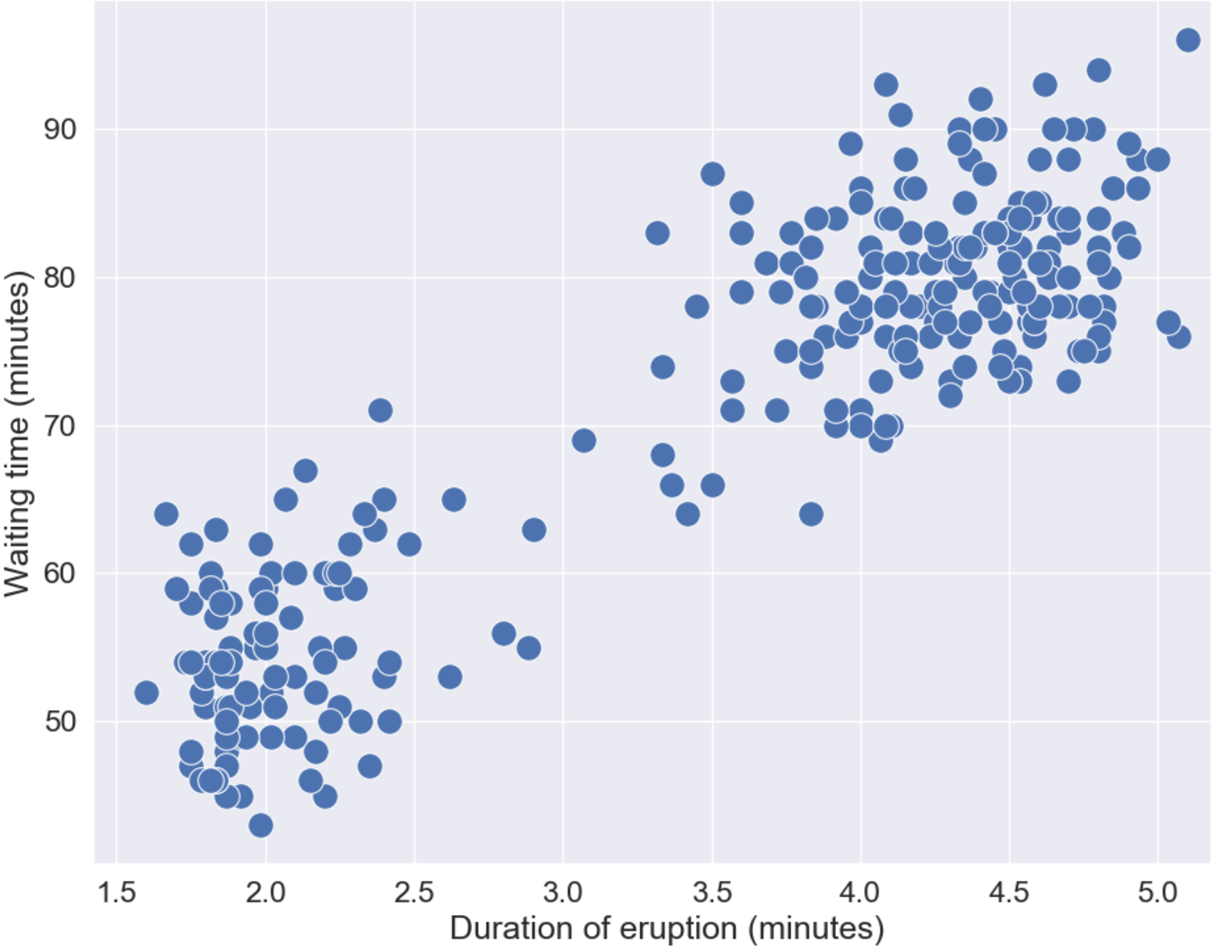
# Machine Learning

## Clustering

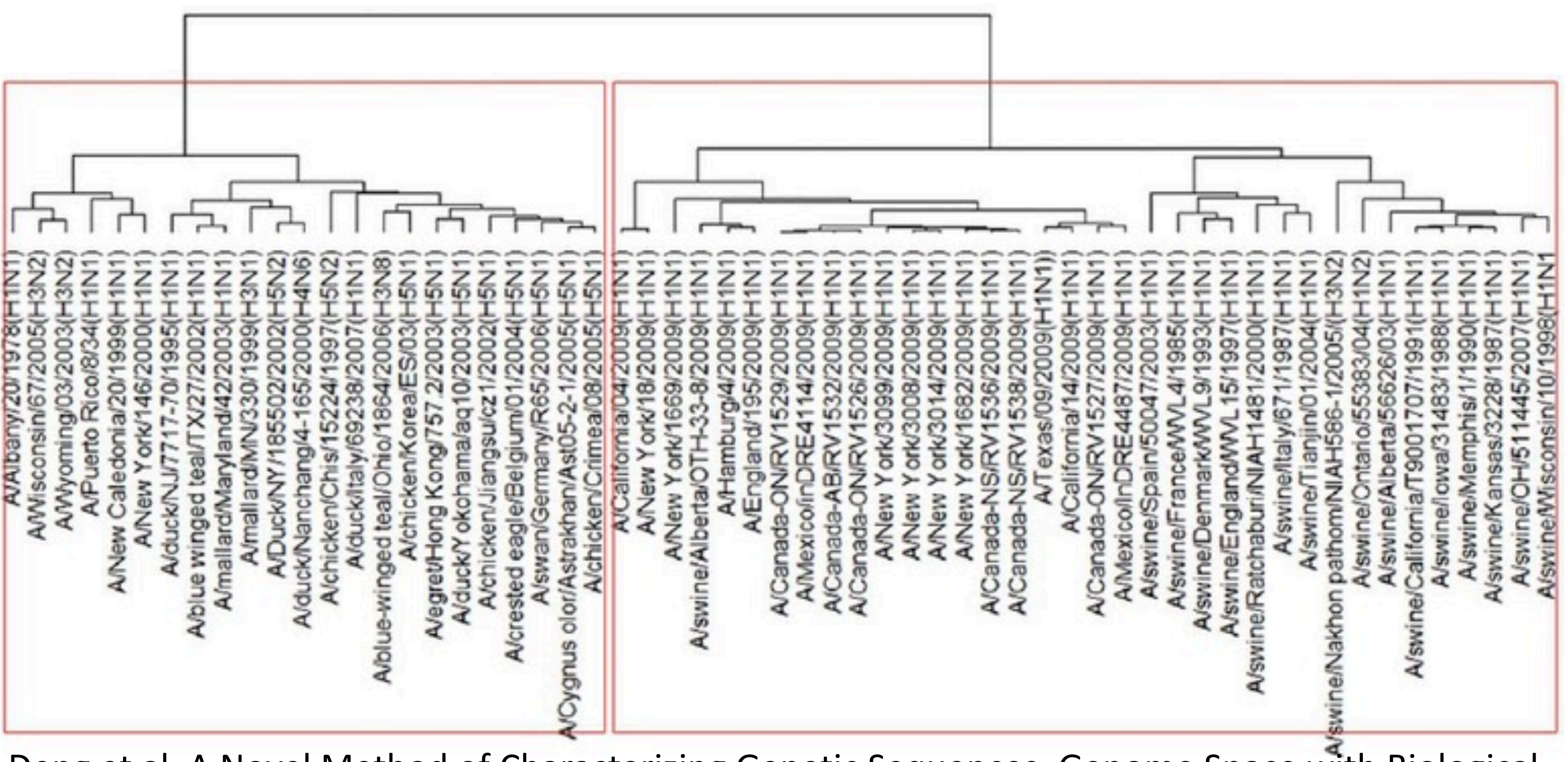
Zach Wood-Doughty and Bryan Pardo
Machine Learning: CS 349 Fall 2021

Some slides borrowed from Mark Dredze and Prem Seetharaman, with inspiration from:
http://www.mit.edu/~9.54/fall14/slides/Class13.pdf and
https://people.eecs.berkeley.edu/~jordan/courses/294-fall09/lectures/clustering/slides.pdf

# Example: Eruptions at Old Faithful Geyser

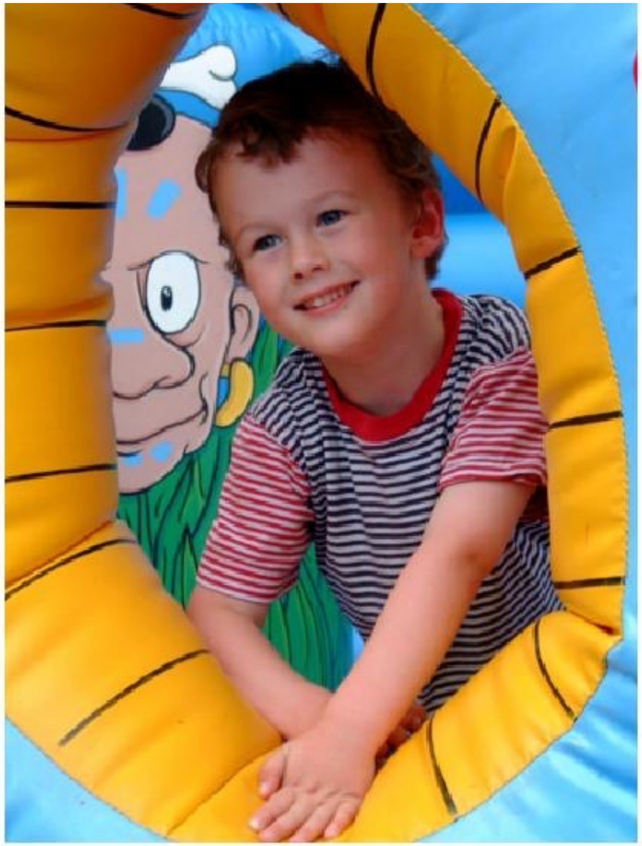


# Example: Clustering H1N1 Genomes

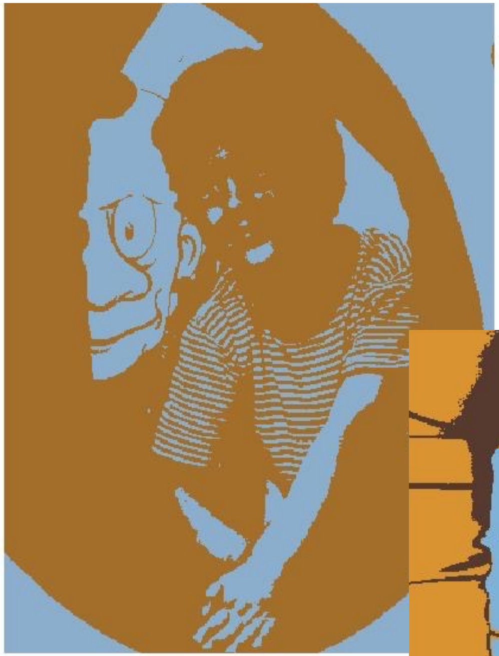

A/Albany/20/1978(H1N1)
A/Wisconsin/67/2005(H3N2)
A/Wyoming/03/2003(H3N2)
A/Puerto Rico/8/34(H1N1)
A/New Caledonia/20/1999(H1N1)
A/New York/146/2000(H1N1)
A/duck/NJ/7717-70/1995(H1N1)
A/blue winged teal/TX/27/2002(H1N1)
A/mallard/Maryland/42/2003(H1N1)
A/mallard/MN/330/1999(H3N1)
A/Duck/NY/185502/2002(H5N2)
A/duck/Nanchang/4-165/2000(H4N6)
A/chicken/Chis/15224/1997(H5N2)
A/duck/Italy/69238/2007(H1N1)
A/blue-winged teal/Ohio/1864/2006(H3N8)
A/chicken/Korea/ES/03(H5N1)
A/egret/Hong Kong/757.2/2003(H5N1)
A/duck/Yokohama/aq10/2003(H5N1)
A/chicken/Jiangsu/cz1/2002(H5N1)
A/crested eagle/Belgium/01/2004(H5N1)
A/swan/Germany/R65/2006(H5N1)
A/Cygnus olor/Astrakhan/Ast05-2-1/2005(H5N1)
A/chicken/Crimea/08/2005(H5N1)
A/California/04/2009(H1N1)
A/New York/18/2009(H1N1)
A/New York/1669/2009(H1N1)
A/swine/Alberta/OTH-33-8/2009(H1N1)
A/Hamburg/4/2009(H1N1)
A/England/195/2009(H1N1)
A/Canada-ON/RV1529/2009(H1N1)
A/Mexico/InDRE4114/2009(H1N1)
A/Canada-AB/RV1532/2009(H1N1)
A/Canada-ON/RV1526/2009(H1N1)
A/New York/3099/2009(H1N1)
A/New York/3008/2009(H1N1)
A/New York/3014/2009(H1N1)
A/New York/1682/2009(H1N1)
A/Canada-NS/RV1536/2009(H1N1)
A/Canada-NS/RV1538/2009(H1N1)
A/Texas/09/2009(H1N1)
A/California/14/2009(H1N1)
A/Canada-ON/RV1527/2009(H1N1)
A/Mexico/InDRE4487/2009(H1N1)
A/swine/Spain/50047/2003(H1N1)
A/swine/France/WVL4/1985(H1N1)
A/swine/Denmark/WVL9/1993(H1N1)
A/swine/England/WVL15/1997(H1N1)
A/swine/Ratchaburi/NIAH1481/2000(H1N1)
A/swine/Italy/671/1987(H1N1)
A/swine/Tianjin/01/2004(H1N1)
A/swine/Nakhon pathom/NIAH586-1/2005/(H3N2)
A/swine/Ontario/55383/04(H1N2)
A/swine/Alberta/56626/03(H1N1)
A/swine/California/T9001707/1991(H1N1)
A/swine/Iowa/31483/1988(H1N1)
A/swine/Memphis/1/1990(H1N1)
A/swine/Kansas/3228/1987(H1N1)
A/swine/OH/511445/2007(H1N1)
A/swine/Wisconsin/10/1998(H1N1

Deng et al. A Novel Method of Characterizing Genetic Sequences: Genome Space with Biological Distance and Applications.

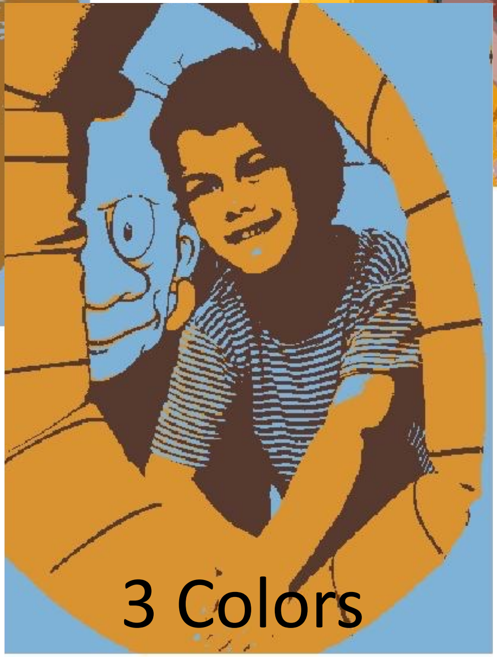
# Example: Color Segmentation



Original Image
16M colors

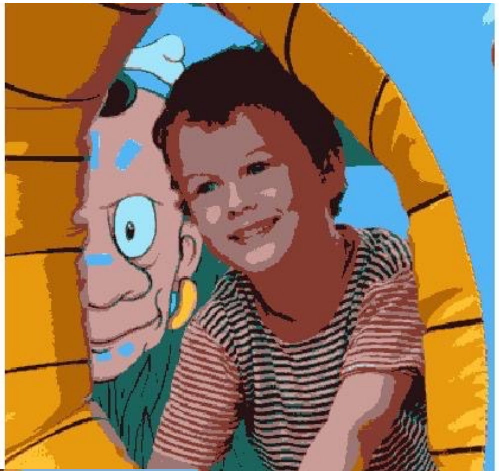


2 Colors



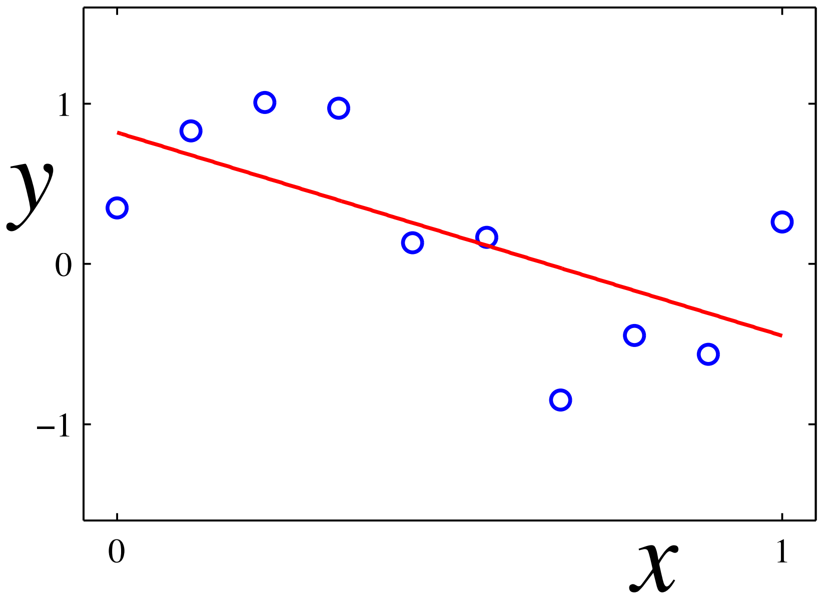
3 Colors

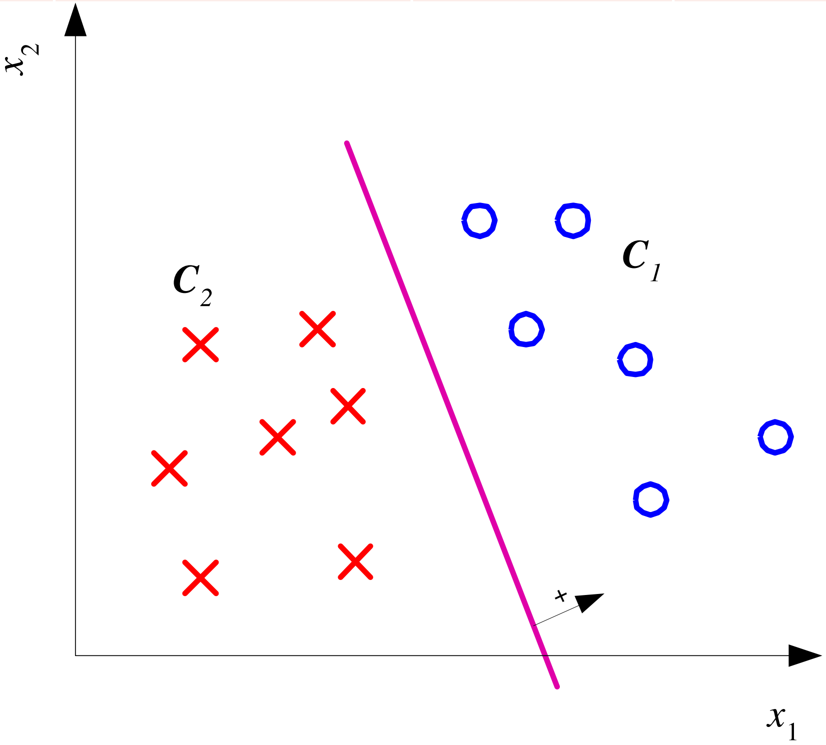
10 Colors



# Recap: Supervised Learning

| Number of Feet | Fur | Size | Has wings | Warm Blood | f(x) |
|---|---|---|---|---|---|
| 2 | No | S | Yes | Yes | 0 |
| **4** | **Yes** | **S** | **No** | **Yes** | **1** |





# Recall: Supervised Learning Tasks

There is a set of possible examples $X = \{\mathbf{x_1}, \ldots \mathbf{x_n}\}$

Each example is a **vector** of d **real valued attributes**

$$\mathbf{x_i} = \langle x_{i,1}, \ldots x_{i,d} \rangle$$

A target function maps *X* onto some **real or categorical value** *Y*

$$f : X \rightarrow Y$$

The DATA is a set of tuples <example, response value>

$$\{<\mathbf{x}_1, y_1>, \ldots <\mathbf{x_n}, y_n>\}$$

Find a hypothesis ***h*** such that...

$$\forall \mathbf{x}, h(\mathbf{x}) \approx f(\mathbf{x})$$

# Unsupervised Learning

- We no longer have labels!
- What can we do?

- We still can have a notion of **groups**
- Task: divide things into piles of similar things
- Classification found patterns that explained a label
  - We can find patterns that separate the data

# Clustering

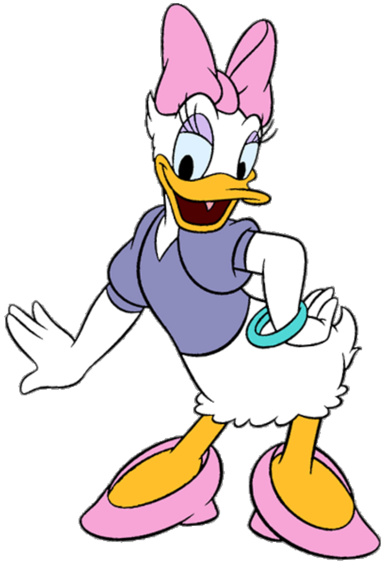
- Sort the data into clusters (groups)
- Examples that are in the same group are similar
  - Items in cluster are more similar to one another than to items not in the cluster
  - Ideally clusters correspond to (unknown) labels
- We don’t know what we will get!
  - What does it mean for two examples to be similar?
  - How do we measure the quality of our clusters?

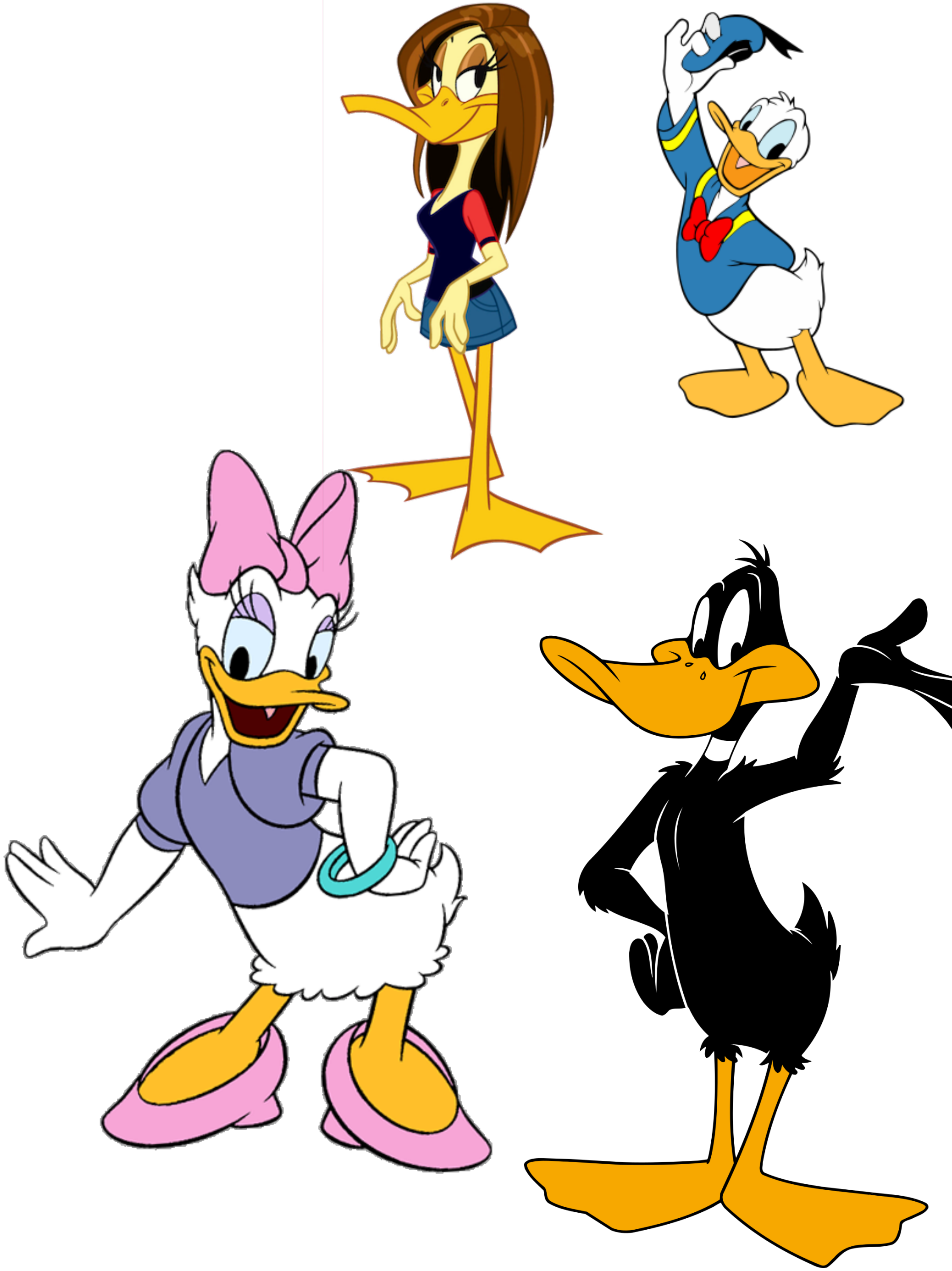
# Unsupervised Learning Tasks

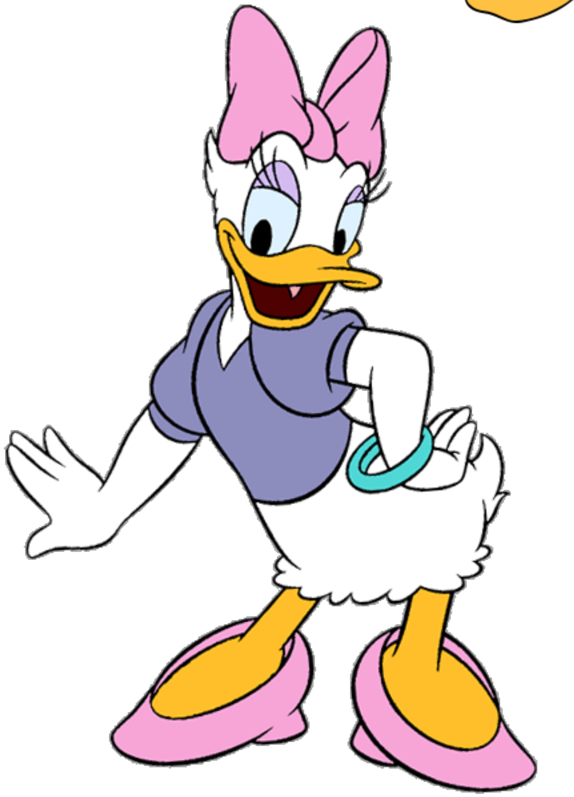
There is a set of possible examples $X = \{\mathbf{x_1}, \ldots \mathbf{x_n}\}$

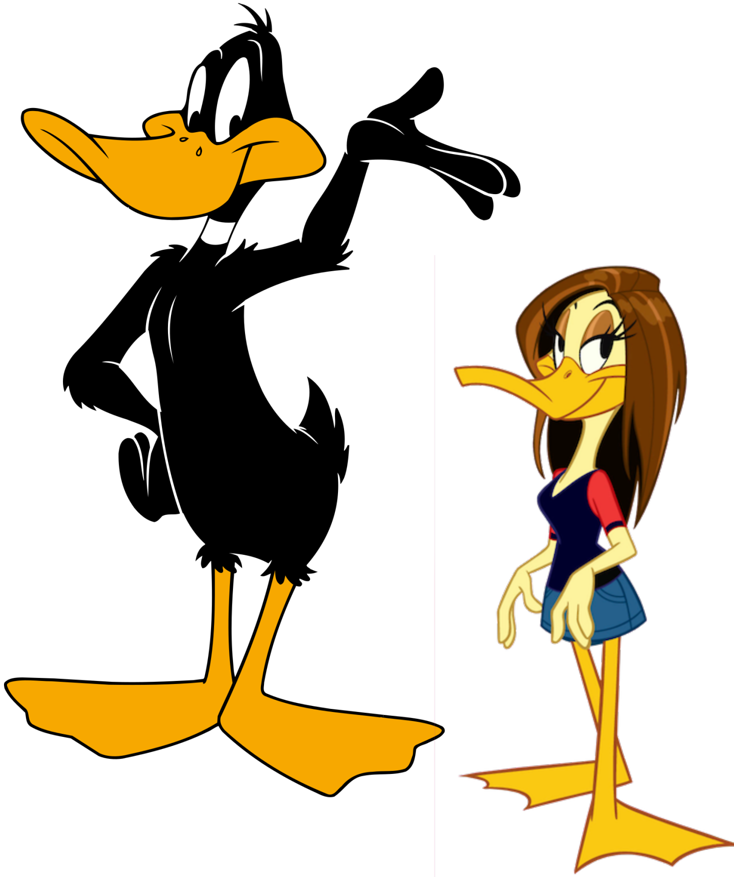
Each example is a **vector** of d **real valued attributes**

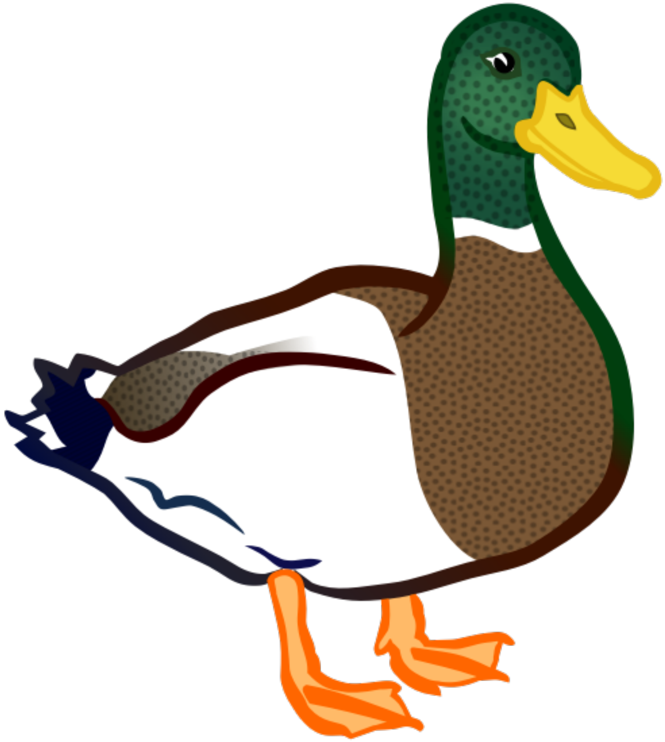
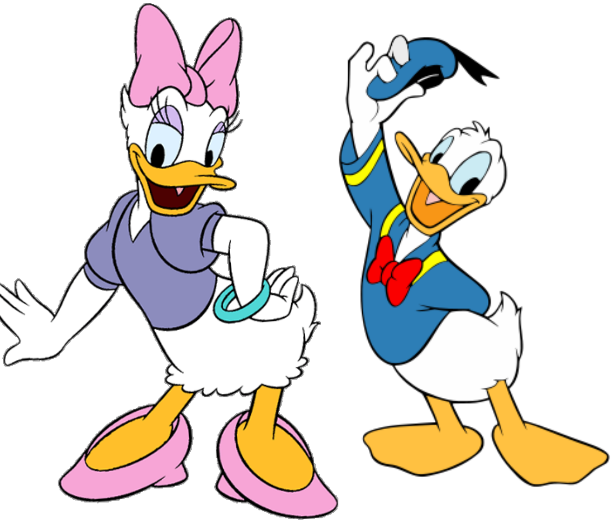
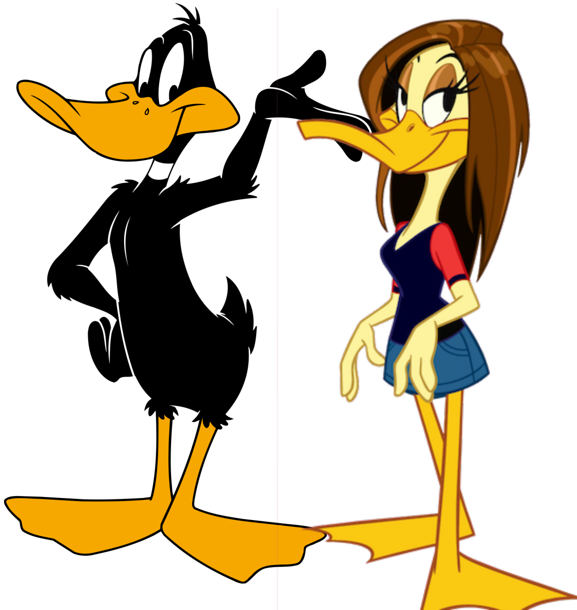
$$\mathbf{x_i} = \langle x_{i,1}, \ldots x_{i,d} \rangle$$









# Unsupervised Learning Tasks

There is a set of possible examples $X = \{\mathbf{x_1}, \ldots \mathbf{x_n}\}$

Each example is a **vector** of d **real valued attributes**

$$\mathbf{x_i} = \langle x_{i,1}, \ldots x_{i,d} \rangle$$

Assume some latent variable(s) z that correspond to the observed data

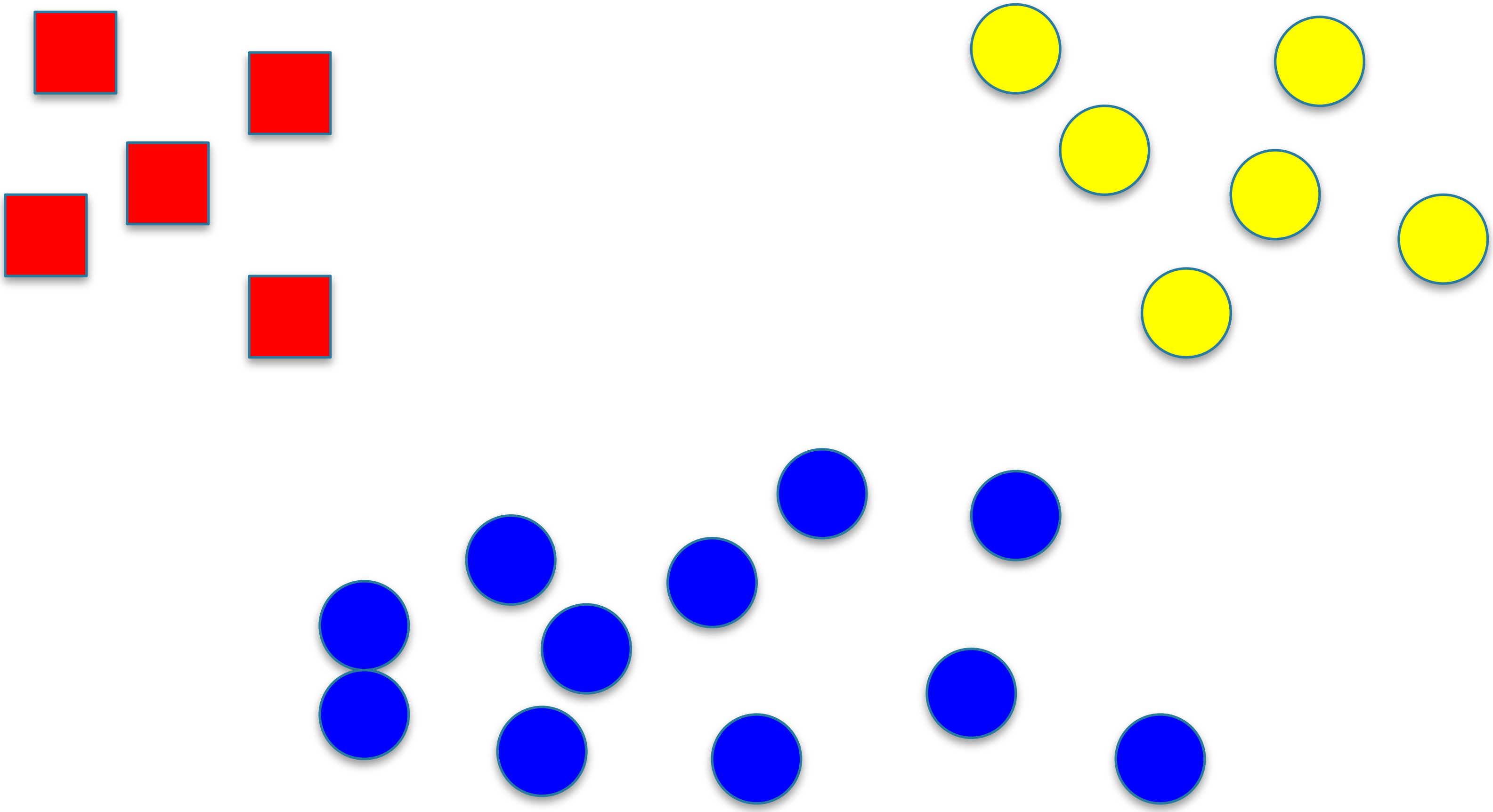
$$\{\langle \mathbf{x_1}, z_1 \rangle, \ldots \langle \mathbf{x_n}, z_n \rangle\}$$

Learn a way to assign examples to clusters such that both:

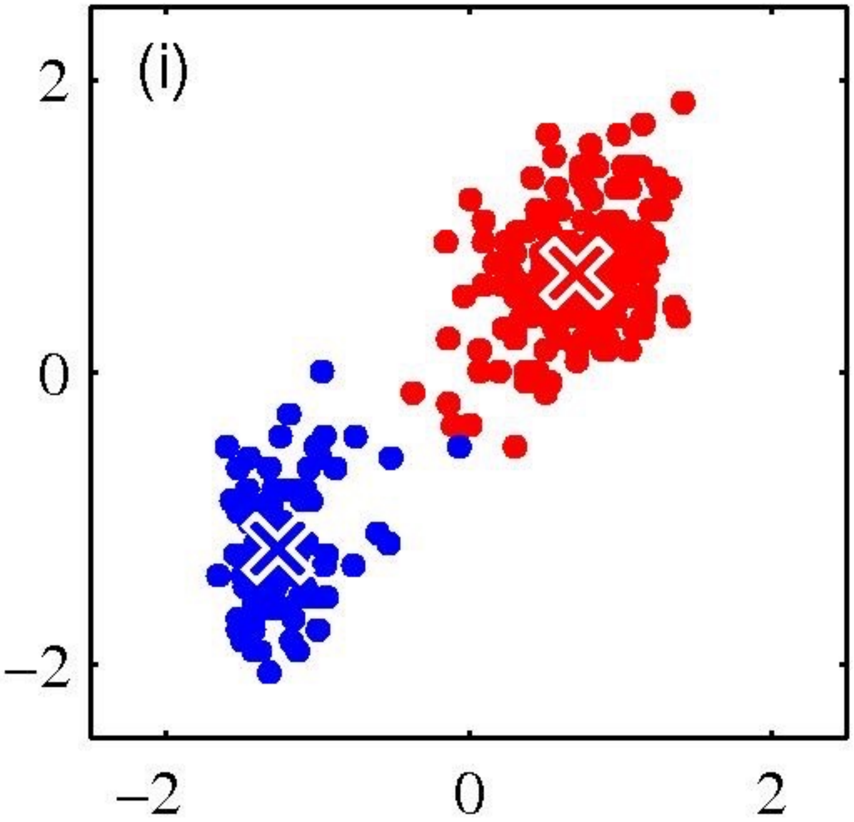
$$d(x_i, x_j) < \epsilon \Rightarrow z_i = z_j$$

$$d(x_i, x_j) > \epsilon \Rightarrow z_i \neq z_j$$

# Geometric Model



# Visualization: 2 Clusters



# Defining Clusters

- A cluster is a group of similar examples
- Define z as an indicator:

$$z_{n,k} \in \{0,1\}$$

- Value of 1 means that example n is in cluster k
- Define cluster k by a prototype:

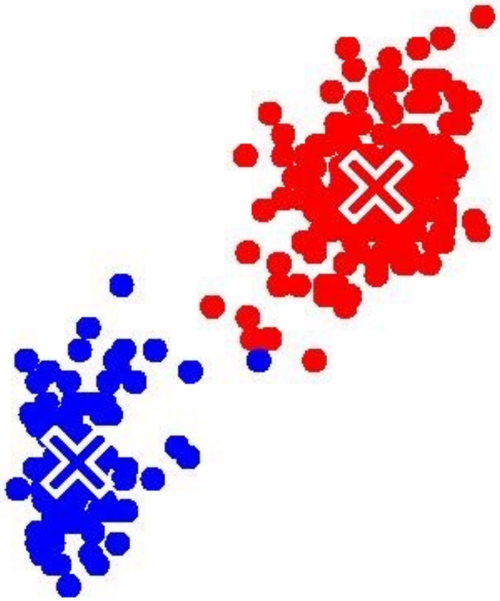
$$\mu_{\mathbf{k}} = \frac{\sum_{n=1}^{N} z_{n,k} \cdot \mathbf{x_n}}{\sum_{n=1}^{N} z_{n,k}}$$

# Clustering objective function

- Objective: maximize the similarity of every cluster
  - Each example in a cluster should be close to its prototypical example

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{n,k} \cdot d(\mathbf{x_n}, \mu_{\mathbf{k}})$$

# Learning

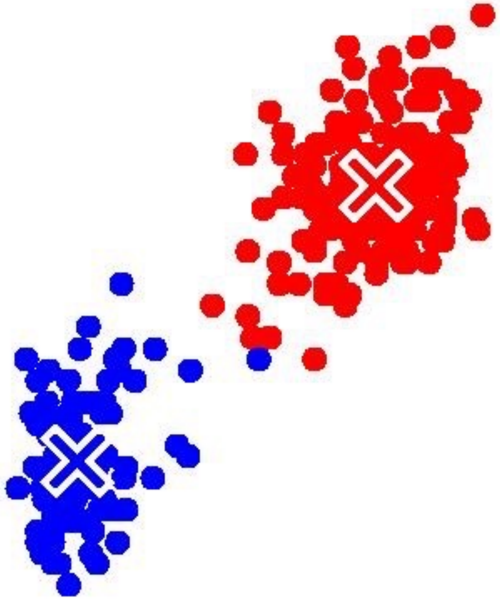


$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{n,k} \cdot d(\mathbf{x_n}, \mu_\mathbf{k})$$

- We’ll typically assume d is Euclidean distance
  - But it doesn’t have to be!
- We have two parameters: z and µ
- Want to pick those parameters to minimize J

# Learning

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{n,k} \cdot d(\mathbf{x_n}, \mu_\mathbf{k})$$



- Our two parameters depend on each other
- If we knew z we could set μ
    - Compute a cluster’s mean from its assigned examples
- If we knew μ we could set z
    - Assign each point to closest cluster

# Update Rules

$$z_{n,k} = \begin{cases} 1 & k = \arg\min_j d(\mathbf{x_n}, \mu_j) \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_{\mathbf{k}} = \frac{\sum_{n=1}^{N} z_{n,k} \cdot \mathbf{x_n}}{\sum_{n=1}^{N} z_{n,k}}$$

# Optimization and convergence

- Each update reduces the value of J
  - Therefore, algorithm will converge
  - (How would you prove this?)
- Note: J is non-convex
  - Not guaranteed to find an optimal clustering
  - Initial values matter, so random restarts may help

# Algorithm: K-Means

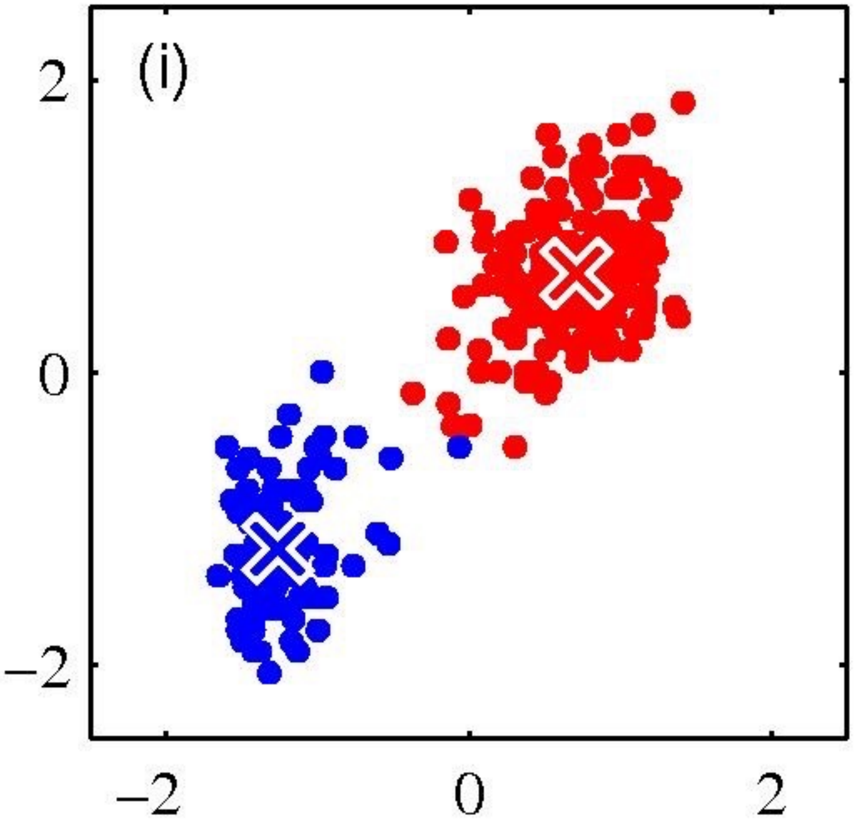
$$X = \{\mathbf{x_1}, \ldots \mathbf{x_n}\}$$
$$\mathbf{x_i} = \langle x_{i,1}, \ldots x_{i,d} \rangle$$

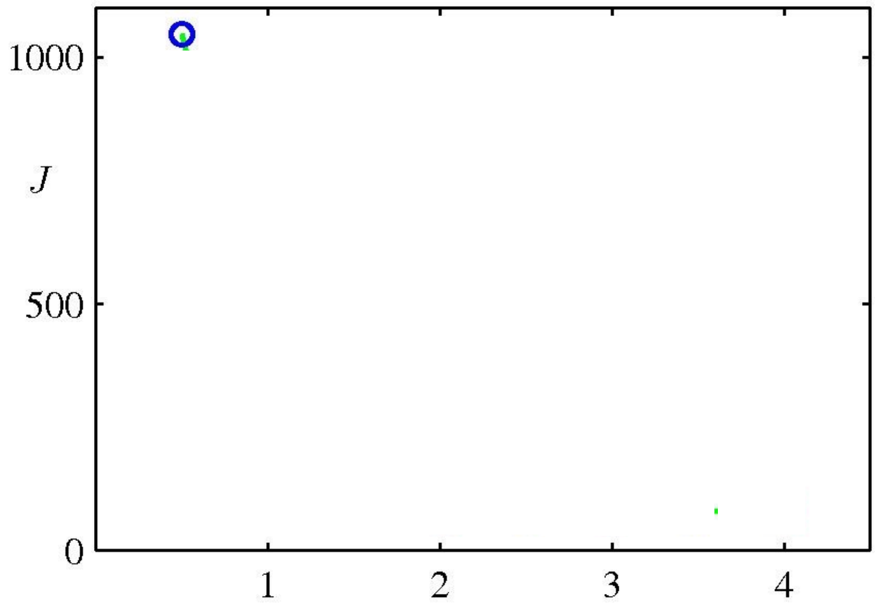
- Input data X and initialize μ
- Iteratively update until convergence:

$$z_{n,k} = \begin{cases} 1 & k = \arg\min_j d(\mathbf{x_n}, \mu_j) \\ 0 & \text{otherwise} \end{cases}$$

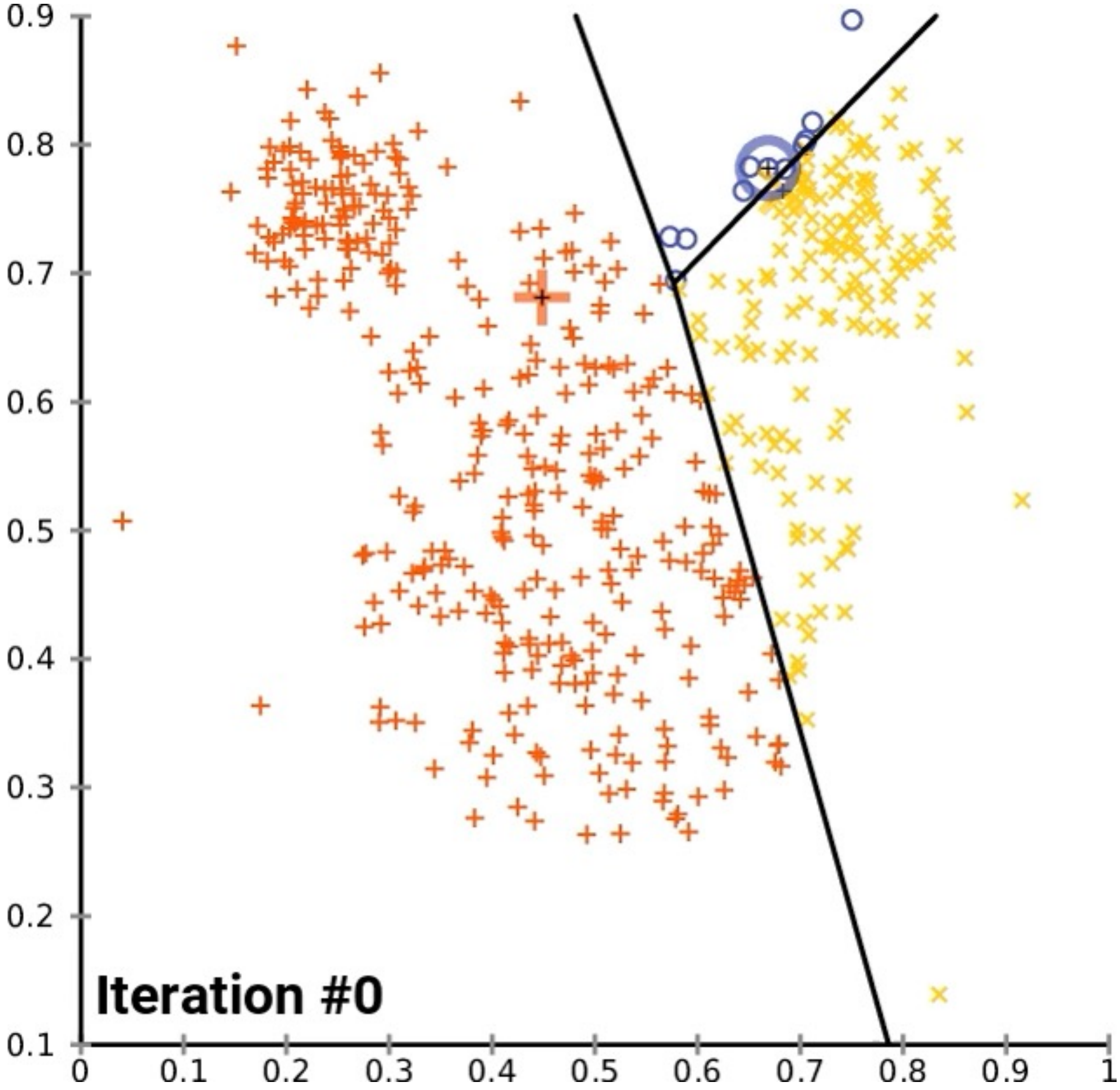
$$\mu_{\mathbf{k}} = \frac{\sum_{n=1}^{N} z_{n,k} \cdot \mathbf{x_n}}{\sum_{n=1}^{N} z_{n,k}}$$

# Visualization: 2 Clusters





https://upload.wikimedia.org/wikipedia/commons/e/ea/K-means_convergence.gif



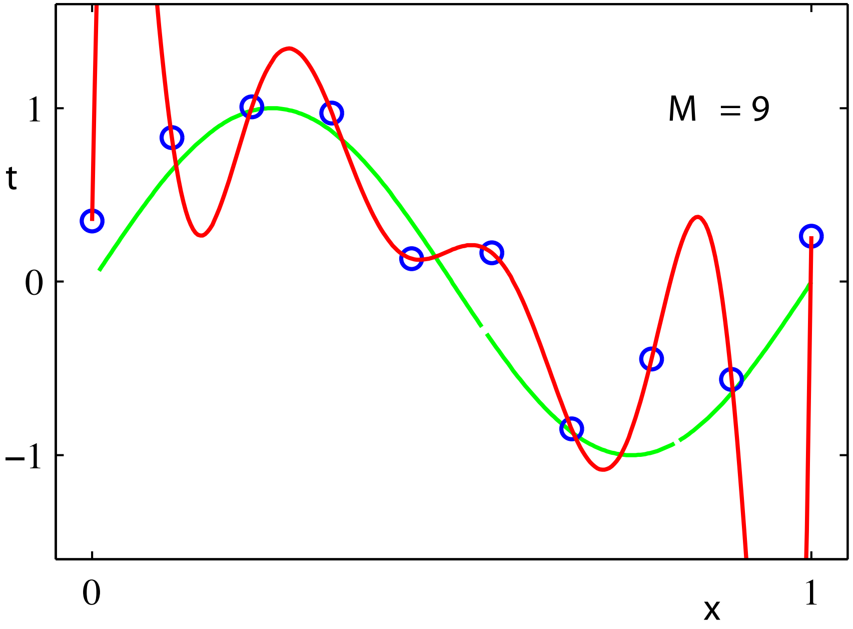
# K-Means is just one algorithm

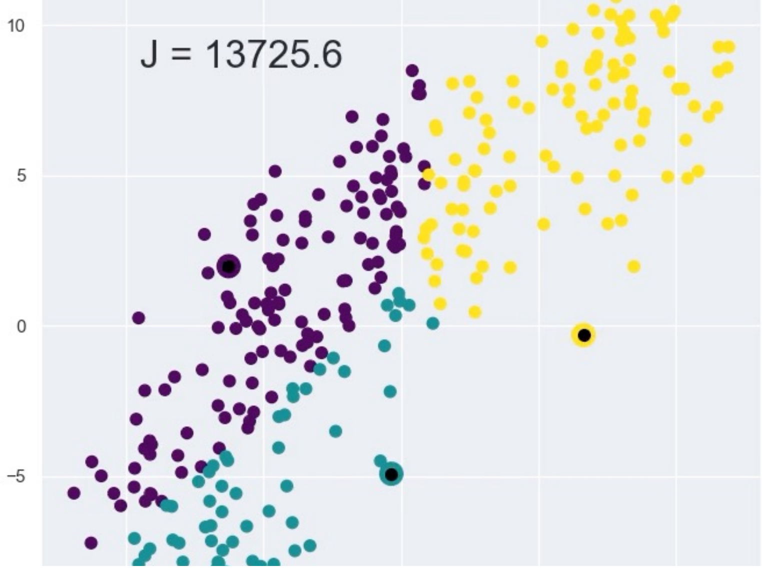
- Many approaches to defining clustering algorithms
- Let's start by understanding limitations of K-means
  - Optimal clustering is NP hard; random restarts needed
  - Choice of K may be important
  - Cluster centers are sensitive to outliers
  - Works poorly on non-convex clusters
  - Assumes spherical, equally likely clusters
  - Hard assignment of example to clusters

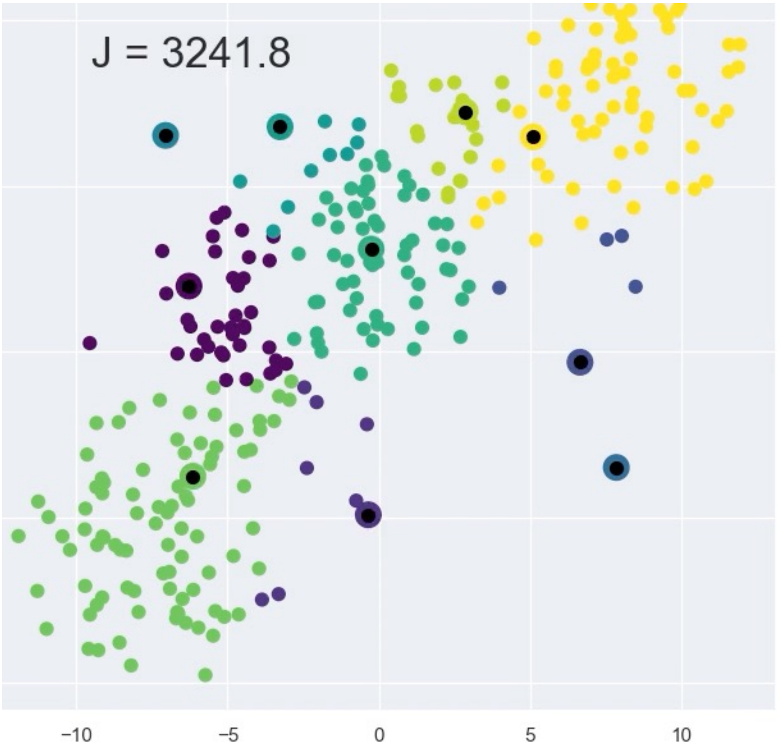
NP-Hard proof: https://cseweb.ucsd.edu/~avattani/papers/kmeans_hardness.pdf

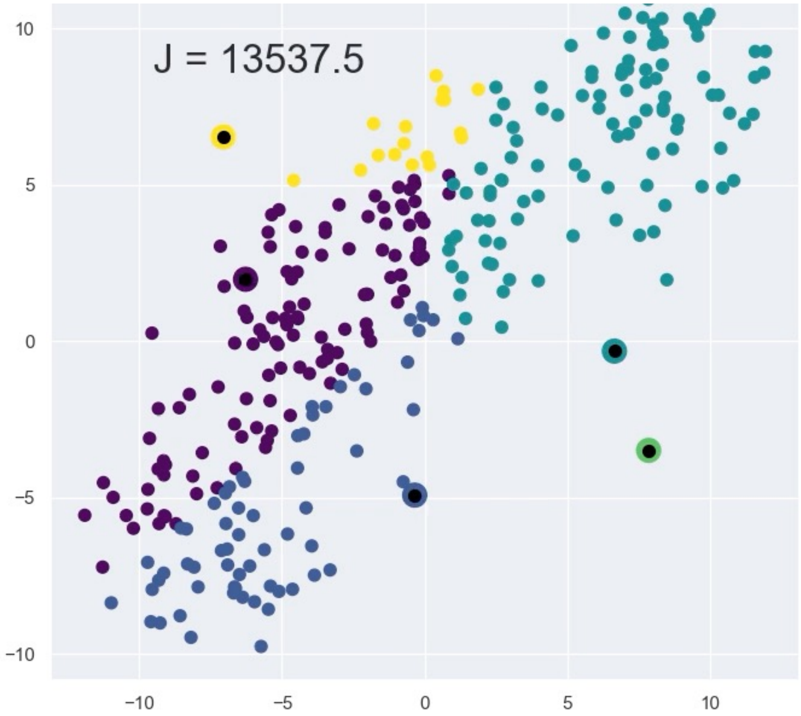
# How many clusters?

- What's going to happen if we keep increasing the number of clusters?
- What happened when we kept increasing the degree of a polynomial regression?
- Will this happen with K-Means?

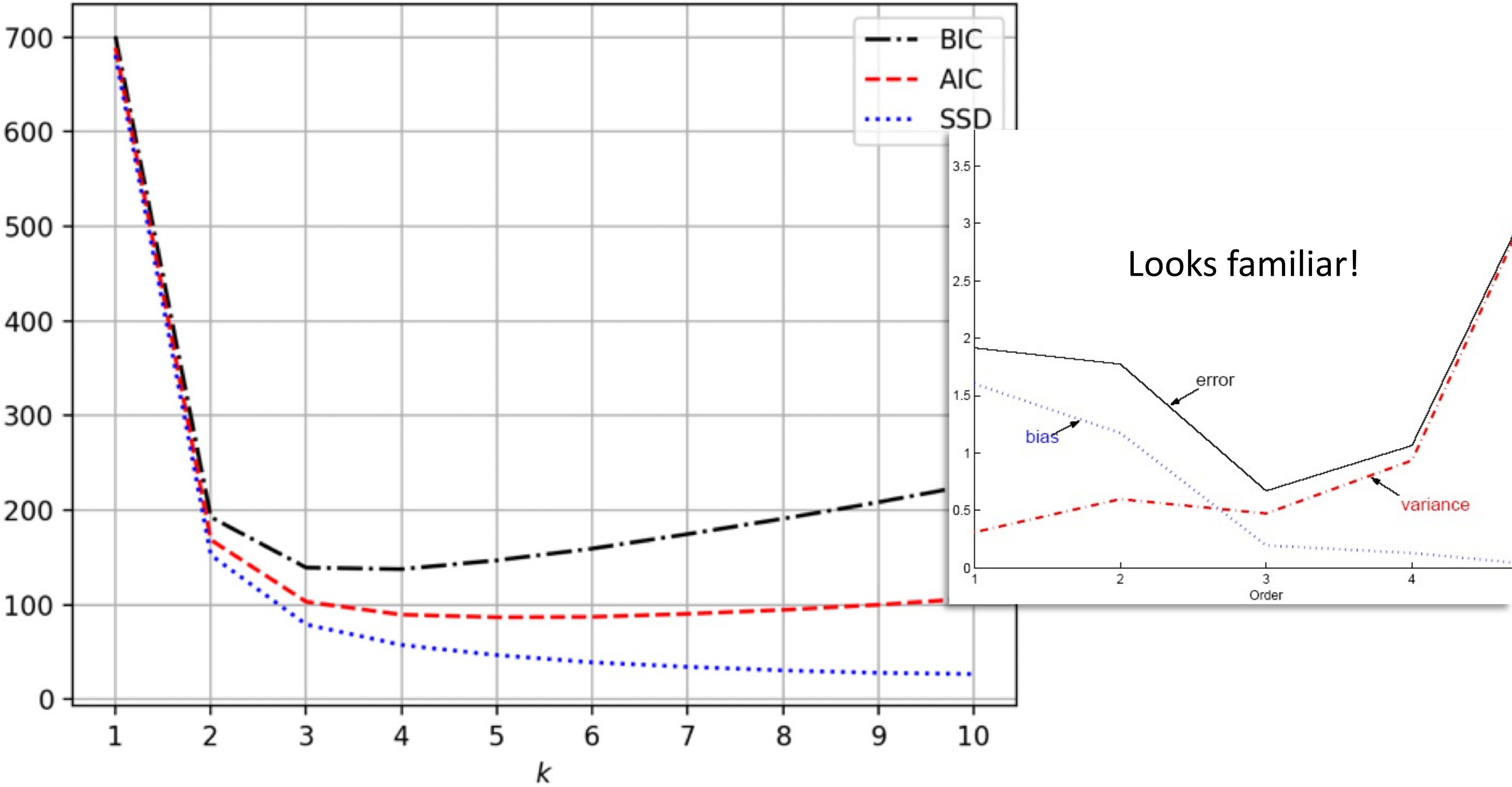



J = 13725.6
10
5
0
−5


J = 3241.8
−10
−5
0
5
10


J = 13537.5
10
5
0
−5
−10
−10
−5
0
5
10

# How many clusters?

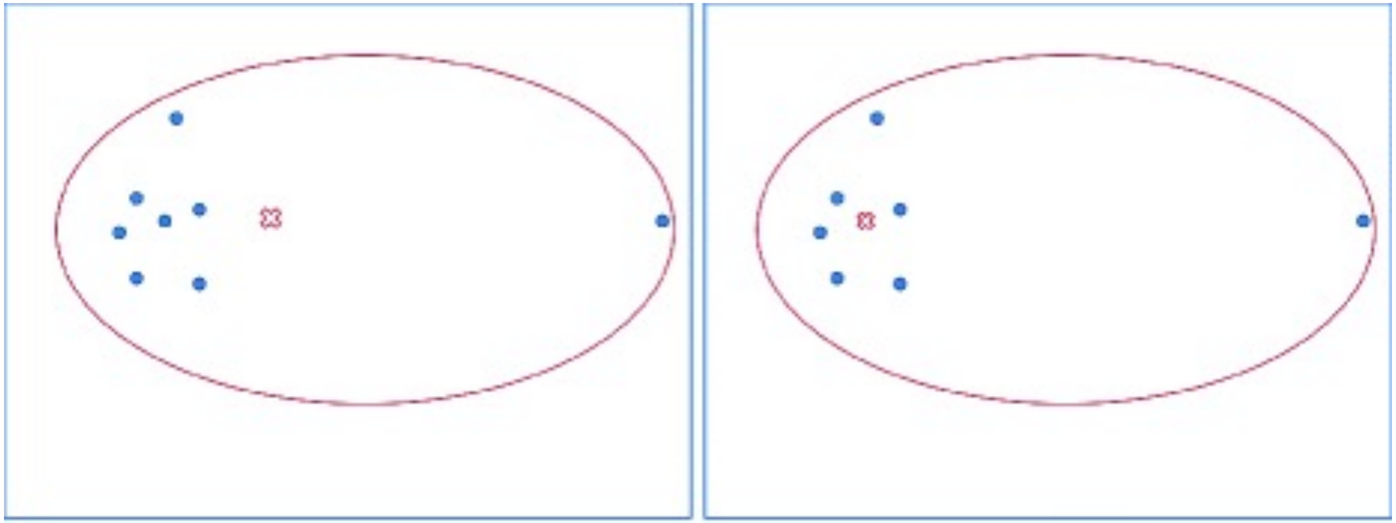


Looks familiar!

SSD: Sum of squared distances (our standard clustering loss)
AIC: Akaike information criterion
BIC: Bayesian information criterion

# K-Means is sensitive to outliers

- Means are sensitive to outliers, which can give bad cluster centers
- Solution: switch to medians



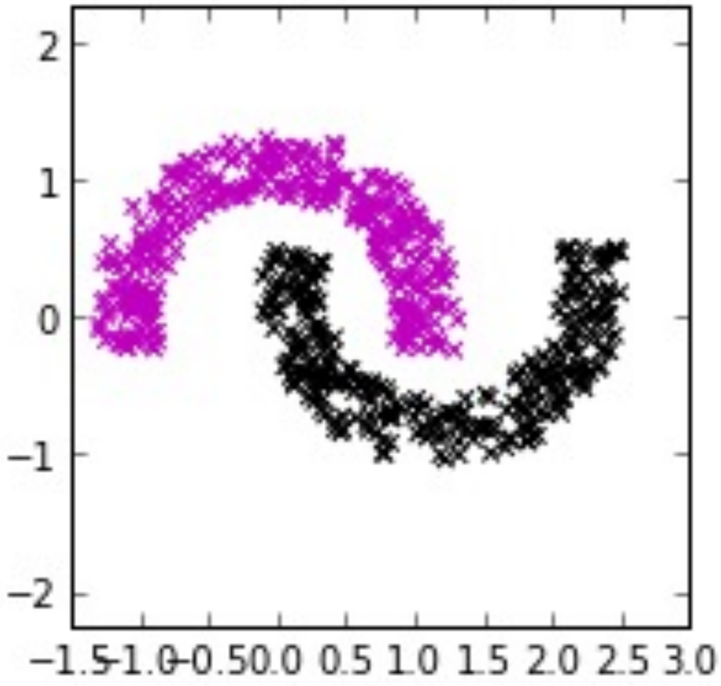
(a) Mean

(b) Medoid

https://link.springer.com/referenceworkentry/10.1007%2F978-0-387-30164-8_426

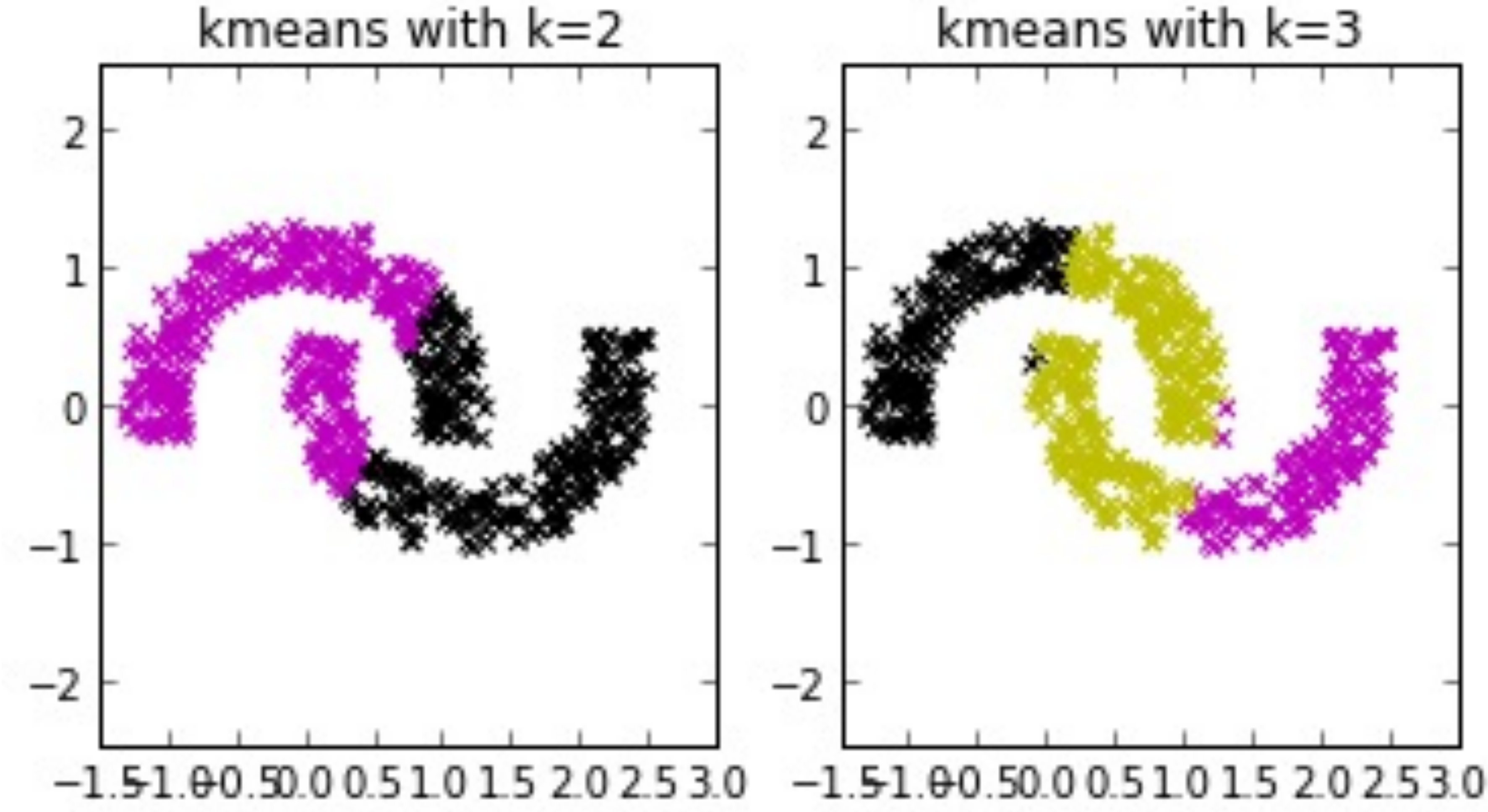
# K-Means and non-convex clusters

- Not all clusters are spherical
- How will k-means do on this data?



https://pafnuty.wordpress.com/2013/08/14/non-convex-sets-with-k-means-and-hierarchical-clustering/

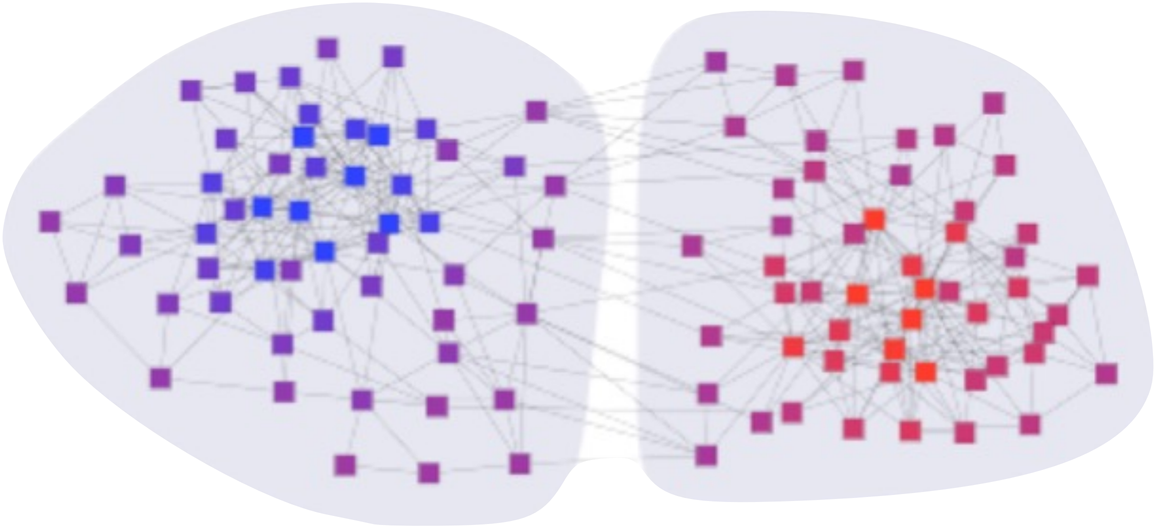
# K-Means and non-convex clusters



https://pafnuty.wordpress.com/2013/08/14/non-convex-sets-with-k-means-and-hierarchical-clustering/

# Spectral Clustering

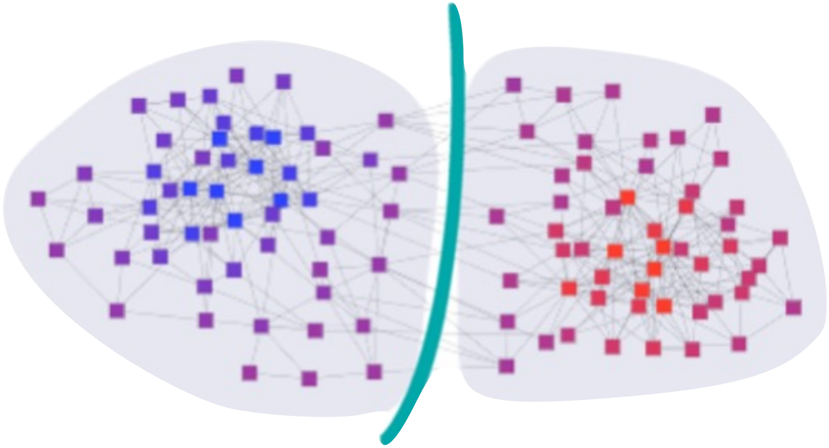
- Partitional but non-spherical clustering
- Construct a graph G from the data
  - Vertices are still examples
  - Edges are weighted similarity between examples
    - Weights may depend on the application

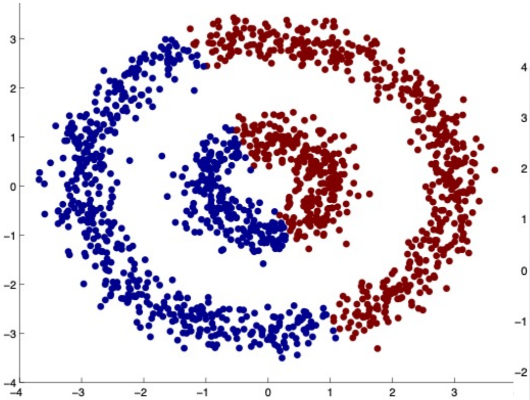


https://people.eecs.berkeley.edu/~jordan/courses/294-fall09/lectures/clustering/slides.pdf

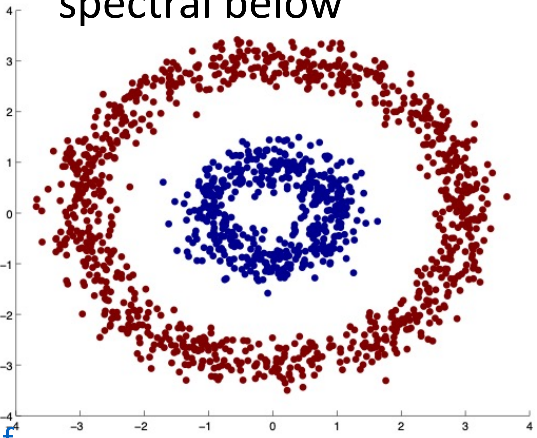
# Spectral Clustering

- Goal of clustering: Partition the vertices of the graph
- Loss function: measured by a cut of the graph
  - Minimize Cut (min-cut): the weight of the edges “cut” by partitioning vertices into different clusters
  - Requires normalization to force meaningful cuts
  - Minimizing normalized cut is still NP-hard





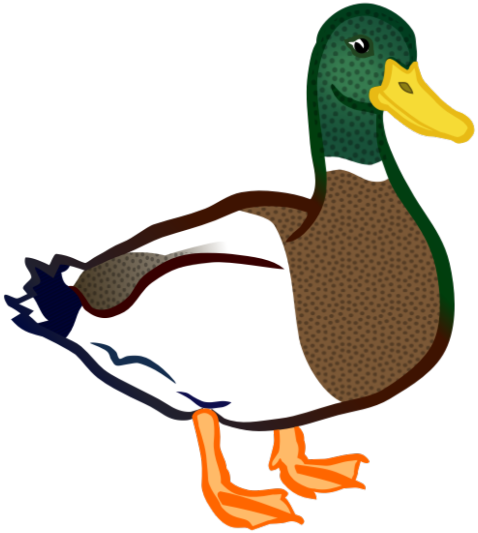
K-means on left; spectral below



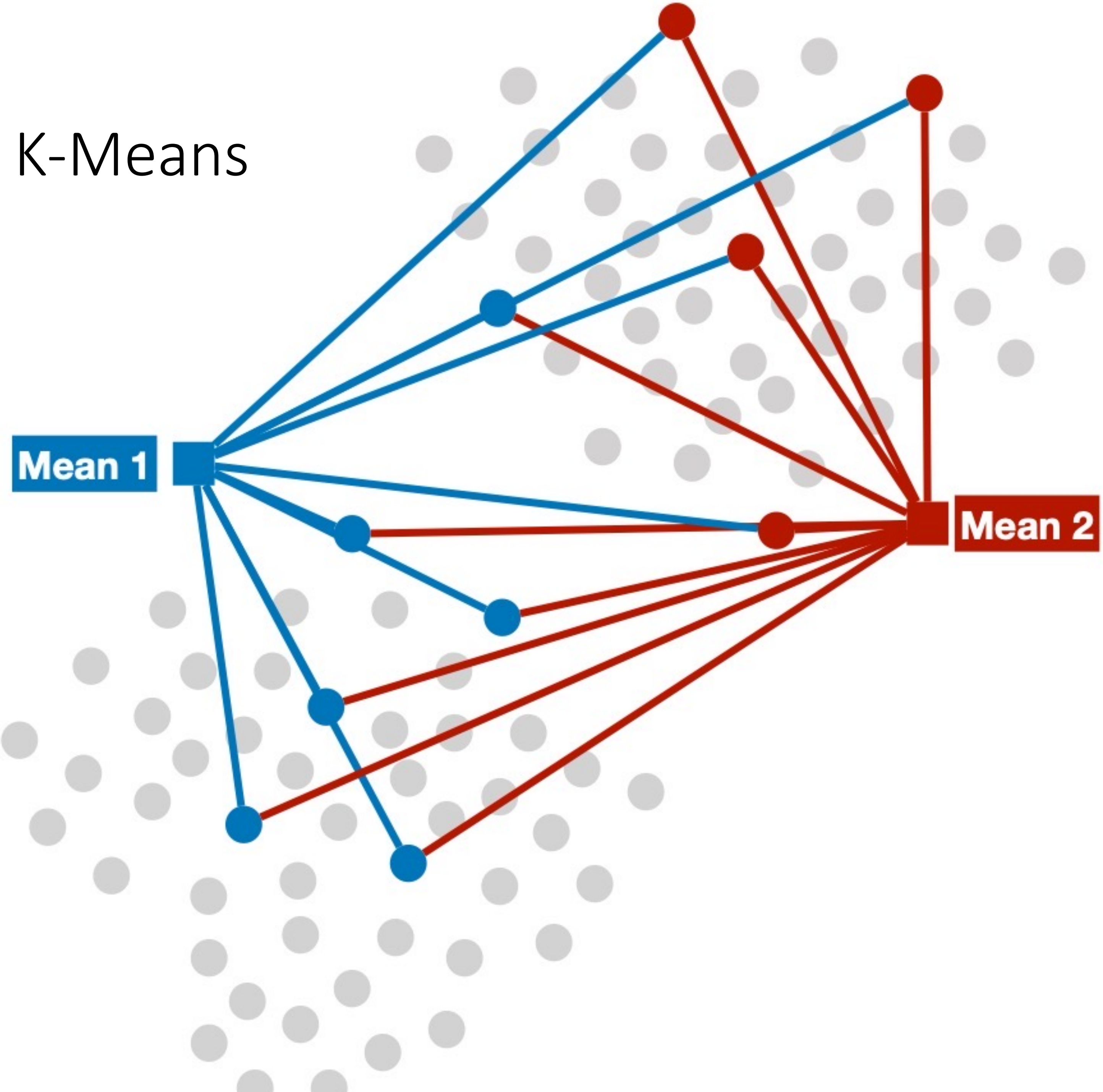
https://people.eecs.berkeley.edu/~jordan/courses/294-fall09/lectures/clustering/slides.pdf

# Relaxing K-Means

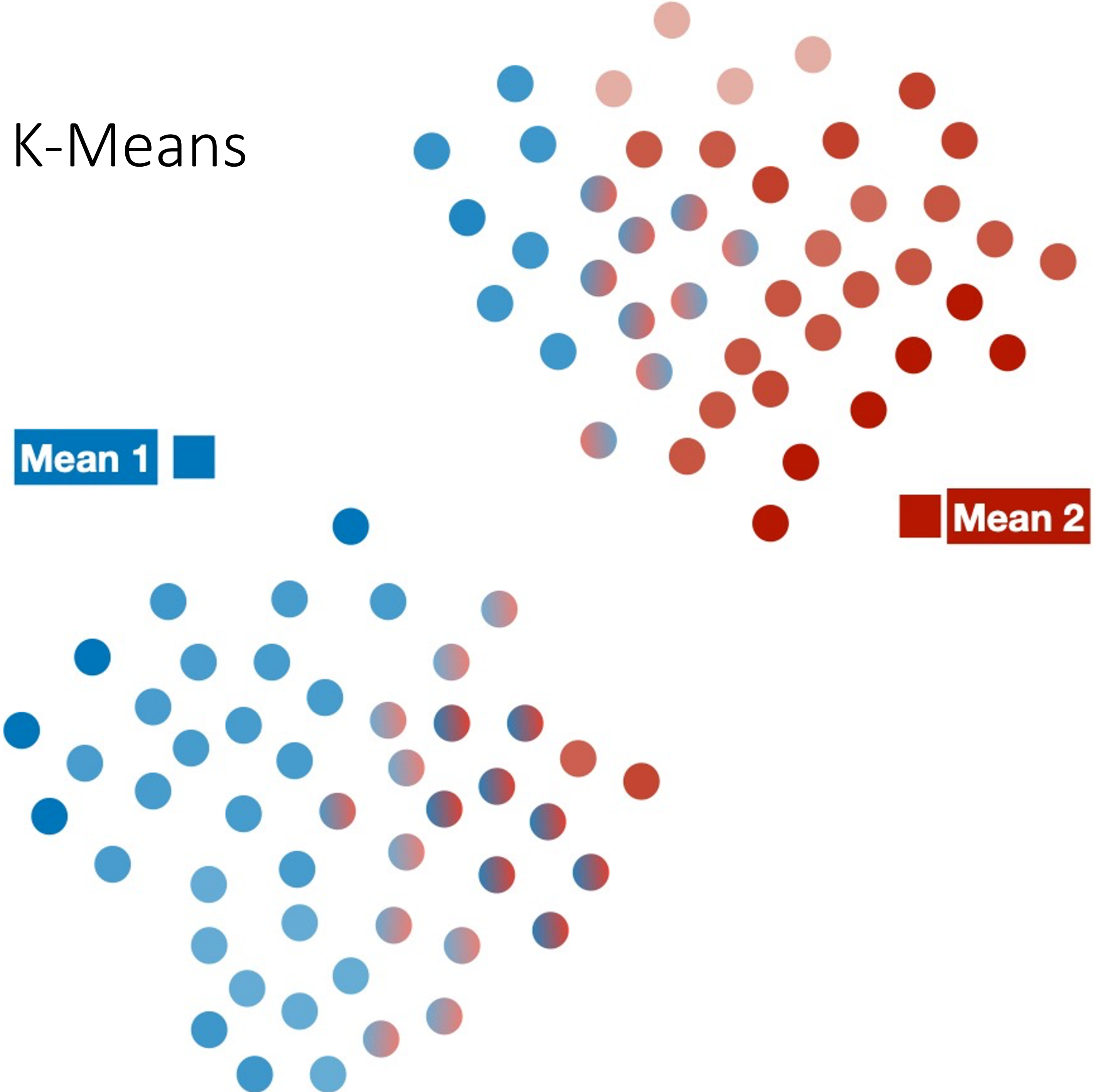
- K-means assumes spherical, equally likely clusters
- An example **must** belong to a single cluster
  - Introduces instability between training iterations as examples “jump” between clusters
- Solution: Relax this constraint to allow a more flexible notion of cluster membership



# Relaxing K-Means



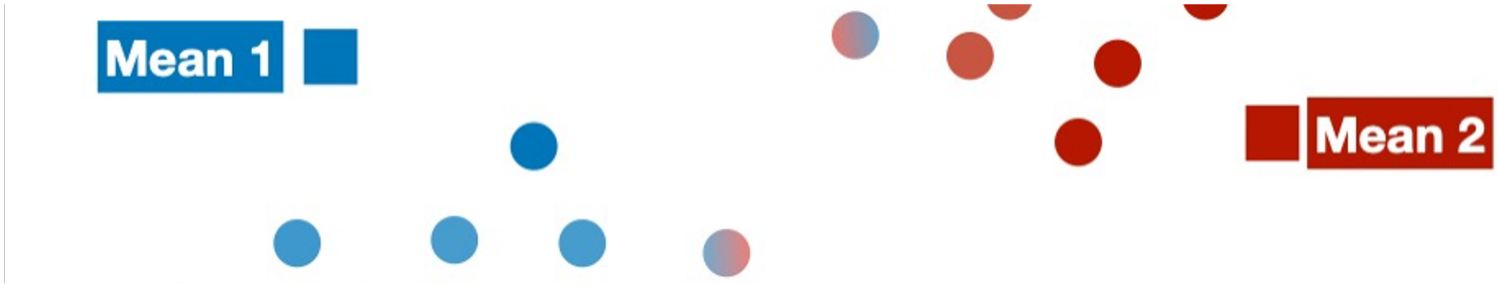
# Relaxing K-Means



# Recall: K-Means Updates

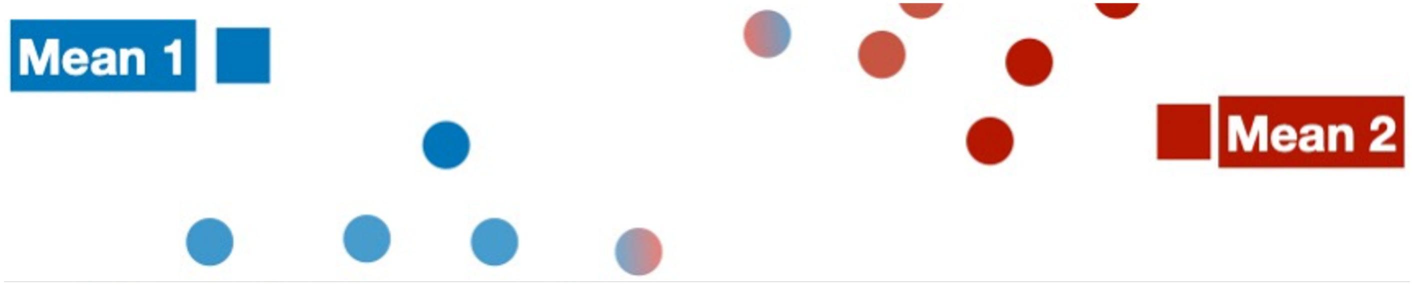
$$z_{n,k} = \begin{cases} 1 & k = \arg\min_j d(\mathbf{x_n}, \mu_j) \\ 0 & \text{otherwise} \end{cases}$$

$$\mu_{\mathbf{k}} = \frac{\sum_{n=1}^{N} z_{n,k} \cdot \mathbf{x_n}}{\sum_{n=1}^{N} z_{n,k}}$$



$$z_{n,k} = \begin{cases} 1 & k = \arg\min_j d(\mathbf{x_n}, \mu_j) \\ 0 & \text{otherwise} \end{cases}$$

# Picking a new update rule

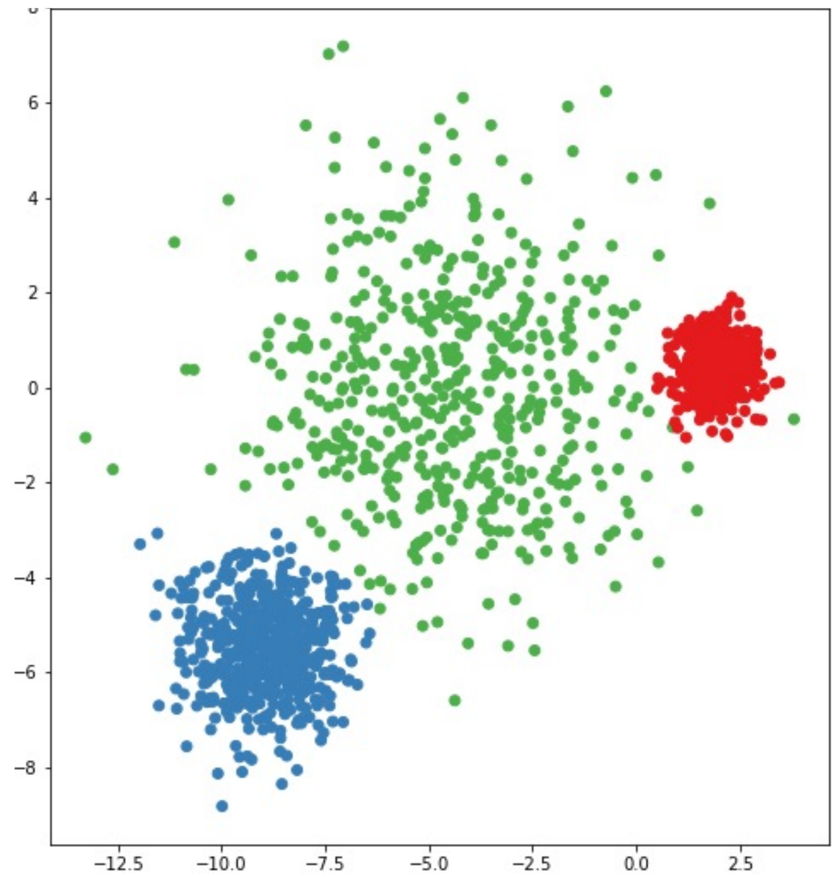


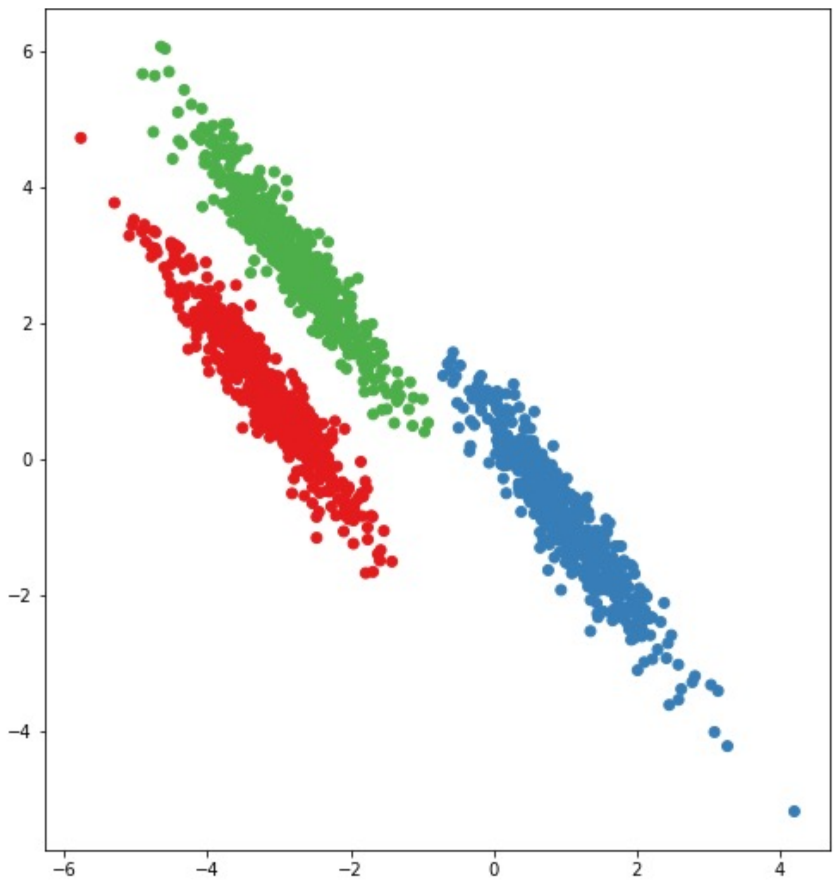
$$z_{n,1} = \frac{-d(\mathbf{x_n}, \mu_1)}{-d(\mathbf{x_n}, \mu_1) - d(\mathbf{x_n}, \mu_2)}$$

$$z_{n,k} = \frac{-d(\mathbf{x_n}, \mu_k)}{-\sum_{j=1}^{K} d(\mathbf{x_n}, \mu_j)}$$

# Picking a new update rule

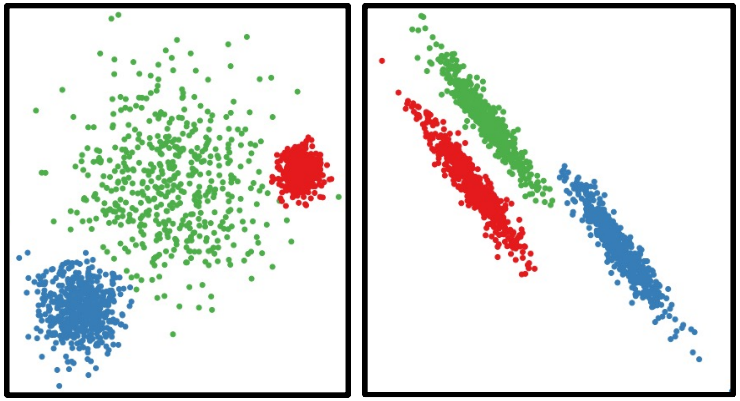
$$z_{n,k} = \frac{d(\mathbf{x_n}, \mu_k)}{\sum_j d(\mathbf{x_n}, \mu_j)}$$





https://towardsdatascience.com/identifying-restaurant-hotspots-with-a-gaussian-mixture-model-2a840ab0c782

# Picking a new update rule



$$z_{n,k} = \frac{-\left\|\mathbf{x_n} - \mu_k\right\|^2}{-\sum_{j=1}^{K}\left\|\mathbf{x_n} - \mu_j\right\|^2}$$

$$z_{n,k} = \frac{-(\mathbf{x_n} - \mu_k)^\top \mathbf{I_K}(\mathbf{x_n} - \mu_k)}{\sum_{j=1}^{K} -(\mathbf{x_n} - \mu_j)^\top \mathbf{I_K}(\mathbf{x_n} - \mu_j)}$$

$$z_{n,k} = \frac{-(\mathbf{x_n} - \mu_k)^\top \mathbf{\Sigma_k}(\mathbf{x_n} - \mu_k)}{\sum_{j=1}^{K} -(\mathbf{x_n} - \mu_j)^\top \mathbf{\Sigma_j}(\mathbf{x_n} - \mu_j)}$$

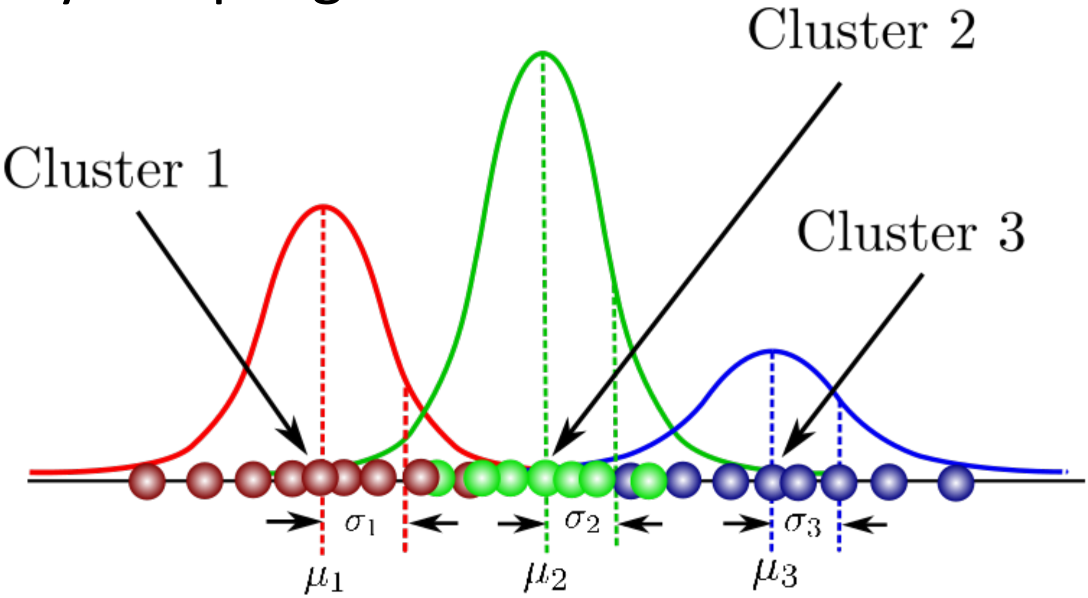
$$z_{n,k} = \frac{\mathcal{N}(\mathbf{x_n} \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K}\mathcal{N}(\mathbf{x_n} \mid \mu_j, \Sigma_j)}$$

$$z_{n,k} = \frac{\exp\left(-(\mathbf{x_n} - \mu_k)^\top \mathbf{\Sigma_k}(\mathbf{x_n} - \mu_k)\right)}{\sum_{j=1}^{K}\exp\left(-(\mathbf{x_n} - \mu_j)^\top \mathbf{\Sigma_j}(\mathbf{x_n} - \mu_j)\right)}$$

$$z_{n,k} = \frac{(2\pi)^{-K/2}\sqrt{\det(\Sigma_k)}\exp\left(-(\mathbf{x_n} - \mu_k)^\top \mathbf{\Sigma_k}(\mathbf{x_n} - \mu_k)/2\right)}{\sum_{j=1}^{K}(2\pi)^{-K/2}\sqrt{\det(\Sigma_j)}\exp\left(-(\mathbf{x_n} - \mu_j)^\top \mathbf{\Sigma_j}(\mathbf{x_n} - \mu_j)/2\right)}$$

# Generative Clustering Model

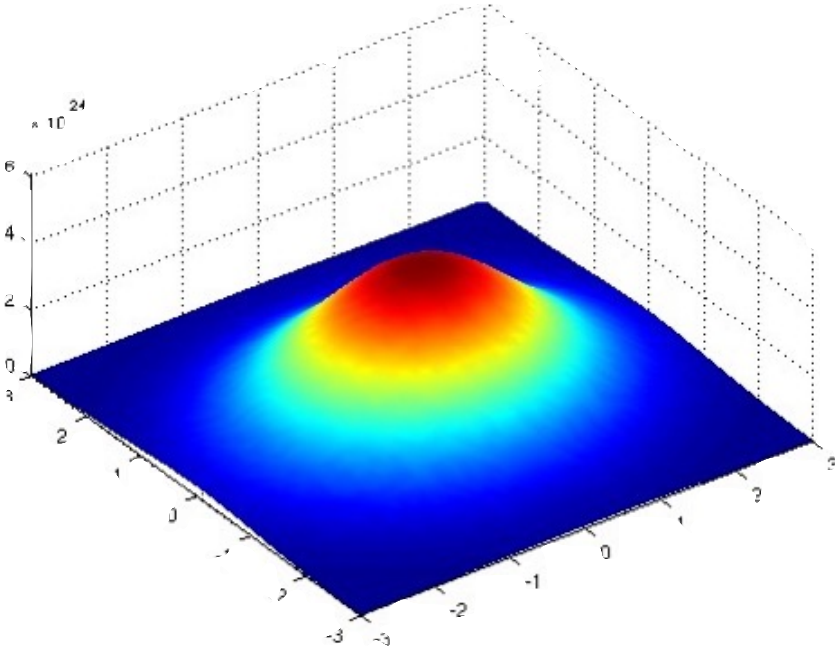
- Assume we have K clusters
- Each cluster represented by a multivariate Gaussian
- Generative process
  - Select a cluster (a Gaussian distribution)
  - Generate an example by sampling from the Gaussian



https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95

# Gaussian Mixtures

- Since we have multiple Gaussians generating points, we call the model Gaussian Mixture Model
- Why Gaussians?
  - Captures intuition about clusters
  - Examples are more likely to be near center of cluster



# Gaussian Mixture Model

- Cluster Responsibilities

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

- Cluster means, variances, and weight coefficients

$$N_k = \sum_n \gamma(z_{nk})$$

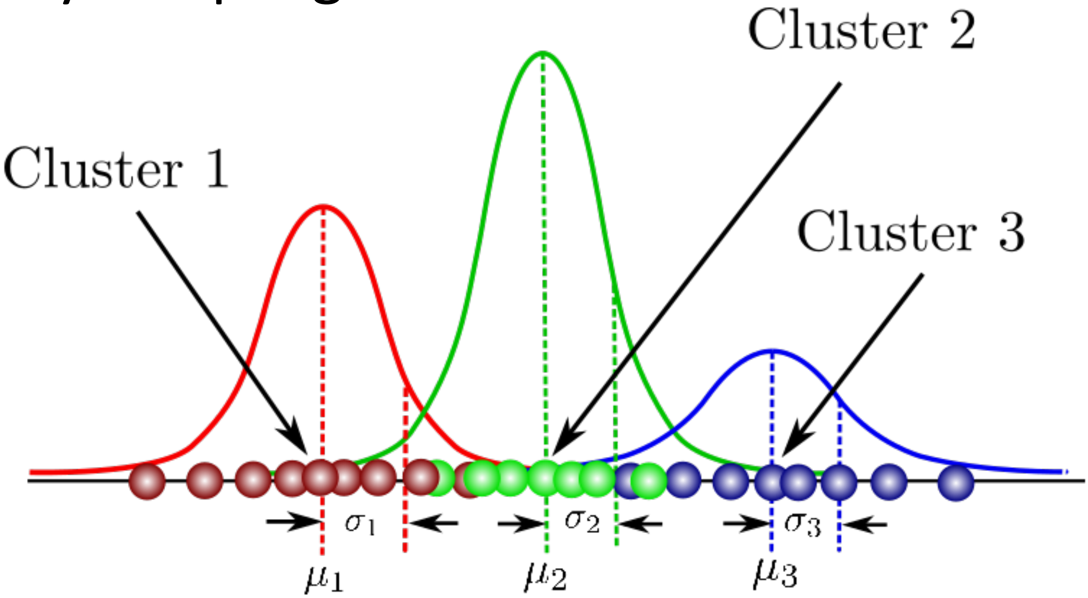
$$\pi_k = \frac{N_k}{N} = \frac{N_k}{\sum_k N_k}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

# Generative Clustering Model

- Assume we have K clusters
- Each cluster represented by a multivariate Gaussian
- Generative process
  - Select a cluster (a Gaussian distribution)
  - Generate an example by sampling from the Gaussian



https://towardsdatascience.com/gaussian-mixture-models-explained-6986aaf5a95

# Problems with GMMs

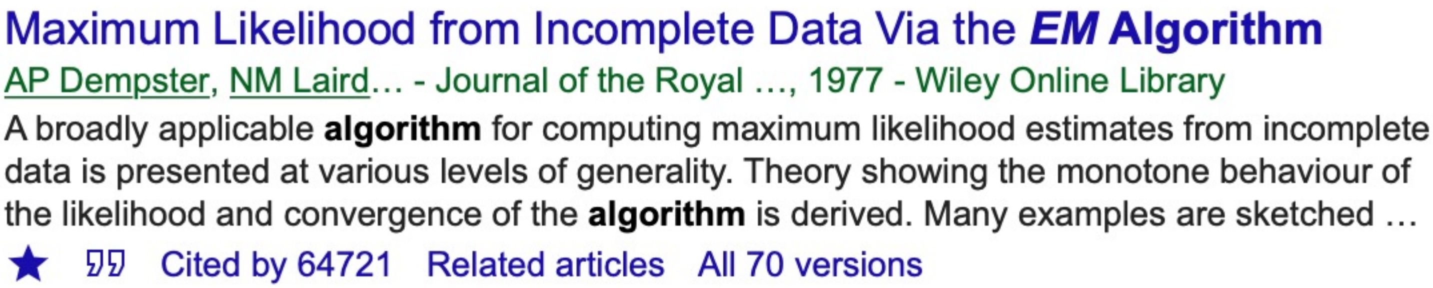
- Mode collapse: cluster with a single example
    - Undefined variance: catch this and reset that cluster
- Non-convex likelihood: K! equivalent solutions
    - Random restarts may still be helpful
- Slower: requires more iterations than K-Means
    - And each iteration is more computationally expensive

# Next time: Expectation Maximization

- K-Means and GMMs share a general algorithm:

- Initialize parameters that describe the data
- Repeat until converged:
    1. Compute assignment for every data point
    2. Update parameters based on those assignments

- What else can this algorithm do?



Maximum Likelihood from Incomplete Data Via the ***EM*** **Algorithm**

AP Dempster, NM Laird… - Journal of the Royal …, 1977 - Wiley Online Library

A broadly applicable **algorithm** for computing maximum likelihood estimates from incomplete data is presented at various levels of generality. Theory showing the monotone behaviour of the likelihood and convergence of the **algorithm** is derived. Many examples are sketched …

★ ❞ Cited by 64721 Related articles All 70 versions