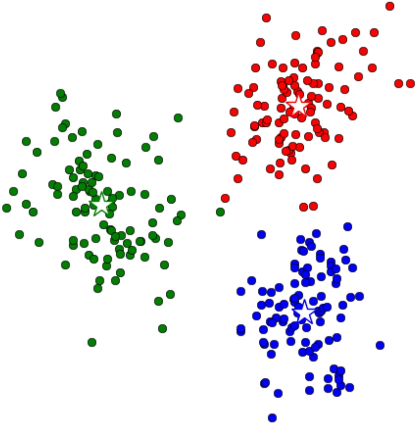
Iteration 2



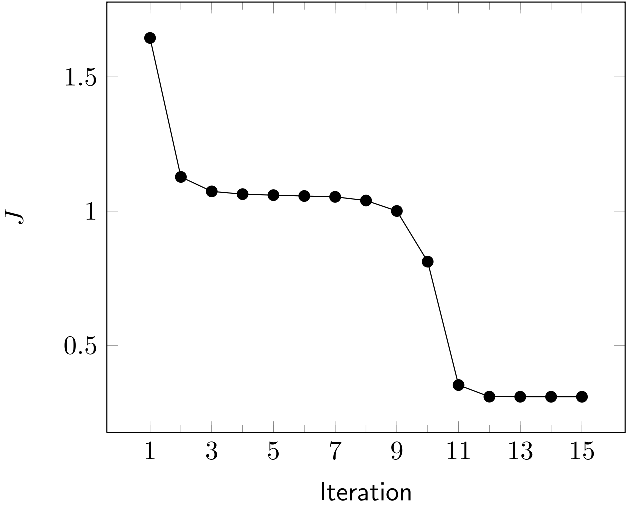
Iteration 10



Final clustering



Convergence



Outline

Clustering

Algorithm

Examples

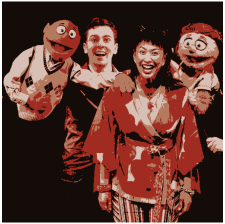
Applications

Color compression

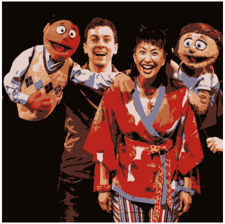
- 3-vectors $x_1, \dots x_N$ represent RGB values of each pixel in an image
- in 24-bit color representation $(x_i)_m \in \{1, 2, \dots, 256\}$
- total of $256^3 \approx 1.7 \times 10^7$ possible colors
- compress color vectors to k colors using k-means

Compressed image for various values of k

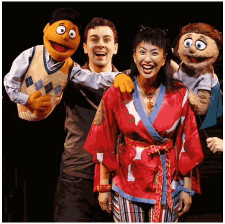
k=4



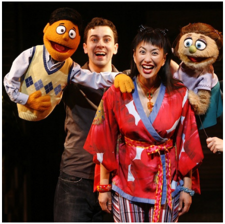
k=8



k=64

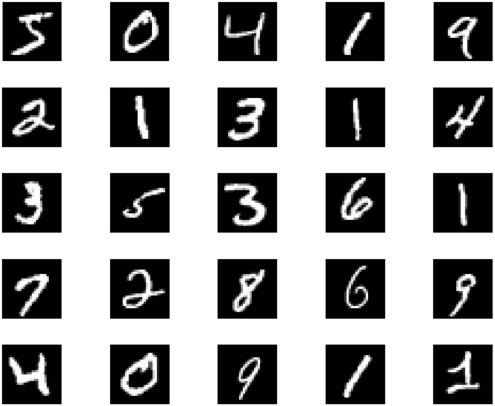


Original image



Handwritten digit image set

- MNIST images of handwritten digits (via Yann Lecun)
- $N = 60,000$ 28×28 images, represented as 784-vectors x_i
- 25 examples shown below



k-means image clustering

- run k-means with $k = 20$
- representatives shown as images below
- k-means has ‘discovered’ the digits (mostly)