Computer Vision: Fall 2022 — Lecture 4

Dr. Karthik Mohan

Univ. of Washington, Seattle

October 11, 2022

Weekly Logistics

	Day	Timings	Class type
Lecture 1 (In-person)	T	4 pm - 6 pm	(In-person)
Lecture 2	Th	4 pm - 6 pm	Zoom
Office Hours Karthik	T	5:50 - 6:20 pm	In-person/Zoom
Office Hours Ayush	Fri	5-6 pm	Zoom
Quiz Section Ayush	Mon	5-6 pm	Zoom

Late Days for Assessments

Late Submission Policy

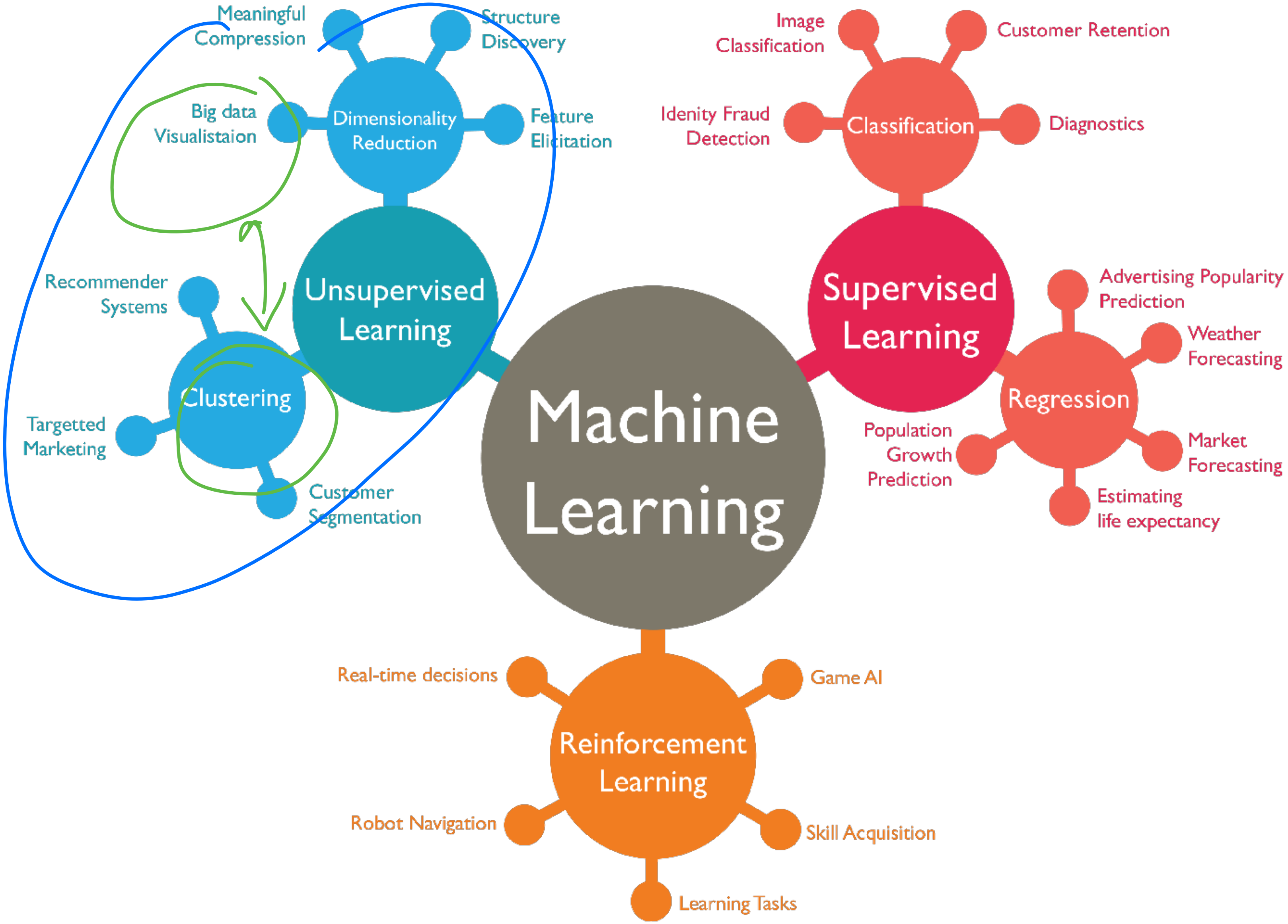
Maximum of 5 Late days allowed across the quarter. If you exceed your late days and your submission is late - Incur a 10% penalty on your grade.

Today

1. Introduction to clustering and kMeans
2. kMeans and kMeans++
3. tSNE for Image Visualization

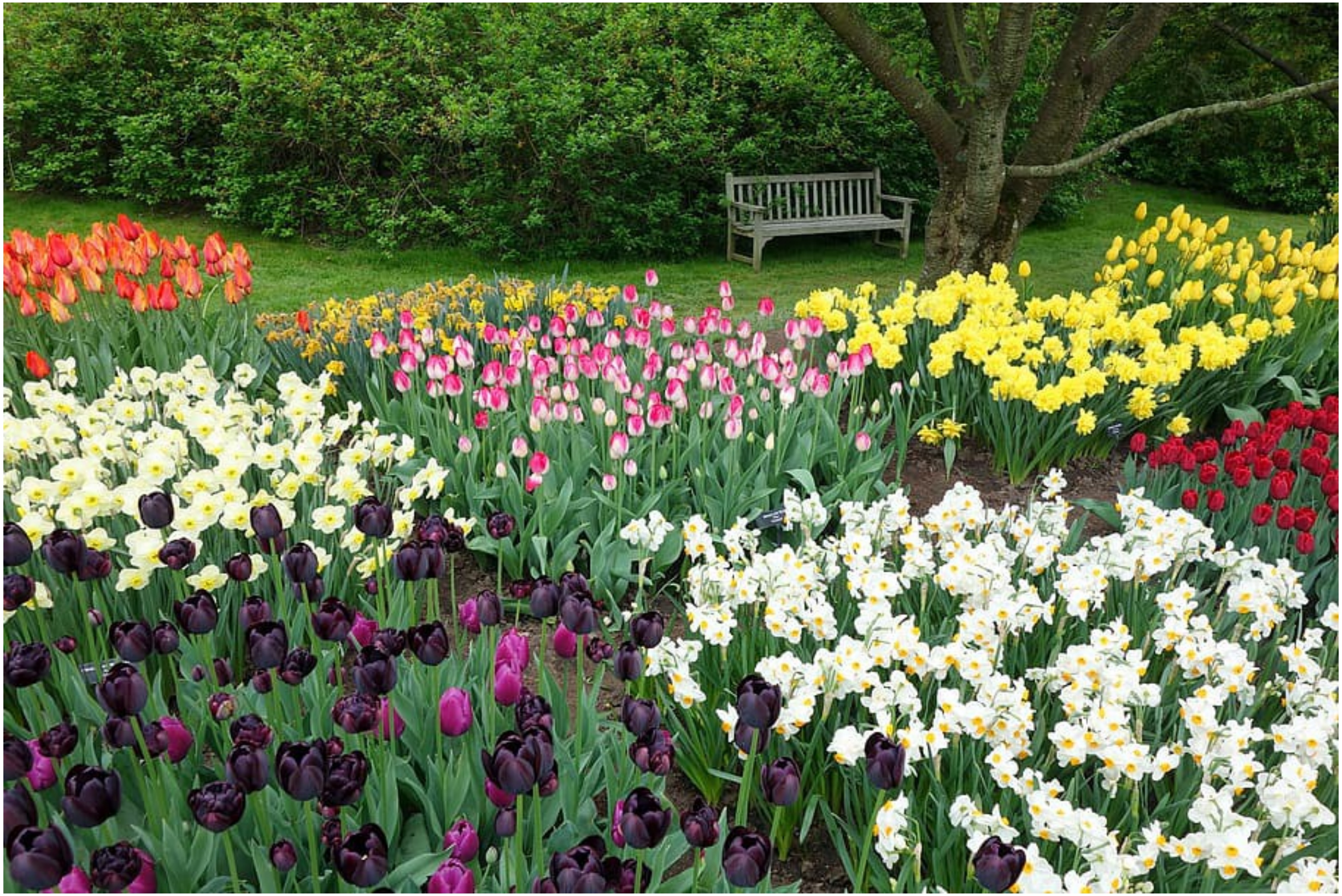
Next Topic: Clustering of Data/Images

Big Picture


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big data Visualistaion
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Idenity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Population Growth Prediction
Market Forecasting
Estimating life expectancy
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

The clustering problem

k?
=
#clusters

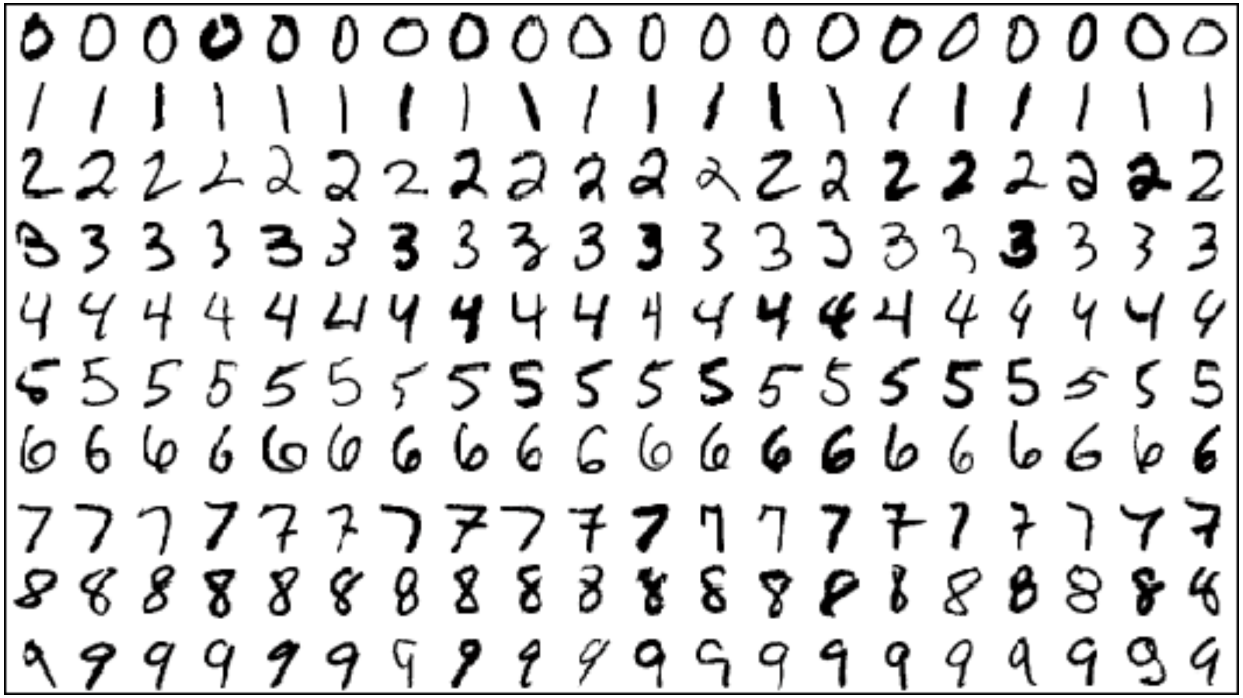


Clustering vs Classification

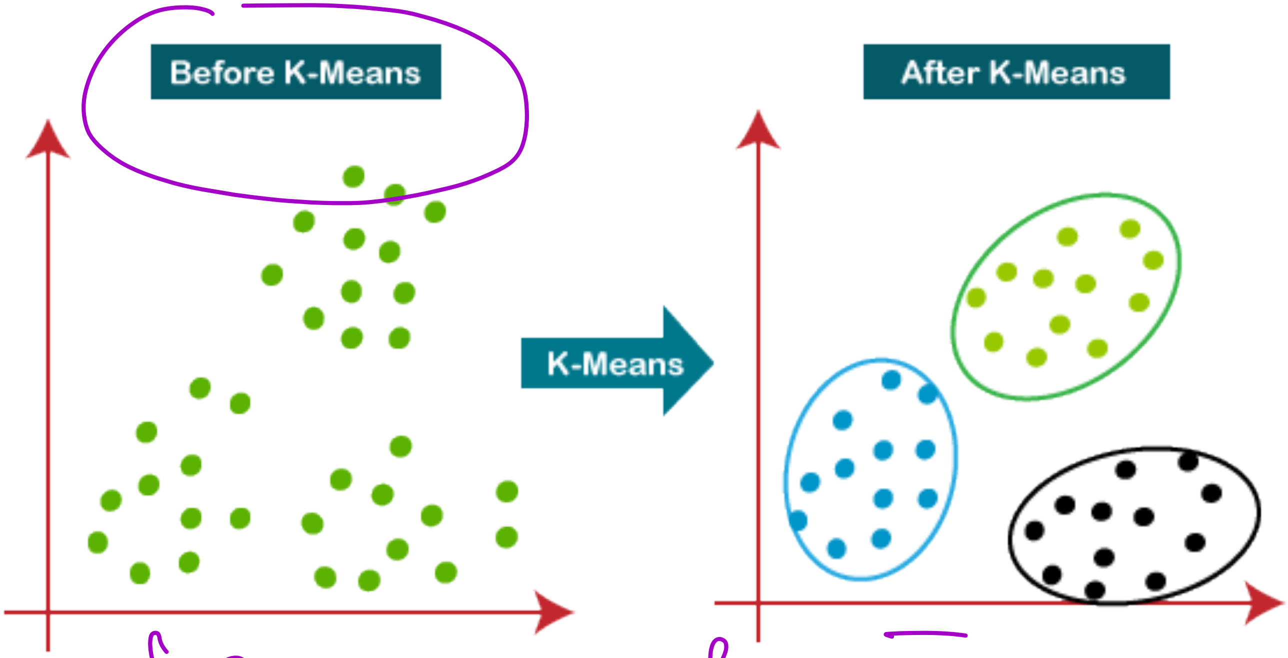
Difference

In the classification problem, you are given (x^i, y_i) - I.e. both the data point i and its true label y_i for training purposes. Example - a flower i and its label (flower type). Whereas in clustering problem, you are just given the data points, i.e. x^i. However, you still want to break up the data points into clusters - where each cluster has relatively similar data points.

Digits Clustering



Clustering of data points



K = 3

Data points in $\mathbb{R}^2$ → 2 Dims
- Not clustered

Clustering for News

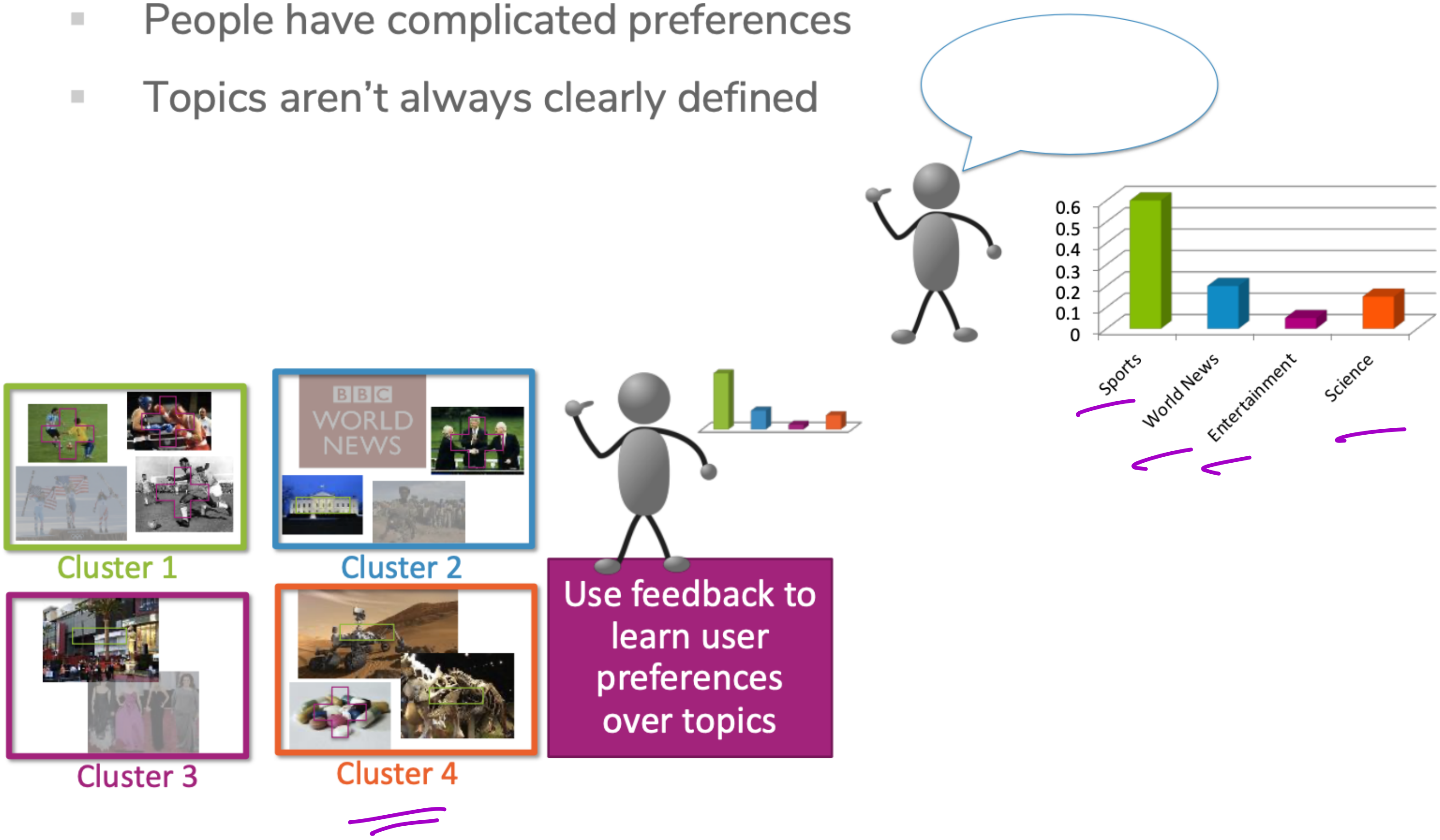
SPORTS

WORLD NEWS

Clustering for 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



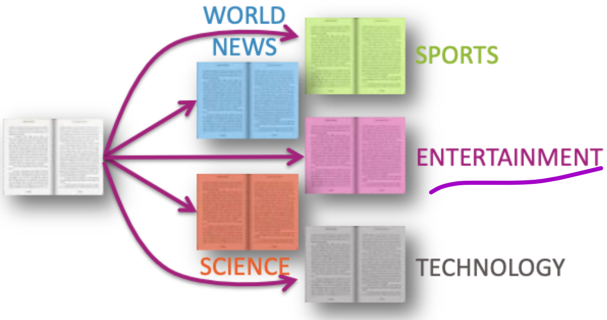
Use feedback to learn user preferences over topics

Clustering for News

What if the labels are known? Given labeled training data

Can do multi-class classification methods to predict label

Supervised Learning



Clustering Basics

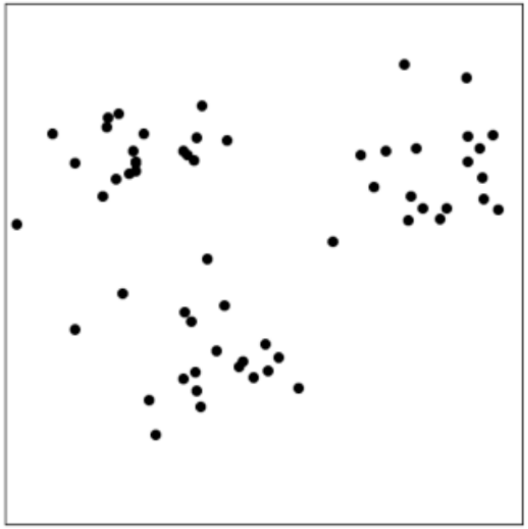
- In many real world contexts, there aren't clearly defined labels so we won't be able to do classification
- We will need to come up with methods that uncover structure from the (unlabeled) input data X.
- **Clustering** is an automatic process of trying to find related groups within the given dataset.

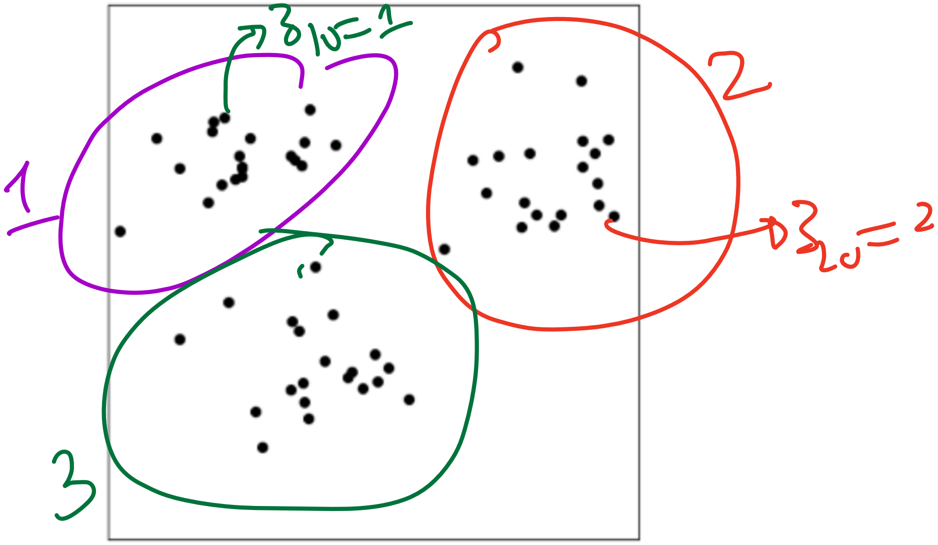
Input 1 Input 2

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

cluster Assignment

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





Clustering Basics

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

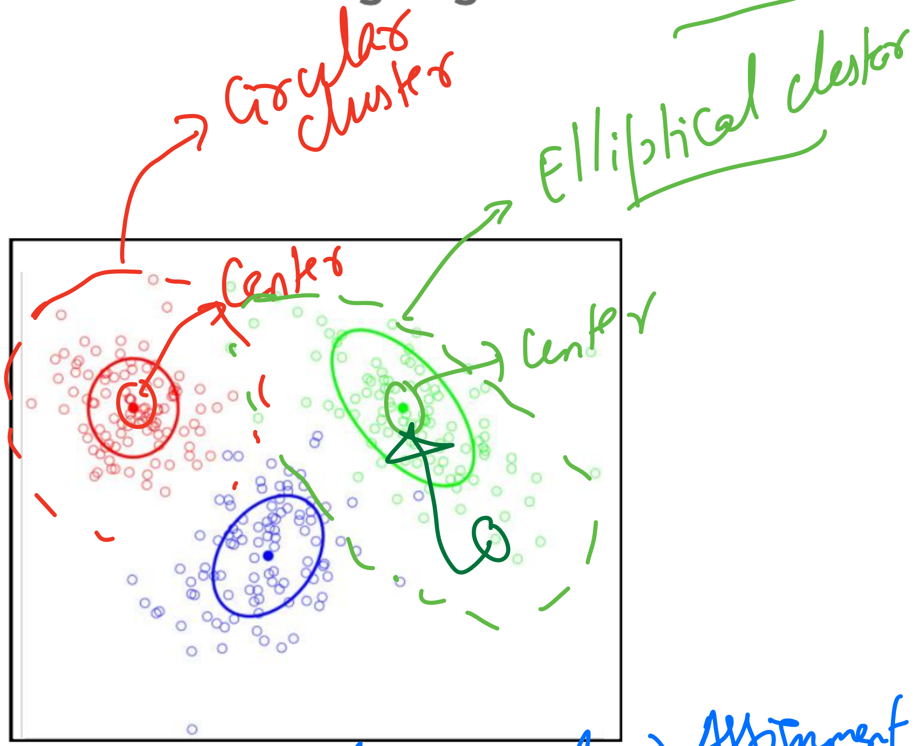
- The location of its center (**centroid**)
- Shape and size of its **spread**

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

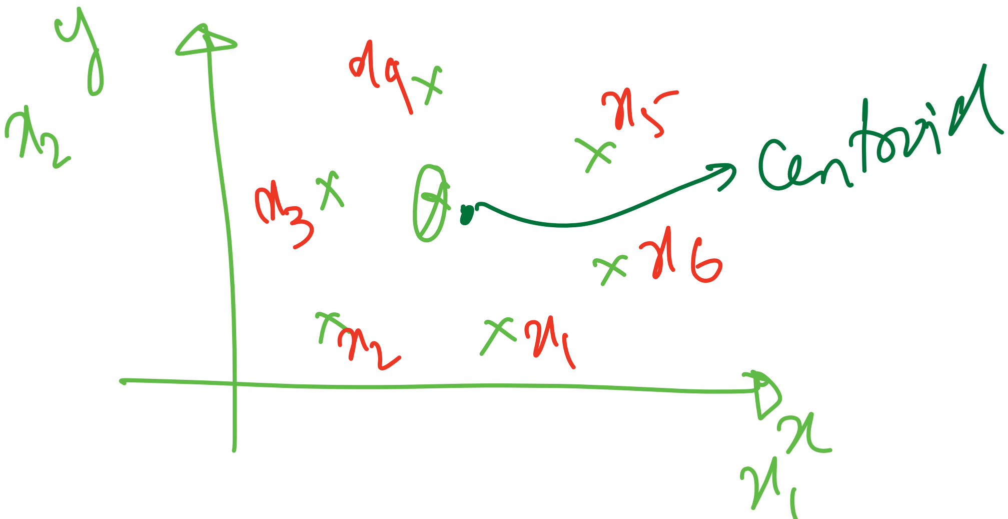


k - means : 1) Identify good Centroids 2) Assignment

Centroid of a cluster

Centroid

The centroid of a cluster of points is the point that is closest to all the points on an average. If we use Euclidean distance, the centroid is just the average of the points! *Why?*



$$\text{centroid} = \frac{x_1 + x_2 + x_3 + x_4 + x_5 + x_6}{6}$$

ICE #1: Centroid of a cluster

Centroid

The centroid of a cluster of points is the point that is closest to all the points on an average. If we use Euclidean distance, the centroid is just the average of the points!

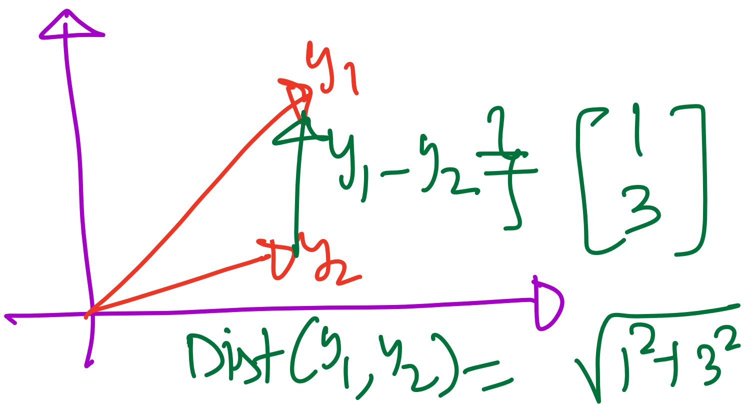
ICE #1

Consider a cluster, $C_3 = [(3,5),(3,4),(5,6)]$ with 3 data points, where each data point is in 2 dimensions. Visualize the data points by plotting it on a x-y graph. The centroid of C_3 is given by:

1. $(5, 11/3)$
2. $(11, 5/3)$
3. $(5/3, 11)$
4. $(11/3, 5)$

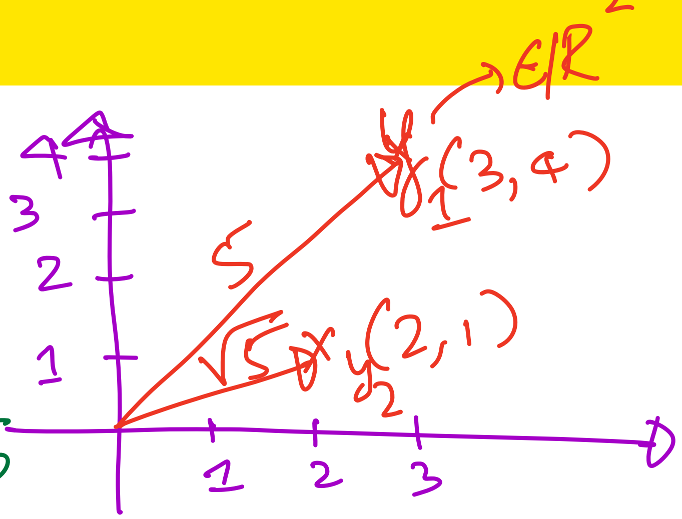
Also plot the centroid on your x-y graph.

Distance typically used



$y_1 - y_2 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$

$\text{Dist}(y_1, y_2) = \sqrt{1^2 + 3^2} = \sqrt{10}$



Euclidean Distance

Distance between two points, x_1, x_1 is given by:

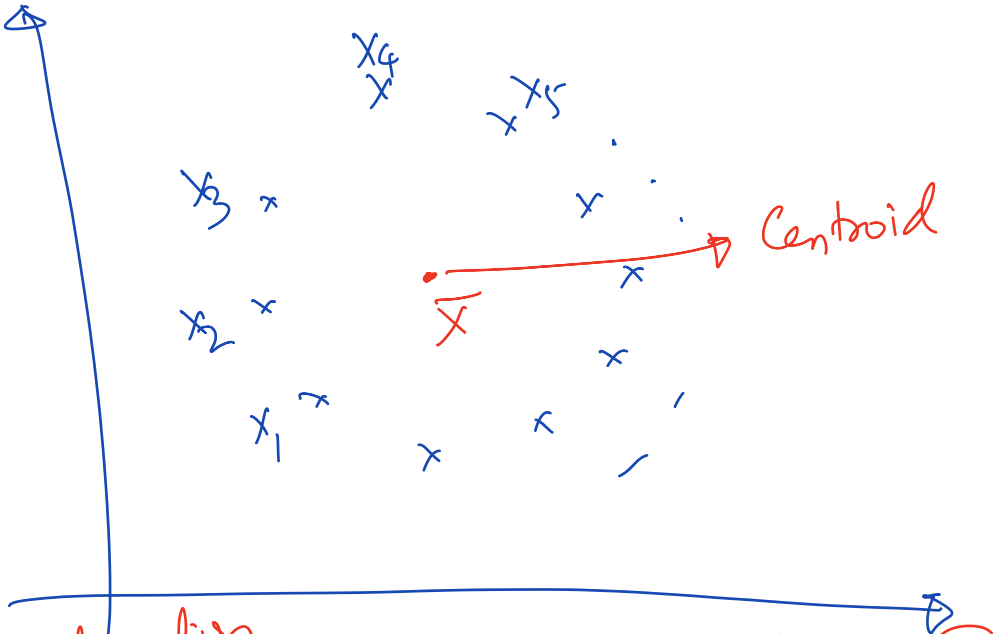
$$\|x_1 - x_2\|_2$$

ℓ_2 norm of $x_1 - x_2$

magnitude for a vector

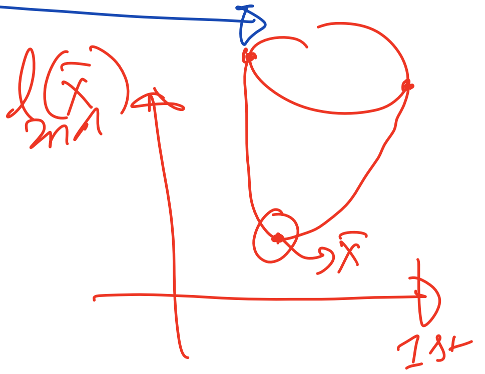
$$\| y \|_2 = \sqrt{\sum_i y_i^2}$$

$y = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$ $\quad \|y\|_2 = \sqrt{3^2 + 4^2} = \sqrt{25} = 5$



Loss function

$$l(\bar{x}) = \min_{\bar{x}} \sum_{i=1}^{10} \underbrace{\|\bar{x} - x_i\|_2^2}$$



$$\nabla_{\bar{x}} l(\bar{x}) = 0$$

$$\Rightarrow \sum_{i=1}^{10} 2(\bar{x} - \underline{x_i}) = 0$$

$$\Rightarrow 10\bar{x} = \sum_i x_i \Rightarrow \bar{x} = \frac{\sum_i x_i}{10}$$

$l(\underline{y}) = \|y\|_2^2$ ↳ $\in \mathbb{R}^d$

$\nabla l = ?$

$l(y) = y_1^2 + y_2^2 + y_j^2 + y_d^2$

$$(\nabla l)_j = \frac{\partial l}{\underline{\partial y_j}} = \frac{\partial (y_j^2)}{\partial y_j} = 2y_j$$

Put it together

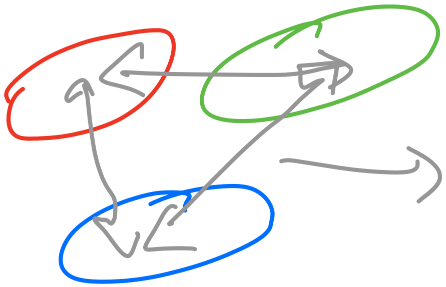
$$(\nabla l) = \begin{bmatrix} 2y_1 \\ 2y_2 \\ \vdots \\ 2y_j \\ \vdots \\ 2y_d \end{bmatrix} = 2y$$

↳ $\in \mathbb{R}^d$

Clustering Distances

→ well-separated clusters

Inter-cluster distance



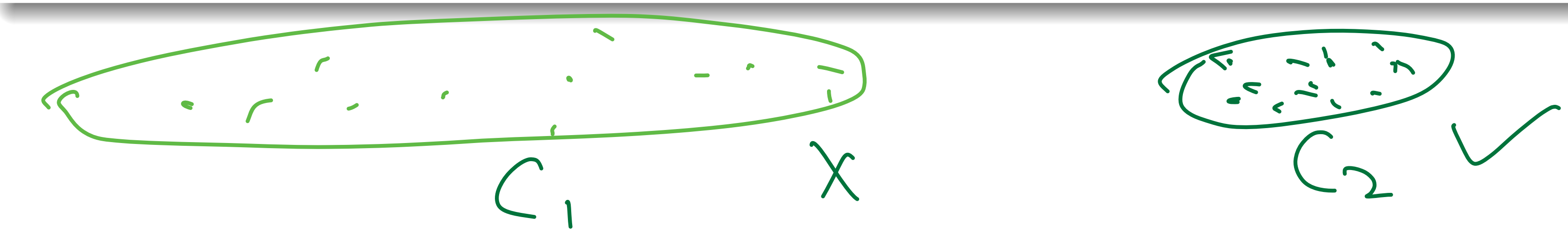
→ inter-cluster distance

The distance between different clusters is called the inter-cluster distance. We typically want clusters to capture different elements and hence a good clustering would have a larger inter-cluster distance

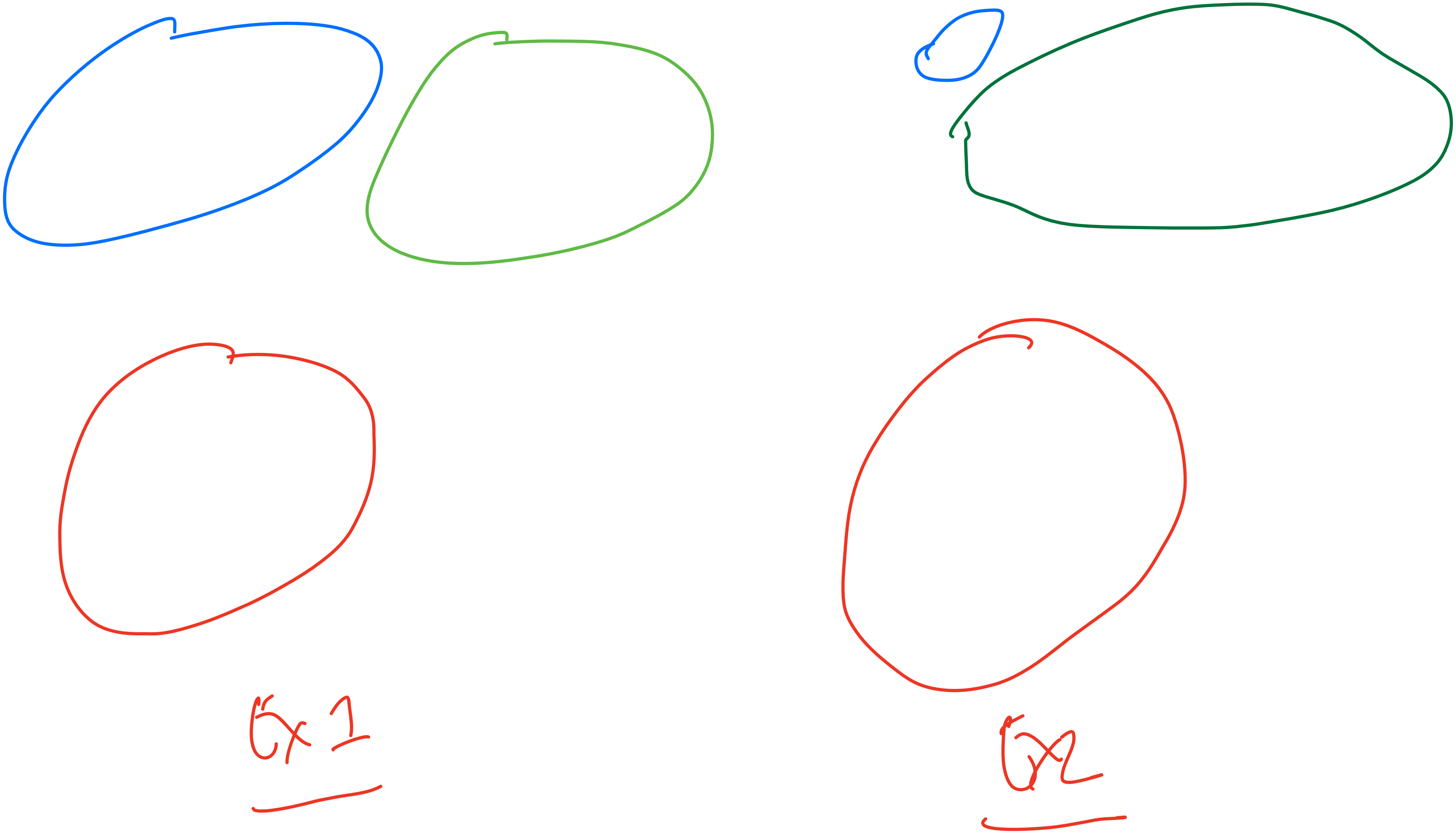
Intra-cluster distance

→ Cohesive clusters

The distance between points inside a cluster is called the intra cluster distance. We typically want a cluster to be cohesive and close to a center and hence a good clustering would have a smaller intra-cluster distance for each cluster.



Inter and Intra Cluster Distances Visual



ICE #2: Inter-cluster Distance

Inter-cluster distance

Consider 3 clusters of points where $C_1 = [(1,2),(2,3),(2,1)]$, $C_2 = [(4,5),(5,7),(6,4)]$ and $C_3 = [(3,5),(3,4),(5,6)]$. Let the distance between two clusters be defined as the "Euclidean Distance" between the centroids of the two clusters. The distance between clusters C_1 and C_3 (inter-cluster distance) is closest to:

1. 1
2. 2
3. 3
4. 4

$\sqrt{2^2+3^2} = \sqrt{13}$

Centroid $C_1 = \begin{bmatrix} 5/3 \\ 2 \end{bmatrix}$

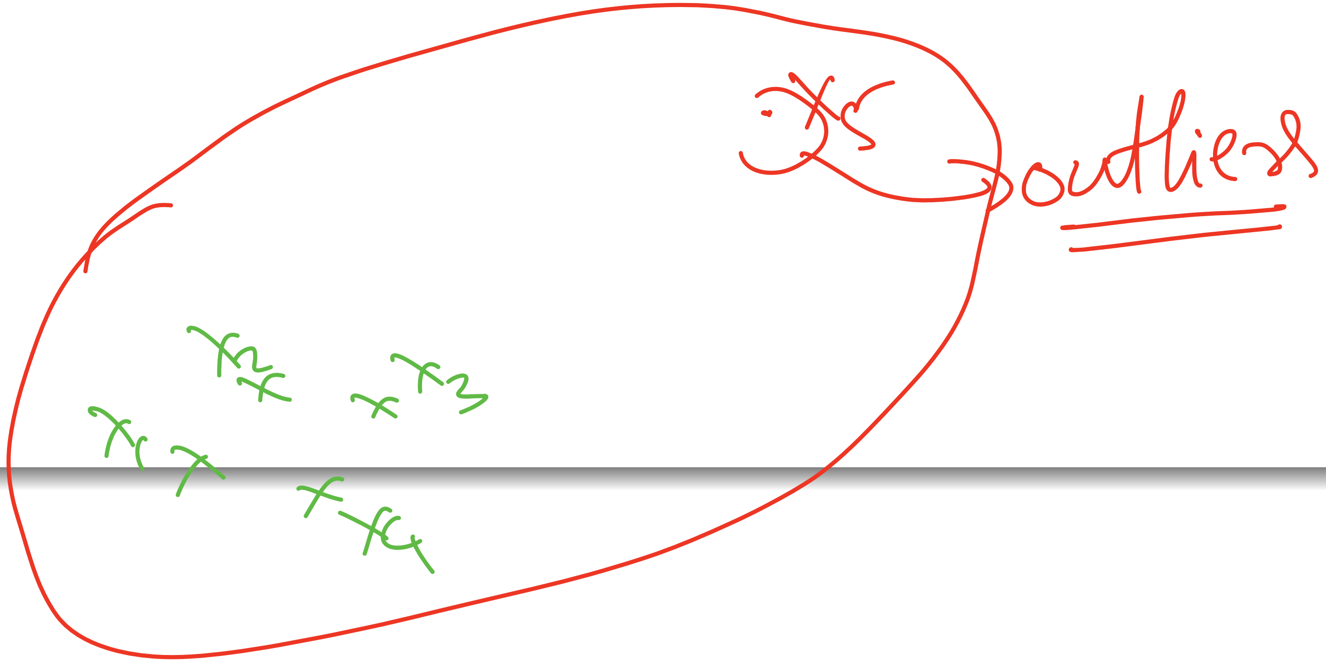
Centroid $C_3 = \begin{bmatrix} 11/3 \\ 5 \end{bmatrix}$

ICE #3: Intra-cluster Distance

Intra-cluster distance

Consider 3 clusters of points where $C_1 = [(1,2),(2,3),(2,1)]$, $C_2 = [(4,5),(5,7),(6,4)]$ and $C_3 = [(3,5),(3,4),(5,6)]$. Let the intra-cluster distance for a cluster be defined as the maximum euclidean distance between any two points in the cluster. The intra cluster distance of C_2 is closest to:

1. 1
2. 2
3. 3
4. 4

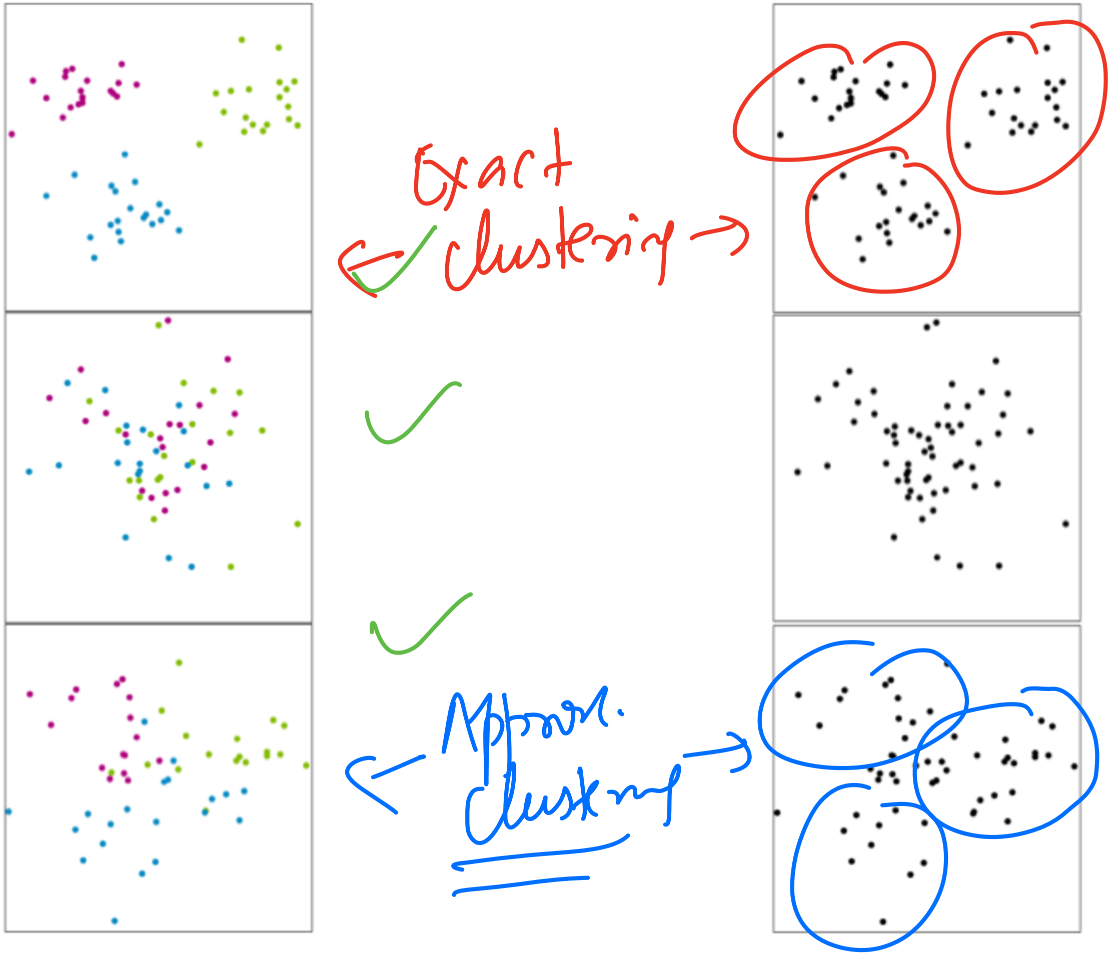


Clustering on different Data sets

Clustering is easy when distance captures the clusters

Ground Truth (not visible)

Given Data



Text Book

Reality

Exact clustering

Approx. clustering

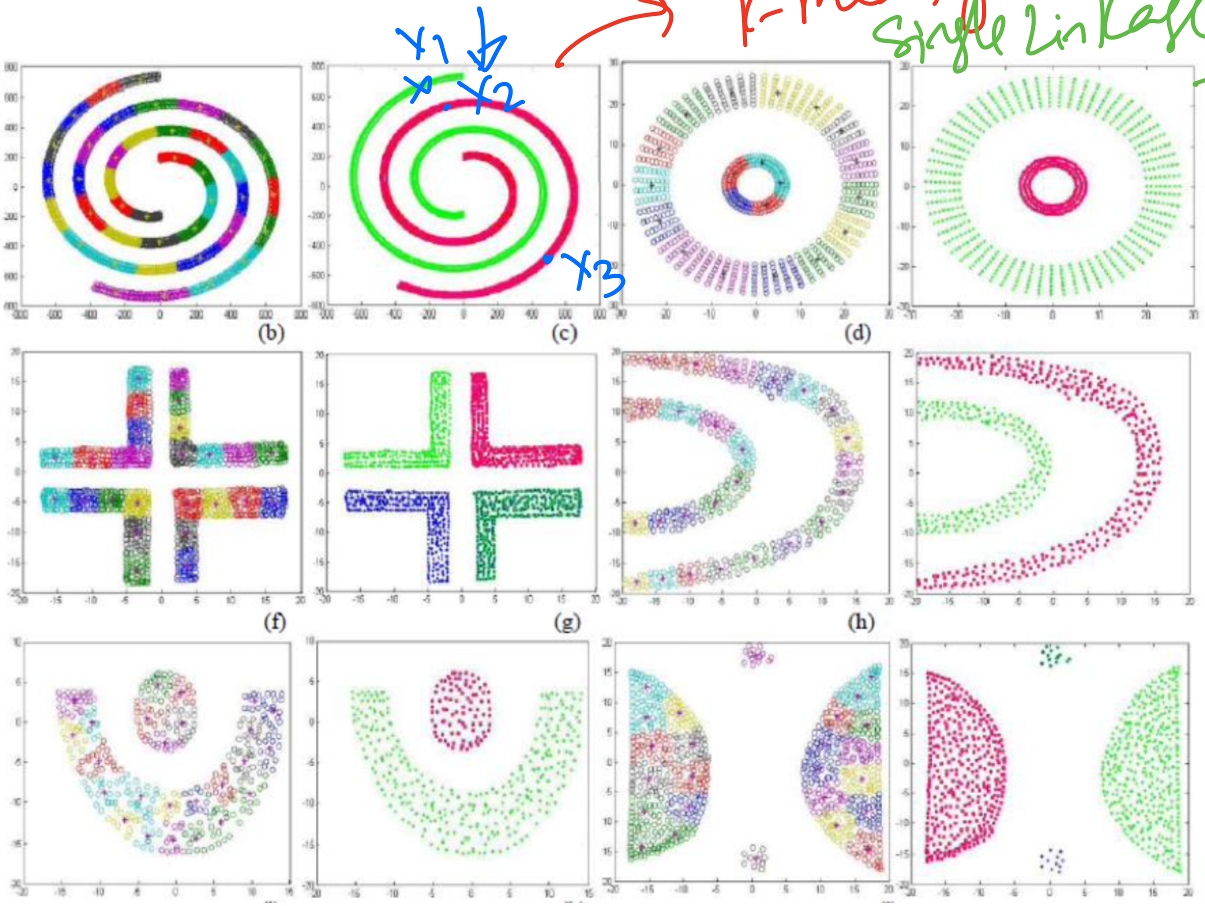
Clustering - Hard cases

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

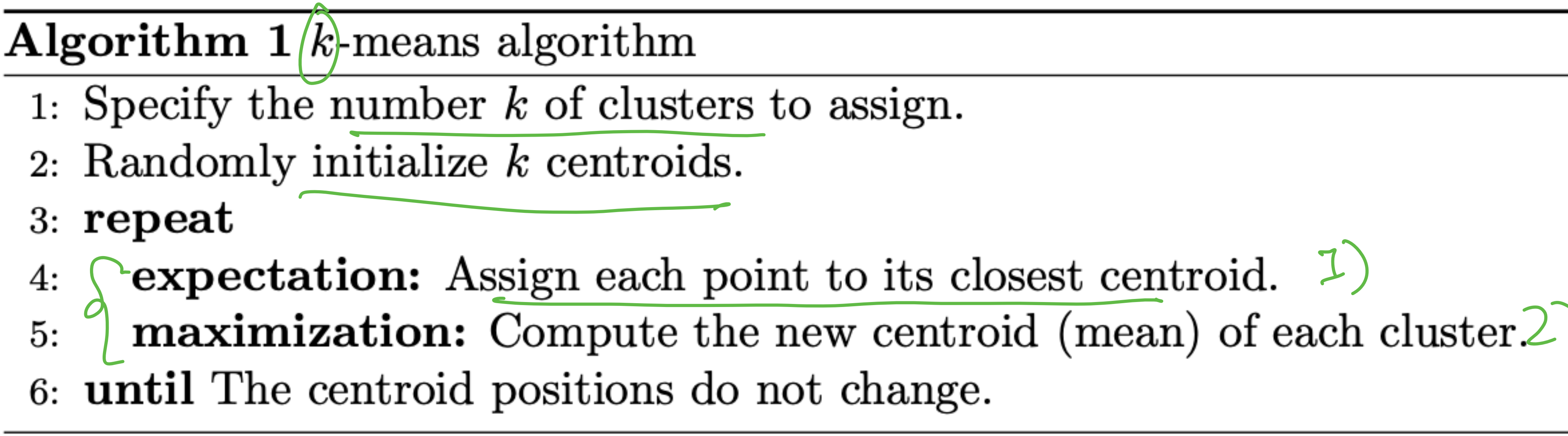
- Distance does not determine clusters

K-means fails

single linkage clustering works



k-means



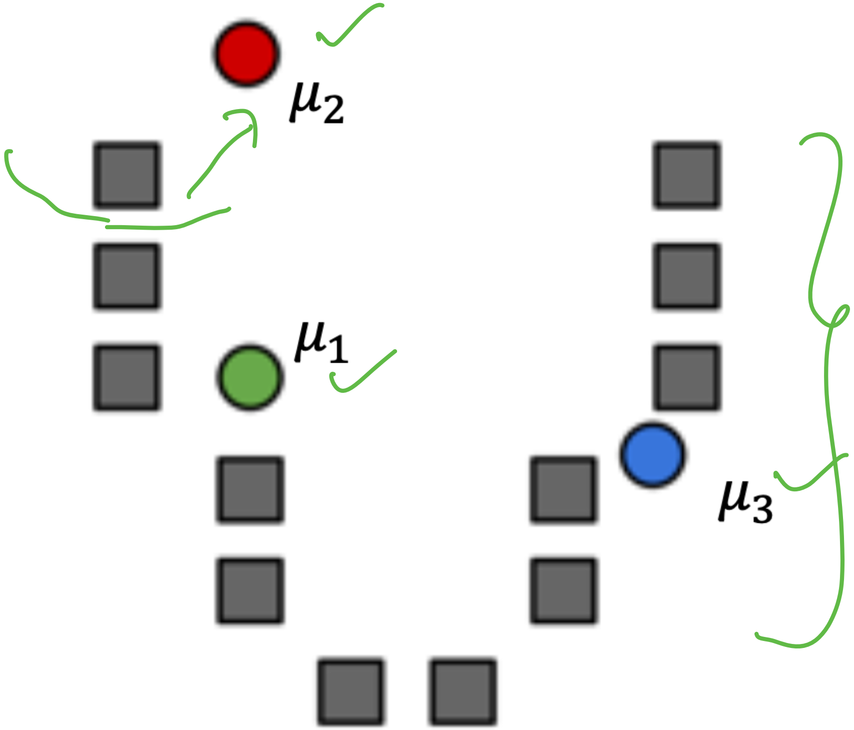
Algorithm 1 k-means algorithm

```
Specify the number $k$ of clusters to assign.
Randomly initialize $k$ centroids.
repeat
    expectation: Assign each point to its closest centroid. 1)
    maximization: Compute the new centroid (mean) of each cluster. 2)
until The centroid positions do not change.
```

k-means Clustering

Start by choosing the initial cluster centroids

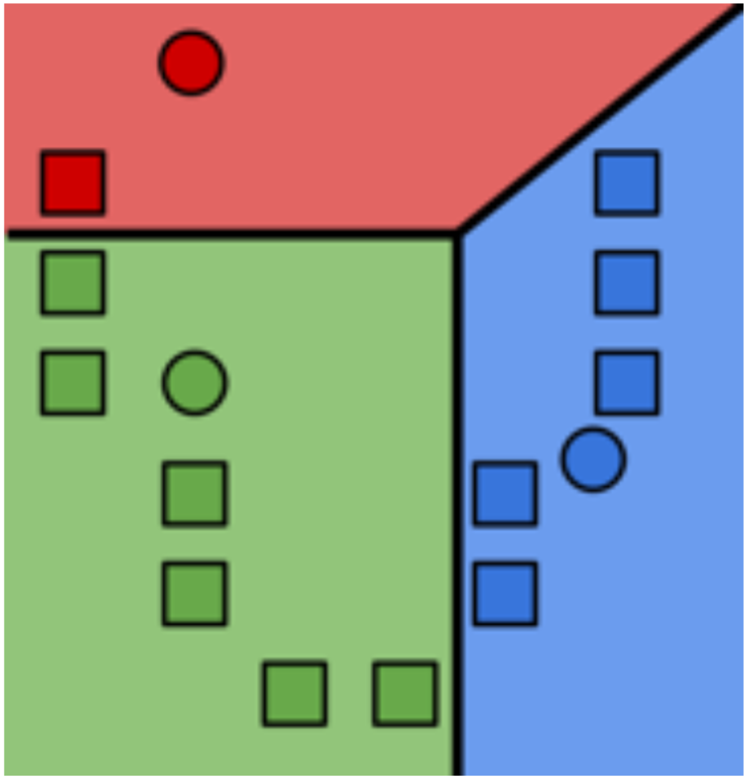
- A common default choice is to choose centroids at random
- Will see later that there are smarter ways of initializing



k-means Clustering

Assign each example to its closest cluster centroid

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

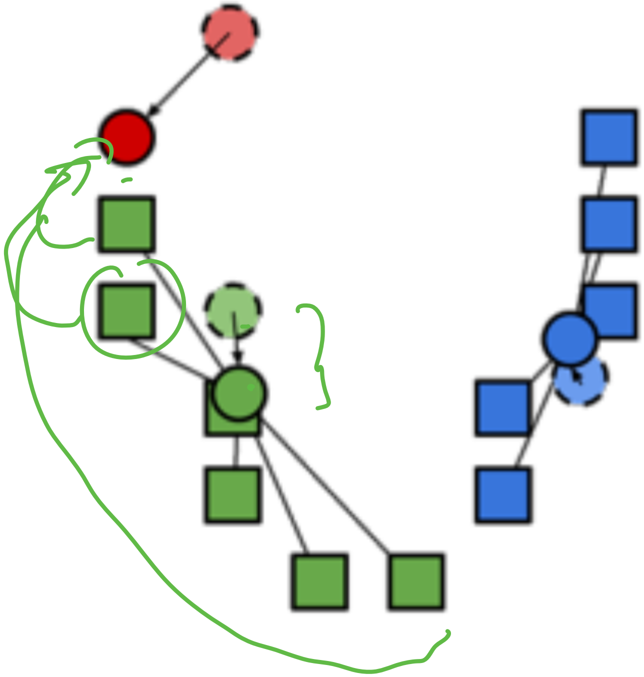


k-means Clustering

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!

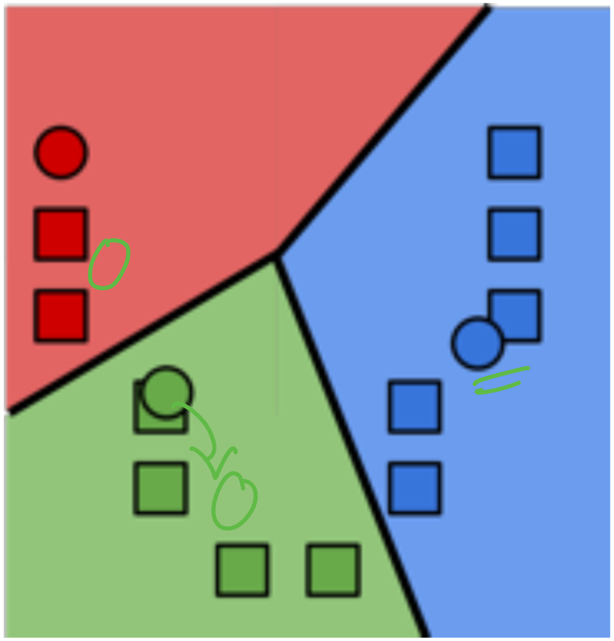


k-means Clustering

Repeat Steps 1 and 2 until convergence

Will it converge? Yes! Stop when

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



What will it converge to?

- Global optimum
- [✓ Local optimum]
- Neither

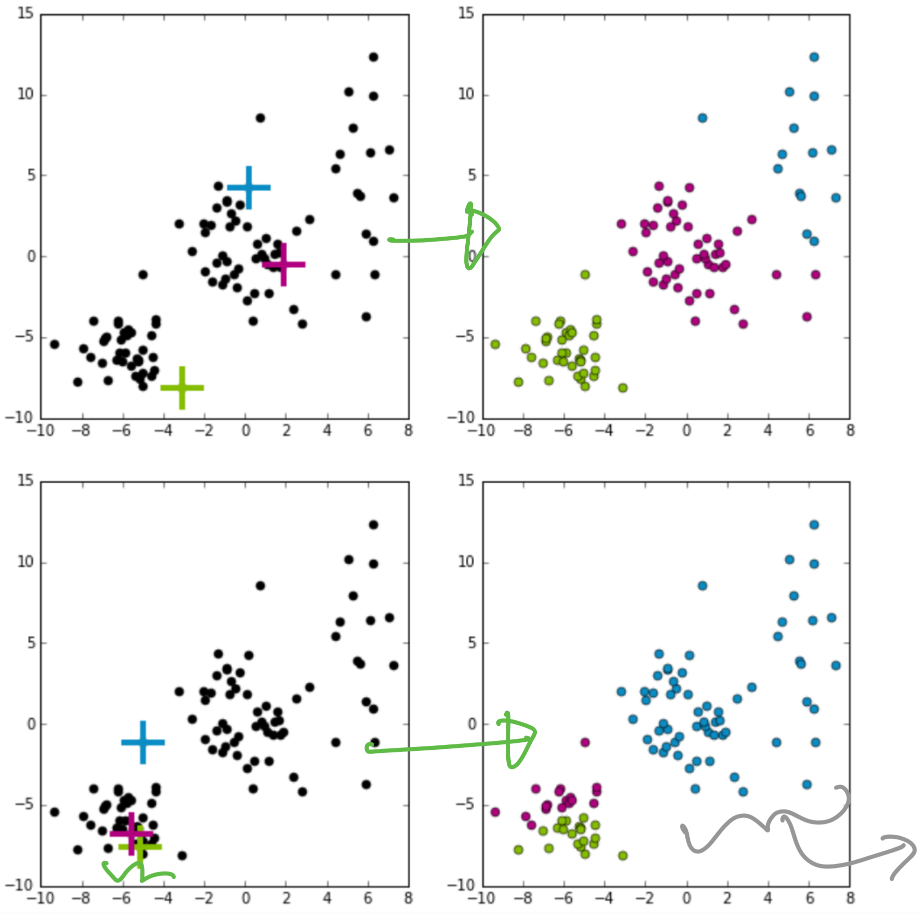
k-means Local Optima

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

- Initial settings will greatly impact results!

1

2



Large Intra-Cluster Distance

ICE #4: Which cluster is better?

Intra-cluster distance

Consider that k-means was applied with two different starting points and produced the following two sets of clusters: $C_1 = [(1,2),(2,3),(2,1)]$, $C_2 = [(4,5),(5,7),(6,4)]$ and $C_3 = [(3,5),(3,4),(5,6)]$ and $D_1 = [(1,2),(4,5),(3,5)]$, $D_2 = [(2,3),(2,1),(6,4)]$, $D_3 = [(5,7),(3,4),(5,6)]$. Which of these two algorithms has a lower $\frac{\text{Inter-cluster Distance}}{\text{Intra-cluster Distance}}$. Let inter-cluster distance of an algorithm be given by the minimum inter-cluster distance of any two clusters in the clustering and intra-cluster distance of an algorithm be given by the maximum of the intra-cluster distance of any two clusters in the clustering.

1. Algorithm 1
2. Algorithm 2
3. Both are equally good
4. Neither

$$\frac{\min_{i} IC(C_i, C_j)}{\max_{k} Intra(C_k)}$$

Improving kMeans - kMeans++

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

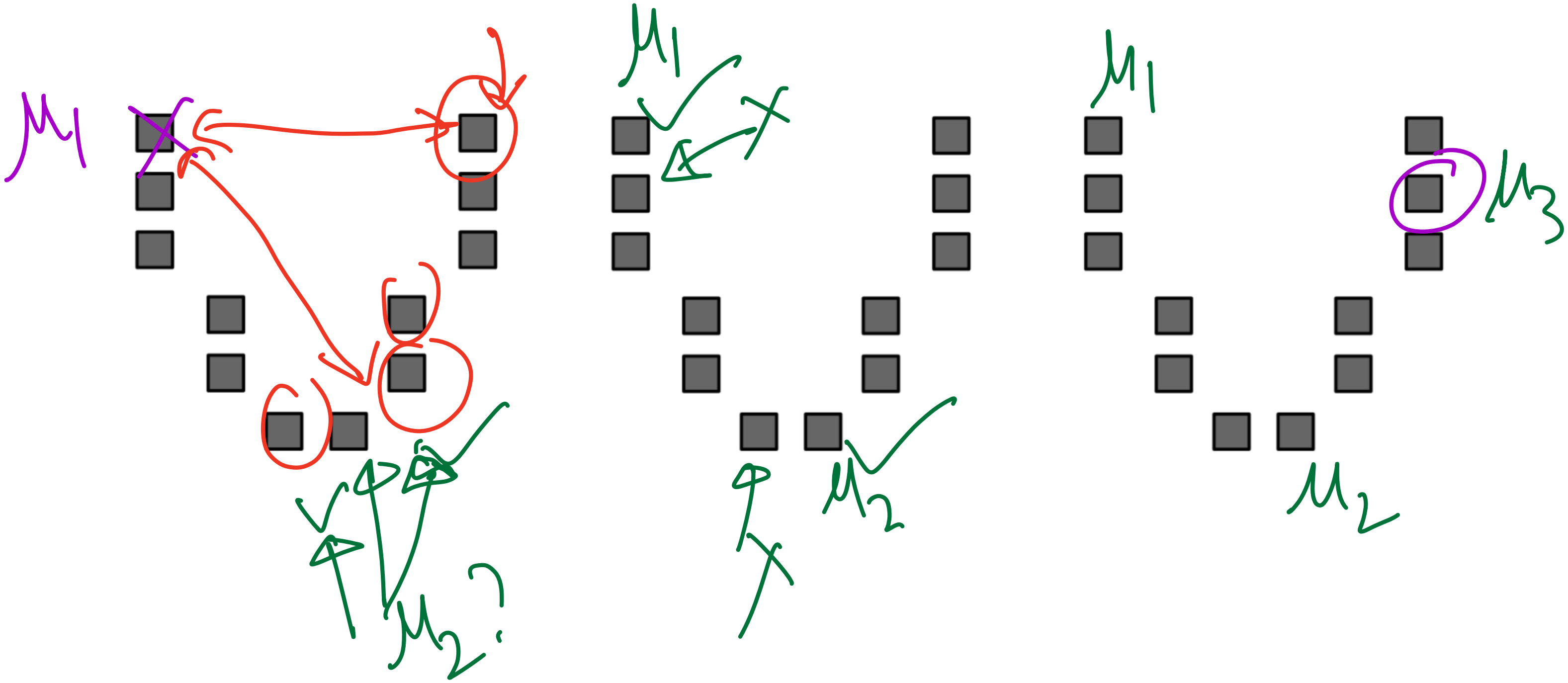
μ_1 - centroid 1

Improving kMeans - kMeans++

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 summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.

k-means summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.

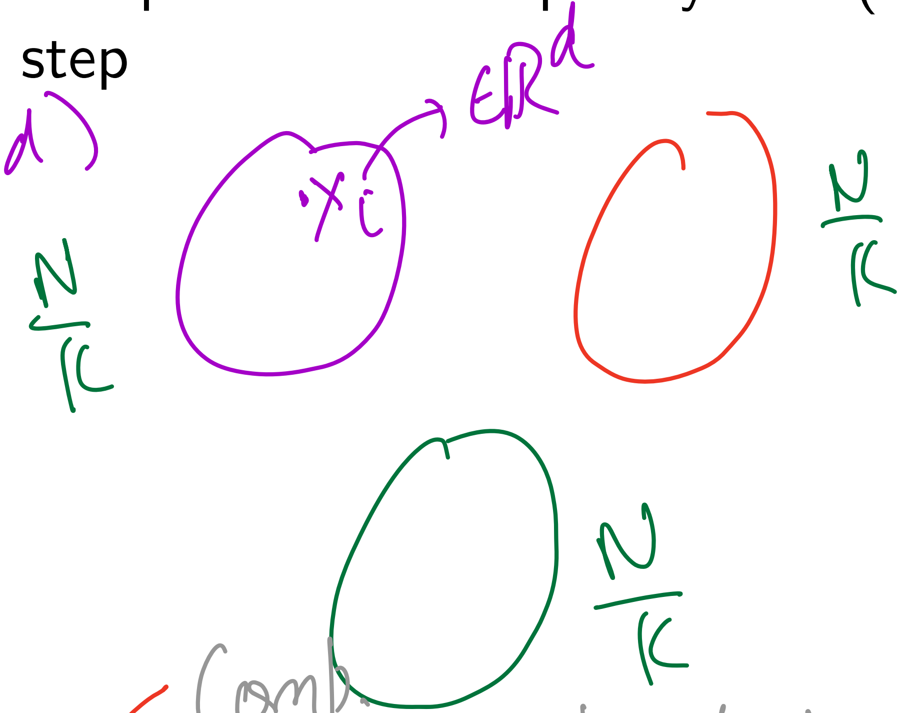
k-means summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.
3. k-means has a computational complexity of $O(Nd)$ for $\underline{\underline{C}}$ step and $O(Nkd)$ for A step

C-step (Centroid)

$\left(\frac{N}{K} d\right) \times K$

$O(Nd)$



N data points

K clusters

(K=3)

A-step

$O(Ndk)$ ✓ → Comp.

Complexity of k-means

k-means summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.
3. k-means has a computational complexity of $O(Nd)$ for C step and $O(Nkd)$ for A step
4. Counter examples of clusters that may not be clustered well by k-means. Can use other methods for this.

k-means summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.
3. k-means has a computational complexity of $O(Nd)$ for C step and $O(Nkd)$ for A step
4. Counter examples of clusters that may not be clustered well by k-means. Can use other methods for this.
5. k-means++ - Improvement on k-means and yields better quality clusters

k-means summary

1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.
3. k-means has a computational complexity of $O(Nd)$ for C step and $O(Nkd)$ for A step
4. Counter examples of clusters that may not be clustered well by k-means. Can use other methods for this.
5. k-means++ - Improvement on k-means and yields better quality clusters
6. Both k-means and k-means++ suffer from the clusters being spherical in nature. What if the true cluster shapes look different? Next lecture: Kernel k-means and Agglomerative clustering!

k-means summary

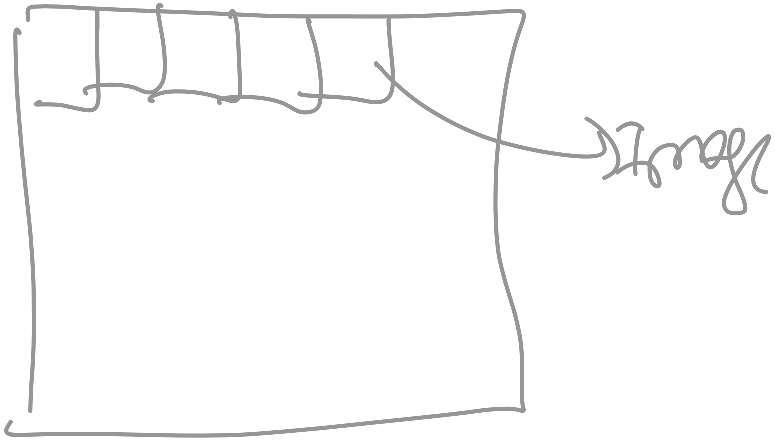
1. k-means - A generic clustering algorithm that can take N data points and group them into K clusters based on Euclidean distance.
2. Clusters make sense if cluster division makes sense based on Euclidean distance.
3. k-means has a computational complexity of $O(Nd)$ for C step and $O(Nkd)$ for A step
4. Counter examples of clusters that may not be clustered well by k-means. Can use other methods for this.
5. k-means++ - Improvement on k-means and yields better quality clusters
6. Both k-means and k-means++ suffer from the clusters being spherical in nature. What if the true cluster shapes look different? Next lecture: Kernel k-means and Agglomerative clustering!
7. Clustering can help with cold-start problem. E.g. recommending new products!

Clustering in 2 dimensions - tSNE!

Clustering for Data Visualization

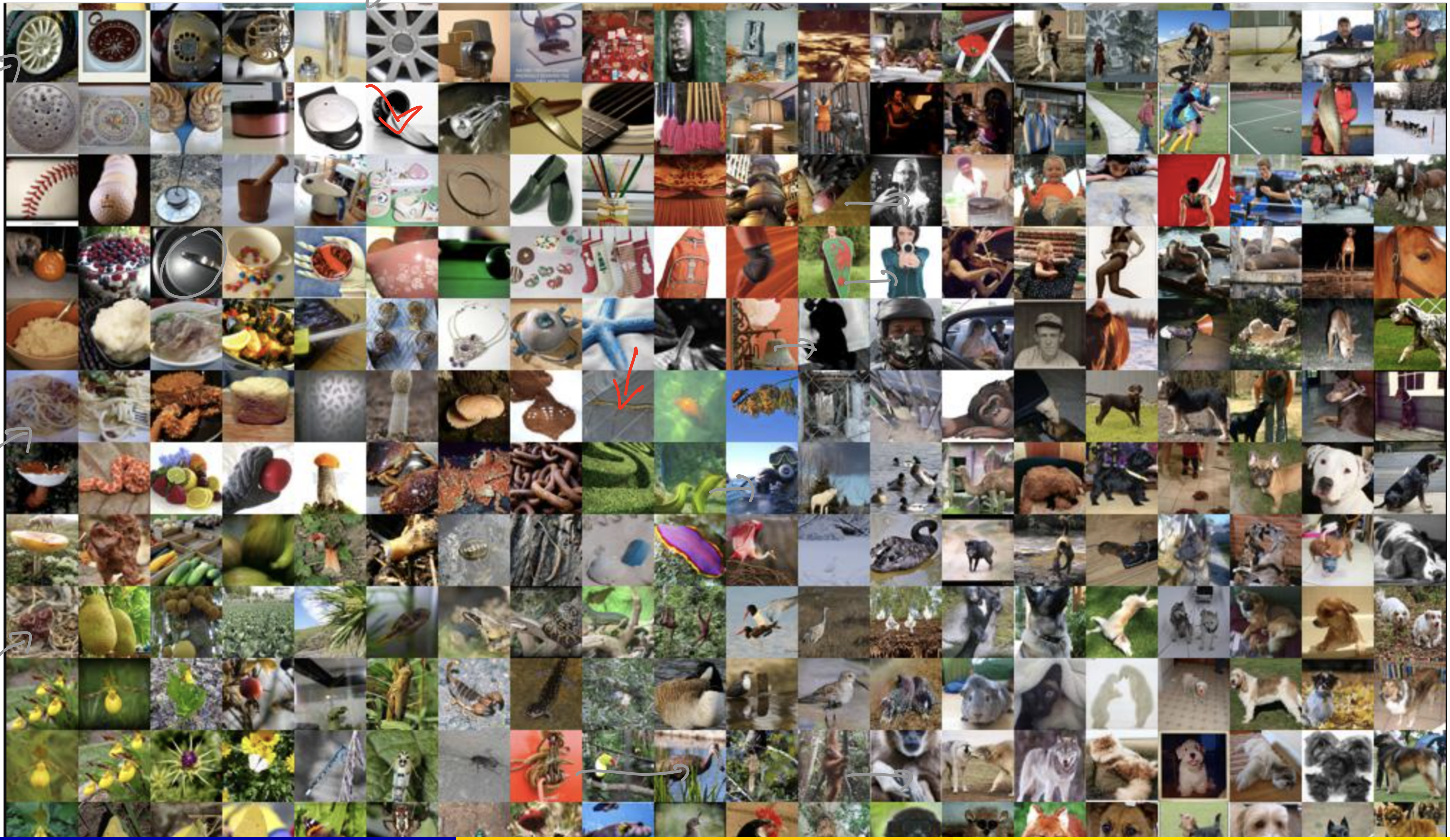
Images

Let's say we had 1000 images and wanted to "cluster" them onto a super-grid of images so that similar images are closely placed on the super-grid and dis-similar are placed further away. k-means clustering will only get us half-way there!



Data Visualization: Stochastic Neighborhood Embeddings (SNE)!

(tSNE) t-Stochastic NE



SNE

1000x1000

High-level Idea

Find an embedding of images in 2 dimensions that put similar images close to each other and dis-similar images further away from each other.

SNE

High-level Idea

Find an embedding of images in 2 dimensions that put similar images close to each other and dis-similar images further away from each other.

Soft clustering

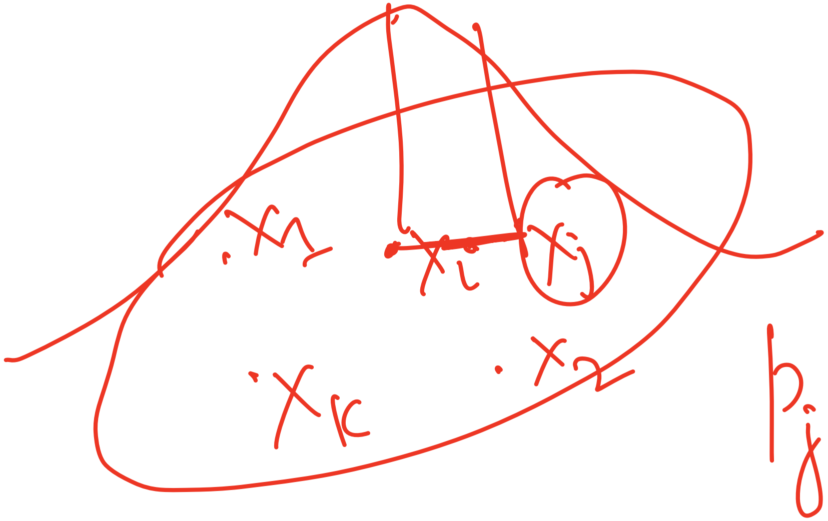
We don't have a K here. But if you look at any neighborhood of the super grid of images - ~~They~~ will look similar! We can call this soft-clustering.

SNE

Similarity measure through Probabilities

Let $x_1, x_2, \ldots$ represent features of the data in their original dimensions (e.g. images).

$$p_{j|i} = \underbrace{\frac{e^{-\|x_i - x_j\|_2^2/2\sigma_i^2}}{\sum_{k \neq i} e^{-\|x_i - x_k\|_2^2/2\sigma_i^2}}}_{} \rightarrow \text{normal / Gaussian}$$



$$e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$p_{j|i} \propto e^{-\frac{\|x_i - x_j\|_2^2}{2\sigma_i^2}}$$

SNE

Similarity measure through Probabilities

Let $x_1, x_2, \ldots$ represent features of the data in their original dimensions (e.g. images).

x_i – Image i

$$p_{j|i} = \frac{e^{-\|x_i - x_j\|_2^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|x_i - x_k\|_2^2/2\sigma_i^2}}$$

Low-dimensional embedding Probabilities

y_i – Low dim embed of Image i

Let $y_1, y_2, \ldots$ represent features of the data in lower (embedded) dimensions (e.g. 2 dimensions).

Low-dimensional

2 dimensional

$$q_{j|i} = \frac{e^{-\|y_i - y_j\|_2^2/2\sigma_i^2}}{\sum_{k\neq i} e^{-\|y_i - y_k\|_2^2/2\sigma_i^2}}$$

Estimating low-dimensional embeddings in SNE

A similarity measure for Probabilities - KL Divergence

$$KL(\underline{p}||\underline{q}) = \sum_{i=1}^{d} p_i \log \frac{p_i}{q_i}$$

Estimating low-dimensional embeddings in SNE

A similarity measure for Probabilities - KL Divergence

$$KL(p||q) = \sum_{i=1}^{d} p_i \log \frac{p_i}{q_i}$$

Loss function

$$\underline{L}(y_1, y_2, \ldots, y_N) = \sum_{i=1}^{N} KL(\underline{P_i}||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

Gradient and GD

Gradient

$$\frac{\partial L}{\partial y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Gradient and GD

Gradient

$$\frac{\partial L}{\partial y_i} = 2\sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

GD

$$\underline{\underline{y^{t+1}}} = y^t - \eta \frac{\partial L}{\partial y}(y^t)$$

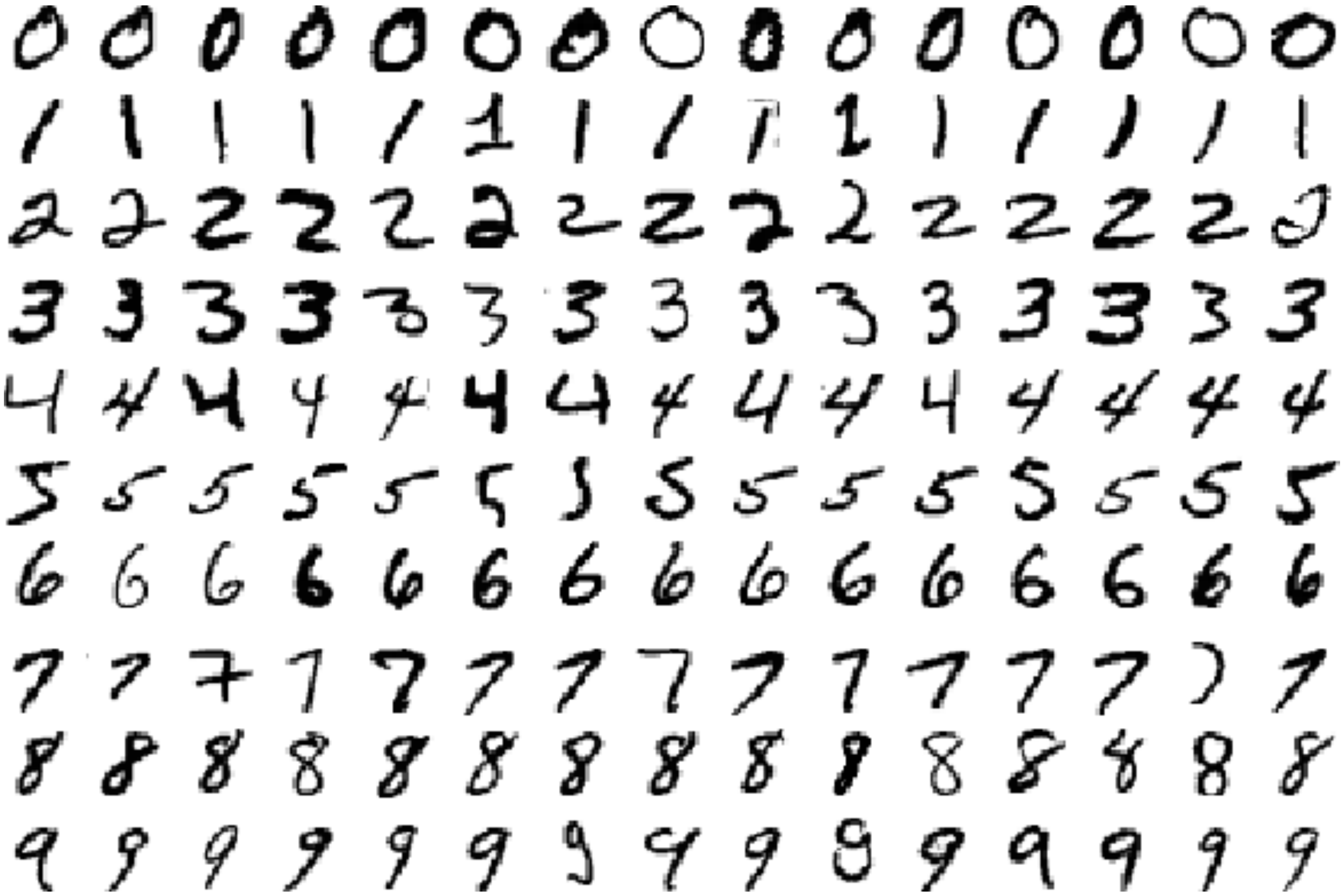
Image Chain

ICE #5 (3 mins break out)

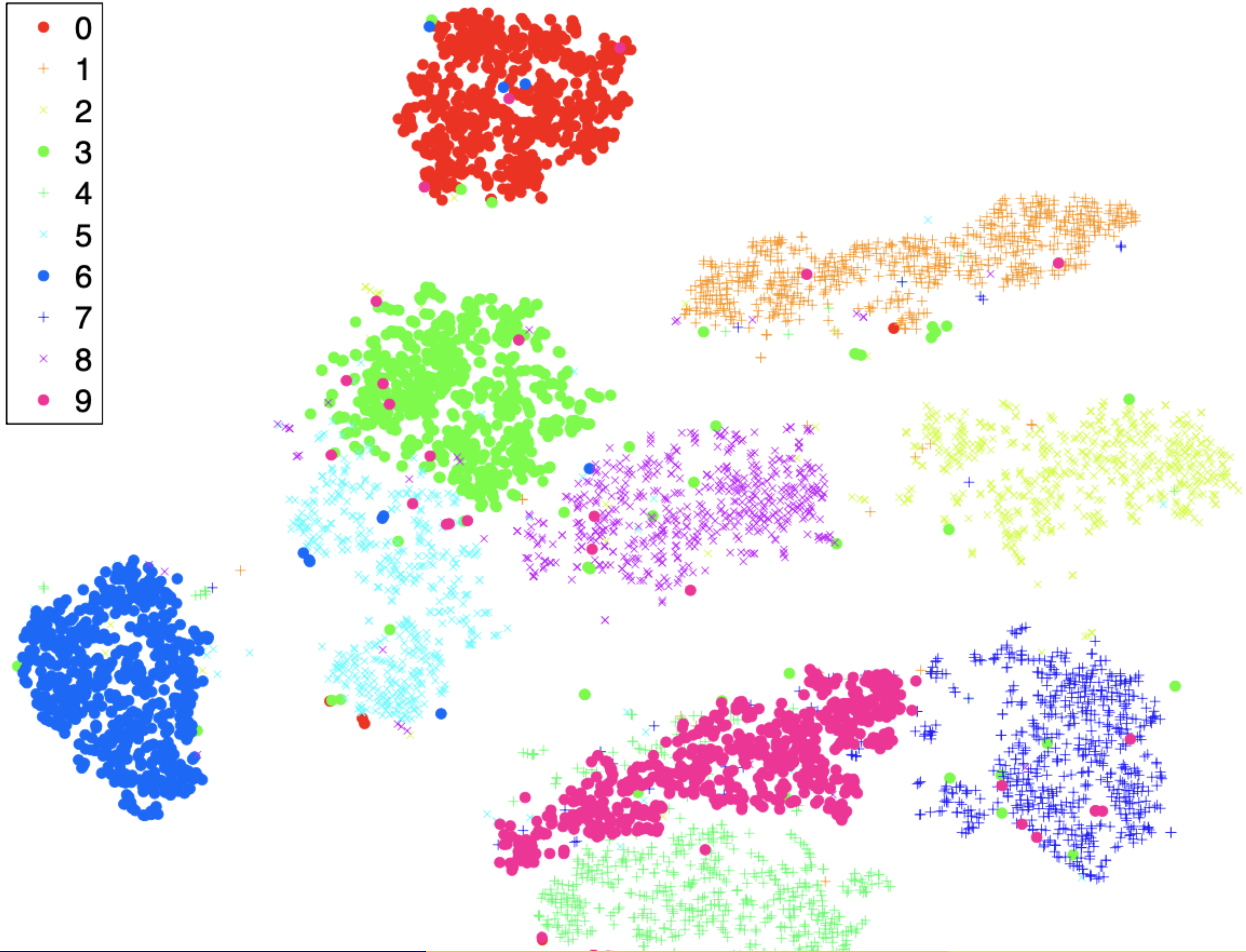
Let's say you want to create a video that has 1000 images (e.g. the one we looked at earlier) in a sequence so that the images in the video transforms smoothly from one to the next. How would you go about doing this if you learned a tSNE representation for the images?

How do we create this grid?

MNIST digits data set



MNIST tSNE embeddings



Summary

1. k-Means and k-Means++ for regular clustering
2. For hierarchies and free-form clustering - Agglomerative clustering methods
3. tSNE - Soft-clustering of images (We will have this be part of Assignment 2)