Lecture 16 – Clustering

DSC 40A, Fall 2022 @ UC San Diego

Mahdi Soleymani, with help from **many others**

Announcements

- Homework 4 due **Friday at 11:59pm**.
  - Remember to submit Survey 4 after finishing!

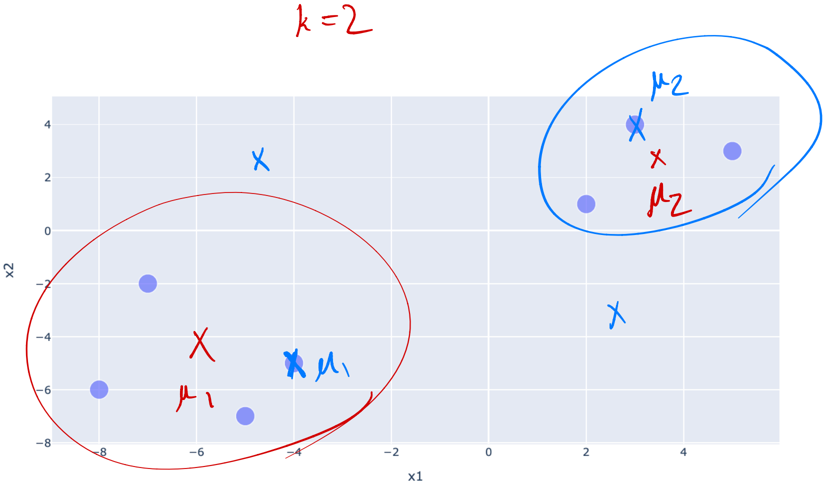
- Groupwork ~~2~~ 4 due **tonight 11/1 at 11:59pm**.

Agenda

- k-Means Clustering algorithm.
- Why does k-Means work?
- Practical considerations.

k-Means Clustering

Question: how might we “cluster” these points into groups?



Problem statement: clustering

Goal: Given a list of n data points, stored as vectors in $\mathbb{R}^d$, $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$, and a positive integer k, **place the data points into k clusters of nearby points**.

- Clusters are defined by **centroids**, $\mu_1, \mu_2, \ldots, \mu_k$. Each data point "belongs" to the group corresponding to the nearest centroid.

- We want to find the centroids that minimize **inertia**:

$C(\mu_1, \mu_2, \ldots, \mu_k) =$ total squared distance of each data point $\vec{x}_i$ to its closest centroid μ_j

- k-Means Clustering is an algorithm that attempts to minimize inertia.

k-Means Clustering, i.e. Lloyd's Algorithm

initialization

belong to dataset

1. Pick a value of k and randomly initialize k centroids.
2. Keep the centroids fixed, and update the groups.
   - Assign each point to the nearest centroid.
3. Keep the groups fixed, and update the centroids.
   - Move each centroid to the center of its group by averaging their coordinates.
4. Repeat steps 2 and 3 until the centroids stop changing.

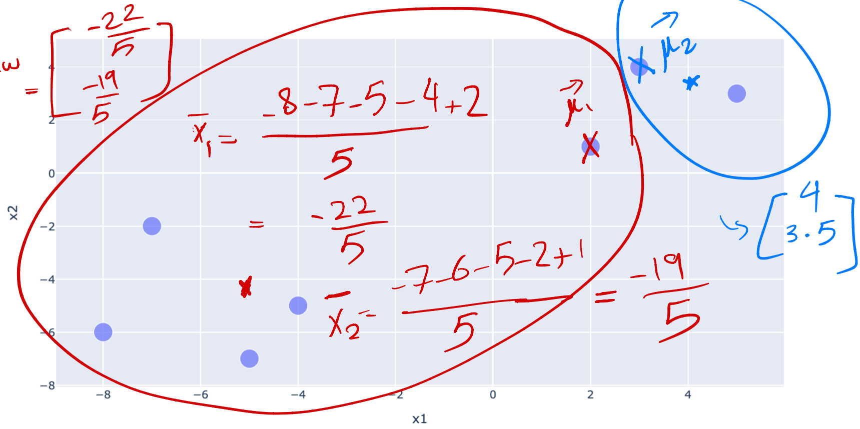
Example

See the following site for an interactive visualization of k-Means Clustering: `https://tinyurl.com/40akmeans`

An example by-hand

Suppose we choose the initial centroids $\mu_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ and $\mu_2 = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$.

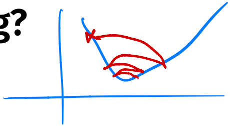
Where will the centroids move to after one iteration of k-Means Clustering?



Follow along with the demo by clicking the **code** link on the course website next to Lecture 16.

Why does k-Means work?

What is the goal of k-Means Clustering?

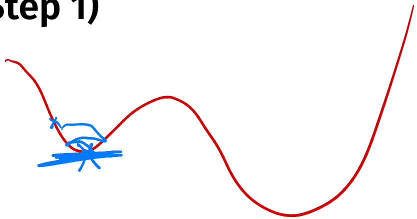


- Recall, our goal is to find the centroids $\mu_1, \mu_2, ..., \mu_k$ that minimize inertia:

$$C(\mu_1, \mu_2, ..., \mu_k) = \text{total squared distance of each data point } \vec{x}_i \text{ to its closest centroid } \mu_j$$

- Let's argue that each step of the k-Means Clustering algorithm reduces inertia.
  - After enough iterations, inertia will be small enough.

Why does k-Means work? (Step 1)

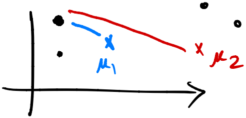


Let's look at each step one at a time.
Step 1: Pick a value of *k* and randomly initialize *k* centroids.

- After initializing our *k* centroids, we have an initial value of inertia. We are going to argue that this only decreases.

$C(\mu_1, \mu_2, \dots, \mu_k)$

Why does k-Means work? (Step 2)



Step 2: Keep the centroids fixed, and update the groups by assigning each point to the nearest centroid.

- Assuming the centroids are fixed, for each $\vec{x}_i$ we have a choice — which group should it be a part of?
- Whichever group we choose, inertia will be calculated using the squared distance between $\vec{x}_i$ and that group's centroid.
- Thus, to minimize inertia, we assign each $\vec{x}_i$ to the group corresponding to the closest centroid.

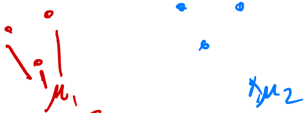
Note that this analysis holds every time we're at Step 2, not just the first time.

Why does k-Means work? (Step 3)

Step 3: Keep the groups fixed, and update the centroids by moving each centroid to the center of its group (by averaging coordinates).

- Before we justify why this is optimal, let's re-visit inertia.

Aside: separating inertia



- Inertia:

  $$C(\mu_1, \mu_2) = C(\mu_1) + C(\mu_2)$$

  $C(\mu_1, \mu_2, \ldots, \mu_k)$ = total squared distance of each data point $\vec{x}_i$ to its closest centroid μ_j

- Note that an equivalent way to write inertia is

  $C(\mu_1, \mu_2, \ldots, \mu_k) = C(\mu_1) + C(\mu_2) + \ldots + C(\mu_k)$ where

  $C(\mu_j)$ = total squared distance of each data point $\vec{x}_i$ in group j to centroid μ_j

- What's the point?

Why does k-Means work? (Step 3)

$$C(\mu_1, \mu_2, \ldots, \mu_k) = C(\mu_1) + C(\mu_2) + \ldots + C(\mu_k) \quad \text{where}$$
$$C(\mu_j) = \text{total squared distance of each data point } \vec{x}_i$$
$$\text{in group } j \text{ to centroid } \mu_j$$

Step 3: Keep the groups fixed, and update the centroids by moving each centroid to the center of its group (by averaging coordinates).

- Let's argue why this minimizes $C(\mu_j)$, for each group j.

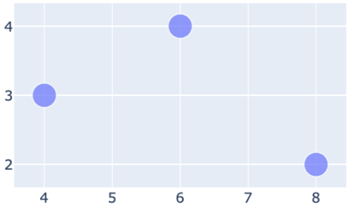
Why does k-Means work? (Step 3)

$C(\mu_1, \dots, \mu_k) = C(\mu_1) + \dots + C(\mu_k)$

$C(\mu_j)$ = total squared distance of each data point $\vec{x}_i$ in group j to centroid μ_j

Suppose group j contains the points $(4, 3)$, $(6, 4)$, and $(8, 2)$.

Where should we put $\mu_j = \begin{bmatrix} a \\ b \end{bmatrix}$ to minimize $C(\mu_j)$?



j

$$C(a,b) = (4-a)^2 + (3-b)^2 + (6-a)^2 + (4-b)^2 + (8-a)^2 + (2-b)^2$$

$$a = \frac{4+6+8}{3} \qquad b = \frac{3+4+2}{3}$$

Why does k-Means work? (Step 3)

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - h)^2 \longrightarrow h = \text{Mean}(x_i\text{'s})$$

$$C(a,b) = (4-a)^2 + (3-b)^2 + (6-a)^2 + (4-b)^2 + (8-a)^2 + (2-b)^2$$

$$\frac{\partial C}{\partial a} = 2(4-a)(-1) + 2(6-a)(-1) + 2(8-a)(-1)$$

$= 0$

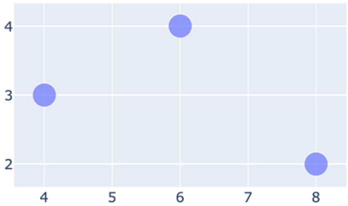
$\Rightarrow$ solve for a

Why does k-Means work? (Step 3)

$C(\mu_j)$ = total squared distance of each data point $\vec{x}_i$ in group j to centroid μ_j

Suppose group j contains the points $(4,3)$, $(6,4)$, and $(8,2)$.

Where should we put $\mu_j = \begin{bmatrix} a \\ b \end{bmatrix}$ to minimize $C(\mu_j)$?



Cost and empirical risk

- On the previous slide, we saw a function of the form

$$\begin{aligned} C(\mu_j) = C(a, b) &= (4 - a)^2 + (3 - b)^2 \\ &+ (6 - a)^2 + (4 - b)^2 \\ &+ (8 - a)^2 + (2 - b)^2 \end{aligned}$$

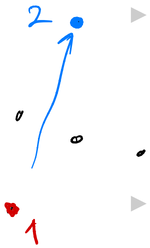
- $C(a, b)$ can be thought of as the sum of two separate functions, $f(a)$ and $g(b)$.
  - $f(a) = (4 - a)^2 + (6 - a)^2 + (8 - a)^2$ computes the total squared distance of each x_1 coordinate to a.
  - From earlier in the course, we know that $a^* = \frac{4+6+8}{3} = 6$ minimizes $f(a)$.

Practical considerations

Initialization

- Depending on our initial centroids, k-Means may "converge" to a clustering that doesn't actually have the lowest possible inertia.
  - In other words, like gradient descent, k-Means can get caught in a **local minimum**.

- Some solutions:
  - Run k-Means several times, each with different randomly chosen initial centroids. Keep track of the inertia of the final result in each attempt. Choose the attempt with the lowest inertia.



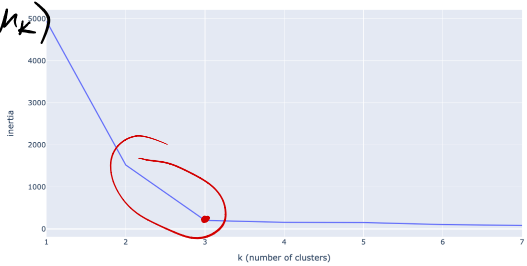
  - **k-Means++**: choose one initial centroid at random, and choose the remaining initial centroids by maximizing distance from all other centroids.

Choosing k

- Note that as k increases, inertia decreases.
  - Intuitively, as we add more centroids, the distance between each point and its closest centroid will drop.
- But the goal of clustering is to put data points into groups, and having a large number of groups may not be meaningful.
- This suggests a tradeoff between k and inertia.

The “elbow” method

- Strategy: run k-Means Clustering for many choices of k (e.g. $k = 1, 2, 3, \dots, 8$).
- Compute the value of inertia for each resulting set of centroids.
- Plot a graph of inertia vs k.
- Choose the value of k that appears at an “elbow”.



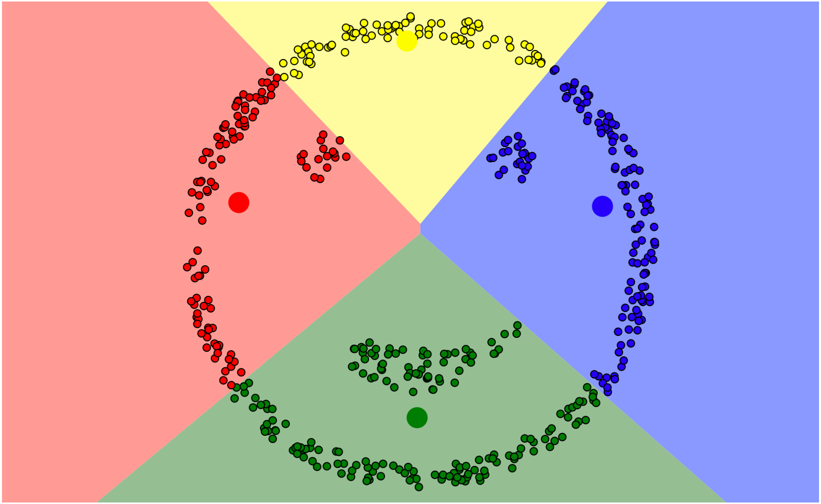
See the notebook for a demo.

Low inertia isn't everything!

- Even if k-Means works as intended and finds the choice of centroids that minimize inertia, the resulting clustering may not look "right" to us humans.
  - Recall, inertia measures the total squared distance to centroids.

  - This metric doesn't always match our intuition.

- Let's look at some examples at `https://tinyurl.com/40akmeans`.
  - Go to "I'll Choose" and "Smiley Face". Good luck!



Other clustering techniques

- k-Means Clustering is just one way to cluster data.
- There are many others, each of which work differently and produce different kinds of results.
- Another common technique: **agglomerative clustering**.
  - High level: start out with each point being in its own cluster. Repeatedly combine clusters until only k are left.
- Check out **this chart**.

Summary, next time

Summary

- k-Means Clustering attempts to minimize inertia.
  - We showed that it minimizes inertia on each step, but it's possible that it converges to a local minimum.
  - Different initial centroids can lead to different clusterings.
- To choose k, the number of clusters, we can use the elbow method.
- Other clustering techniques may work better than k-Means Clustering in certain cases.
- Outcomes, sample spaces, and events are the "building blocks" of probability.

Next time

- A deep-dive on the fundamentals rules of probability.
- **Important:** We've posted **many** probability resources on the resources tab of the course website. These will no doubt come in handy.
  - No more DSC 40A-specific readings.