CSE/STAT 416

K-Means Clustering

Amal Nanavati
University of Washington
Aug 1, 2022

Adapted from Hunter Schafer's slides

Administrivia

- We have now finished out study on supervised learning!
- **This Week**: Clustering with text data
- **Next Week**: Dimensionality Reduction, Recommender Systems
- **Next-Next Week**: Course Wrap-Up & Final
- Deadlines:
  - HW5 due TOMORROW, Tues 8/2 11:59PM
    - Submit Concept & Programming on Gradescope
  - HW6 Released Wed 8/3
  - LR 7 due Fri 8/5 11:59PM
- Notes on the end of the quarter
  - Guest Panel Extra Credit: Mon 8/15 (during lecture)
  - HW7 due Tues 8/16, **NO LATE DAYS**
  - Take-Home Final Exam: Wed 8/17 – Thurs 8/18

Addressing LR Questions

Poll Everywhere

Think

3 min

What is the result of applying a convolution using this kernel on this input image?

Use 1x1 zero padding and a 2x2 stride

Result: 3x3

Image

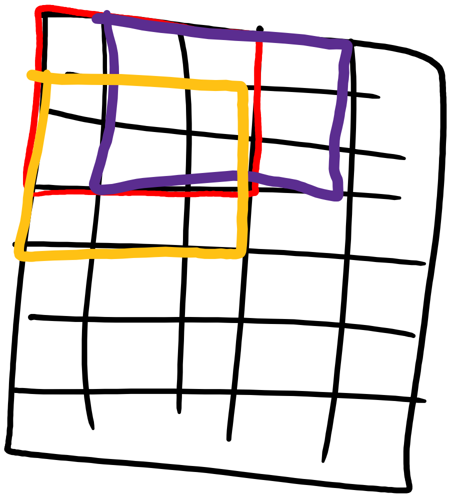
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Kernel

1	1
0	2

$$= \begin{pmatrix} 2 & 6 & 0 \\ 23 & 35 & 8 \\ 13 & 29 & 16 \end{pmatrix}$$

Kernel 3 x 3, stride of 1



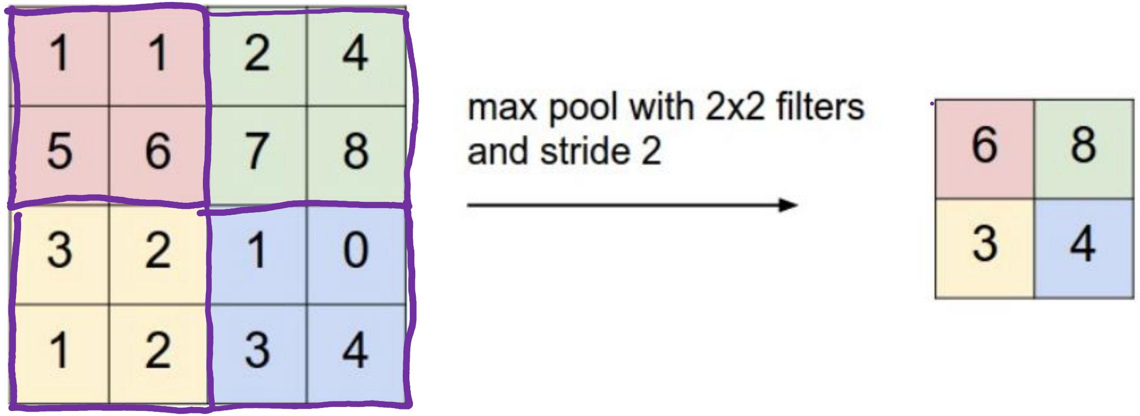
Pooling

Another core operation that is similar to a convolution is a **pool**.

- Idea is to down sample an image using some operation
- Combine local pixels using some operation (e.g. max, min, average, median, etc.)

Typical to use **max pool** with 2x2 filter and stride 2

- Tends to work better than average pool



Total: $250 + 5000 + 27{,}720 = \boxed{32{,}970}$ << 66k!

Weight Sharing

In a conv layer w/ a $k \times k$ kernel, I input channels, and d output channels, the num param to be learnt are

$$k \cdot k \cdot I \cdot d$$

Consider solving a digit recognition task on 28x28 images. Suppose I wanted to use a fully connected hidden layer with 84 neurons

With Convolutions (assume n1=10, n2=20)

inputs flattened:

$4 \cdot 4 \cdot 20 = 320$

FC component:

$320 \cdot 84 + 84 \cdot 10 = 27{,}720$

Conv2:

$5 \cdot 5 \cdot 10 \cdot 20 = 5000$

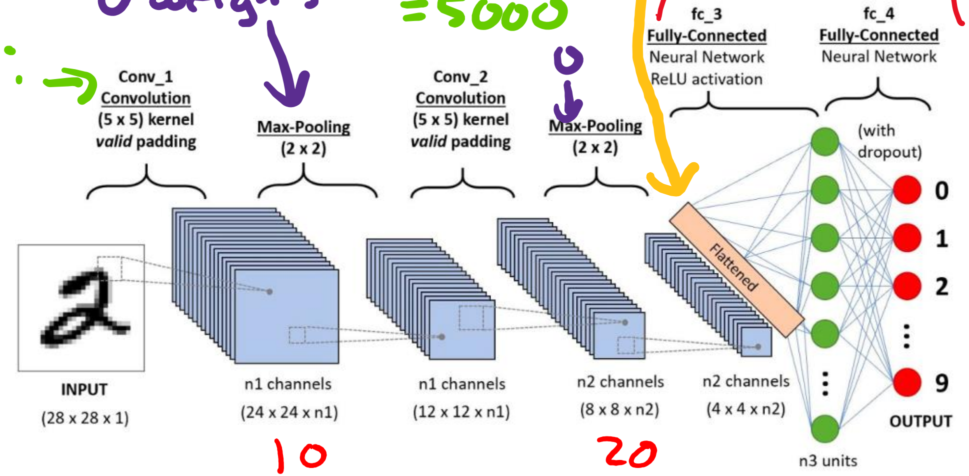
MaxPool:

0 weights

0

Conv1:

$5 \cdot 5 \cdot 10 = 250$



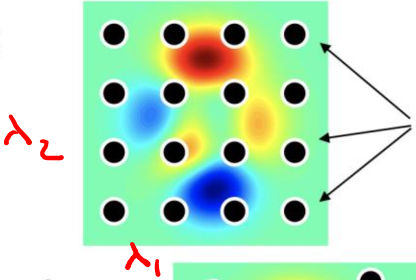
10

20

Hyperparameter Optimization

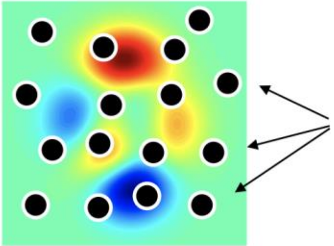
How do we choose hyperparameters to train and evaluate?

Grid search:



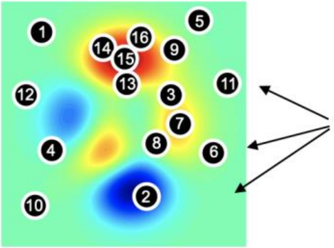
Hyperparameters on 2d uniform grid

Random search:



Hyperparameters randomly chosen

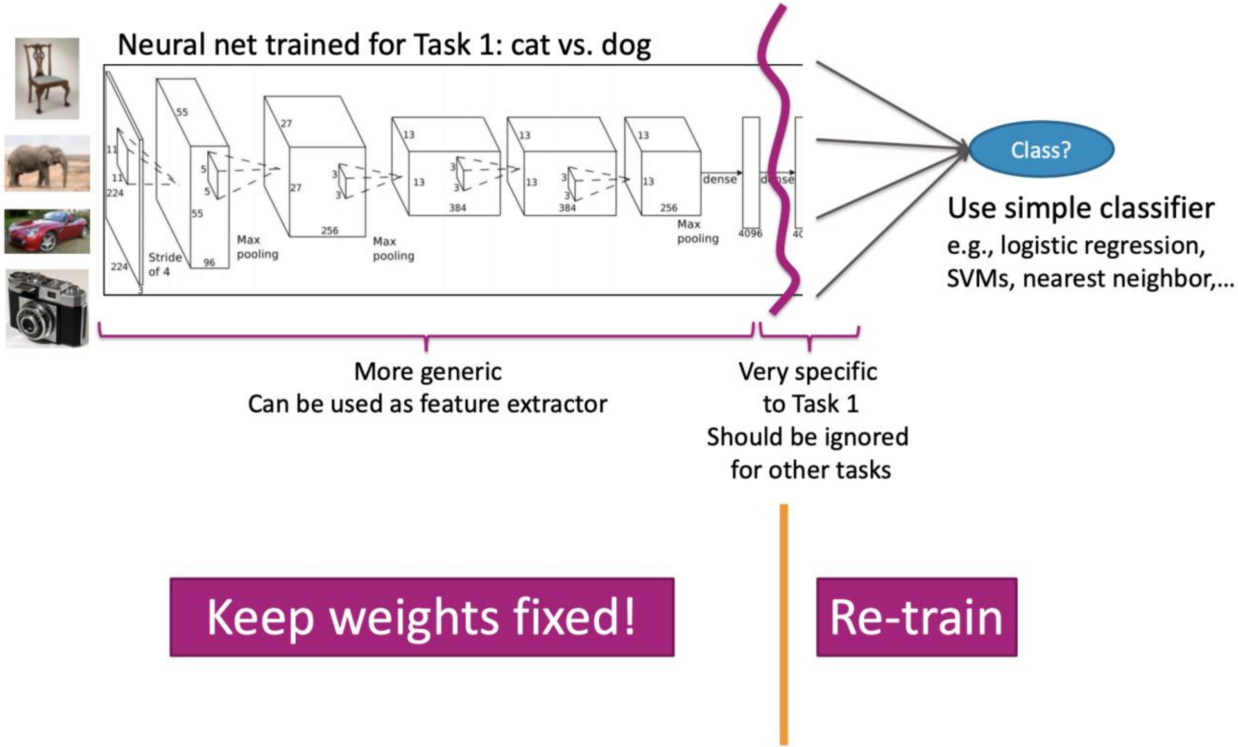
Bayesian Optimization:



Hyperparameters ***adaptively*** chosen

Transfer Learning

Share the weights for the general part of the network



Clustering Overview

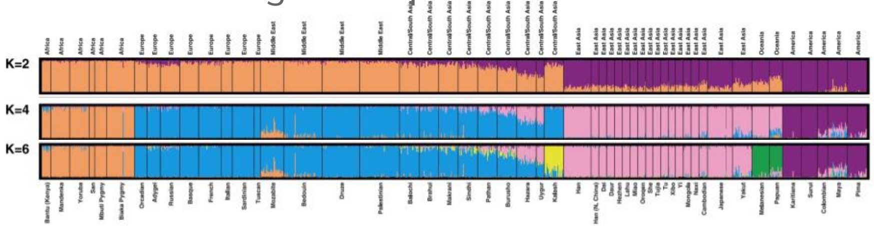
Recap

- For the past 6 weeks, we have covered different **supervised learning** algorithms
- Now, we're going to explore **unsupervised learning** methods where don't have labels / outputs in your datasets anymore.
- Note that several of the concepts you learnt for supervised learning, such as cross-validation, overfitting, bias-variance tradeoff, accuracy, error, etc. no longer apply in unsupervised learning!

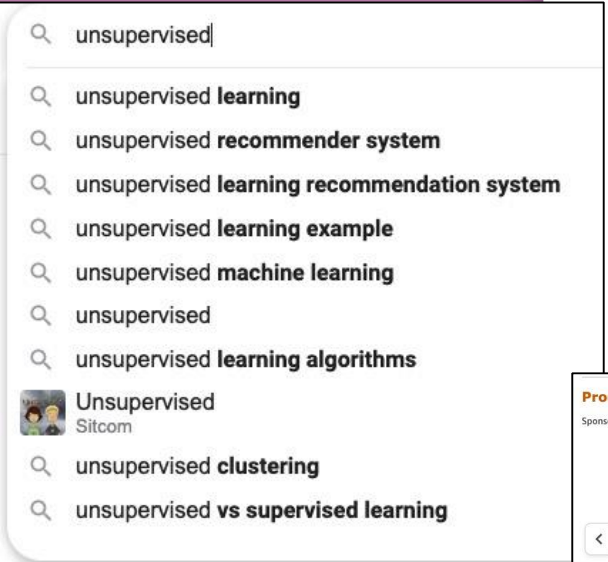
Unsupervised Learning

- A type of machine learning that detects underlying patterns in **unlabeled** data.
- Examples of unsupervised learning tasks:
  - Cluster similar articles together.

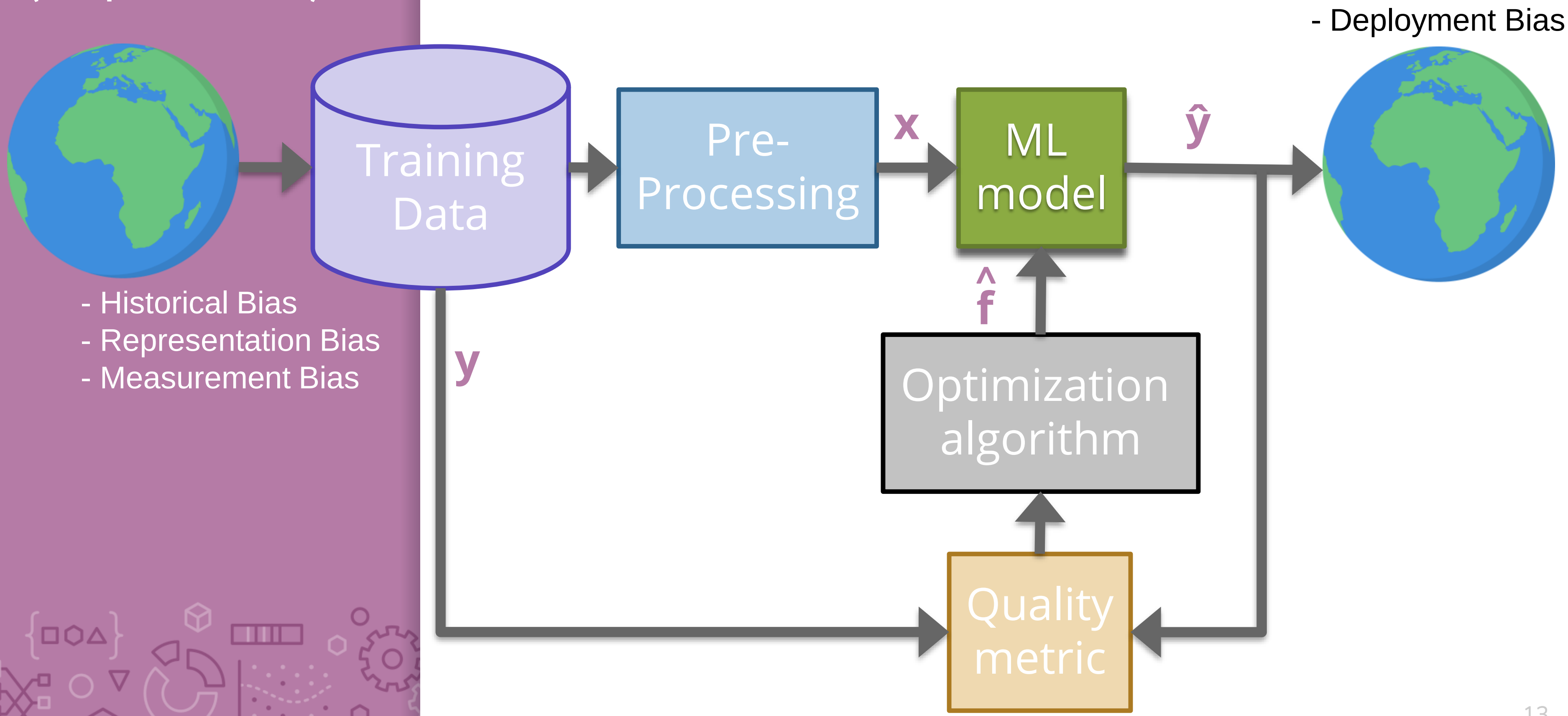
  - Cluster gene sequences.



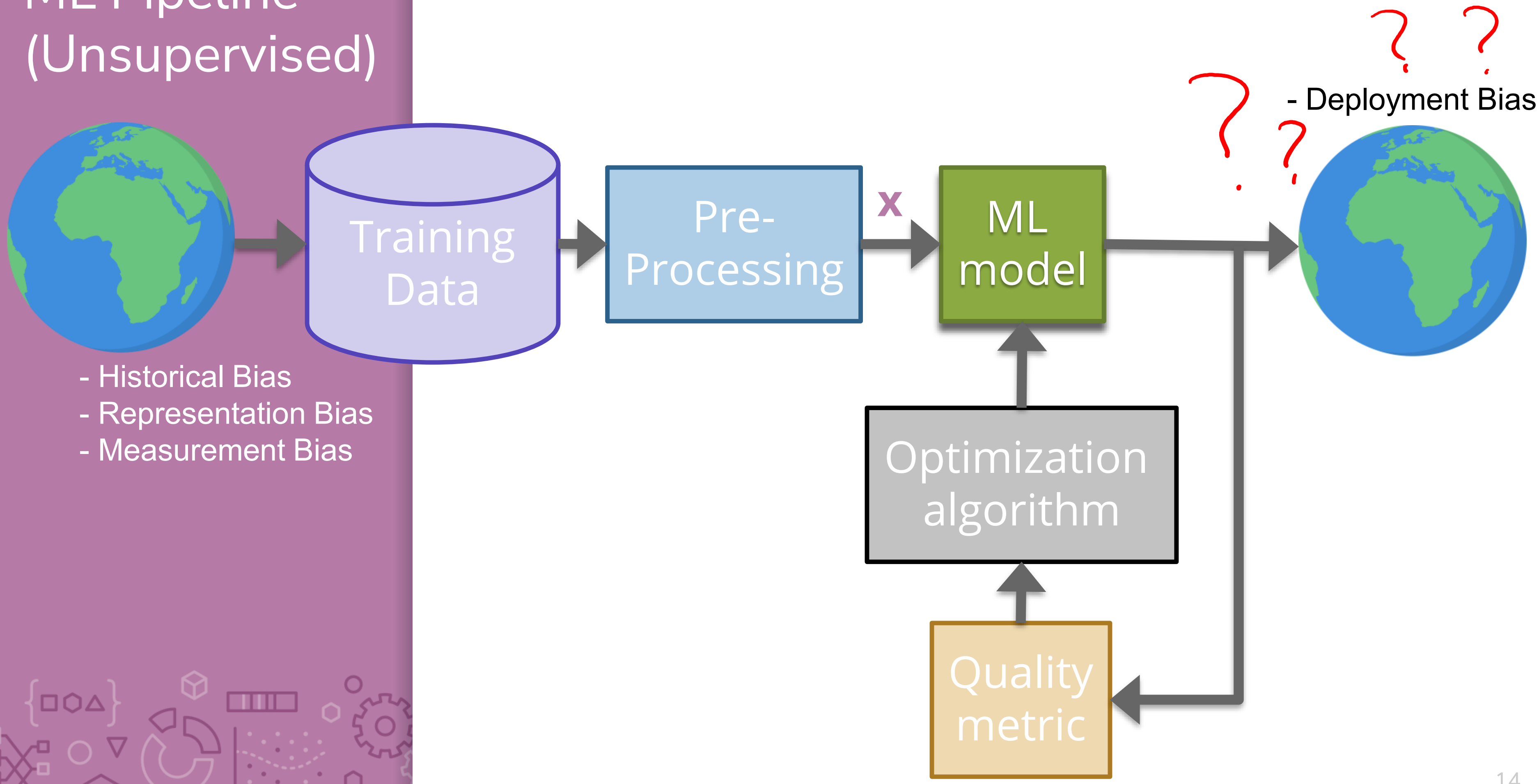
  - Recommend items, searches, movies, etc.



ML Pipeline (Supervised)



ML Pipeline (Unsupervised)



- Historical Bias
- Representation Bias
- Measurement Bias

Clustering

SPORTS

WORLD NEWS

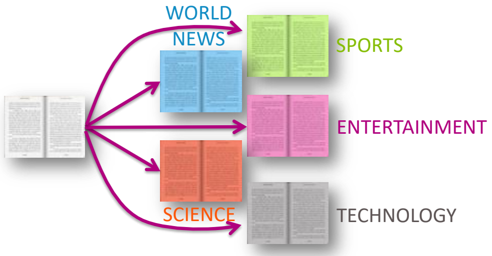
Note that we're not talking about learning user preferences (yet – come back next week ☺).

Our case study is **document retrieval**. Given that someone read a particular article, what similar articles would you recommend (without personalization)?

Labeled Data

What if the labels are known? Given labeled training data.

Can do multi-class classification methods to predict label.

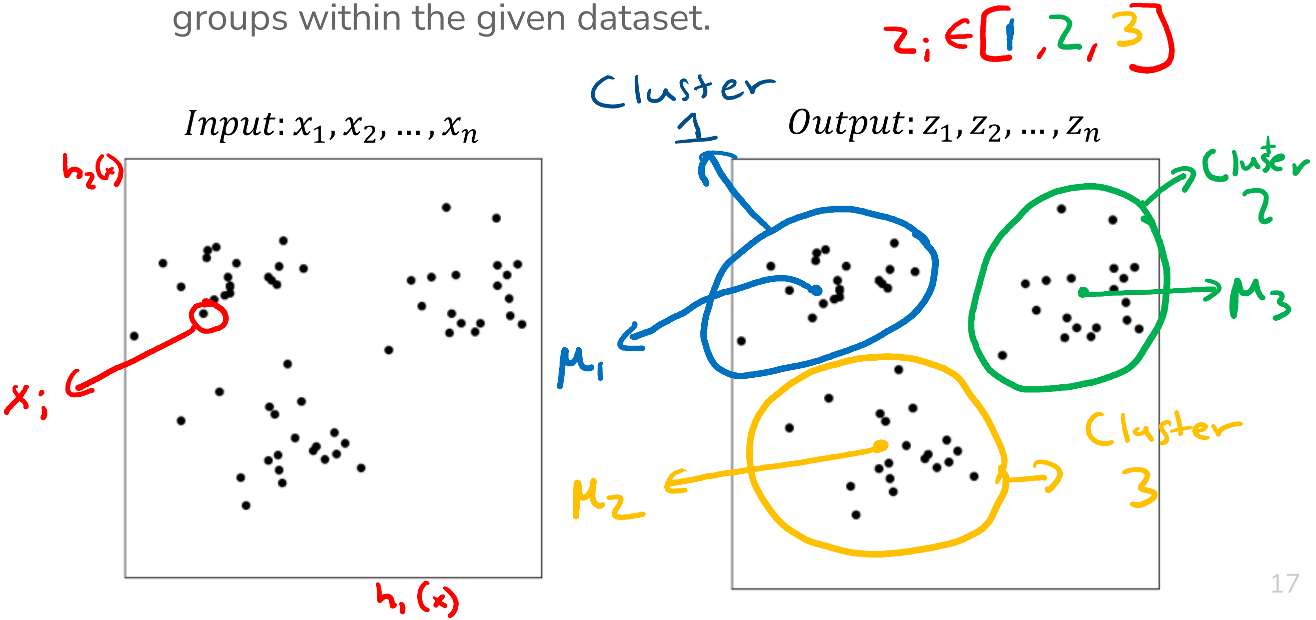


However, not all articles fit cleanly into one label.

Further, oftentimes real-world data doesn't have labels.

Unlabeled Data

- 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.



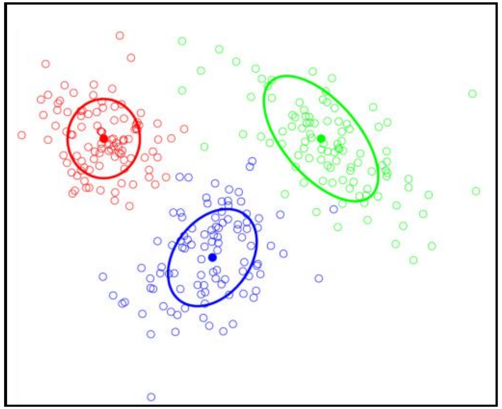
Define Clusters

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 objective function for a clustering that determines how good the assignments are

- Based on distance of assigned examples to each cluster.
- Close distance reflects strong similarity between datapoints.

When Might This Work?

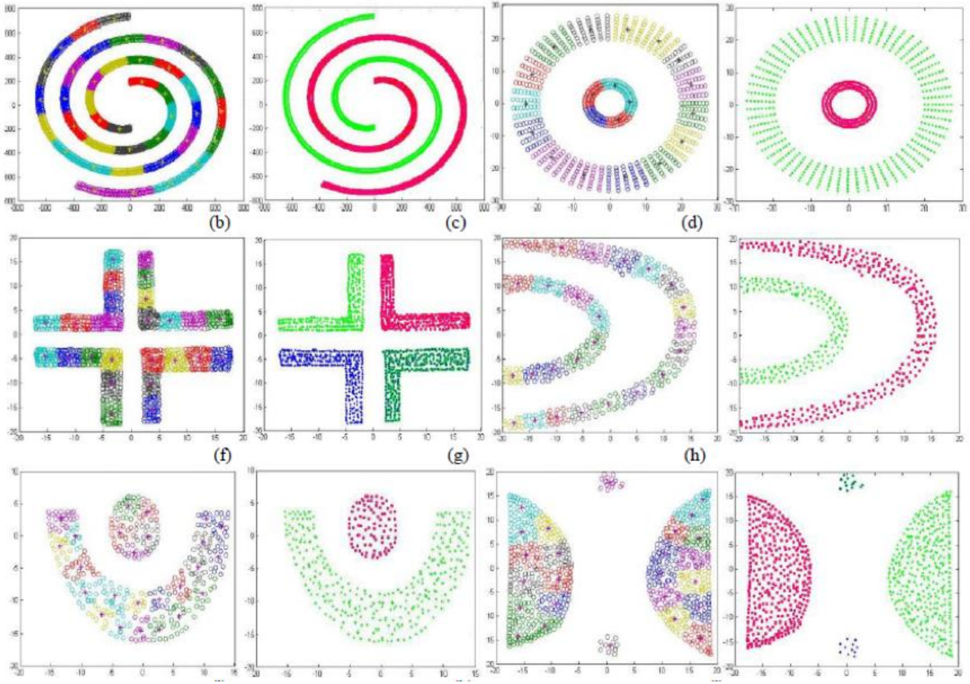
Clustering is easy when distance captures the clusters.

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



Poll Everywhere

Think

1 min

- Think of 1-2 problems that you might want to use clustering for. For each problem, describe:
  - Why unsupervised learning is the right approach.
  - What the input features are for the clustering algorithm.
  - What clusters you hypothesize would emerge.

Poll Everywhere

Group

2 min

- Think of 1-2 problems that you might want to use clustering for. For each problem, describe:
  - Why unsupervised learning is the right approach.
  - What the input features are for the clustering algorithm.
  - What clusters you hypothesize would emerge.

Embedding Text Data Revisited

TF-IDF

Converting Text to Numbers (Vectorizing):

Bag of Words

- **Idea**: One feature per word!

Example: "Sushi was great, the food was awesome, but the service was terrible"

sushi	was	great	the	food	awesome	but	service	terrible
1	3	1	2	1	1	1	1	1

This **has** to be too simple, right?

- Stay tuned (today and Wed) for issues that arise and how to address them ☺

Bag of Words

Pros

- Very simple to describe
- Very simple to compute

Cons

- Common words like “the” and “a” dominate counts of uncommon words
- Often it’s the uncommon words that uniquely define a doc.

TF-IDF (Term Frequency Inverse Document Frequency)

• Corpus of 1000 documents

• "the" occurs in doc 1 200 times

Goal: Emphasize important words

- Appear frequently in the document (common locally)

$TF(\text{"the"}, 1) = \frac{200}{\text{len}(\text{doc } 1)}$

TF= Term frequency = $\frac{\text{word counts}}{\text{num words}}$

- Appears rarely in the corpus (rare globally)

IDF= Inverse doc freq. = $\log \frac{\#\ \text{docs}}{1 + \#\ \text{docs using word}}$

"the" occurs in <u>all</u> docs at least once

↳ $IDF(\text{"the"}) = \log \frac{1000}{1001} \approx 0$

tf * idf

Do a pair-wise multiplication to compute the TF-IDF for each word

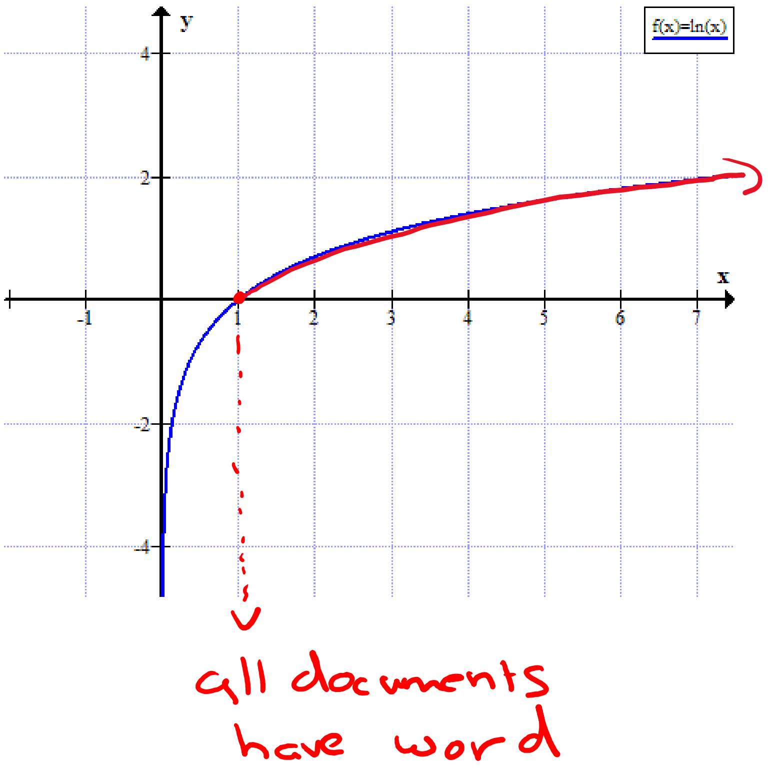
- Words that appear in every document will have a small IDF making the TF-IDF small!

Understanding IDF

IDF goes from 0, when all documents have the word, to log(# docs), when no docs have the word.

$$IDF = \log\left(\frac{\#\text{docs}}{1 + \#\text{docs w/ word}}\right)$$

Assume # docs is large



Recall the Bag of Words Example from Lecture 5

Review
“Sushi was great, the food was awesome, but the service was terrible”
“Terrible food; the sushi was rancid.”

Note that if we divide the Bag of Words embedding by the num words in the document, we get the TF!

Sushi	was	great	the	food	awesome	but	service	terrible	rancid
1	3	1	2	1	1	1	1	1	0
1	1	0	1	1	0	0	0	1	1

Poll Everywhere

Think

1 min

- Which word(s) have the largest IDF? Which word(s) have the smallest IDF?

Review
“Sushi was great, the food was awesome, but the service was terrible”
“Terrible food; the sushi was rancid.”

Note that if we divide the Bag of Words embedding by the num words in the document, we get the TF!

Sushi	was	great	the	food	awesome	but	service	terrible	rancid
1	3	1	2	1	1	1	1	1	0
1	1	0	1	1	0	0	0	1	1

Poll Everywhere

Group

2 min

- Which word(s) have the largest IDF? Which word(s) have the smallest IDF?

Red = low

Green = high

Review
"Sushi was great, the food was awesome, but the service was terrible"
"Terrible food; the sushi was rancid."

Note that if we divide the Bag of Words embedding by the num words in the document, we get the TF!

Sushi	was	great	the	food	awesome	but	service	terrible	rancid
1	3	1	2	1	1	1	1	1	0
1	1	0	1	1	0	0	0	1	1

Brain Break

3:29

Hyperparameter

centroid

K-Means Clustering

K-Means Clustering Algorithm

- We define the criterion of assigning point to a cluster based on **its distance.**
- Shorter distance => Better Clustering

Hyper parameter

Algorithm
Given a dataset of n datapoints and a particular choice of k

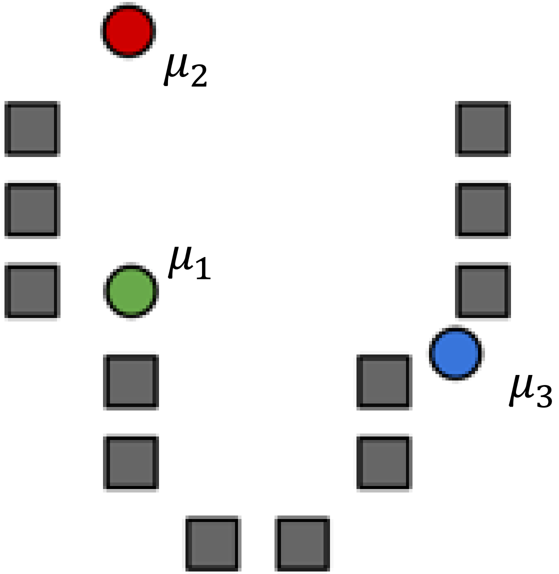
Step 0: Initialize cluster centroids randomly

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 $\mu_1, \ldots, \mu_k$ randomly
- Will see later that there are smarter ways of initializing



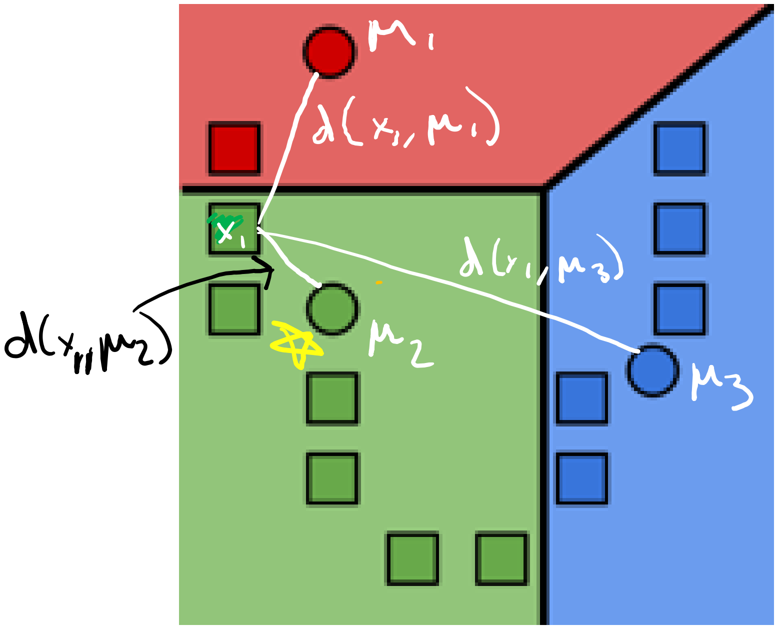
Voronoi Tesselation

Step 1

Assign each example to its closest cluster centroid

For $i = 1$ to n

$$z_i \leftarrow \underset{j \in [k]}{\operatorname{argmin}} \left|\left|\mu_j - x_i\right|\right|_2^2$$



Step 2

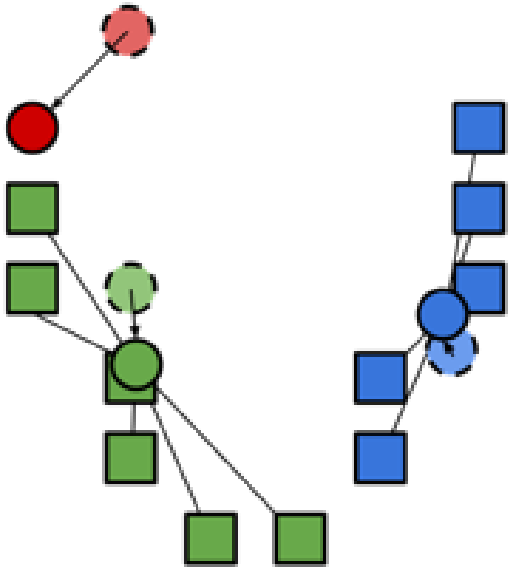
Update the centroids to be the mean of points assigned to that cluster.

$$\mu_j = \frac{\sum_{i=1}^{n} \mathbf{1}\{z_i = j\} x_i}{\sum_{i=1}^{n} \mathbf{1}\{z_i = j\}}$$

= sum of datapoints assigned to cluster j

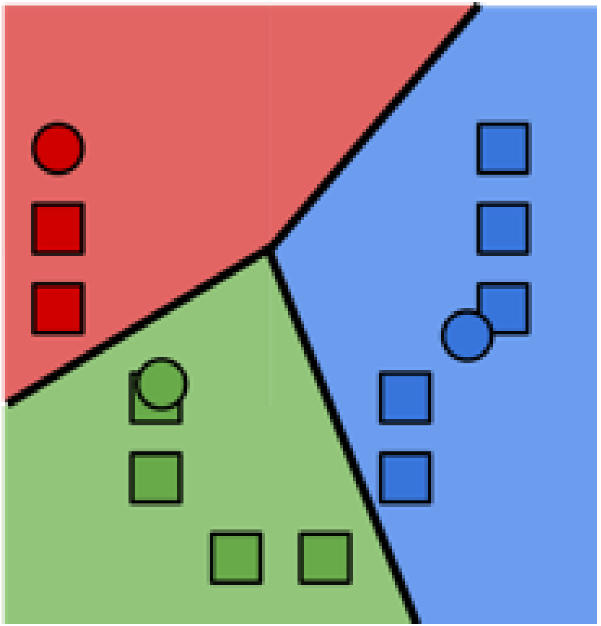
= number of datapoints assigned to cluster j

Computes center of mass for cluster!



Repeat until convergence

Repeat Steps 1 and 2 until convergence



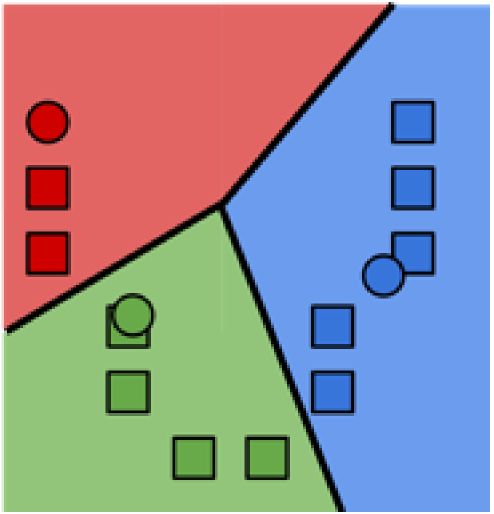
k-Means Stopping Condition

Stopping conditions

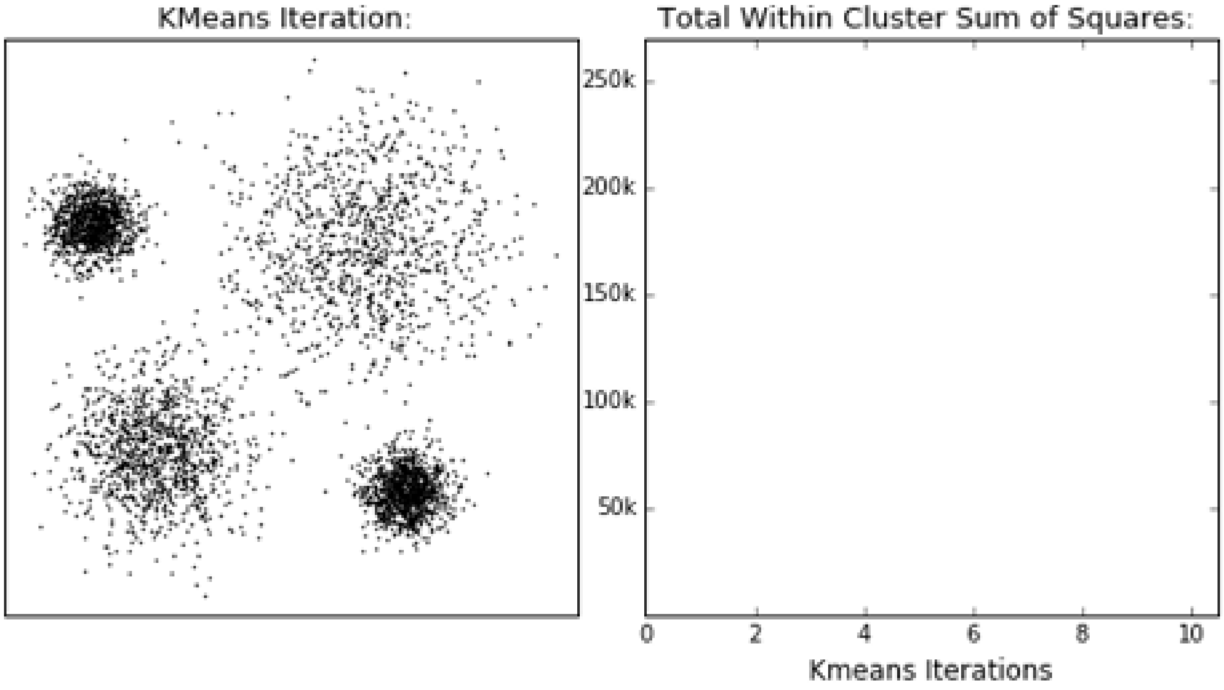
- Cluster assignments haven’t changed
- Centroids haven’t changed
- Some number of max iterations have been passed

What will it converge to?

- Local optima
- ~~Global optima~~
- ~~Neither~~



Visualizing k-Means



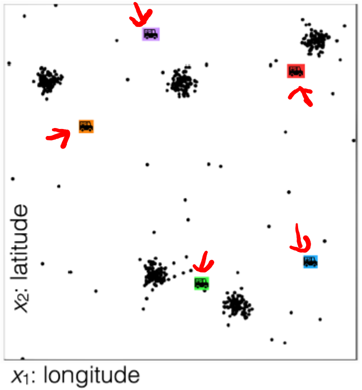
Poll Everywhere

Think

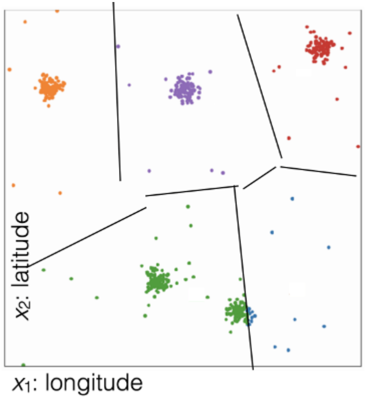
1 min

~~pollev.com/cs416~~

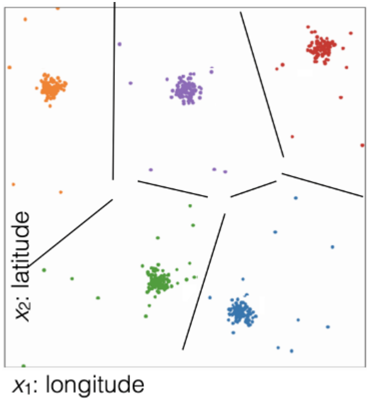
- What cluster assignment would result from these centroids?



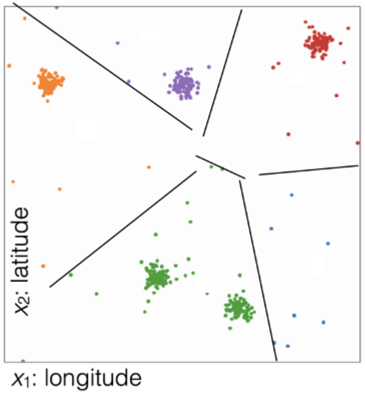
Centroids



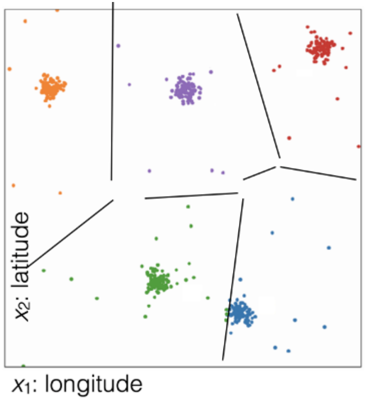
(D)



(A)



(B)



(C)

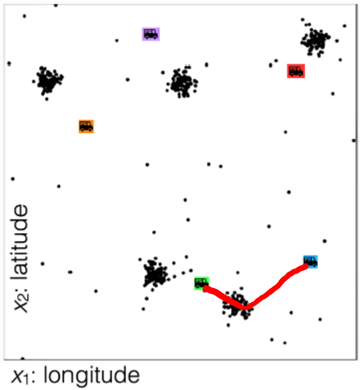
Poll Everywhere

Group

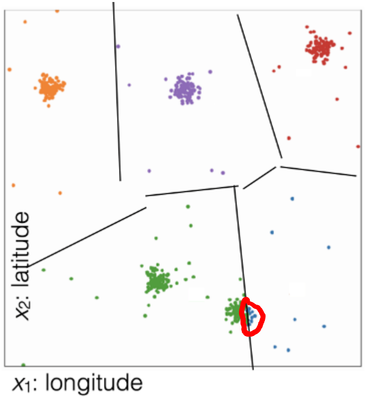
1 min

~~pollev.com/cs416~~

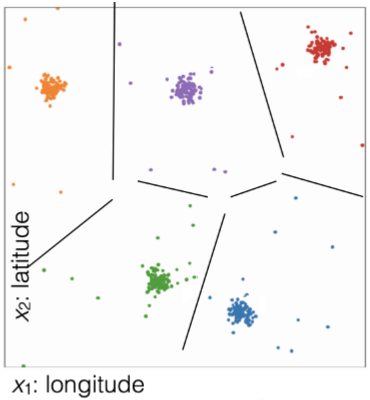
- What cluster assignment would result from these centroids?



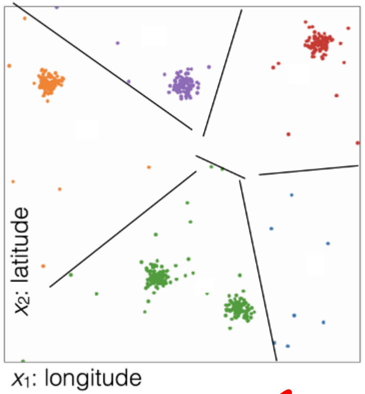
Centroids



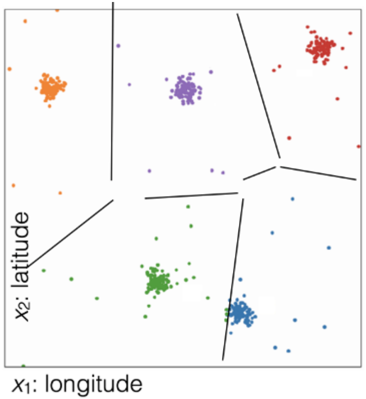
(D)



(A)



(B)



(C)

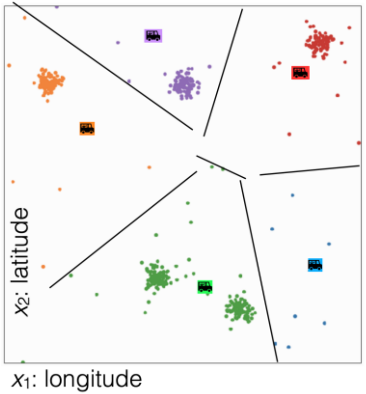
Poll Everywhere

Think

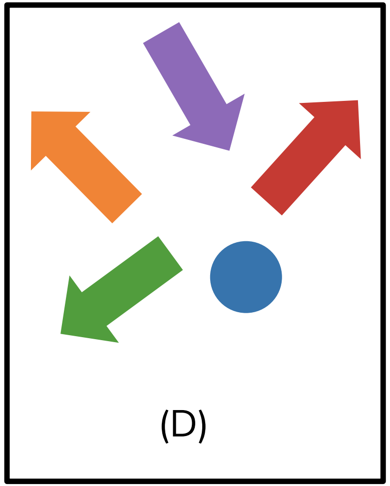
1 min

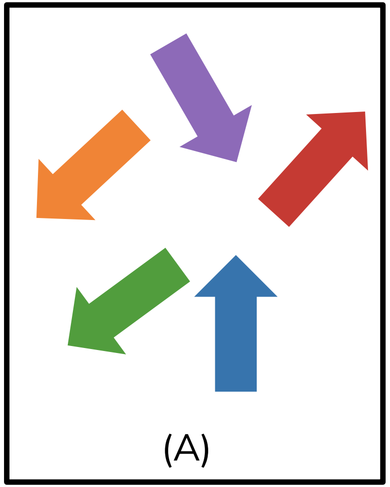
~~pollev.com/cs416~~

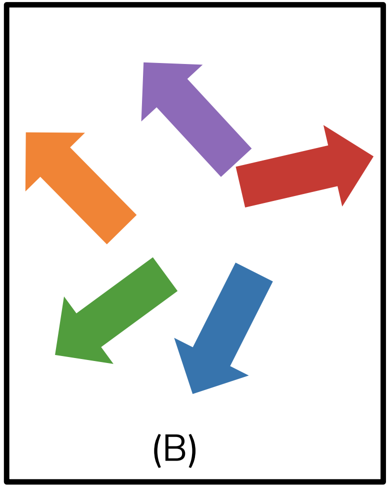
- In what direction would each of the centroids (roughly) move?

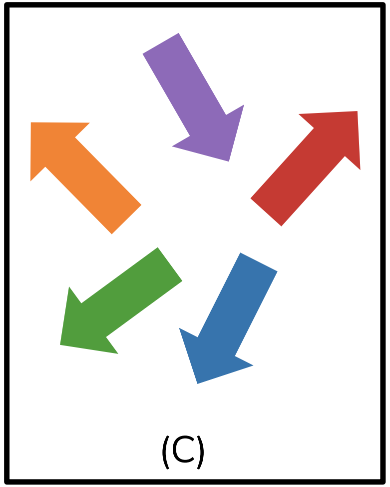


Cluster Assignments









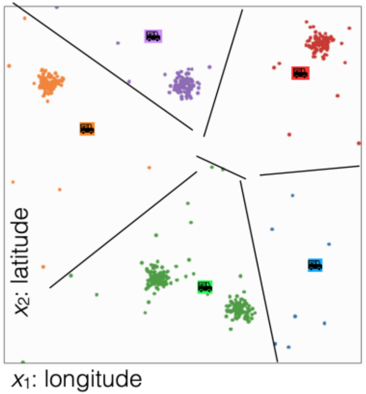
Poll Everywhere

Group

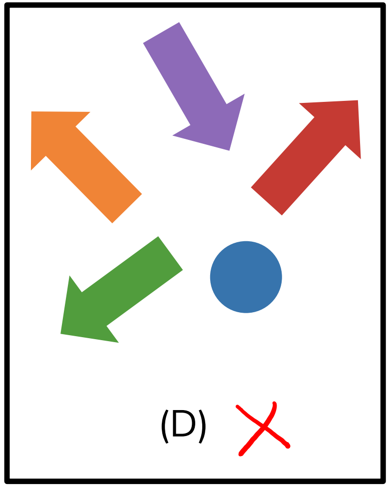
2 min

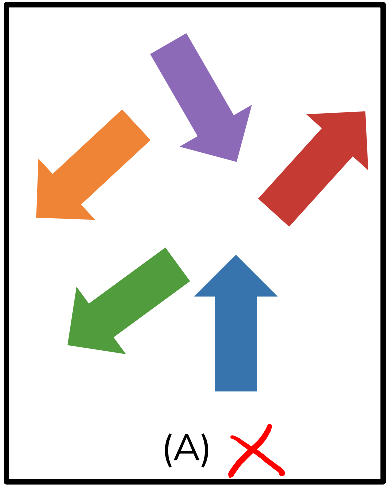
~~pollev.com/cs416~~

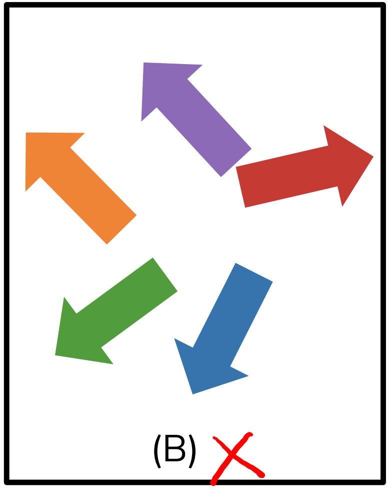
- In what direction would each of the centroids (roughly) move?

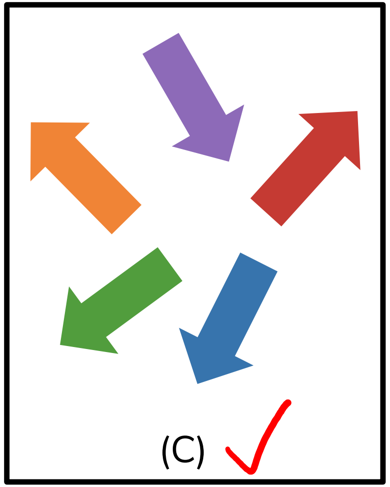


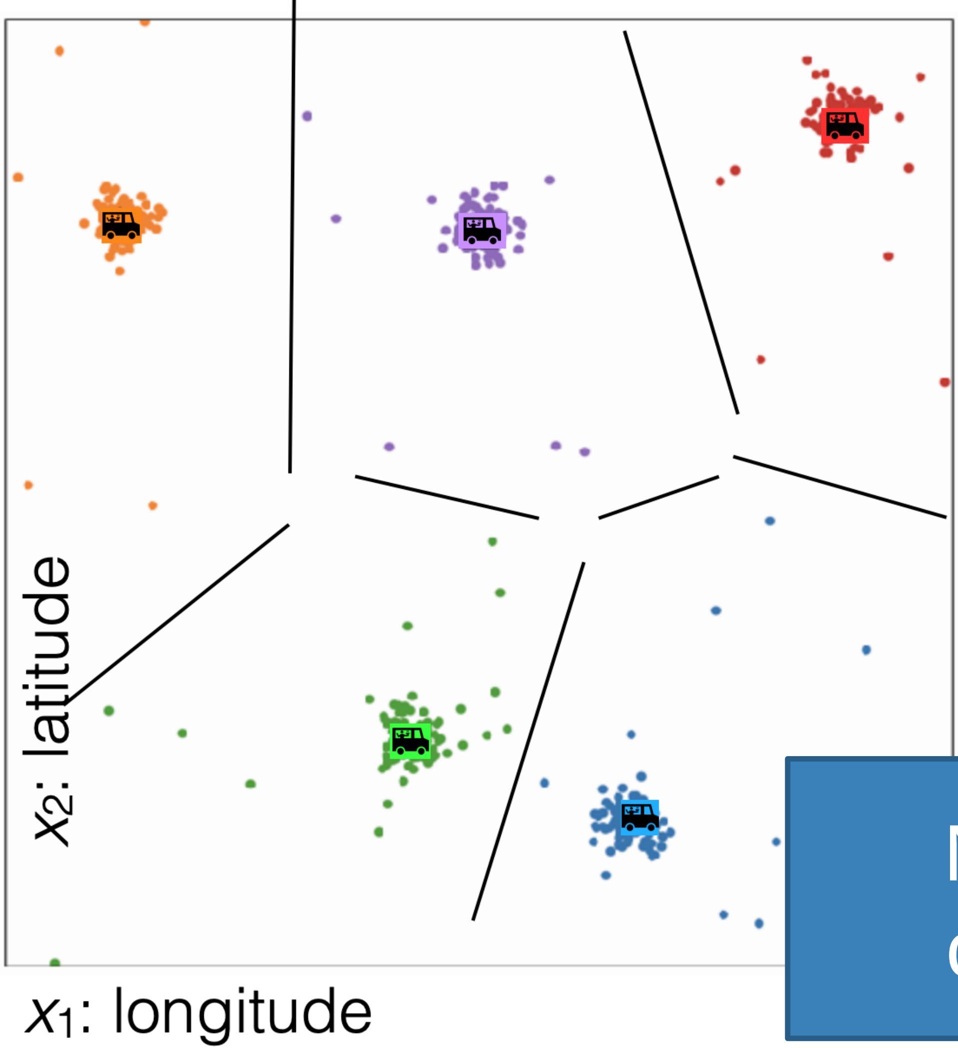
Cluster Assignments









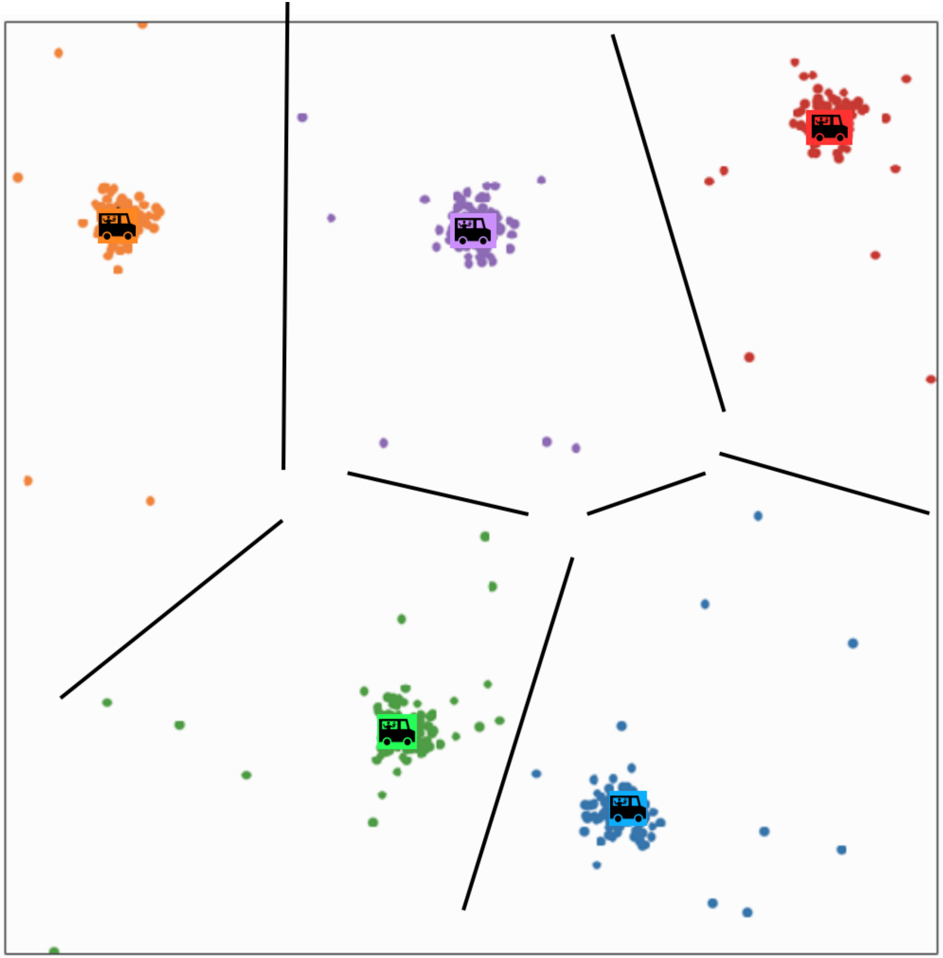
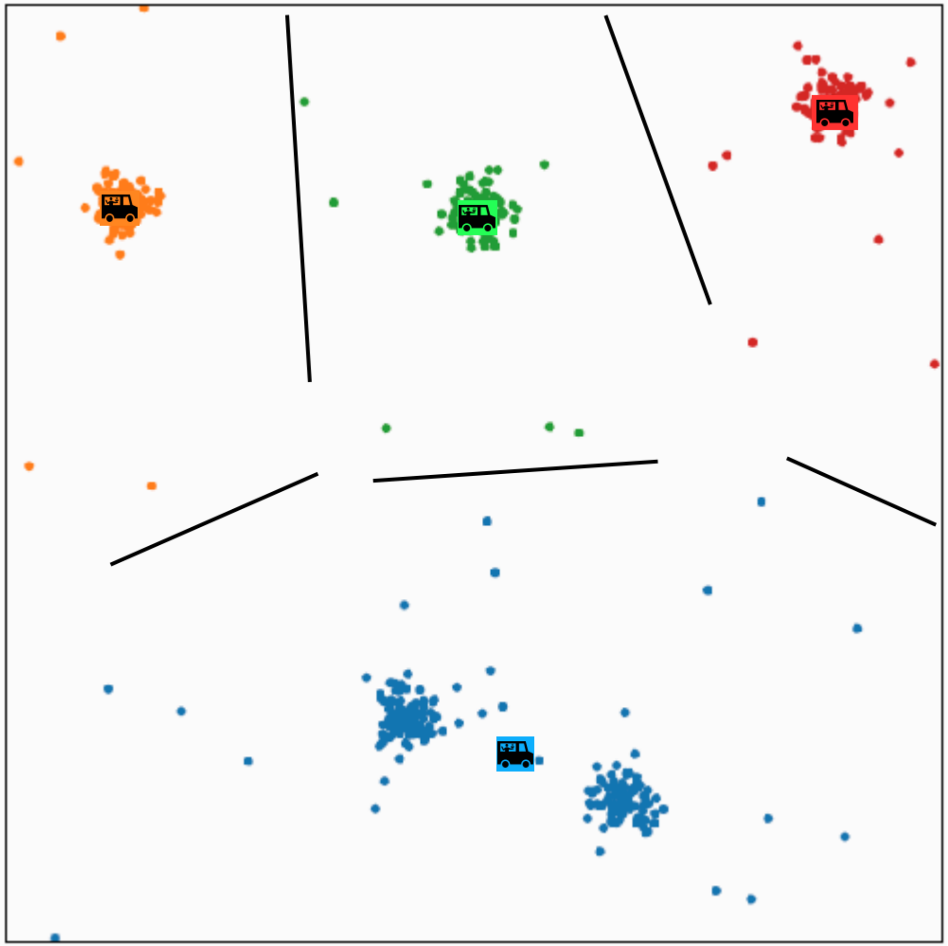


No more changes

Different k gives different results

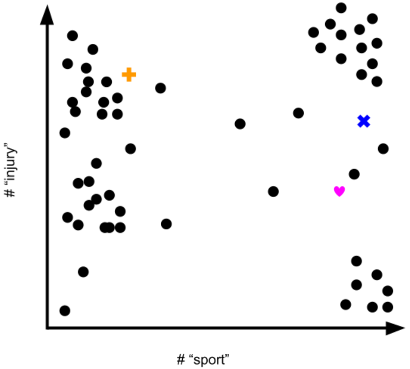
K=5

K=4

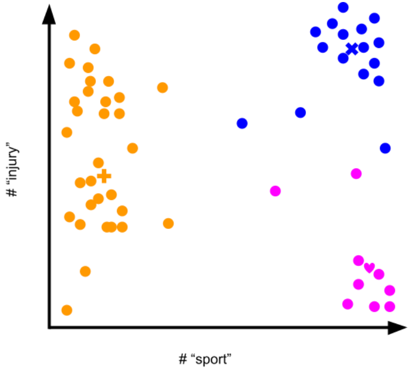



Effect of Initialization

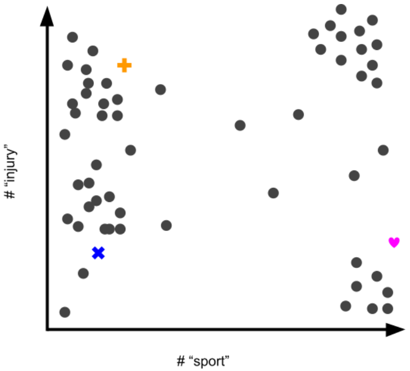
Different initialization can give different results

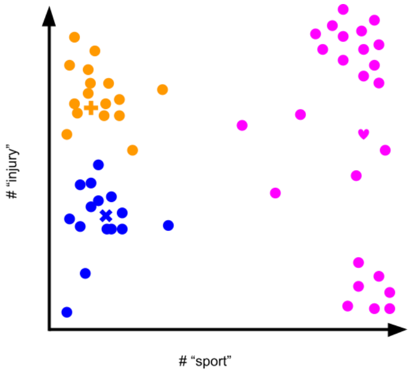


Initialization



Final Clusters



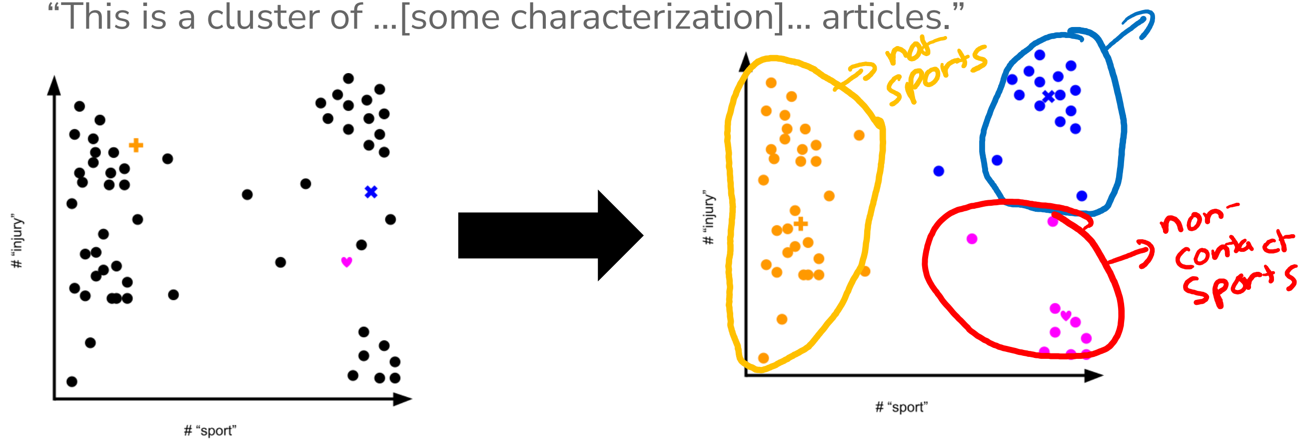


Poll Everywhere

Think

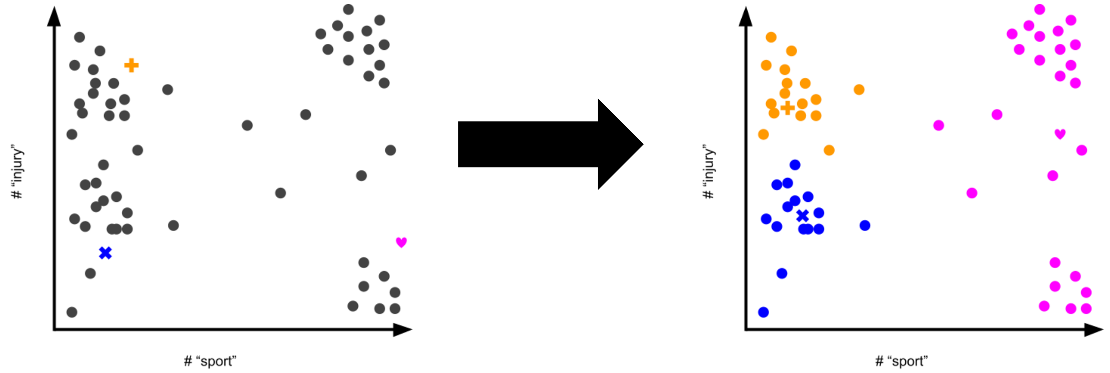
1 min

- You are clustering news articles using the features "# sport" and "# injury." How would you interpret these clusters?
- "This is a cluster of …[some characterization]… articles."



Initialization

Final Clusters

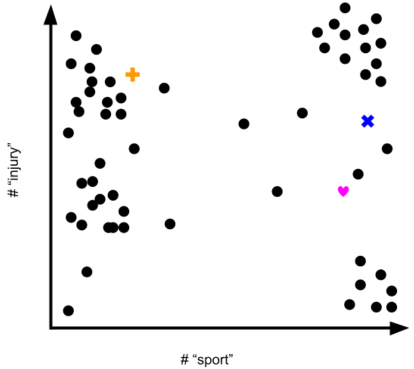


Poll Everywhere

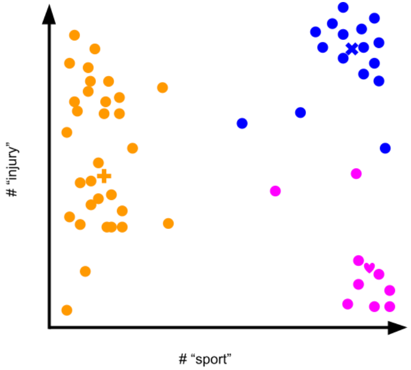
Group

2 min

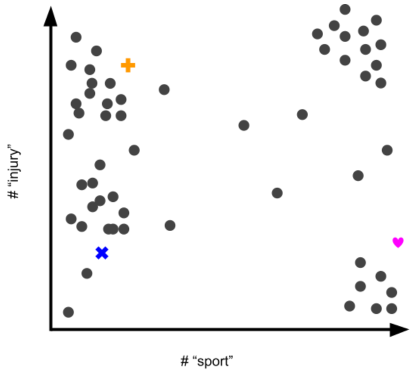
- You are clustering news articles using the features “# sport” and “# injury.” How would you interpret these clusters?
- “This is a cluster of …[some characterization]… articles.”

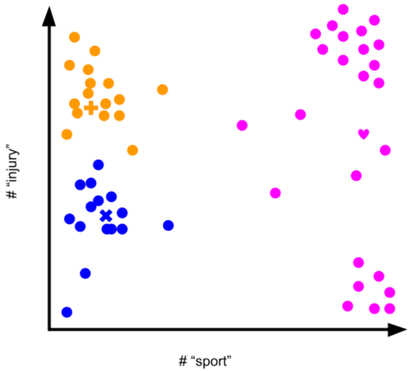


Initialization



Final Clusters

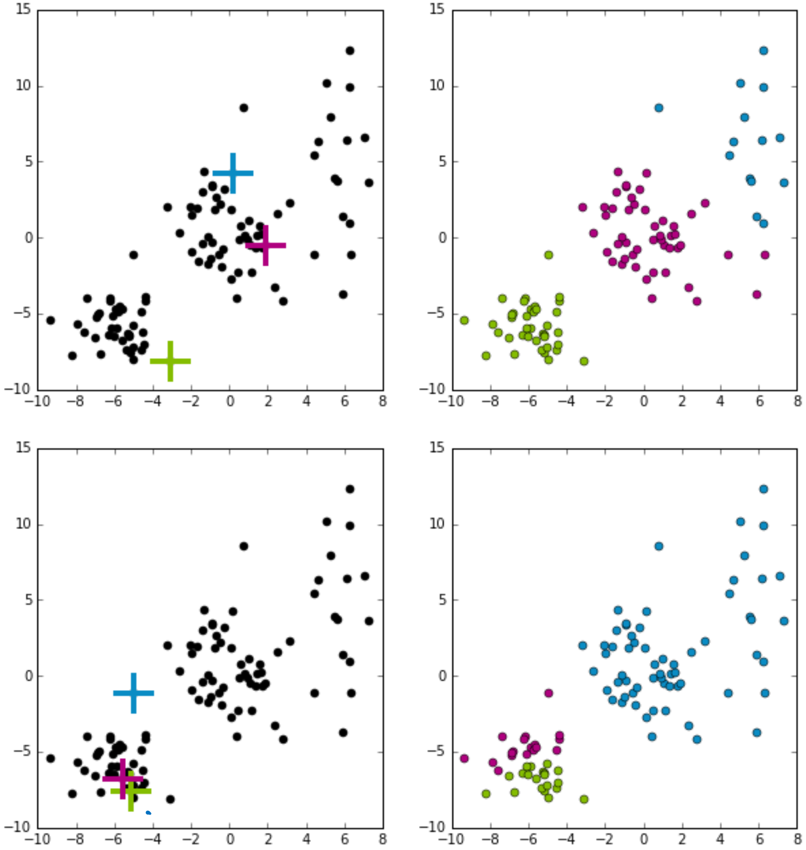




Effect of initialization

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

- Some initialization can be bad and affect the quality of clustering
- Initialization 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

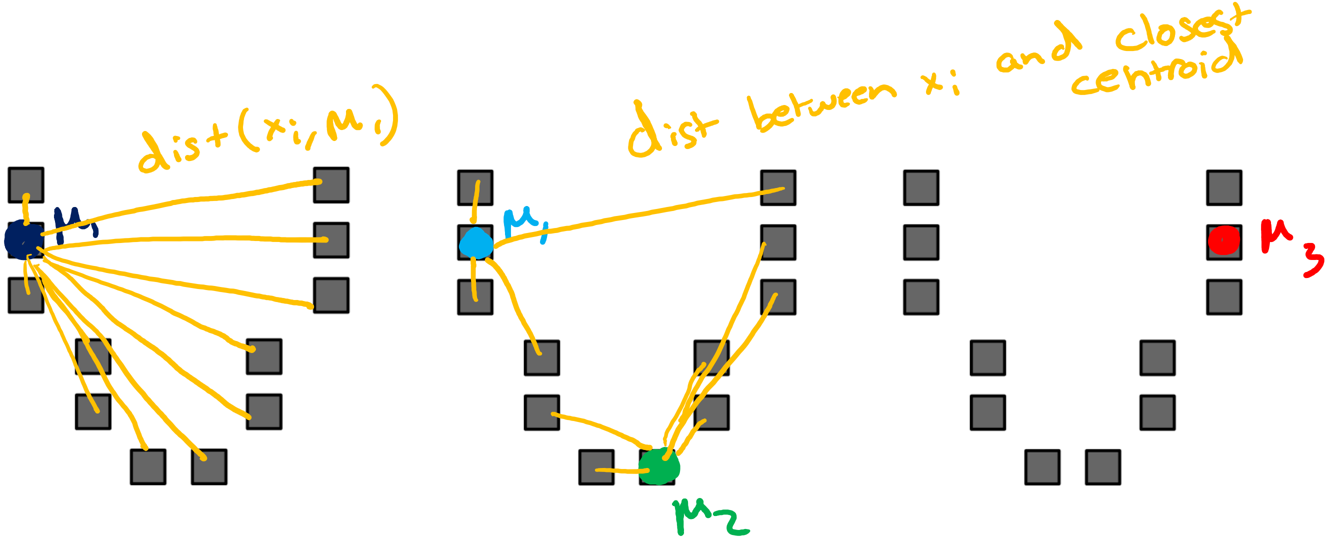
1. Choose first cluster μ_1 from the data uniformly at random
2. For each datapoint x_i, compute the distance between x_i and the closest centroid from the current set of centroids (starting with just μ_i). Denote that distance $d(x_i)$.
3. Choose a new centroid from the remaining data points, where the probability of x_i being chosen is 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

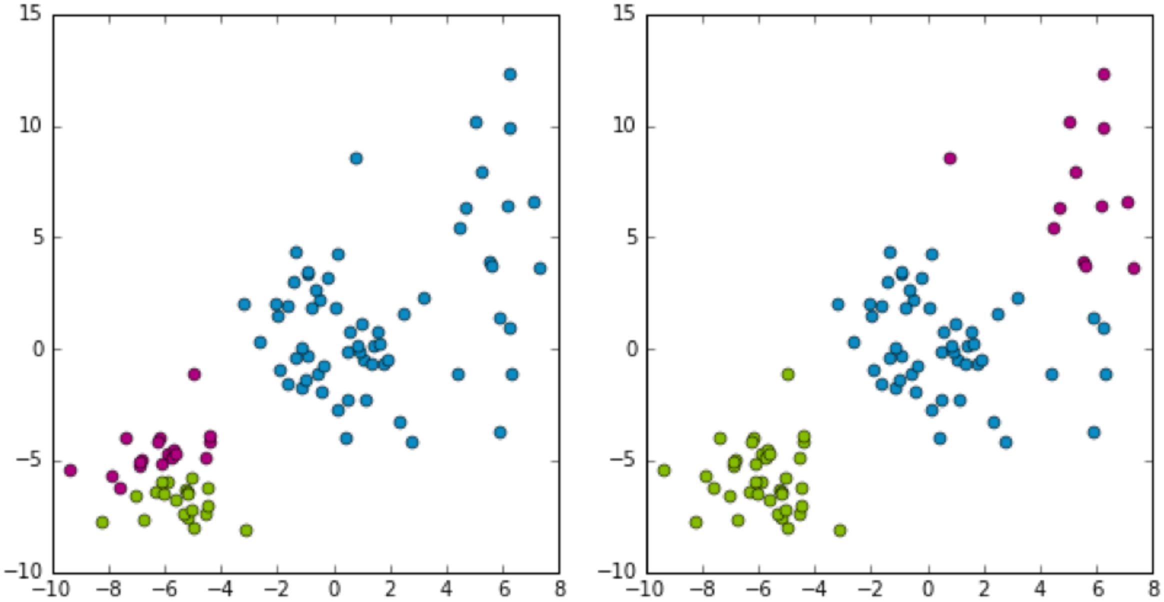
Cons

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

Assessing Performance

Which Cluster?

Which clustering would I prefer?

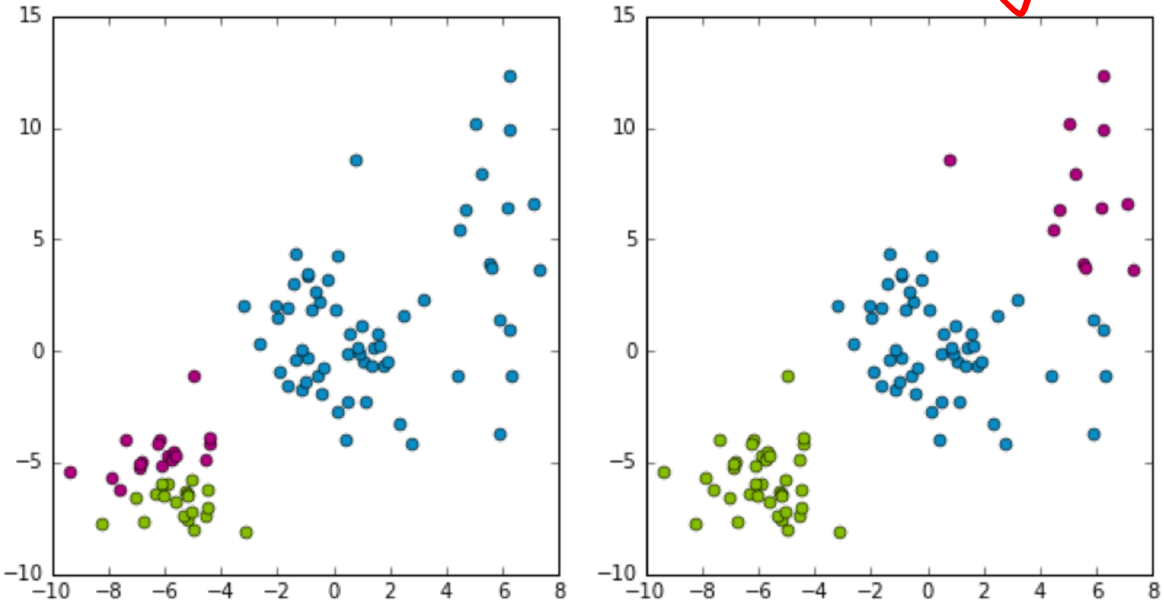


Don't know, there is no "right answer" in clustering 🤷 .

Depends on the practitioner's domain-specific knowledge and interpretation of results!

Which Cluster?

Which clustering does k-means prefer?



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

$$\underset{z,\mu}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{i=1}^{n} \mathbf{1}\{z_i = j\} \left\|\mu_j - x_i\right\|_2^2$$

sum over clusters

total distance between points in that cluster and the centroid

Coordinate Descent

k-means is trying to minimize the heterogeneity objective

$$\underset{z,\mu}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{i=1}^{n} \mathbf{1}\{z_i = j\} \left|\left|\mu_j - x_i\right|\right|_2^2$$

Step 0: Initialize cluster centers

Repeat until convergence: fix μ, minimize z

- 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 fix z, minimize μ

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.

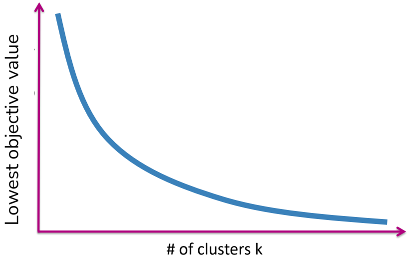
=> Will converge in finite time

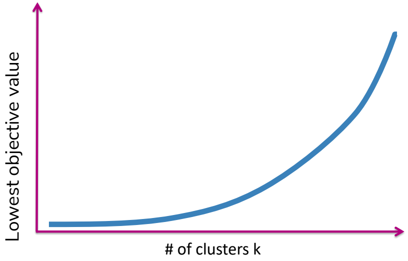
Poll Everywhere

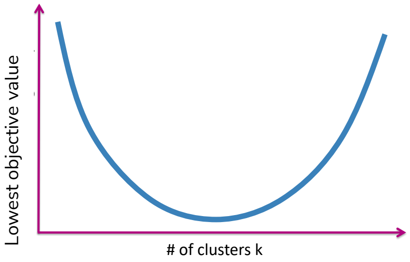
Think

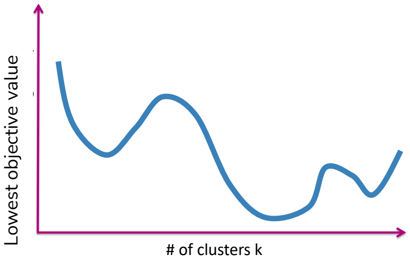
1 min

Consider training k-means to convergence for different values of k. Which of the following graphs shows how the heterogeneity objective will change based on the value of k?









Poll Everywhere

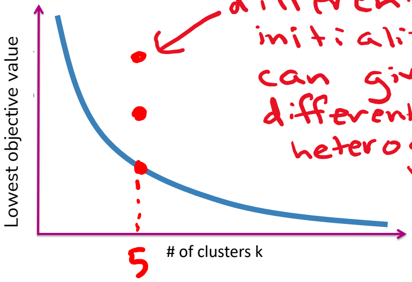
Group

2 mins

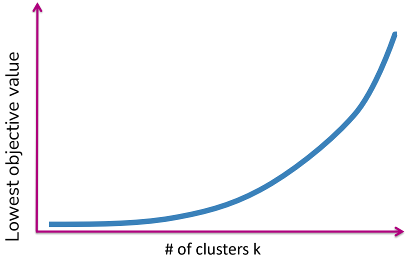
$K = 1$

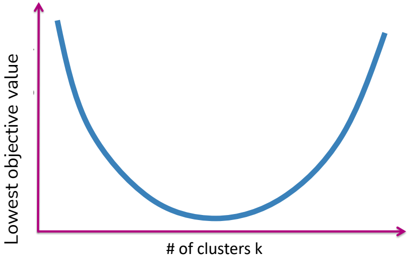
$k = n \Rightarrow$ heterogeneity is 0

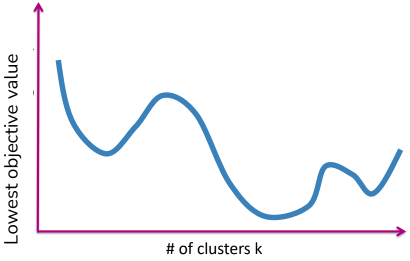
Consider training k-means to convergence for different values of k. Which of the following graphs shows how the heterogeneity objective will change based on the value of k?



different initializations can give different heterogeneity values



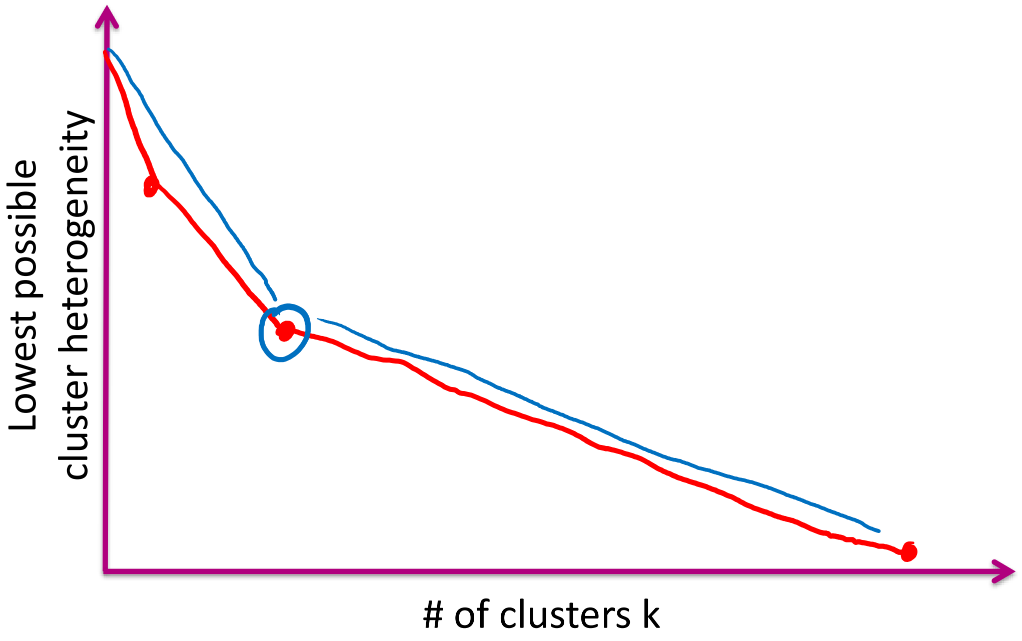




How to Choose k?

No right answer! Depends on your application.

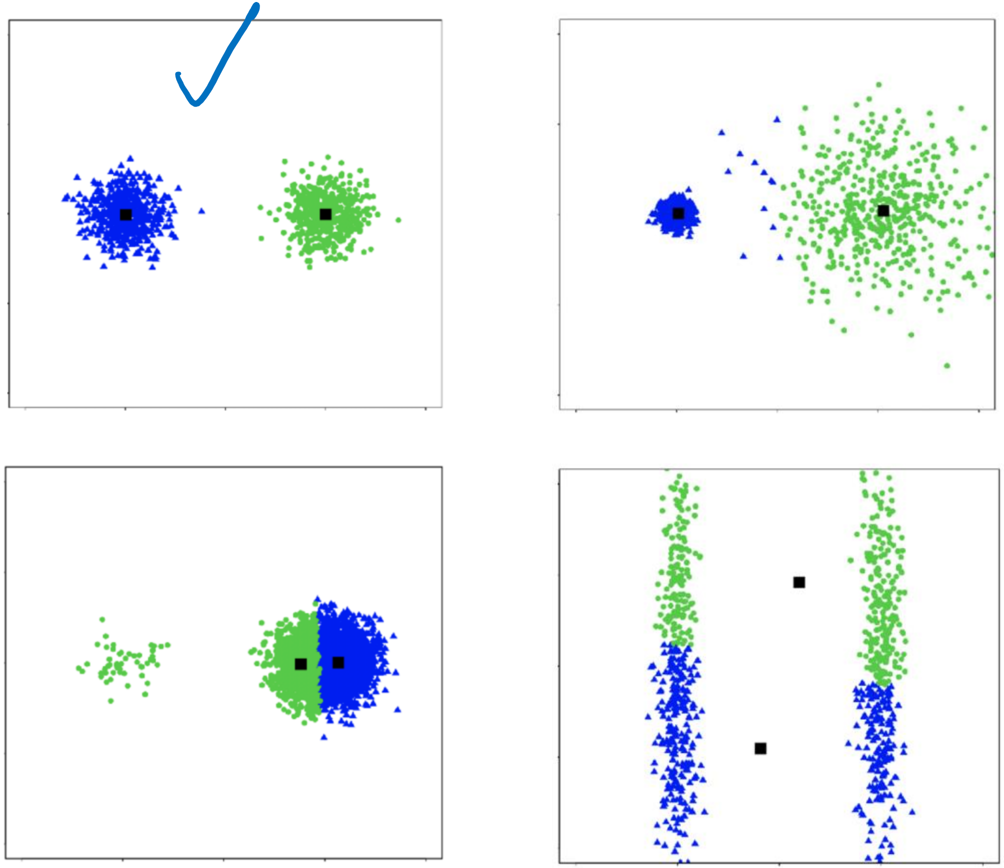
- General, look for the "elbow" in the graph



Note: You will usually have to run k-means multiple times for each k

Cluster shape

- k-means works well for well-separated **hyper-spherical** clusters of the same size



Poll Everywhere

Think

1 min

SKIPPED

- Identify the similarities and differences between the following:
  - k-means & k nearest neighbors
  - clustering & classification

Poll Everywhere

Group

2 min

SKIPPED

- Identify the similarities and differences between the following:
  - k-means & k nearest neighbors
  - clustering & classification

Will pick up on Wed

Clustering vs Classification

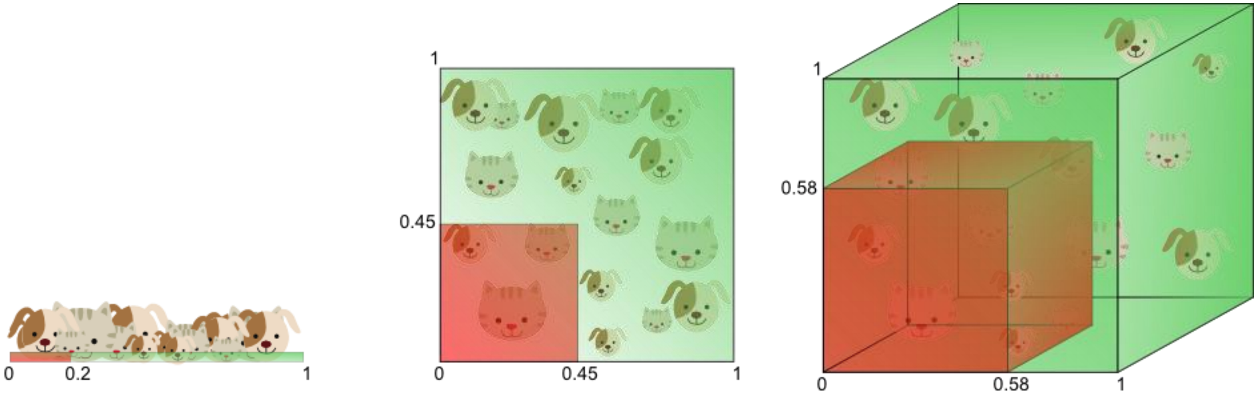
- Clustering looks like we assigned labels (by coloring or numbering different groups) but we didn't use any **labeled** data.
- In clustering, the "labels" don't have meaning. To give meaning to the labels, human inputs is required
- Classification learns from minimizing the error between a prediction and an actual **label**.
- Clustering learns by minimizing the distance between points in a cluster.
- Classification quality metrics (accuracy / loss) do not apply to clustering (since there is no label).
- You can't use validation set / cross-validation to choose the best choice of k for clustering.

Curse of Dimensionality

High Dimensions

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

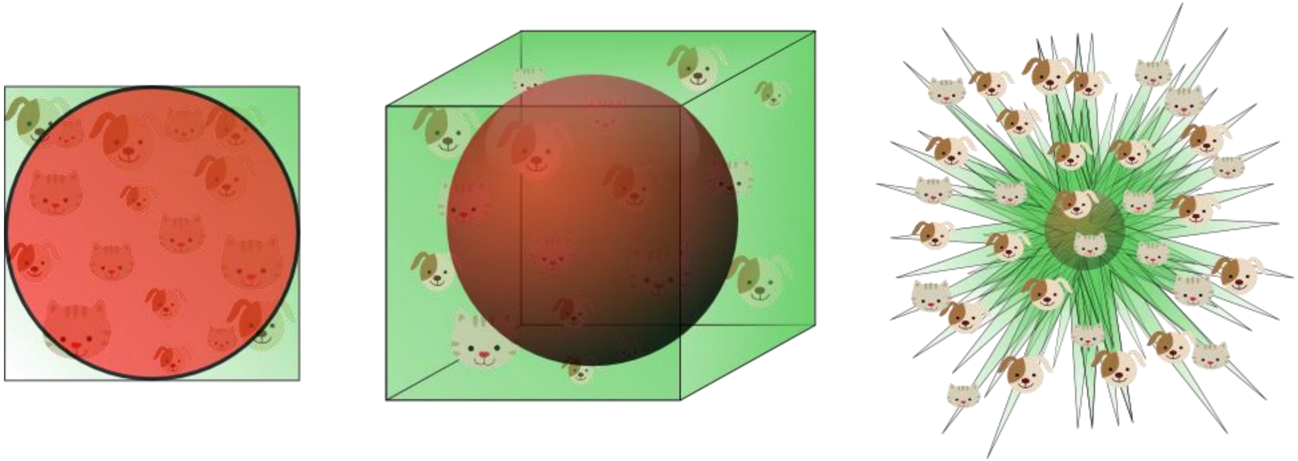
As the number of dimensions grow, the data gets sparser!



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

Data Moves Farther Apart in Higher Dimensions

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.
- The distance between points gets really high!

Practicalities

Have to pay attention to the number of dimensions with distance-based methods (k-means clustering, also k nearest neighbors).

- 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.

Recap

- Differences between classification and clustering
- What types of clusters can be formed by k-means
- K-means algorithm
- Convergence of k-means
- How to choose k
- Better initialization using k-means++