CSC 411: Introduction to Machine Learning

Lecture 15: K-Means

Mengye Ren and Matthew MacKay

University of Toronto

Motivating Examples

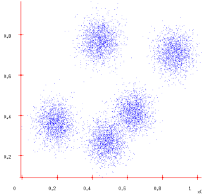
- Some examples of situations where you'd use unupservised learning
  - You want to understand how a scientific field has changed over time. You want to take **a large database of papers** and model how the **distribution of topics changes** from year to year. But **what are the topics**?
  - You're a biologist studying animal behavior, so you want to infer a high-level description of their behavior from video. You don't know **the set of behaviors** ahead of time.
  - You want to reduce your **energy consumption**, so you take a time series of your energy consumption over time, and try to break it down into **separate components** (refrigerator, washing machine, etc.).
- Common theme: you have some data, and you want to infer the causal structure underlying the data.
- This structure is **latent**, which means it's never observed.

Overview

- In last lecture, we looked at density modeling where all the random variables were fully observed.
- The more interesting case is when some of the variables are latent, or never observed. These are called **latent variable models**.
  - This lecture: K-means, a simple algorithm for **clustering**, i.e. grouping data points into clusters
  - Next lecture: Gaussian mixture models

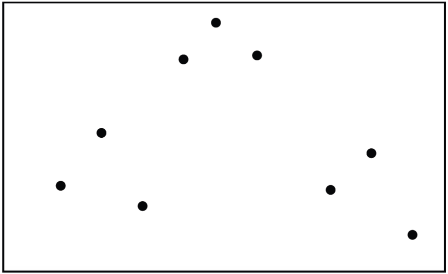
Clustering

- Sometimes the data form clusters, where examples within a cluster are similar to each other, and examples in different clusters are dissimilar:



- Such a distribution is **multimodal**, since it has multiple **modes**, or regions of high probability mass.
- Grouping data points into clusters, with no labels, is called **clustering**
- E.g. clustering machine learning papers based on topic (deep learning, Bayesian models, etc.)

Clustering



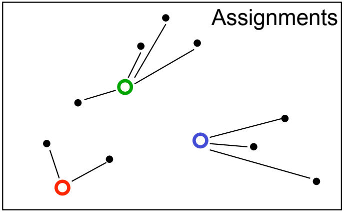
- Assume the data $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ lives in a Euclidean space, $\mathbf{x}^{(n)} \in \mathbb{R}^d$.
- Assume the data belongs to K classes (patterns)
- Assume the data points from same class are similar, i.e. close in Euclidean distance.
- How can we identify those classes (data points that belong to each class)?

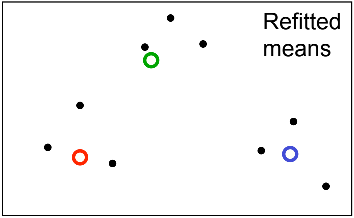
K-means intuition

- K-means assumes there are k clusters, and each point is close to its cluster center (the mean of points in the cluster).
- If we knew the cluster assignment we could easily compute means.
- If we knew the means we could easily compute cluster assignment.
- Chicken and egg problem.
- It is NP hard.
- Very simple (and useful) heuristic - start randomly and alternate between the two.

K-means

- **Initialization**: randomly initialize cluster centers
- The algorithm iteratively alternates between two steps:
  - **Assignment step**: Assign each data point to the closest cluster
  - **Refitting step**: Move each cluster center to the center of gravity of the data assigned to it





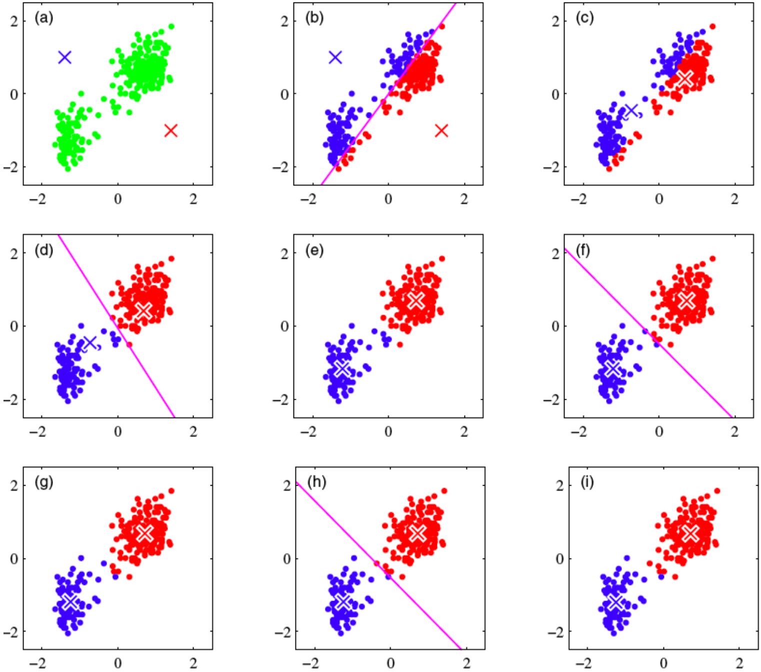


Figure from Bishop

Simple demo: http://syskall.com/kmeans.js/

K-means Objective

What is actually being optimized?

K-means Objective:
Find cluster centers $\mathbf{m}$ and assignments $\mathbf{r}$ to minimize the sum of squared distances of data points $\{\mathbf{x}^{(n)}\}$ to their assigned cluster centers

$$\min_{\{\mathbf{m}\},\{\mathbf{r}\}} J(\{\mathbf{m}\},\{\mathbf{r}\}) = \min_{\{\mathbf{m}\},\{\mathbf{r}\}} \sum_{n=1}^{N}\sum_{k=1}^{K} r_k^{(n)} ||\mathbf{m}_k - \mathbf{x}^{(n)}||^2$$

$$\text{s.t.} \sum_k r_k^{(n)} = 1, \forall n, \quad \text{where} \quad r_k^{(n)} \in \{0,1\}, \forall k, n$$

where $r_k^{(n)} = 1$ means that $\mathbf{x}^{(n)}$ is assigned to cluster k (with center $\mathbf{m}_k$)

- **Optimization method** is a form of coordinate descent ("block coordinate descent")
  - Fix centers, optimize assignments (choose cluster whose mean is closest)
  - Fix assignments, optimize means (average of assigned datapoints)

The K-means Algorithm

- **Initialization**: Set K cluster means $\mathbf{m}_1, \dots, \mathbf{m}_K$ to random values
- Repeat until convergence (until assignments do not change):
  - **Assignment**: Each data point $\mathbf{x}^{(n)}$ assigned to nearest mean

    $$\hat{k}^n = arg \min_k d(\mathbf{m}_k, \mathbf{x}^{(n)})$$

    (with, for example, L2 norm: $\hat{k}^n = arg \min_k ||\mathbf{m}_k - \mathbf{x}^{(n)}||^2$)

    and **Responsibilities** (1-hot encoding)

    $$r_k^{(n)} = 1 \longleftrightarrow \hat{k}^{(n)} = k$$

  - **Refitting:** Model parameters, means are adjusted to match sample means of data points they are responsible for:

    $$\mathbf{m}_k = \frac{\sum_n r_k^{(n)} \mathbf{x}^{(n)}}{\sum_n r_k^{(n)}}$$

K-means for Vector Quantization

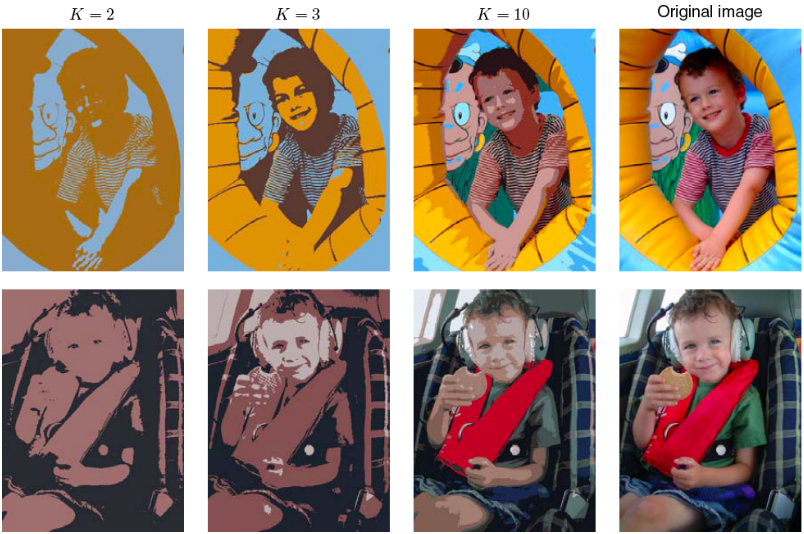
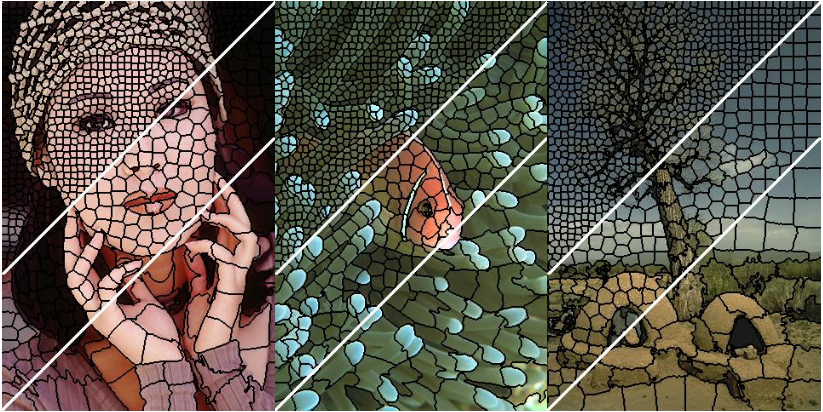


Figure from Bishop

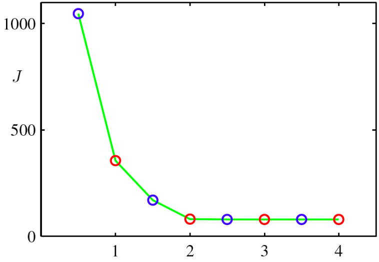
K-means for Image Segmentation



- How would you modify k-means to get superpixels?

Why K-means Converges

- Whenever an assignment is changed, the sum squared distances J of data points from their assigned cluster centers is reduced.
- Whenever a cluster center is moved, J is reduced.
- **Test for convergence**: If the assignments do not change in the assignment step, we have converged (to at least a local minimum).

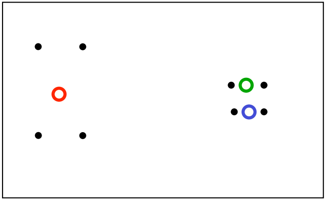


- K-means cost function after each assignment step (blue) and refitting step (red). The algorithm has converged after the third refitting step

Local Minima

- The objective J is non-convex (so coordinate descent on J is not guaranteed to converge to the global minimum)
- There is nothing to prevent k-means getting stuck at local minima.
- We could try many random starting points
- We could try non-local split-and-merge moves:
  - Simultaneously **merge** two nearby clusters
  - and **split** a big cluster into two

A bad local optimum



Soft K-means

- Instead of making hard assignments of data points to clusters, we can make **soft assignments**. One cluster may have a responsibility of .7 for a datapoint and another may have a responsibility of .3.
  - Allows a cluster to use more information about the data in the refitting step.
  - How do we decide on the soft assignments?

Soft K-means Algorithm

- **Initialization**: Set K means $\{\mathbf{m}_k\}$ to random values
- Repeat until convergence (until assignments do not change):
  - **Assignment**: Each data point n given soft "degree of assignment" to each cluster mean k, based on responsibilities

$$r_k^{(n)} = \frac{\exp[-\beta d(\mathbf{m}_k, \mathbf{x}^{(n)})]}{\sum_j \exp[-\beta d(\mathbf{m}_j, \mathbf{x}^{(n)})]}$$

  - **Refitting:** Model parameters, means, are adjusted to match sample means of datapoints they are responsible for:

$$\mathbf{m}_k = \frac{\sum_n r_k^{(n)} \mathbf{x}^{(n)}}{\sum_n r_k^{(n)}}$$

Questions about Soft K-means

Some remaining issues

- How to set β?
- What about problems with elongated clusters?
- Clusters with unequal weight and width

These aren't straightforward to address with K-means. Instead, next lecture, we'll reformulate clustering using a generative model.