10-701: Introduction to Machine Learning Lecture 14: Clustering

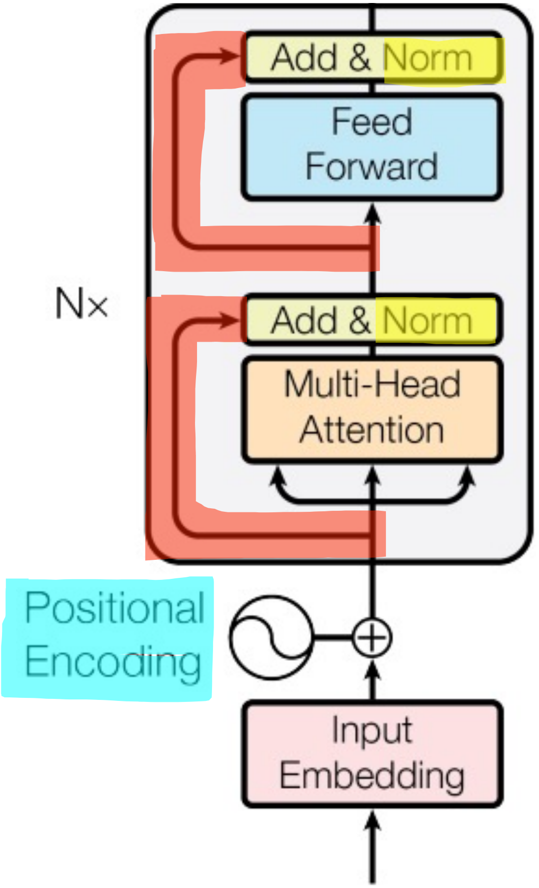
Henry Chai

3/11/24

Front Matter

- Announcements
  - HW4 released 2/28, due 3/15 (Friday) at 11:59 PM
  - Midterm exam on 3/19 from **7 – 9 PM in DH A302**
    - If you have a conflict with this date/time fill out the conflict form on Piazza ASAP
  - Final exam date has been announced: Monday, May 6th from 1 – 4 PM
- Recommended Readings
  - Daumé III, Chapter 15: Unsupervised Learning
  - Murphy, Section 11.1 – 11.4.2

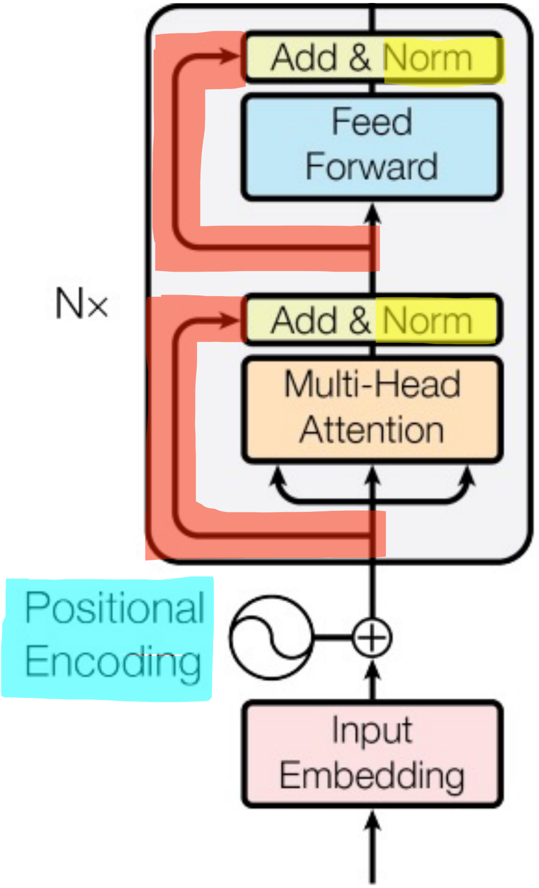
Recall: Transformers



- In addition to multi-head attention, transformer architectures use
  1. Positional encodings
  2. Layer normalization
  3. Residual connections
  4. A fully-connected feed-forward network

Source: https://arxiv.org/pdf/1706.03762.pdf

How on earth do we train these things?



- In addition to multi-head attention, transformer architectures use
    1. Positional encodings
    2. Layer normalization
    3. Residual connections
    4. A fully-connected feed-forward network

Source: https://arxiv.org/pdf/1706.03762.pdf

Learning Paradigms

- Supervised learning - $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}$
  - Regression - $y^{(i)} \in \mathbb{R}$
  - Classification - $y^{(i)} \in \{1, \dots, C\}$
- Unsupervised learning - $\mathcal{D} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$
  - Clustering
  - Dimensionality reduction
- Reinforcement learning
- Active learning
- Semi-supervised learning
- Online learning

Learning Paradigms

- Supervised learning - $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$
  - Regression - $y^{(i)} \in \mathbb{R}$
  - Classification - $y^{(i)} \in \{1, \dots, C\}$
- Unsupervised learning - $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$
  - **Clustering**
  - Dimensionality reduction
- Reinforcement learning
- Active learning
- Semi-supervised learning
- Online learning

Clustering

- Goal: split an unlabeled data set into groups or clusters of "similar" data points
- Use cases:
  - Organizing data
  - Discovering patterns or structure
  - Preprocessing for downstream machine learning tasks
- Applications:

Recall: Similarity for kNN

- Classify a point as the label of the "most similar" training point
- **Idea: given real-valued features, we can use a distance metric to determine how similar two data points are**
- A common choice is Euclidean distance:

$$d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2 = \sqrt{\sum_{d=1}^{D} (x_d - x'_d)^2}$$

- An alternative is the Manhattan distance:

$$d(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_1 = \sum_{d=1}^{D} |x_d - x'_d|$$

Partition-Based Clustering

- Given a desired number of clusters, K, return a partition of the data set into K groups or clusters, $\{C_1, \ldots, C_K\}$, that optimize some objective function

1. What objective function should we optimize?

2. How can we perform optimization in this setting?

Option A

Option B

Which partition is best?

General Recipe for Machine Learning

- Define a model and model parameters

- Write down an objective function

- Optimize the objective w.r.t. the model parameters

Recipe for K-means

- Define a model and model parameters
  - Assume K clusters and use the Euclidean distance
  - Parameters: $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K$ and $z^{(1)}, \dots, z^{(N)}$
- Write down an objective function

$$\sum_{i=1}^{N} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{z^{(i)}} \right\|_2$$

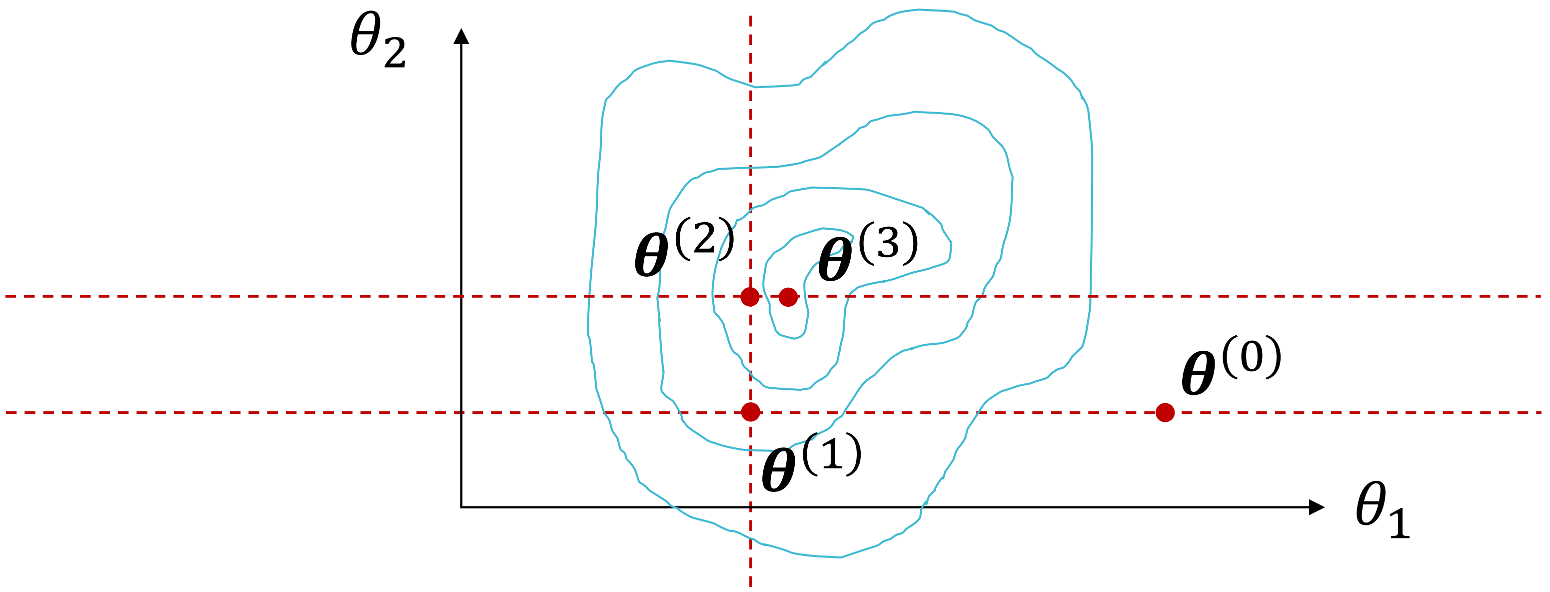
- Optimize the objective w.r.t. the model parameters
  - Use (block) coordinate descent

Coordinate Descent

- Goal: minimize some objective

$$\hat{\boldsymbol{\theta}} = \text{argmin } J(\boldsymbol{\theta})$$

- Idea: iteratively pick one variable and minimize the objective w.r.t. just that variable, *keeping all others fixed*.



Block Coordinate Descent

- Goal: minimize some objective

$$\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}} = \operatorname{argmin}\ J(\boldsymbol{\alpha}, \boldsymbol{\beta})$$

- Idea: iteratively pick one *block* of variables ($\boldsymbol{\alpha}$ or $\boldsymbol{\beta}$) and minimize the objective w.r.t. that block, keeping the other(s) fixed.
  - Ideally, blocks should be the largest possible set of variables that can be efficiently optimized simultaneously

Optimizing the K-means objective

$$\hat{\boldsymbol{\mu}}_1, \ldots, \hat{\boldsymbol{\mu}}_K, z^{(1)}, \ldots, z^{(\mathrm{N})} = \operatorname{argmin} \sum_{i=1}^{N} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{\mu}_{z^{(i)}} \right\|_2$$

- If $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ are fixed

$$\hat{z}^{(i)} = \underset{k \in \{1, \ldots, K\}}{\operatorname{argmin}} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k \right\|_2$$

- If $z^{(1)}, \ldots, z^{(N)}$ are fixed

$$\hat{\boldsymbol{\mu}}_k = \underset{\boldsymbol{\mu}}{\operatorname{argmin}} \sum_{i\,: z^{(i)} = k} \left\| \boldsymbol{x}^{(i)} - \boldsymbol{\mu} \right\|_2$$

$$= \frac{1}{N_k} \sum_{i\,: z^{(i)} = k} \boldsymbol{x}^{(i)}$$

K-means Algorithm

- Input: $\mathcal{D} = \left\{\left(\boldsymbol{x}^{(i)}\right)\right\}_{i=1}^{N}, K$

1. Initialize cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$
2. While NOT CONVERGED
   a. Assign each data point to the cluster with the nearest cluster center:
   $$z^{(i)} = \underset{k}{\operatorname{argmin}} \left\|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\right\|_2$$
   b. Recompute the cluster centers:
   $$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i\,:z^{(i)}=k} \boldsymbol{x}^{(i)}$$
   where N_k is the number of data points in cluster k

- Output: cluster assignments $z^{(1)}, \ldots, z^{(N)}$

K-means: Example ($K = 3$)

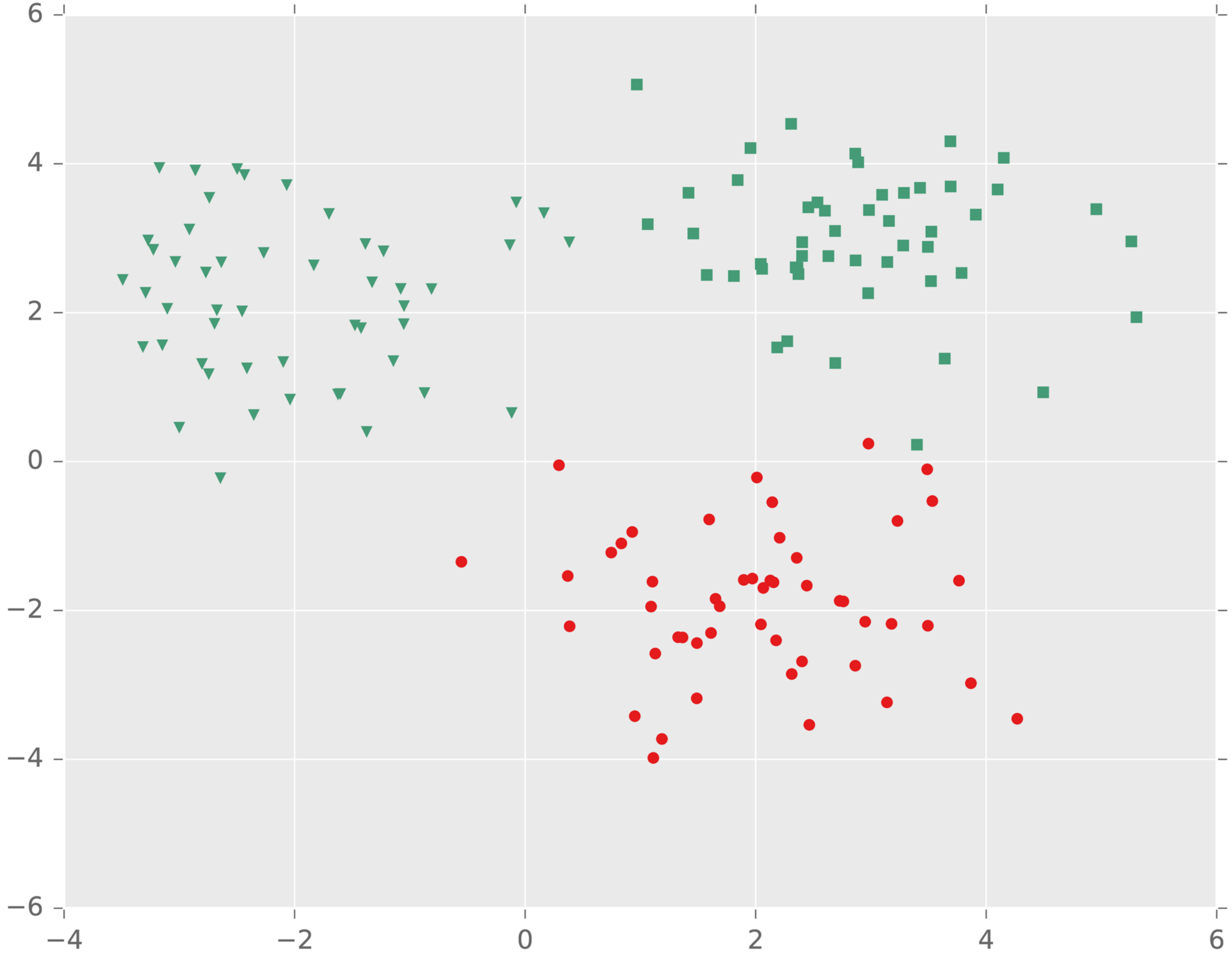


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

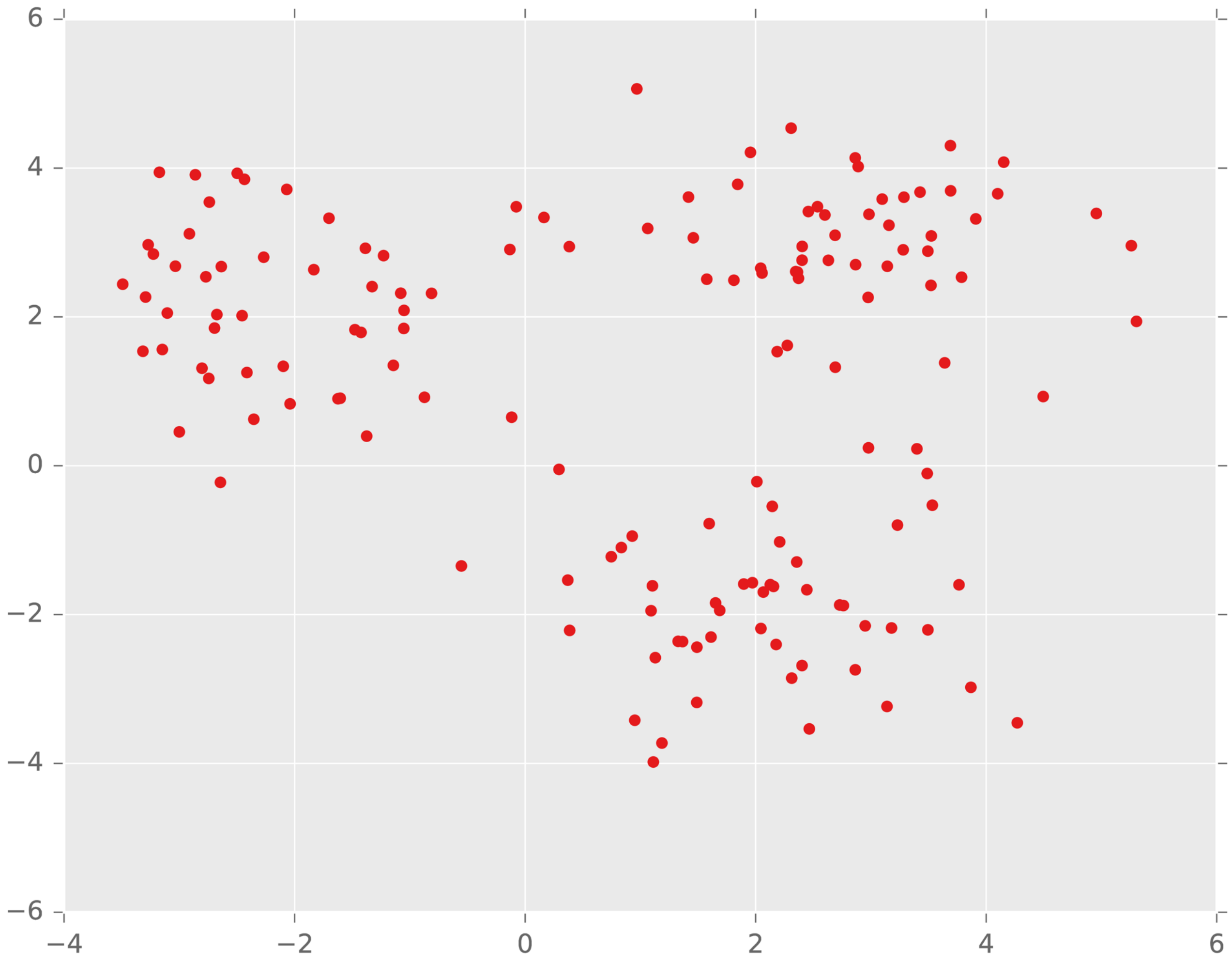


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

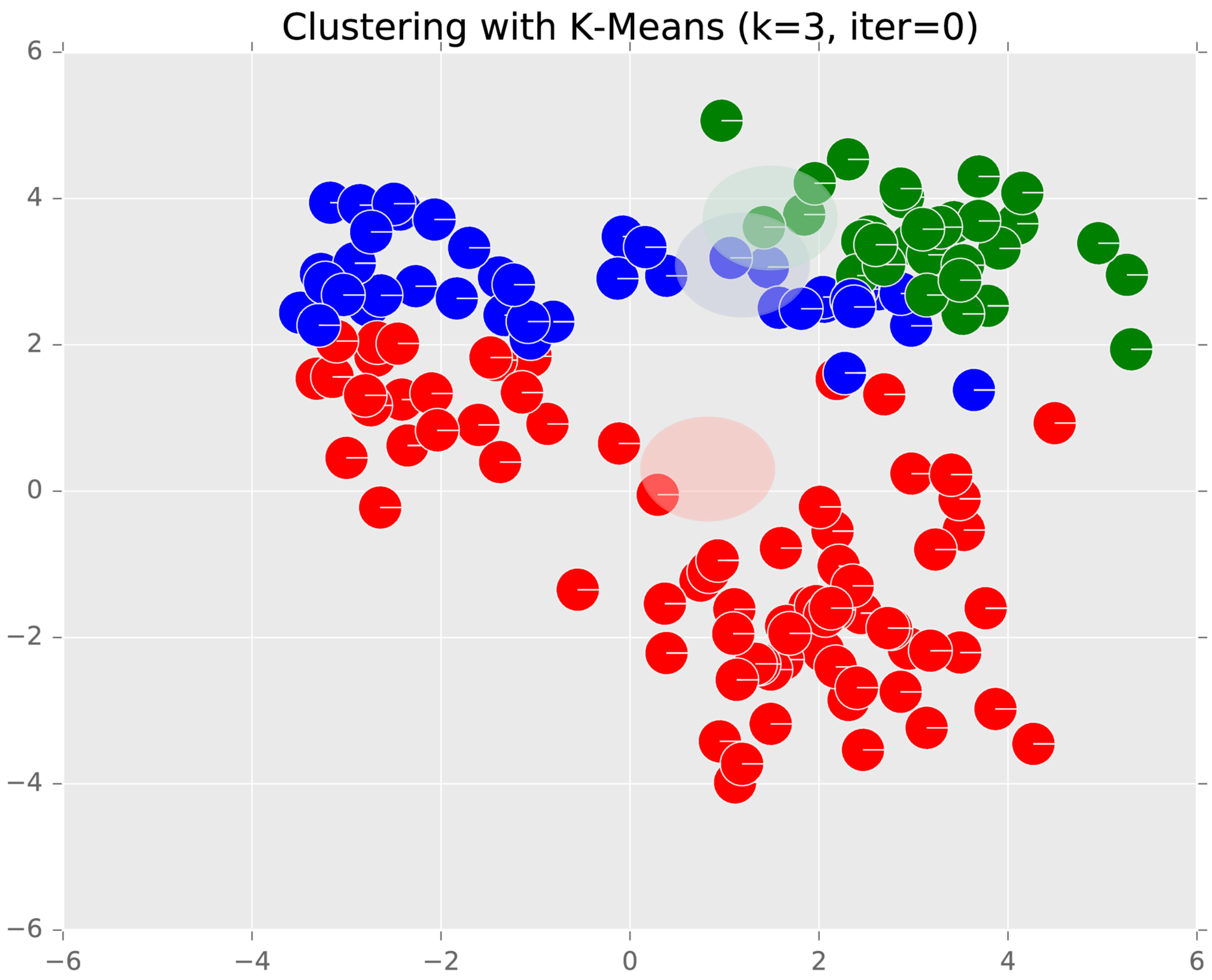


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

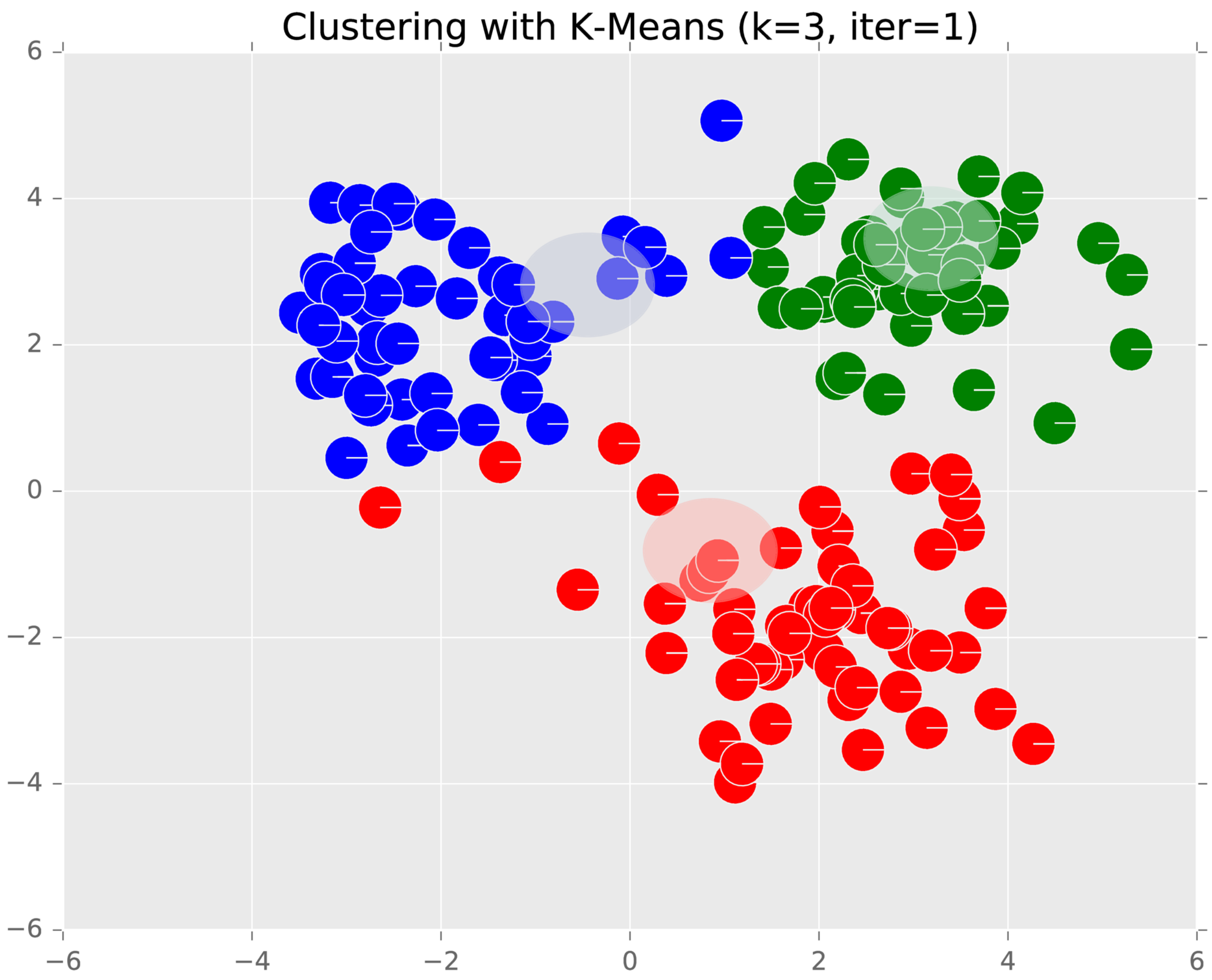


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

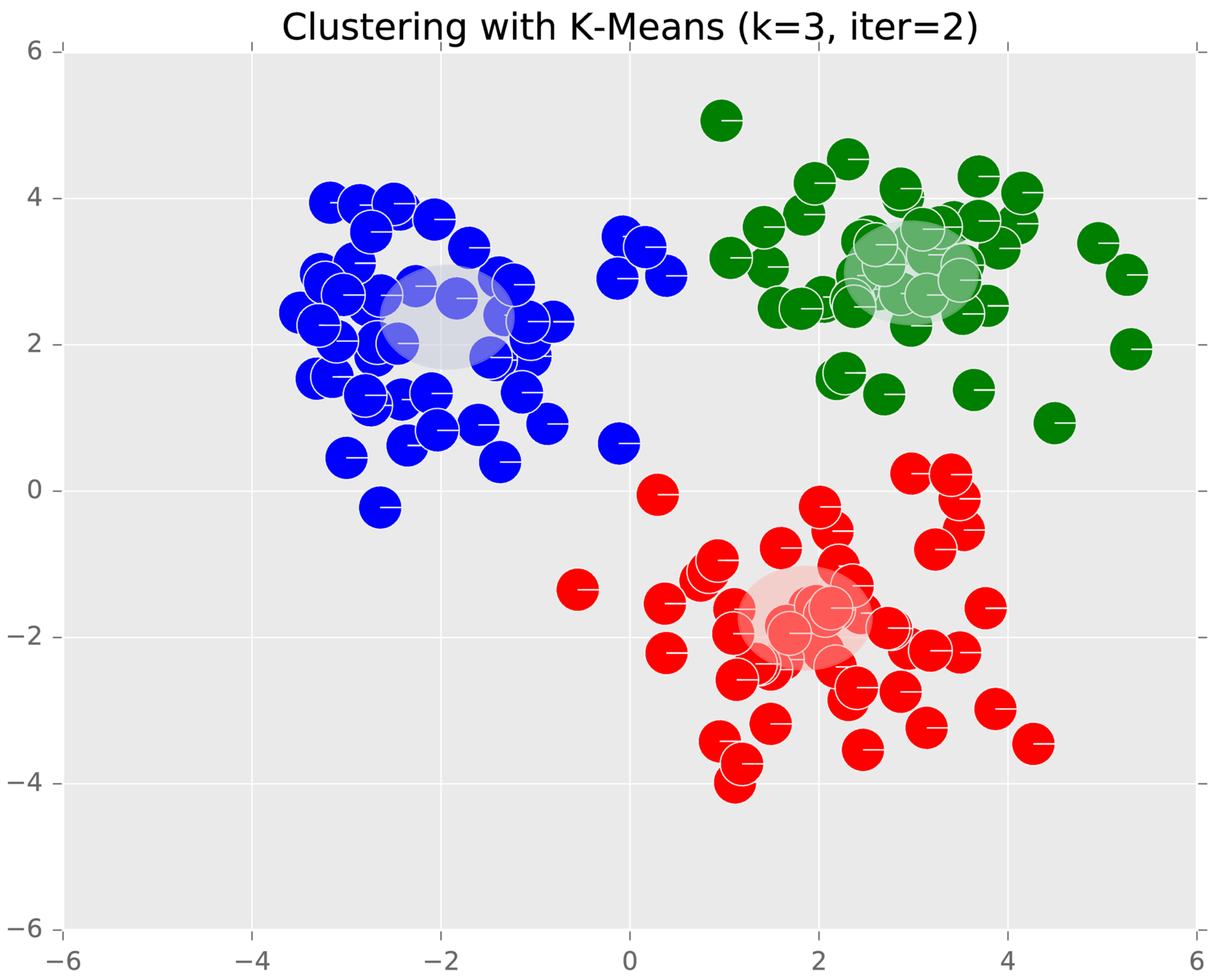


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

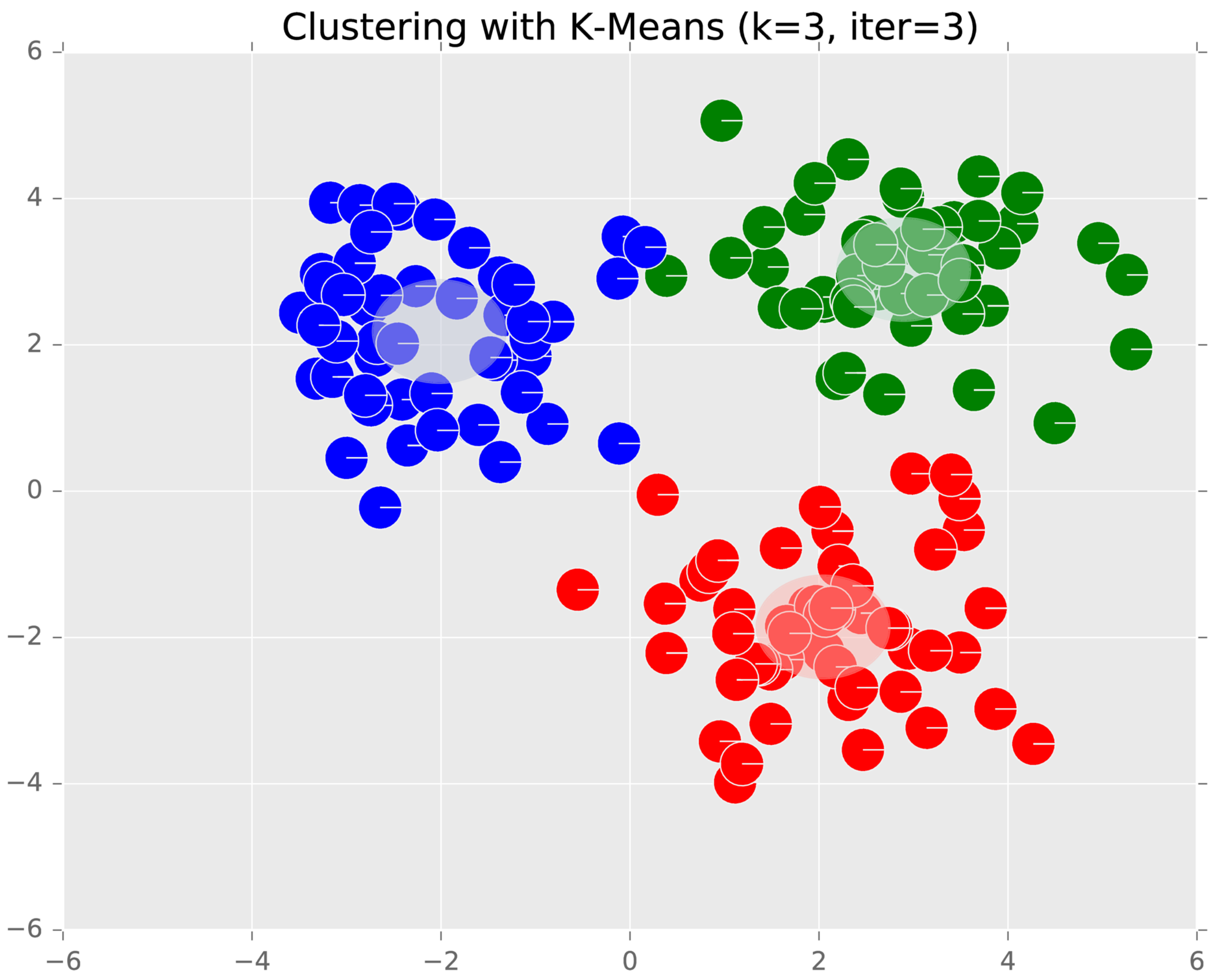


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

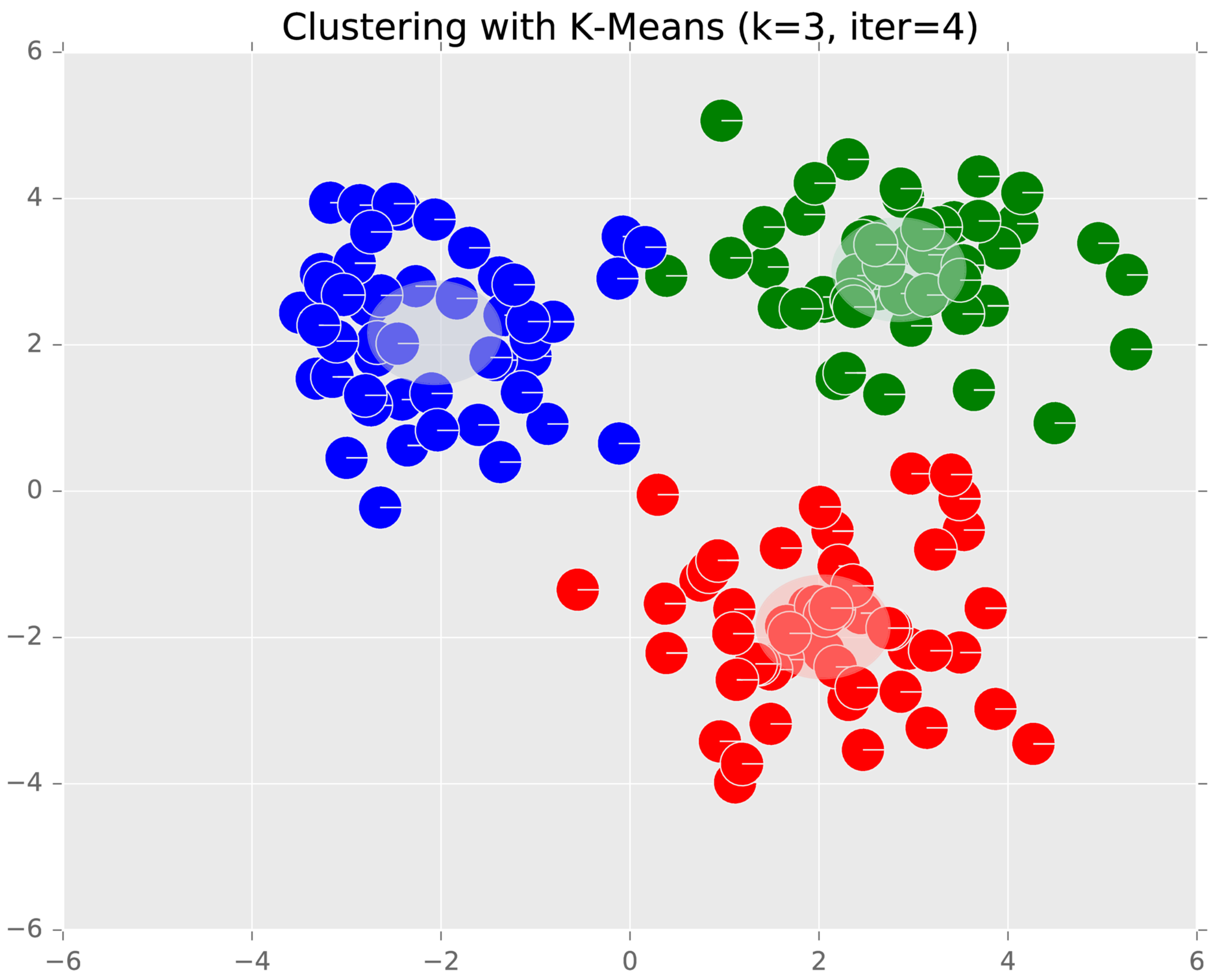


Figure courtesy of Matt Gormley

K-means: Example ($K = 3$)

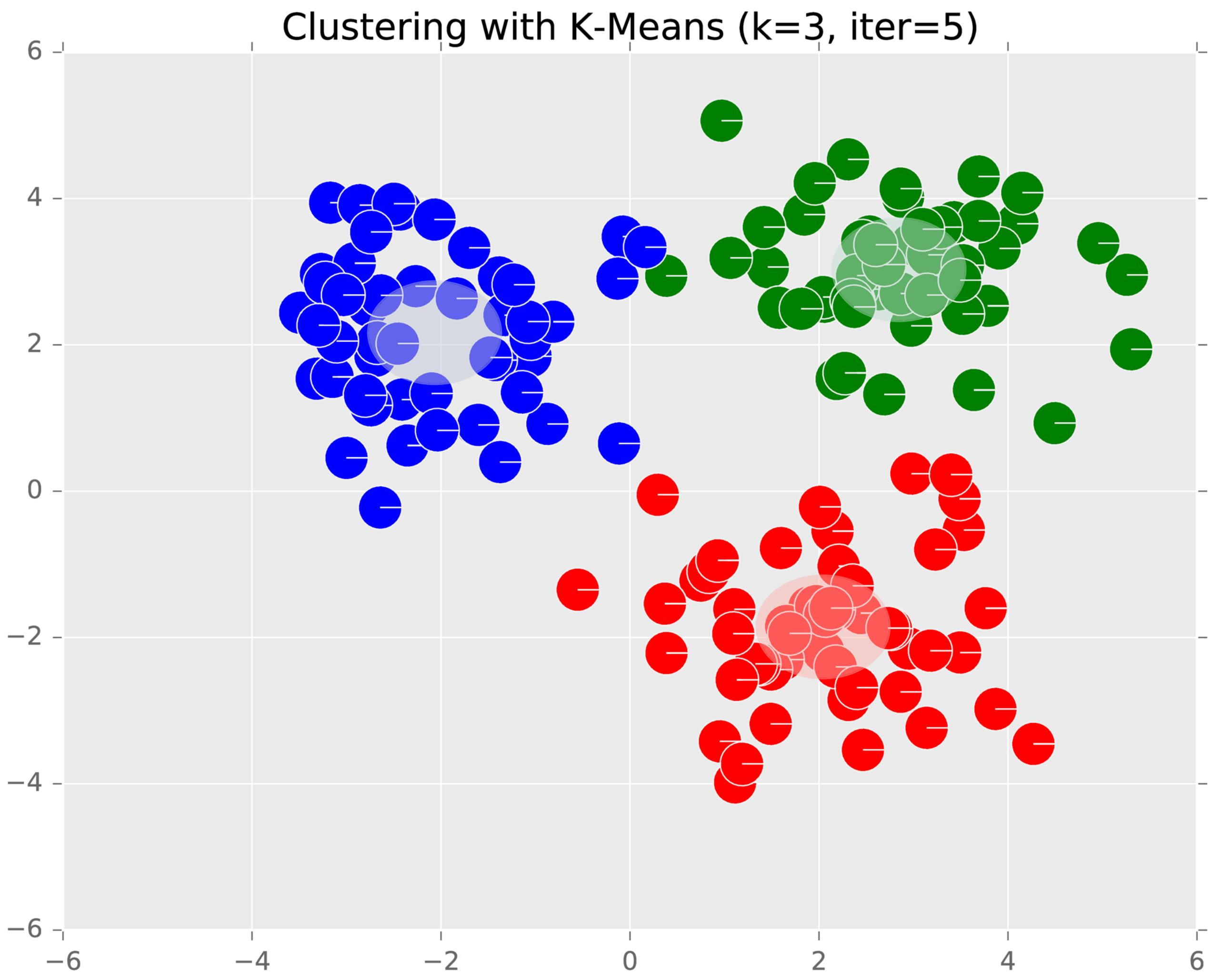


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

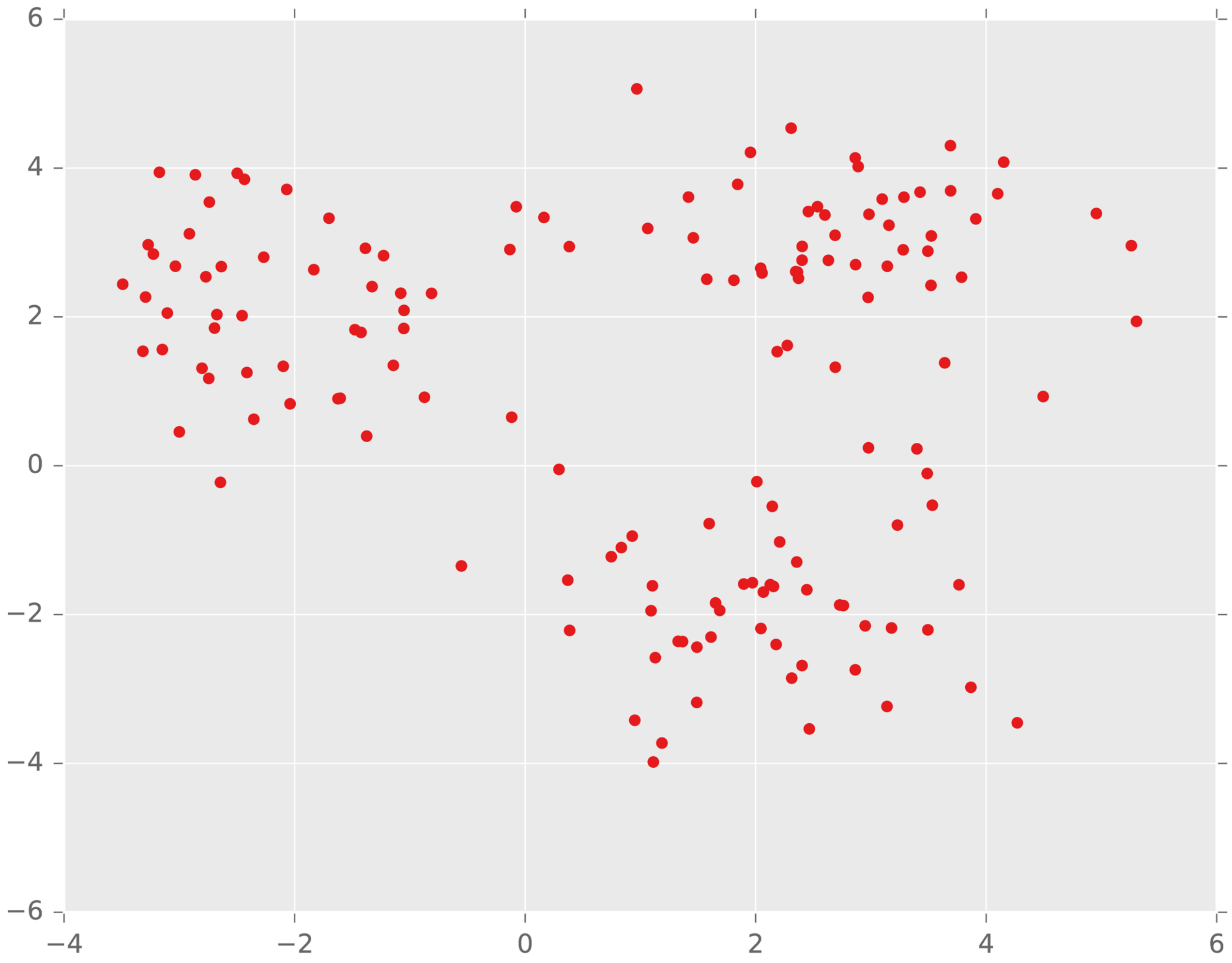


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

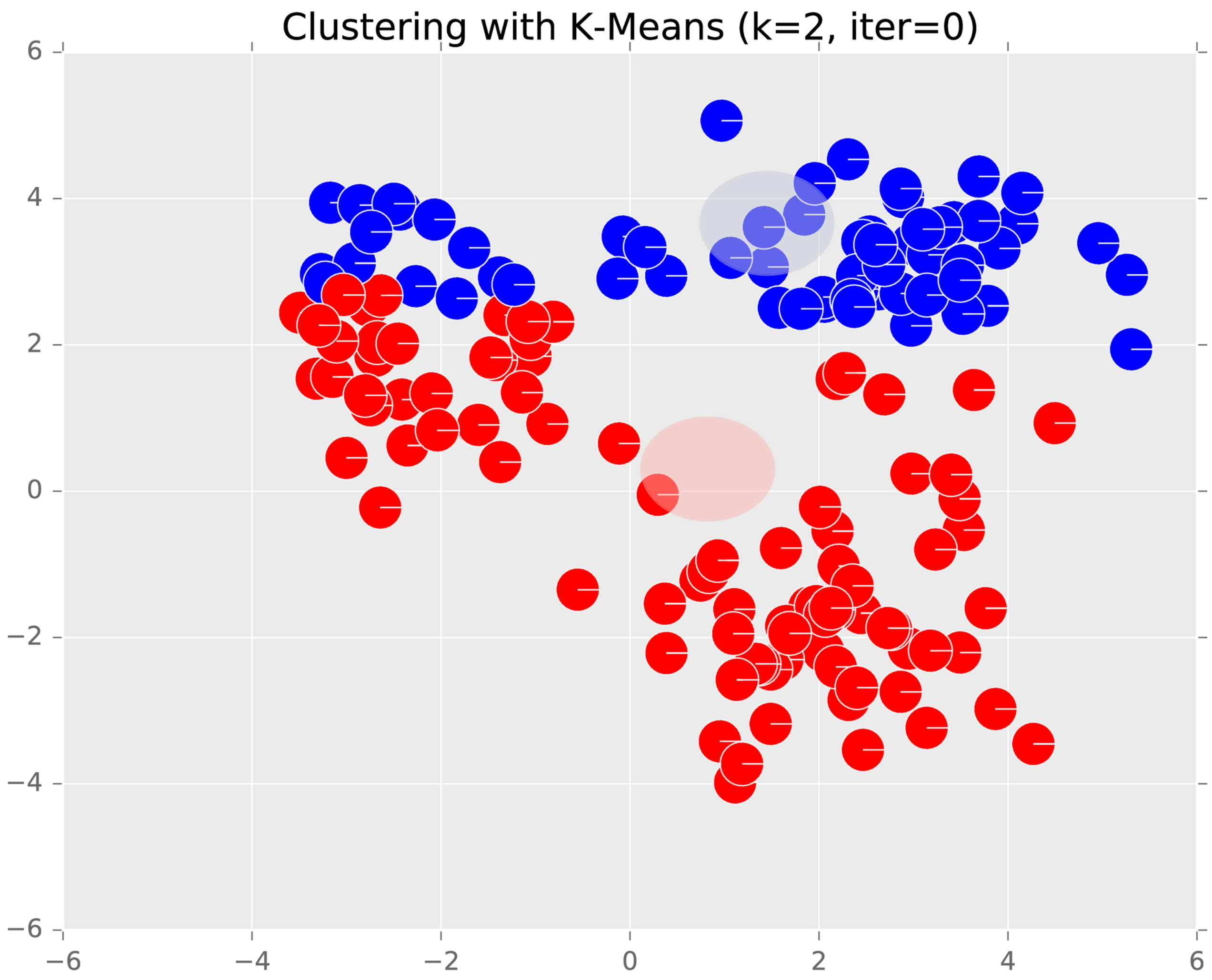


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

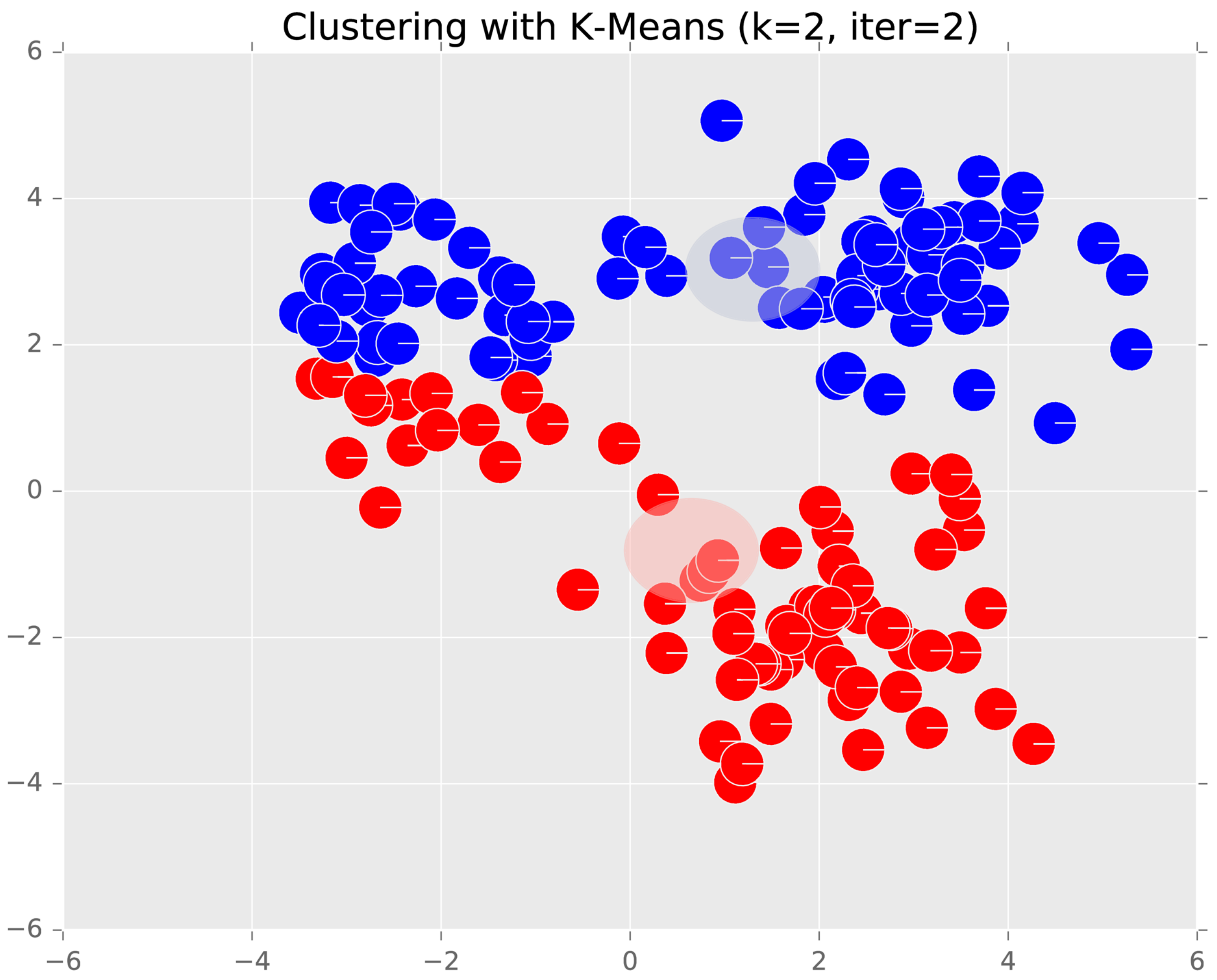


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

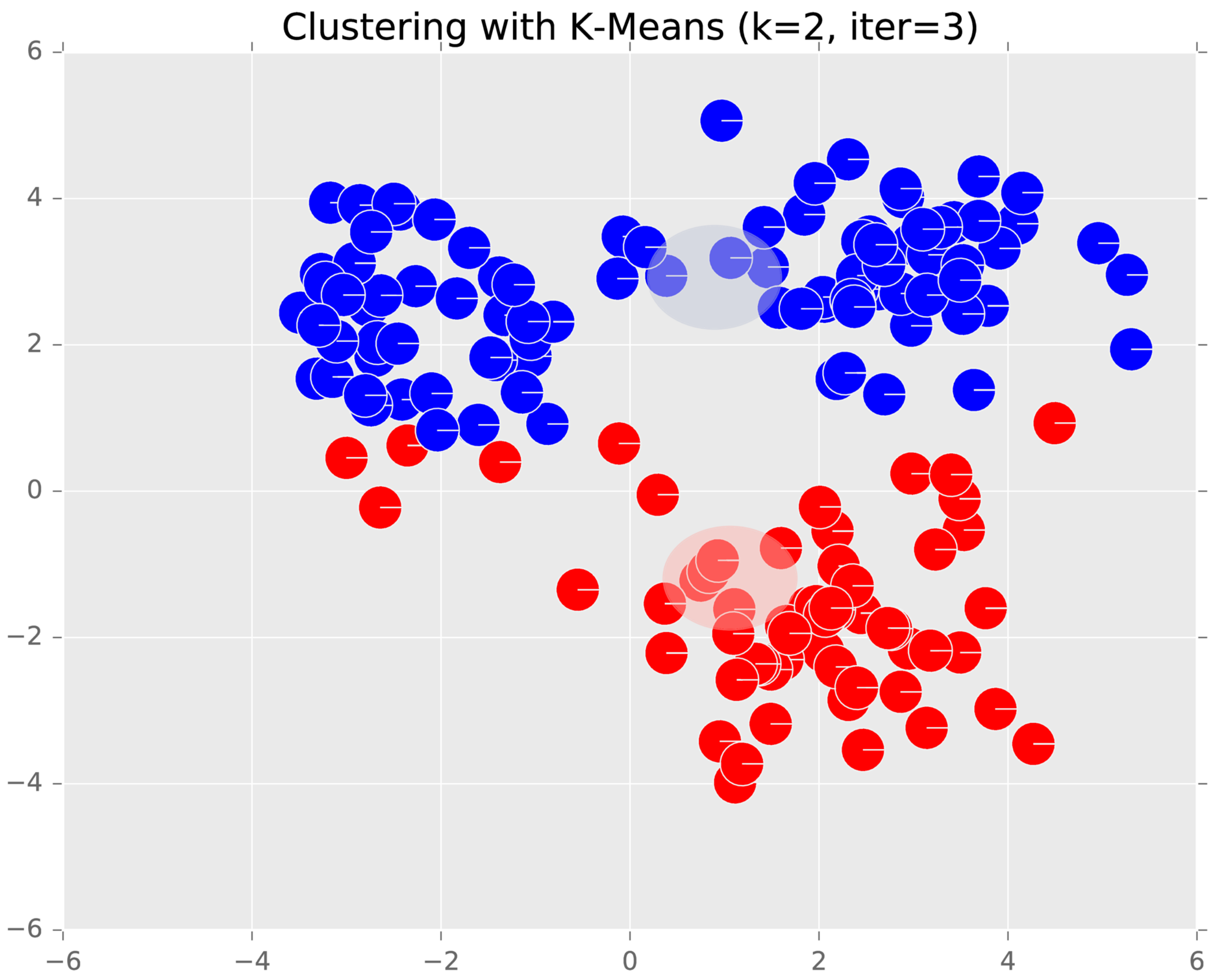


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

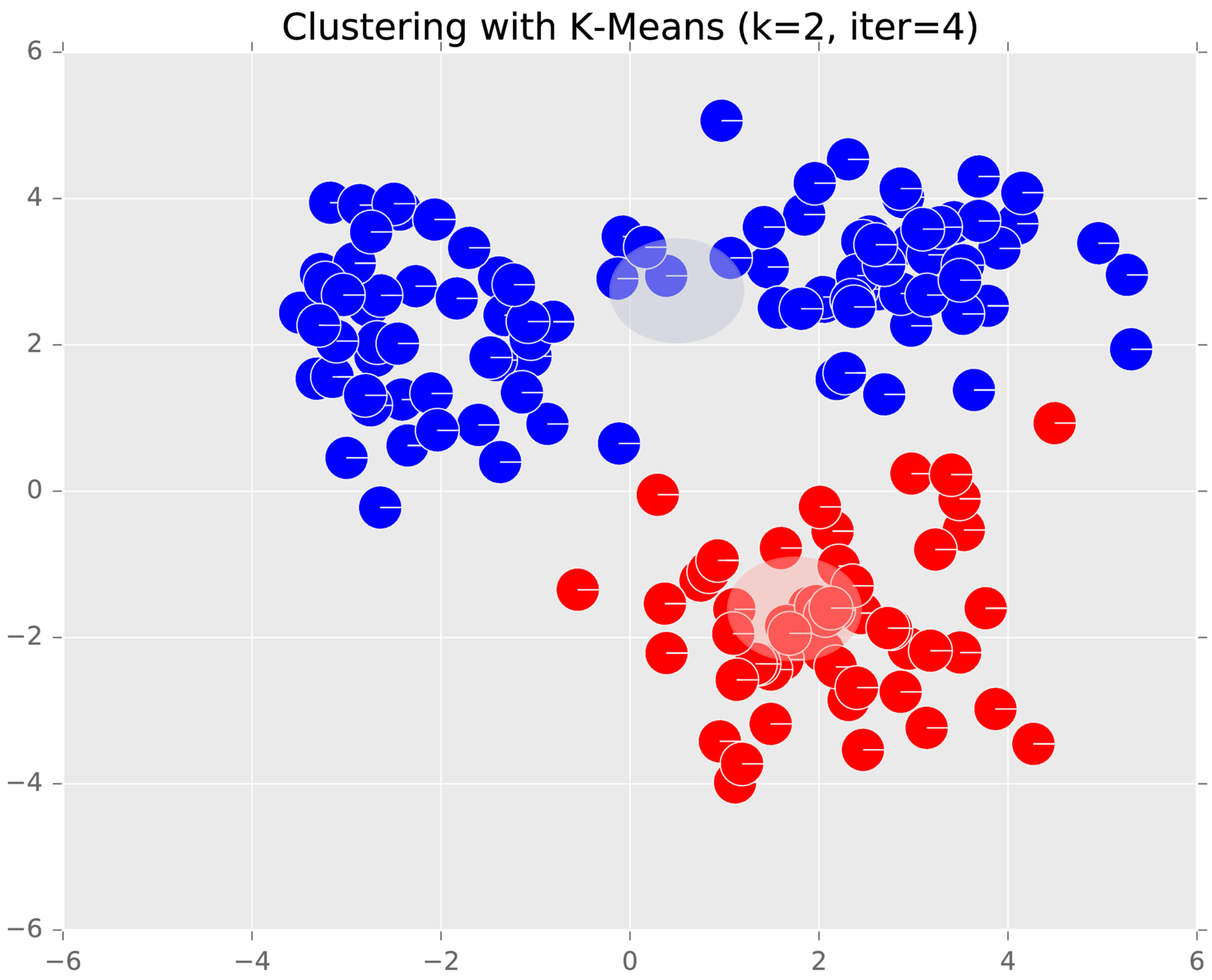


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

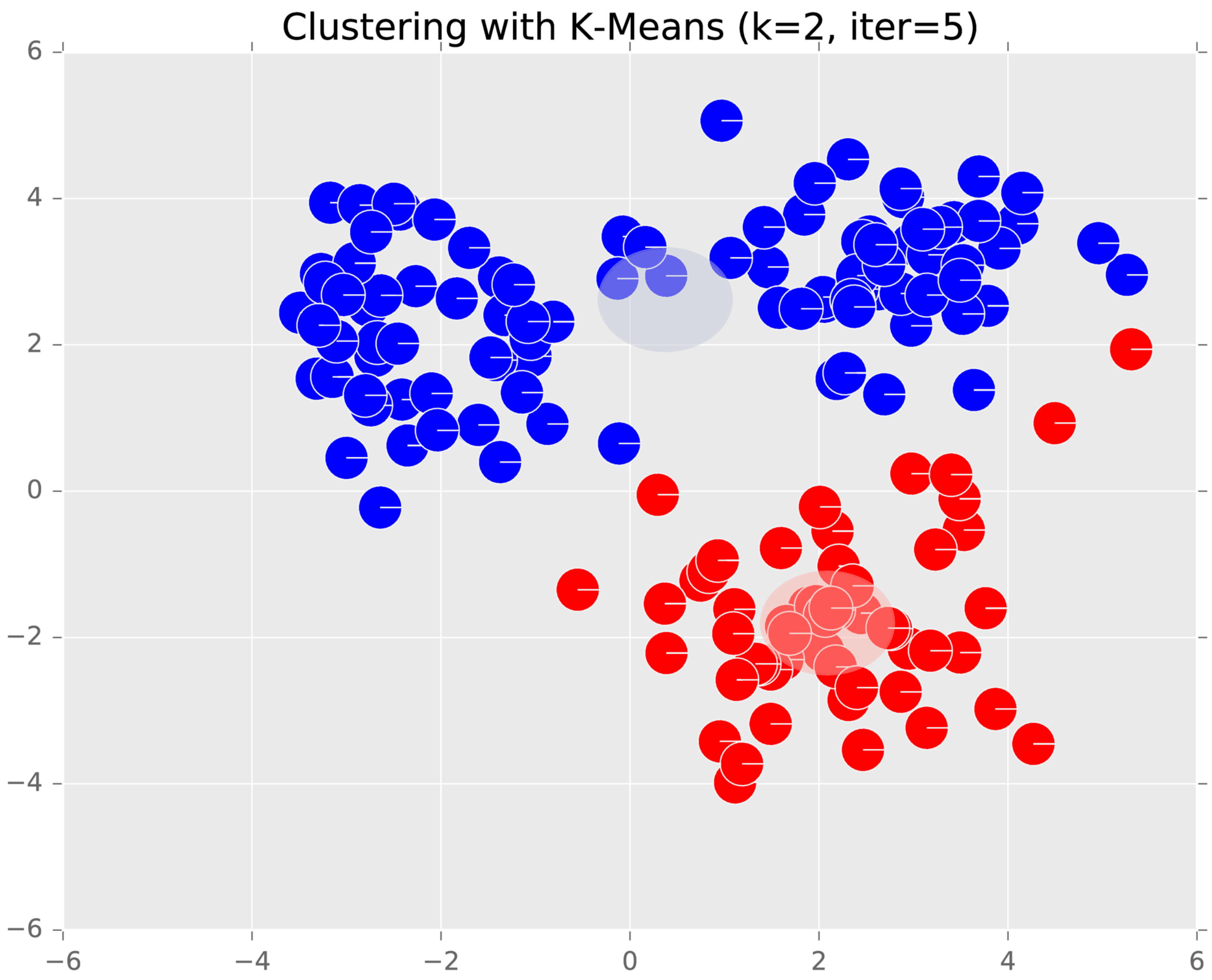


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

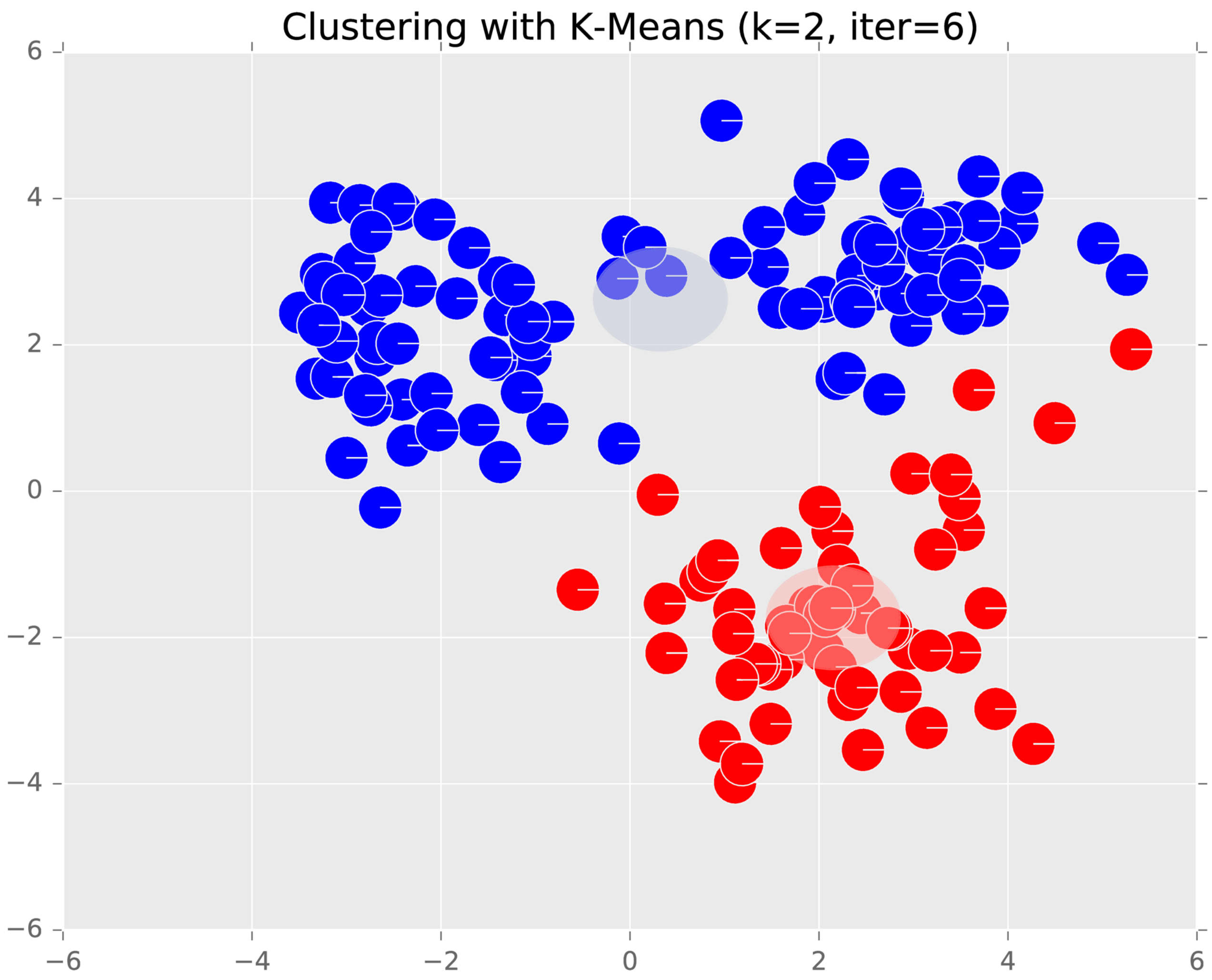


Figure courtesy of Matt Gormley

K-means: Example ($K = 2$)

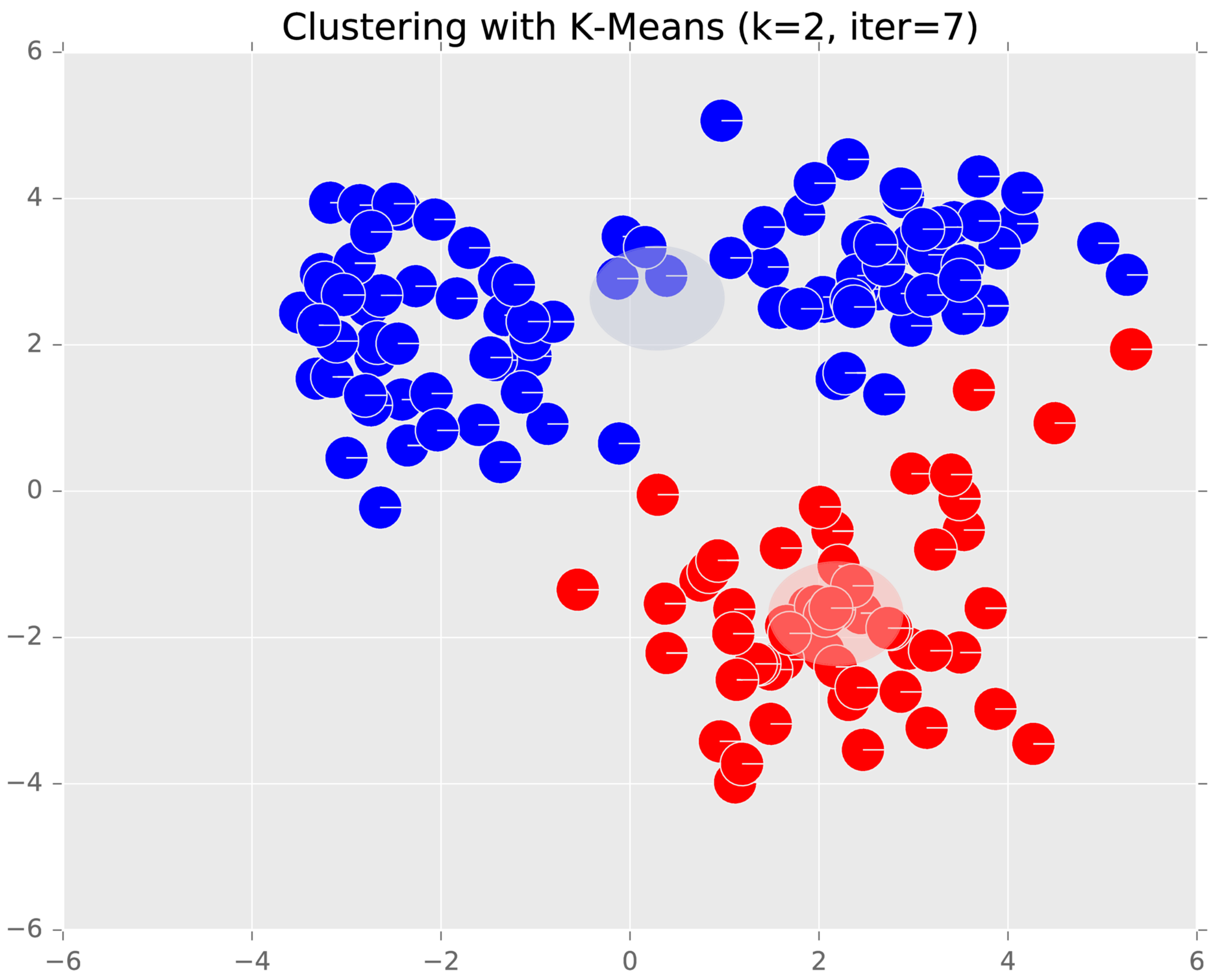
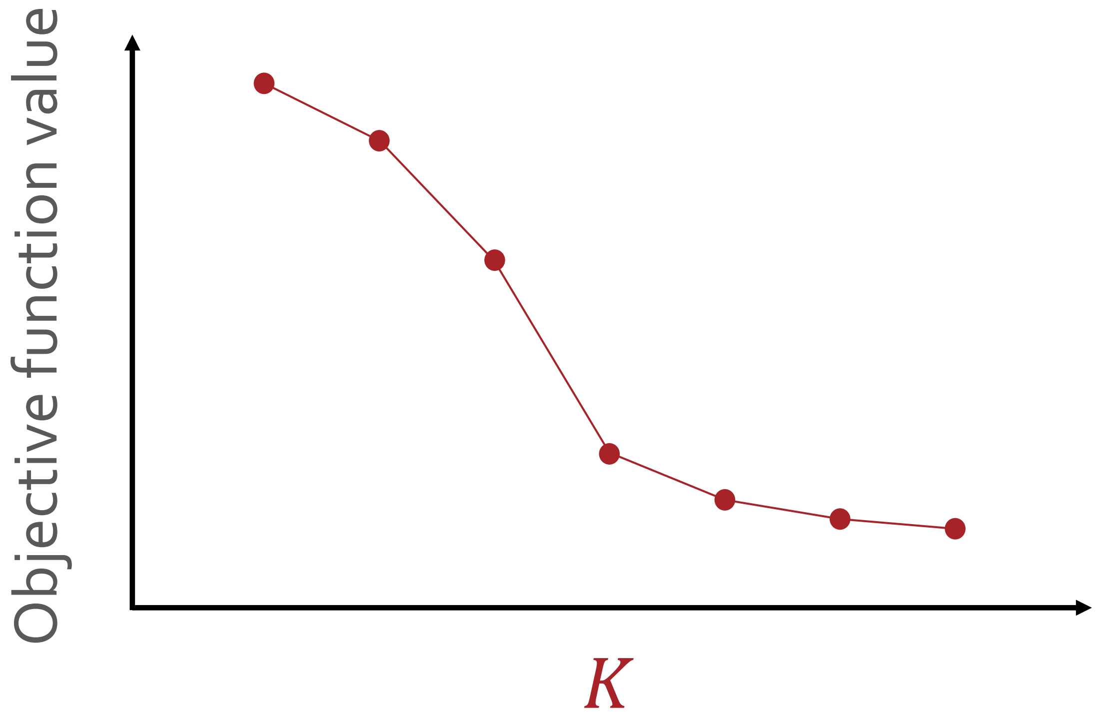


Figure courtesy of Matt Gormley

Setting K

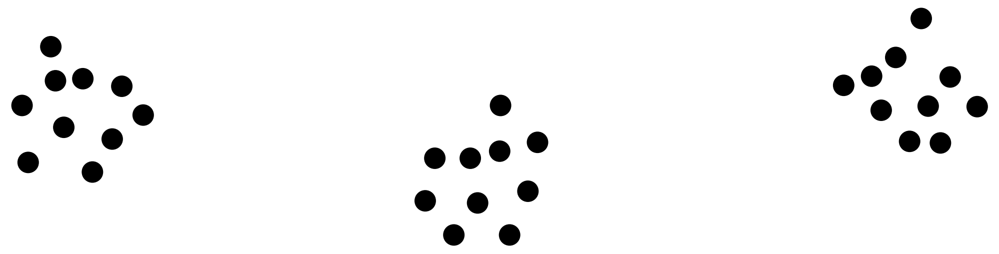
- Idea: choose the value of K that minimizes the objective function



- Better idea: look for the characteristic "elbow" or largest decrease when going from $K - 1$ to K

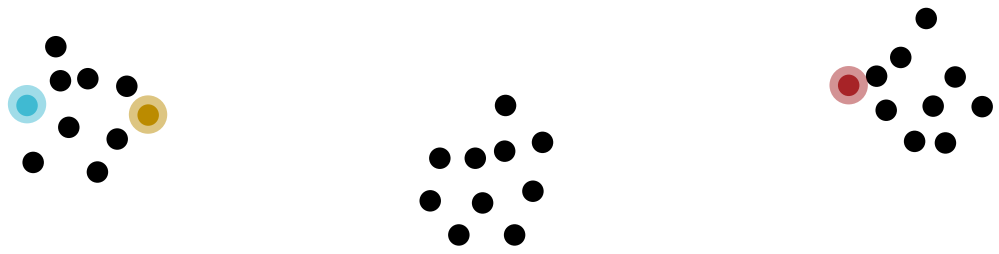
Initializing K-means

- Common choice: choose K data points at random to be the initial cluster centers (Lloyd's method)



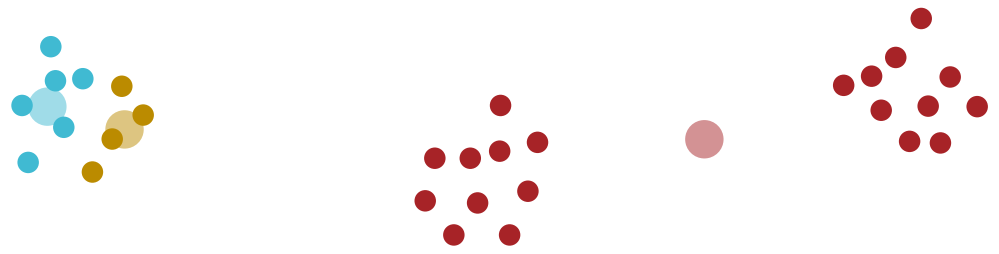
Initializing K-means

- Common choice: choose K data points at random to be the initial cluster centers (Lloyd's method)



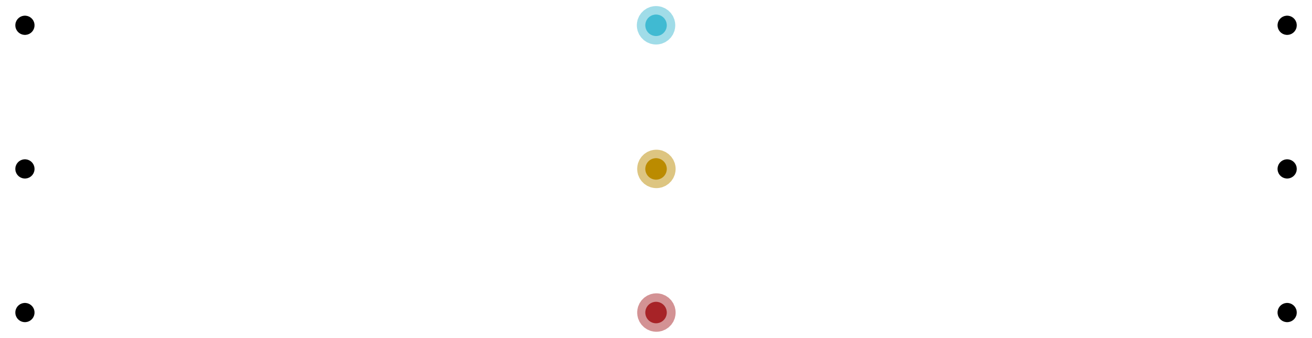
Initializing K-means

- Common choice: choose K data points at random to be the initial cluster centers (Lloyd's method)



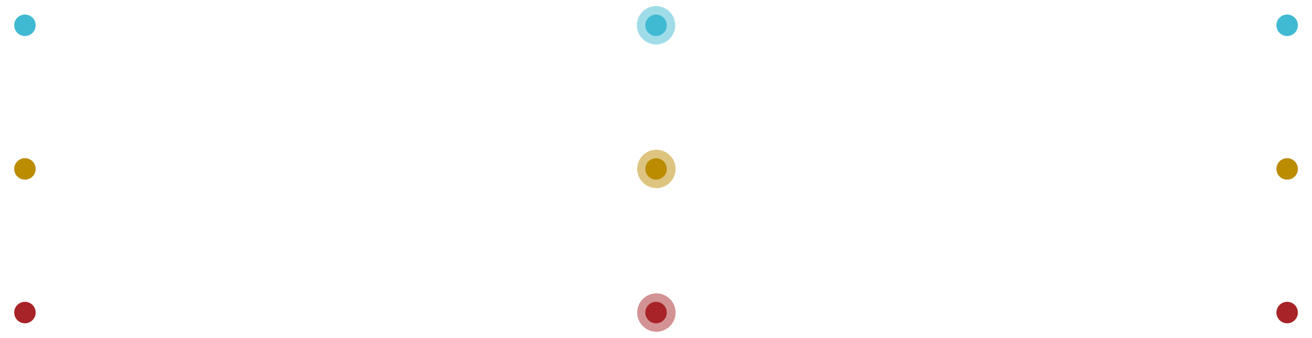
Initializing K-means

- Common choice: choose K data points at random to be the initial cluster centers (Lloyd's method)



Initializing K-means

- Common choice: choose K data points at random to be the initial cluster centers (Lloyd's method)



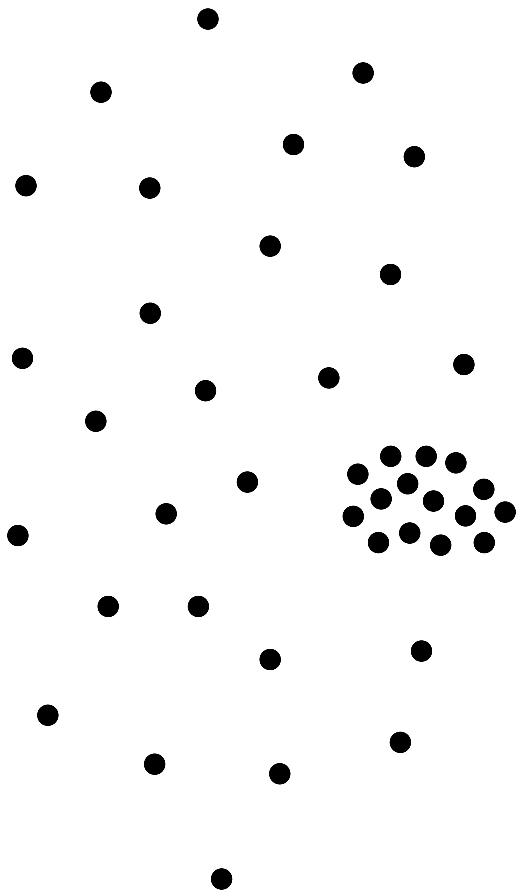
- Lloyd's method converges to a local minimum and that local minimum can be arbitrarily bad (relative to the optimal clusters)
- Intuition: want initial cluster centers to be far apart from one another

K-means++ (Arthur and Vassilvitskii, 2007)

1. Choose the first cluster center randomly from the data points.
2. For each other data point $\boldsymbol{x}$, compute $D(\boldsymbol{x})$, the distance between $\boldsymbol{x}$ and the closest cluster center.
3. Select the next cluster center proportional to $D(\boldsymbol{x})^2$.
4. Repeat 2 and 3 $K - 1$ times.

- K-means++ achieves a $O(\log K)$ approximation to the optimal clustering in expectation
- Both Lloyd's method and K-means++ can benefit from multiple random restarts.

Shortcomings of K-means



- Clusters cannot overlap
- Clusters must all be of the same "width"
- Clusters must be linearly separable

Probabilistic or “Soft” Assignments

- Instead of $z^{(i)}$ being a deterministic scalar, let $\boldsymbol{z}^{(i)}$ be a 1-of-K vector indicating cluster membership
  - For example, $\boldsymbol{z}^{(1)} = [0, 1, 0, \dots, 0]$ indicates that the first data point belongs to the second cluster
  - Let $\pi_k \coloneqq p\left(z_k^{(i)} = 1\right)$

Gaussian Mixture Models (GMMs)

Assume the following data-generating model for our dataset, $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$

1. Sample a cluster at random: $p\left(z_k^{(i)} = 1\right) = \pi_k$
2. Sample a data point from the chosen cluster: $p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1\right) \sim N(\mu_k, \Sigma_k)$

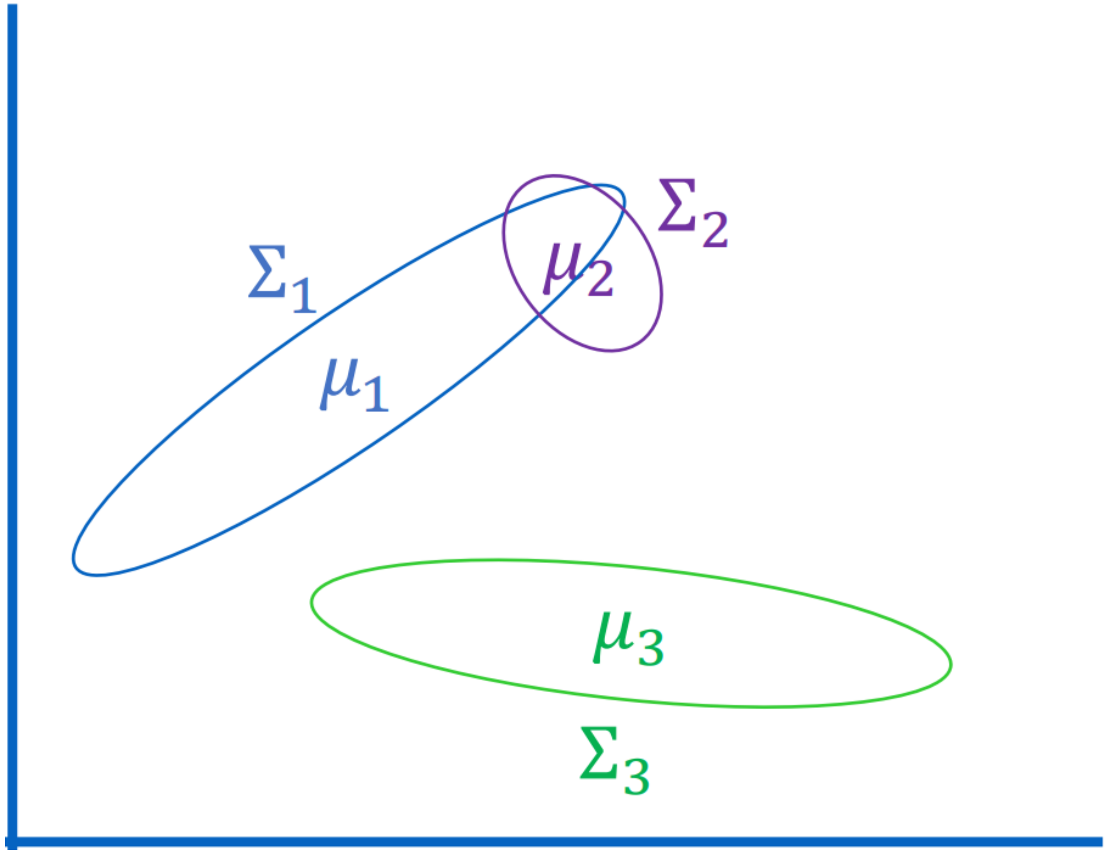
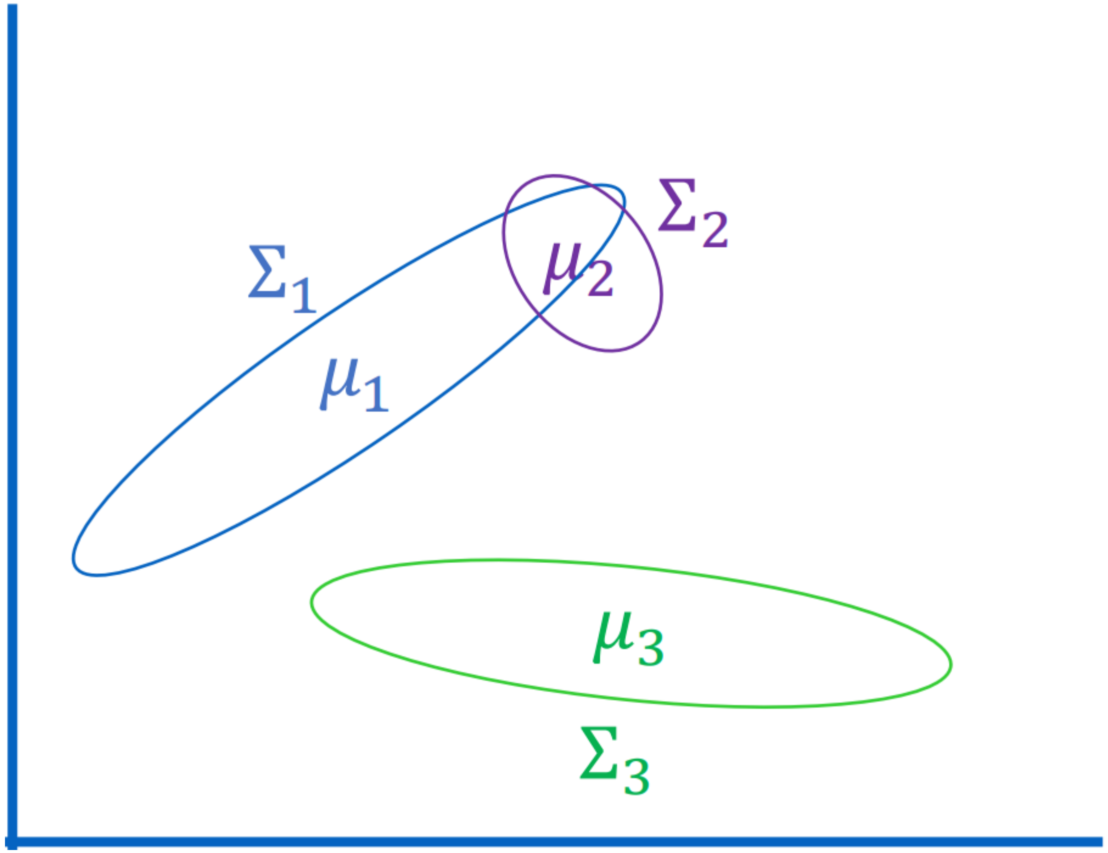


Figure courtesy of Pat Virtue

Gaussian Mixture Models (GMMs)

Assume the following data-generating model for our dataset, $\mathcal{D} = \left\{\boldsymbol{x}^{(i)}\right\}_{i=1}^{N}$

1. Sample a cluster at random: $p\left(z_k^{(i)} = 1\right) = \pi_k$
2. Sample a data point from the chosen cluster: $p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1\right) \sim N(\mu_k, \Sigma_k)$



Let $\theta = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \Sigma_1, \ldots, \Sigma_K, \pi_1, \ldots, \pi_K\}$

Figure courtesy of Pat Virtue

Maximizing the Likelihood?

- The log likelihood of $\mathcal{D} = \{\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}\}_{i=1}^{N}$ is

$$\ell(\theta|\mathcal{D}) = \log \prod_{i=1}^{N} p(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}|\theta) = \sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}|\theta)$$

$$= \sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i)}, \theta) + \log p(\boldsymbol{z}^{(i)}|\theta)$$

$$= \sum_{i=1}^{N} \log \prod_{k=1}^{K} p\left(\boldsymbol{x}^{(i)}\middle|z_k^{(i)} = 1, \theta\right)^{z_k^{(i)}} + \log \prod_{k=1}^{K} p\left(z_k^{(i)} = 1\middle|\theta\right)^{z_k^{(i)}}$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \log p\left(\boldsymbol{x}^{(i)}\middle|z_k^{(i)} = 1, \theta\right) + \sum_{k=1}^{K} z_k^{(i)} \log p\left(z_k^{(i)} = 1\middle|\theta\right)$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \left(\log N(\boldsymbol{x}^{(i)}; \mu_k, \Sigma_k) + \log \pi_k\right)$$

Maximizing the Complete Likelihood

- The log complete likelihood of $\mathcal{D} = \{\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}\}_{i=1}^{N}$ is

$$\ell_c(\theta|\mathcal{D}_c) = \log \prod_{i=1}^{N} p\left(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)} \middle| \theta\right) = \sum_{i=1}^{N} \log p\left(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)} \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \log p\left(\boldsymbol{x}^{(i)} \middle| \boldsymbol{z}^{(i)}, \theta\right) + \log p\left(\boldsymbol{z}^{(i)} \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \log \prod_{k=1}^{K} p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1, \theta\right)^{z_k^{(i)}} + \log \prod_{k=1}^{K} p\left(z_k^{(i)} = 1 \middle| \theta\right)^{z_k^{(i)}}$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \log p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1, \theta\right) + \sum_{k=1}^{K} z_k^{(i)} \log p\left(z_k^{(i)} = 1 \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \left(\log N\left(\boldsymbol{x}^{(i)}; \mu_k, \Sigma_k\right) + \log \pi_k\right)$$

Maximizing the Complete Likelihood is easy but requires $z^{(i)}$!

- The log complete likelihood of $\mathcal{D} = \{\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)}\}_{i=1}^{N}$ is

$$\ell_c(\theta|\mathcal{D}_c) = \log \prod_{i=1}^{N} p\left(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)} \middle| \theta\right) = \sum_{i=1}^{N} \log p\left(\boldsymbol{x}^{(i)}, \boldsymbol{z}^{(i)} \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \log p\left(\boldsymbol{x}^{(i)} \middle| \boldsymbol{z}^{(i)}, \theta\right) + \log p\left(\boldsymbol{z}^{(i)} \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \log \prod_{k=1}^{K} p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1, \theta\right)^{z_k^{(i)}} + \log \prod_{k=1}^{K} p\left(z_k^{(i)} = 1 \middle| \theta\right)^{z_k^{(i)}}$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \log p\left(\boldsymbol{x}^{(i)} \middle| z_k^{(i)} = 1, \theta\right) + \sum_{k=1}^{K} z_k^{(i)} \log p\left(z_k^{(i)} = 1 \middle| \theta\right)$$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} z_k^{(i)} \left(\log N\left(\boldsymbol{x}^{(i)}; \mu_k, \Sigma_k\right) + \log \pi_k\right)$$

- Parameters decoupled → set partial derivatives equal to 0

Maximizing the Marginal Likelihood

- The log *marginal* likelihood of $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ is

$$\ell(\theta|\mathcal{D}) = \log \prod_{i=1}^{N} p(\boldsymbol{x}^{(i)}|\theta) = \sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)}|\theta)$$

$$= \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}^{(i)}} p(\boldsymbol{x}^{(i)}|\boldsymbol{z}^{(i)},\theta)p(\boldsymbol{z}^{(i)}|\theta)$$

$$= \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}^{(i)}} \prod_{k=1}^{K} \left(p\left(\boldsymbol{x}^{(i)}|z_k^{(i)} = 1, \theta\right) p\left(z_k^{(i)} = 1\middle|\theta\right)\right)^{z_k^{(i)}}$$

$$= \sum_{i=1}^{N} \log \sum_{\boldsymbol{z}^{(i)}} \prod_{k=1}^{K} \left(N(\boldsymbol{x}^{(i)}; \mu_k, \Sigma_k)\pi_k\right)^{z_k^{(i)}}$$

- Parameters coupled and *constrained* → gradient ascent is possible but complicated and slow to converge

Recipe for GMMs

- Define a model and model parameters
  - Assume K Gaussian clusters
  - Parameters: $\theta = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K, \Sigma_1, \dots, \Sigma_K, \pi_1, \dots, \pi_K\}$
- Write down an objective function
  - Maximize the log marginal likelihood

$$\ell(\theta|\mathcal{D}) = \log \prod_{i=1}^{N} p\left(\boldsymbol{x}^{(i)}\middle|\theta\right)$$

- Optimize the objective w.r.t. the model parameters
  - Expectation-maximization

Expectation-Maximization for GMMs: Intuition

- Insight: if we knew the cluster assignments, $\boldsymbol{z}^{(i)}$, we could maximize the log complete likelihood instead of the log marginal likelihood
- Idea: replace $\boldsymbol{z}^{(i)}$ in the log complete likelihood with our "best guess" for $\boldsymbol{z}^{(i)}$ given the parameters and the data
- Observation: changing the parameters changes our "best guess" and vice versa
- Approach: iterate between updating our "best guess" and updating the parameters