CSE/STAT 416

Clustering

Vinitra Swamy
University of Washington
July 27, 2020

Recap

Last week, we introduced document retrieval and the algorithms associated with it.

We discussed the k-Nearest Neighbor algorithm and ways of representing text documents / measuring their distances.

We talked about how to use k-NN for classification/regression. We extended this to include other notions of distances by using weighted k-NN and kernel regression.

We also talked about how to efficiently find the set of nearest neighbors using Locality Sensitive Hashing (LSH).

Clustering

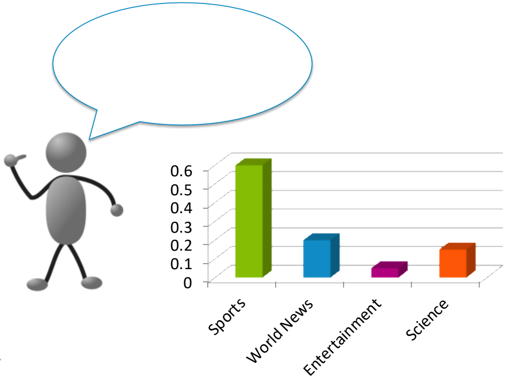
SPORTS

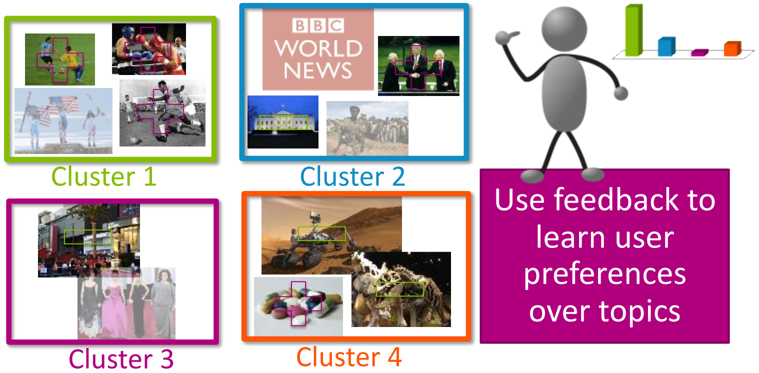
WORLD NEWS

Recommending News

User preferences are important to learn, but can be challenging to do in practice.

- People have complicated preferences
- Topics aren't always clearly defined

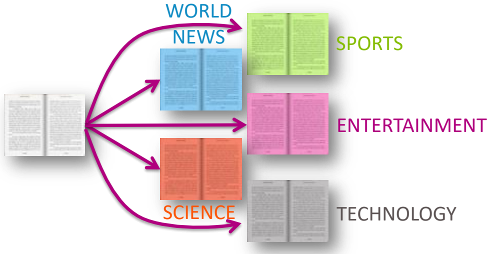




Labeled Data

What if the labels are known? Given labeled training data

Can do multi-class classification methods to predict label



Unsupervised Learning

- In many real world contexts, there aren't clearly defined labels so we won't be able to do classification
- We will need to come up with methods that uncover structure from the (unlabeled) input data X.
- **Clustering** is an automatic process of trying to find related groups within the given dataset.

$Input: x_1, x_2, \ldots, x_n$

$Output: z_1, z_2, \ldots, z_n$

Define Clusters

In their simplest form, a **cluster** is defined by

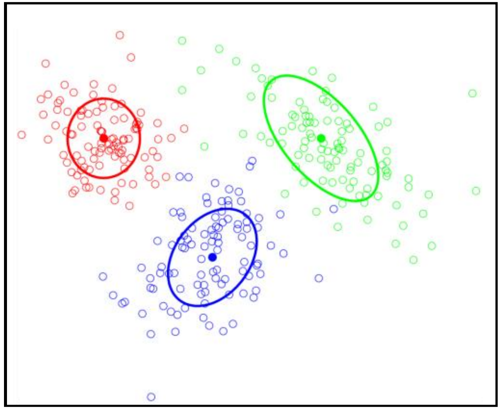
- The location of its center (**centroid**)
- Shape and size of its **spread**

Clustering is the process of finding these clusters and **assigning** each example to a particular cluster.

- x_i gets assigned $z_i \in [1, 2, \ldots, k]$
- Usually based on closest centroid

Will define some kind of score for a clustering that determines how good the assignments are

- Based on distance of assigned examples to each cluster

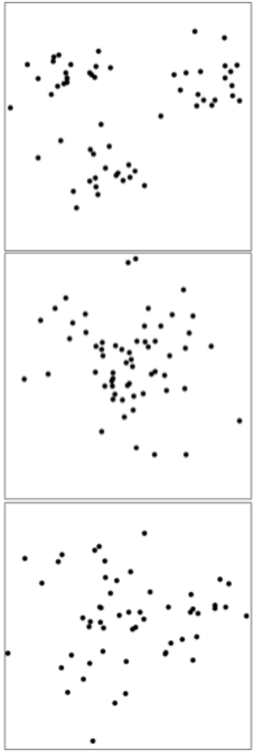


When This Works

Clustering is easy when distance captures the clusters

Ground Truth (not visible)

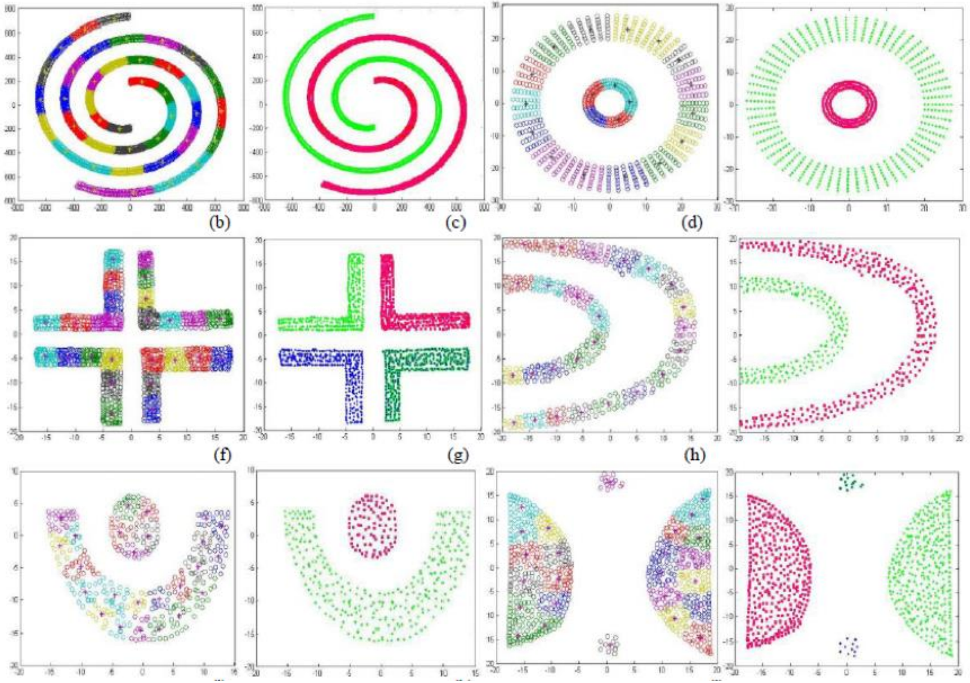
Given Data



Not Always Easy

There are many clusters that are harder to learn with this setup

- Distance does not determine clusters



k-means

Algorithm

Will define the Score for assigning a point to a cluster is

$$Score(x_i, \mu_j) = dist(x_i, \mu_j)$$

Lower score => Better Clustering

k-means Algorithm at a glance

Step 0: Initialize cluster centers

Repeat until convergence:

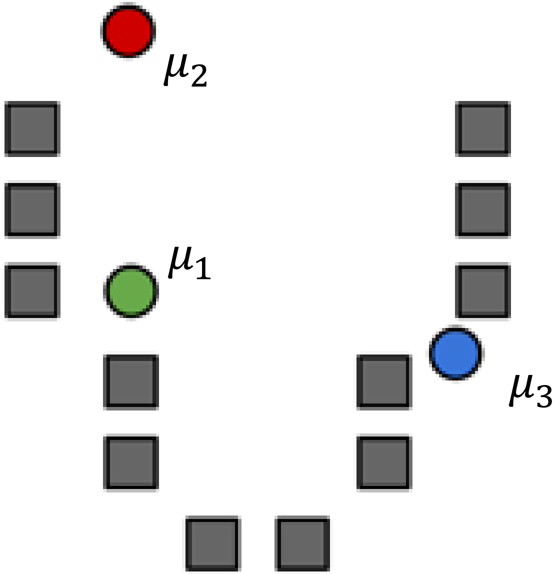
Step 1: Assign each example to its closest cluster centroid

Step 2: Update the centroids to be the average of all the points assigned to that cluster

Step 0

Start by choosing the initial cluster centroids

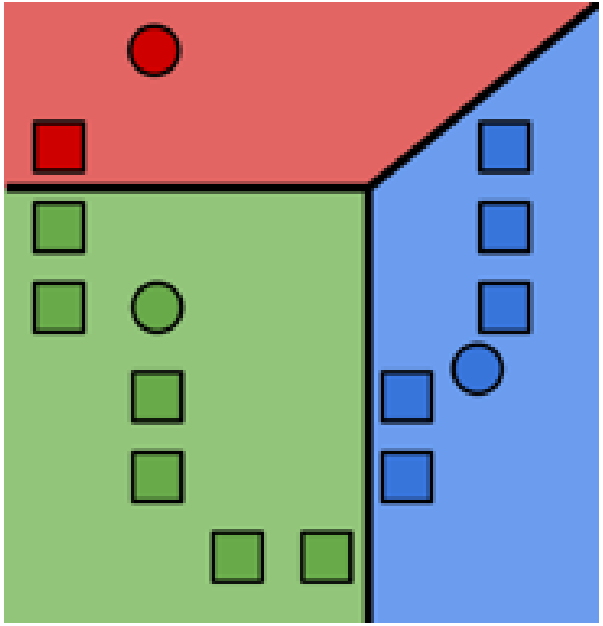
- A common default choice is to choose centroids at random
- Will see later that there are smarter ways of initializing



Step 1

Assign each example to its closest cluster centroid

$$z_i \leftarrow \underset{j \in [k]}{\operatorname{argmin}} \left|\left|\mu_j - x_i\right|\right|^2$$

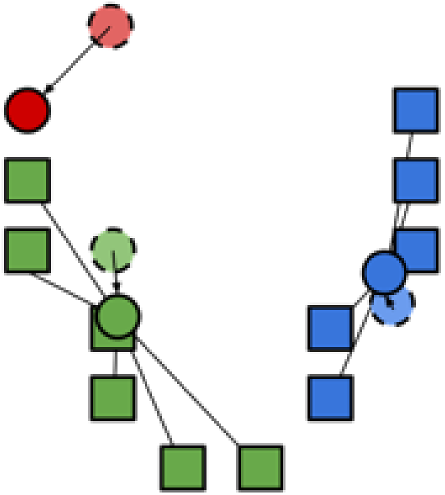


Step 2

Update the centroids to be the mean of all the points assigned to that cluster.

$$\mu_j \leftarrow \frac{1}{n_j} \sum_{i:z_i=j} x_i$$

Computes center of mass for cluster!



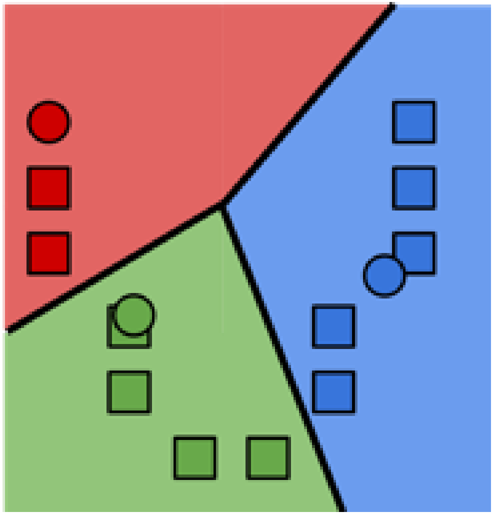
Repeat

Repeat Steps 1 and 2 until convergence

Will it converge? Yes!

What will it converge to?

- Global optimum
- Local optimum
- Neither

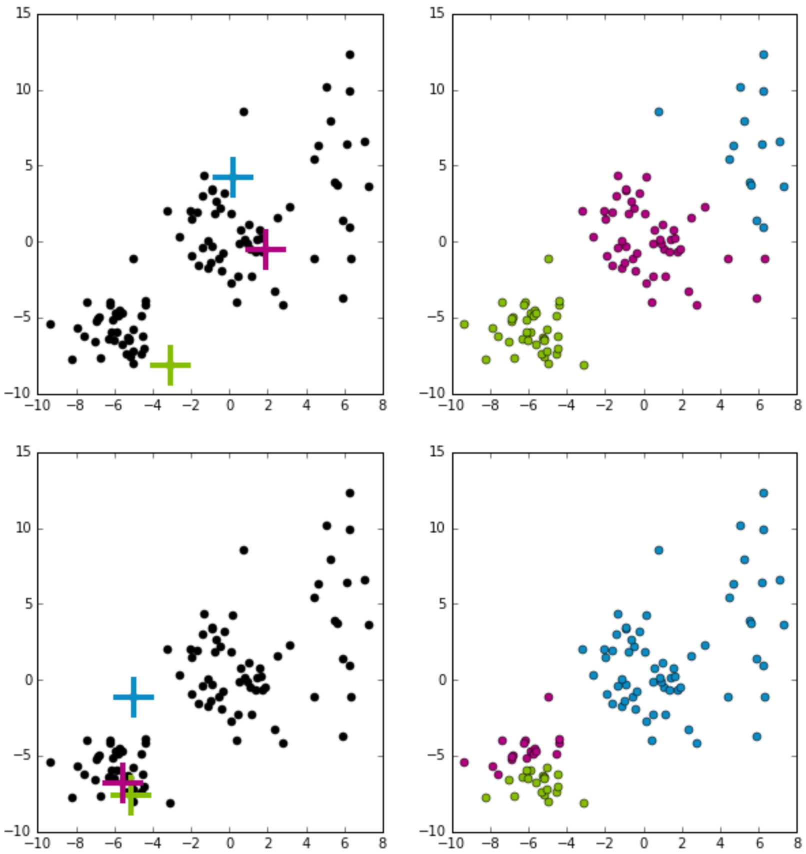


Brain Break

Local Optima

What does it mean for something to converge to a local optima?

- Initial settings will greatly impact results!



Smart Initializing w/ k-means++

Making sure the initialized centroids are "good" is critical to finding quality local optima. Our purely random approach was wasteful since it's very possible that initial centroids start close together.

Idea: Try to select a set of points farther away from each other.

k-means++ does a slightly smarter random initialization

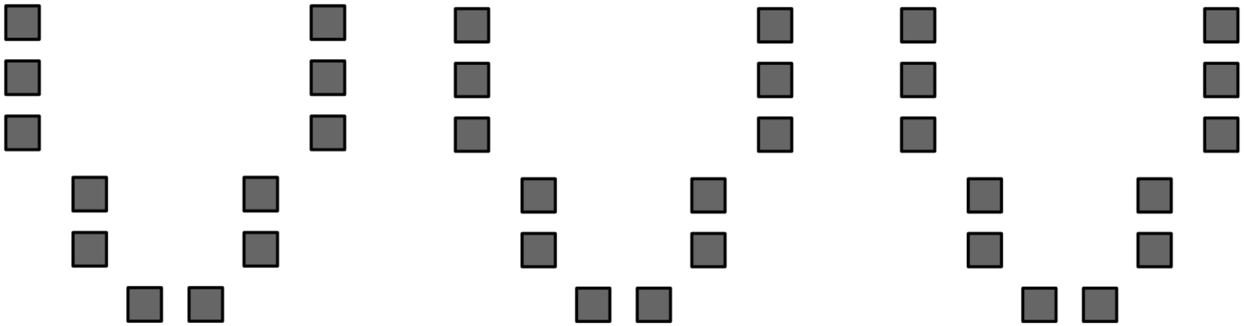
1. Choose first cluster μ_1 from the data uniformly at random
2. For the current set of centroids (starting with just μ_1), compute the distance between each datapoint and its closest centroid
3. Choose a new centroid from the remaining data points with probability of x_i being chosen proportional to $d(x_i)^2$
4. Repeat 2 and 3 until we have selected k centroids

k-means++ Example

Start by picking a point at random

Then pick points proportional to their distances to their centroids

This tries to maximize the spread of the centroids!



k-means++
Pros / Cons

Pros

- Improves quality of local minima
- Faster convergence to local minima

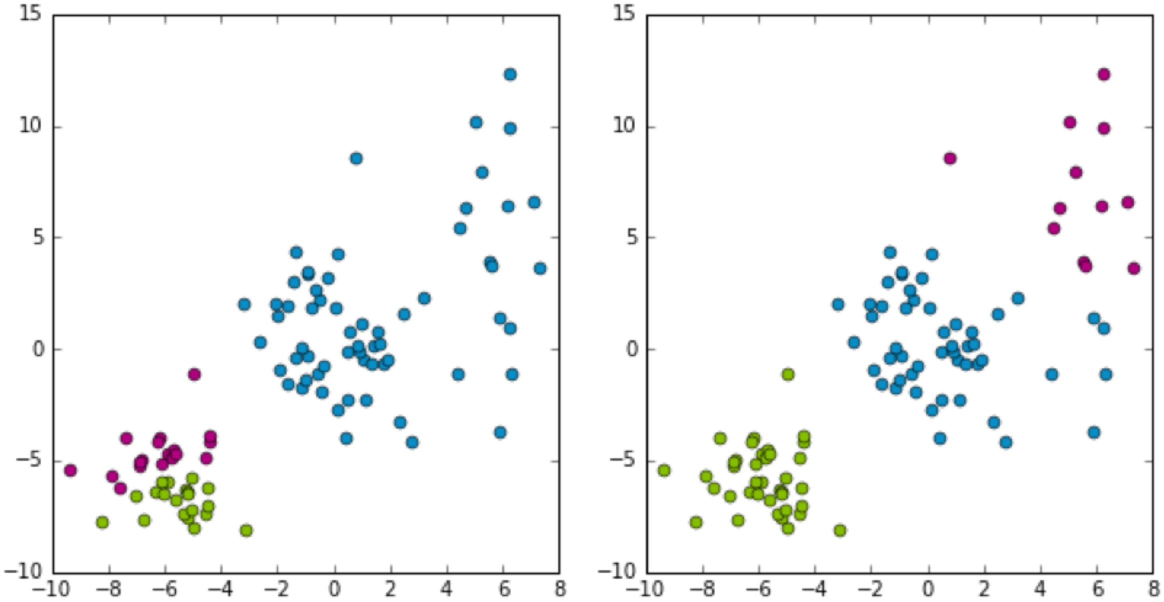
Cons

- Computationally more expensive at beginning when compared to simple random initialization

Assessing Performance

Which Cluster?

Which clustering would I prefer?



k-means is trying to optimize the **heterogeneity** objective

$$\sum_{j=1}^{k} \sum_{i:z_i=j} \left|\left|\mu_j - x_i\right|\right|_2^2$$

Coordinate Descent

k-means is trying to minimize the heterogeneity objective

$$\underset{\mu_1,\ldots,\mu_k,\, z_1,\ldots,\, z_n}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{i:z_i=j} \left|\left|\mu_j - x_i\right|\right|_2^2$$

Step 0: Initialize cluster centers

Repeat until convergence:

Step 1: Assign each example to its closest cluster centroid

Step 2: Update the centroids to be the mean of all the points assigned to that cluster

Coordinate Descent alternates how it updates parameters to find minima. On each of iteration of Step 1 and Step 2, heterogeneity decreases or stays the same.

=> Will converge in finite time

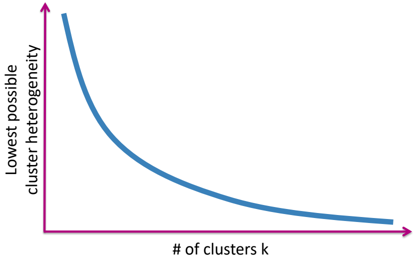
Poll Everywhere

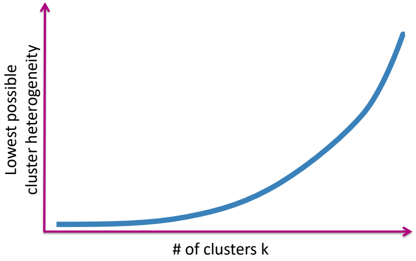
Think

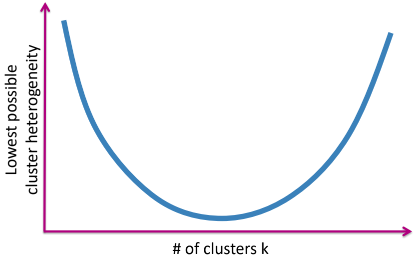
1 min

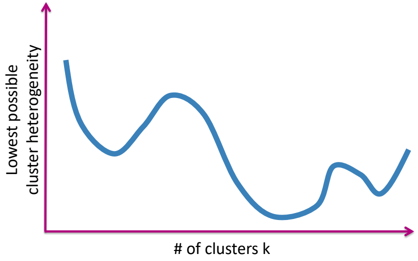
pollev.com/cse416

Consider trying k-means with different values of k. Which of the following graphs shows how the globally optimal heterogeneity changes for each value of k?









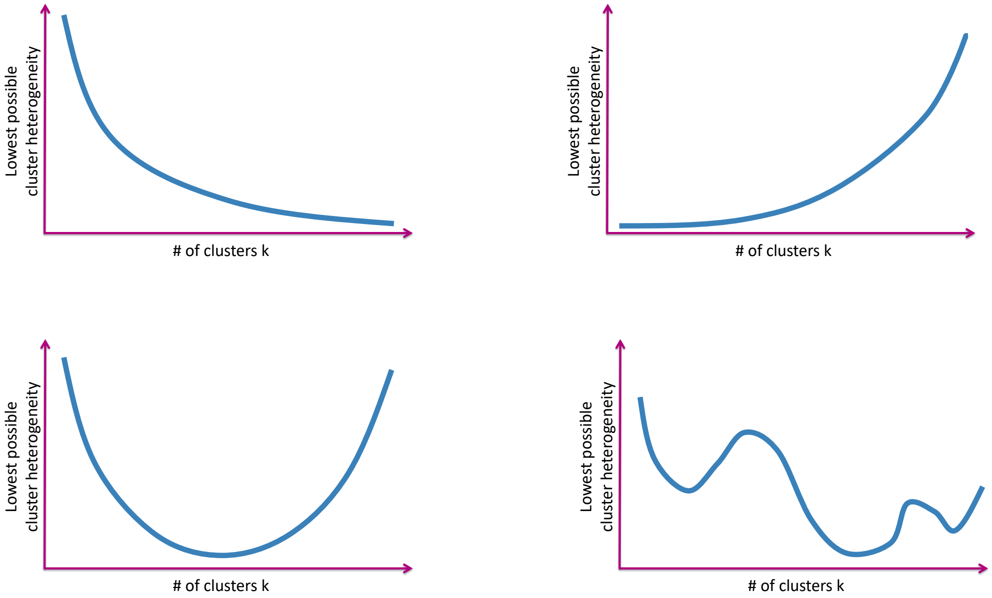
Poll Everywhere

Think

2.5 min

pollev.com/cse416

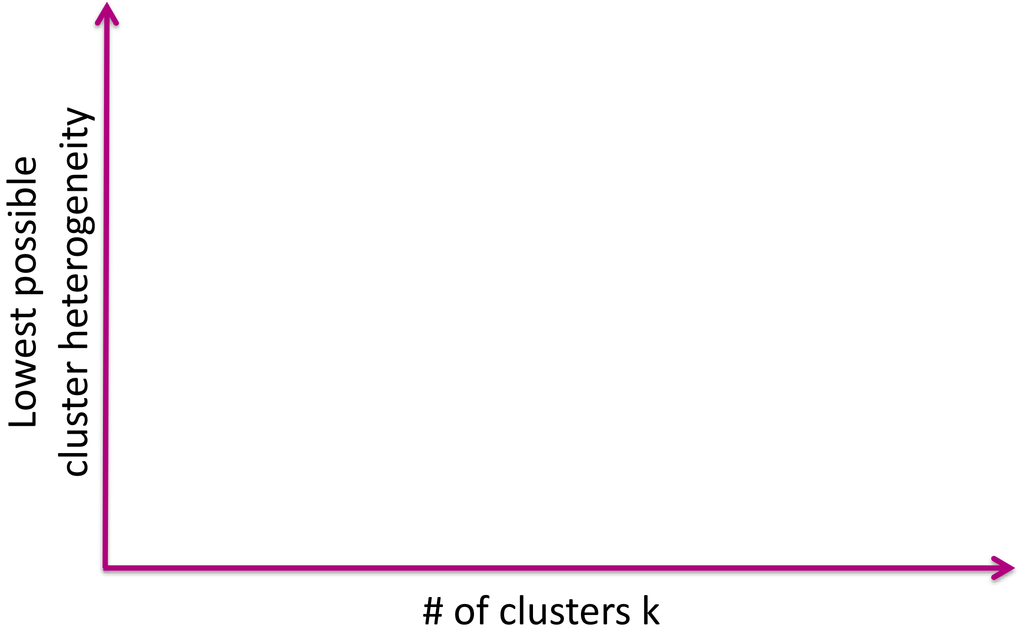
Consider trying k-means with different values of k. Which of the following graphs shows how the globally optimal heterogeneity changes for each value of k?



How to Choose k?

No right answer! Depends on your application.

- General, look for the “elbow” in the graph

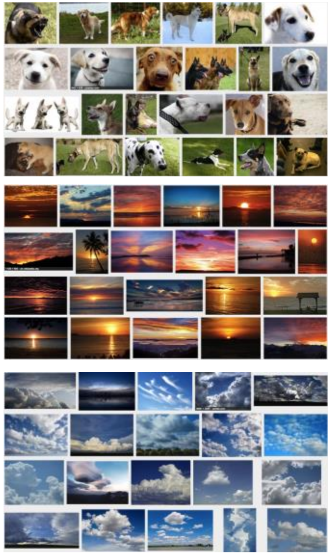


Brain Break

Other Applications

Clustering Images

- For search, group as:
  - Ocean
  - Pink flower
  - Dog
  - Sunset
  - Clouds
  - …



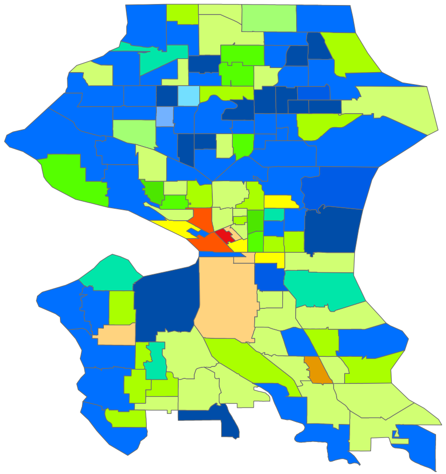
Result Diversity

- Search terms can have multiple meanings
- Example: "cardinal"

- Use clustering to structure output

Share Information

- Task 1: Estimate price at a small regional level
- Challenge:
  - Only a few (or no!) sales in each region per month
- Solution:
  - Cluster regions with similar trends and share information within a cluster



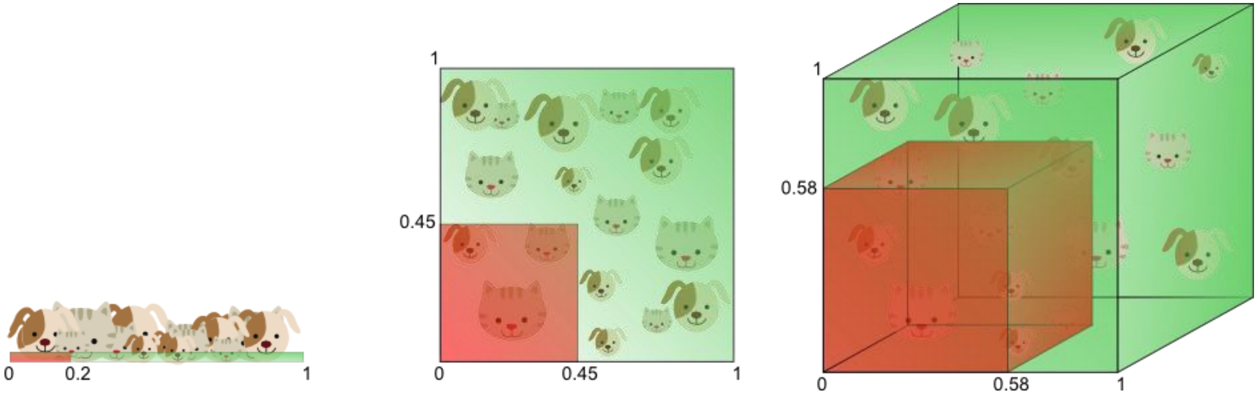
City of Seattle

Curse of Dimensionality

High Dimensions

Methods like k-NN and k-means that rely on computing distances start to struggle in high dimensions.

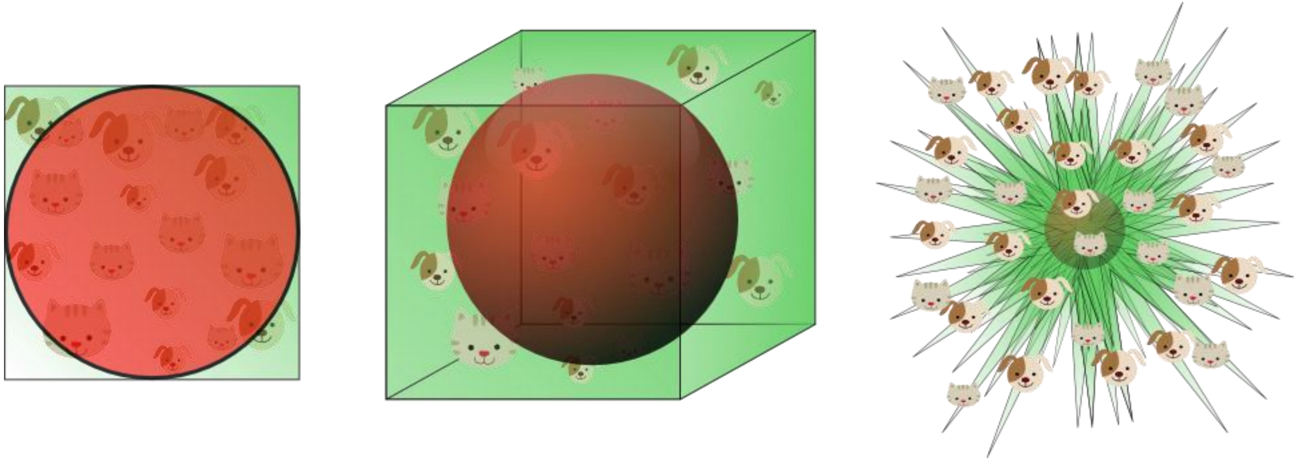
As the number of dimensions grow, the data gets sparser!



Need more data to make sure you cover all the space in high dim.

Even Weirder

It’s believable with more dimensions the data becomes more sparse, but what’s even weirder is the sparsity is not uniform!



As D increases, the “mass” of the space goes towards the corners.

- Most of the points aren’t in the center.
- Your nearest neighbors start looking like your farthest neighbors!

Practicalities

Have you pay attention to the number of dimensions

- Very tricky if $n < D$
- Can run into some strange results if D is very large

Later, we will talk about ways of trying to do dimensionality reduction in order to reduce the number of dimensions here.