CS182: Introduction to Machine Learning –Unsupervised Learning: Kmeans

Yujiao Shi

SIST, ShanghaiTech

Spring, 2025

Clustering, Informal Goals

Goal: Automatically partition unlabeled data into groups of similar data points.

Question: When and why would we want to do this?

Useful for:

- Automatically organizing data.
- Understanding hidden structure in data.
- Preprocessing for further analysis.
  - Representing high-dimensional data in a low-dimensional space (e.g., for visualization purposes).

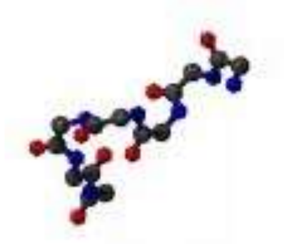
Slide courtesy of Nina Balcan

Applications (Clustering comes up everywhere…)

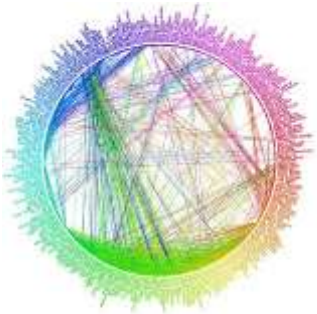
- Cluster news articles or web pages or search results by topic.

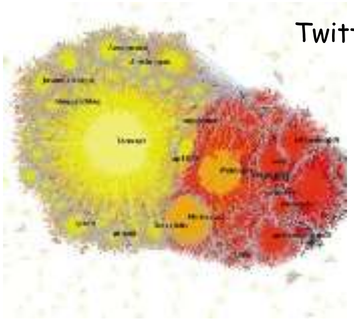



- Cluster protein sequences by function or genes according to expression profile.




- Cluster users of social networks by interest (community detection).


Facebook network


Twitter Network

Slide courtesy of Nina Balcan

Applications (Clustering comes up everywhere…)

- Cluster customers according to purchase history.

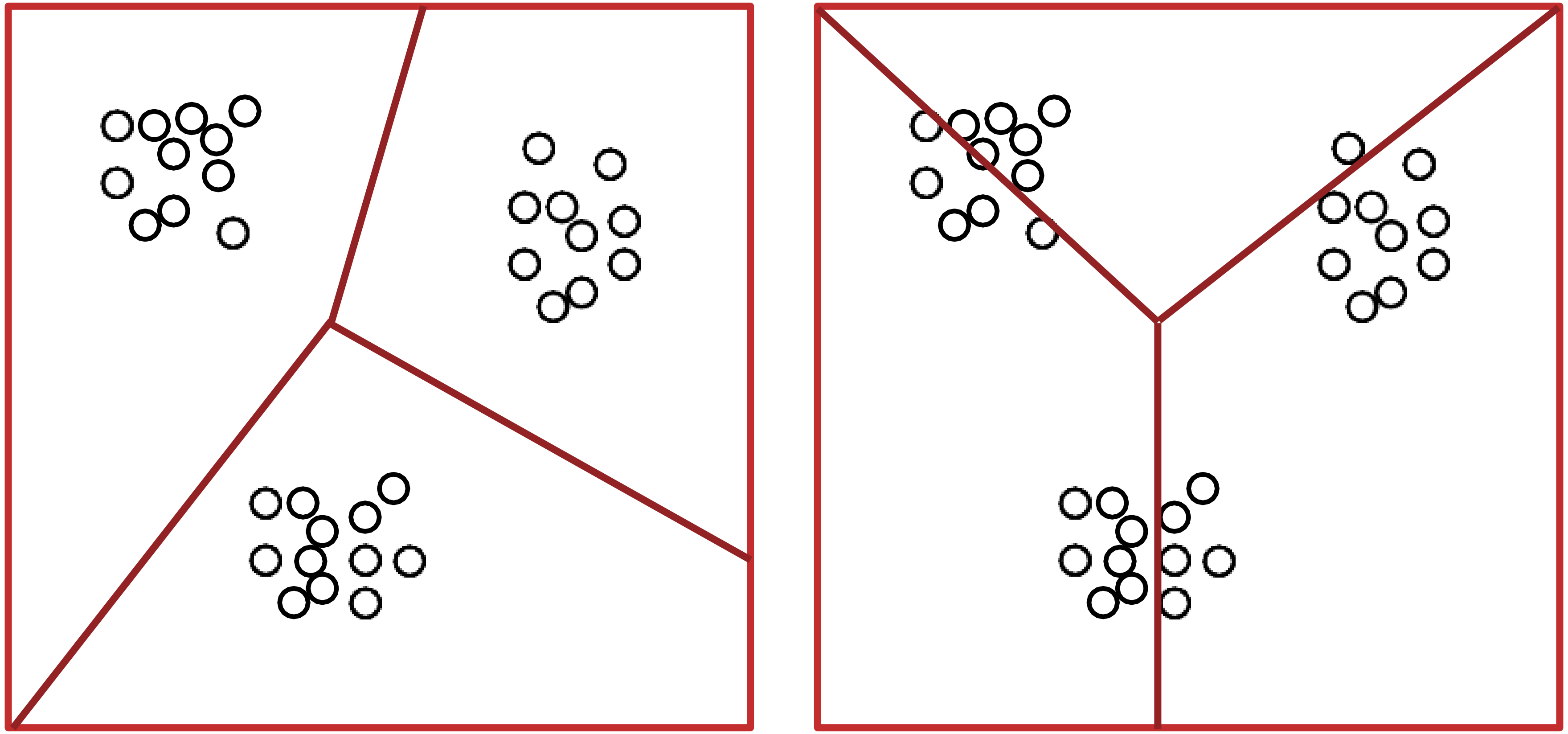
- Cluster galaxies or nearby stars (e.g. Sloan Digital Sky Survey)

- And many many more applications….

Slide courtesy of Nina Balcan

Clustering

Question: Which of these partitions is "better"?



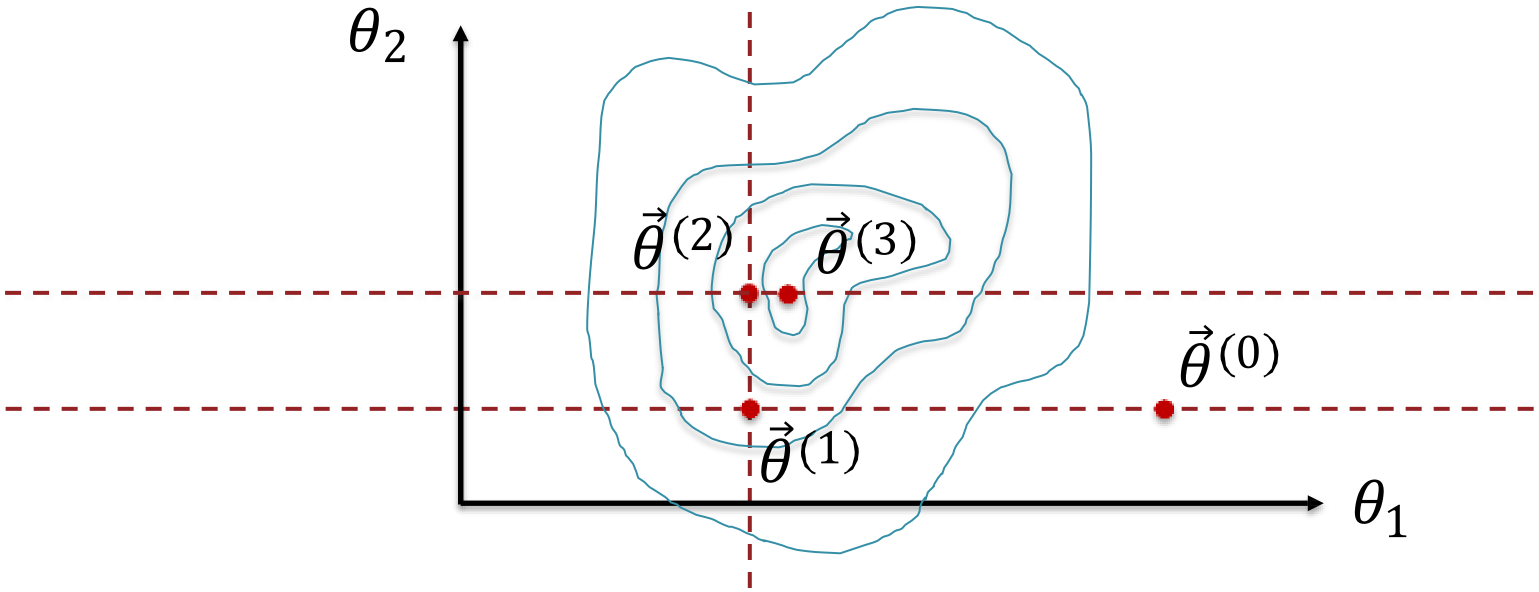
OPTIMIZATION BACKGROUND

Coordinate Descent

- Goal: minimize some objective

$$\vec{\theta}^* = \underset{\vec{\theta}}{\operatorname{argmin}}\, J(\vec{\theta})$$

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



Block Coordinate Descent

上海科技大学 ShanghaiTech University

- Goal: minimize some objective (with 2 blocks)

$$\vec{\alpha}^*, \vec{\beta}^* = \underset{\vec{\alpha}, \vec{\beta}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

- Idea: iteratively pick one *block* of variables ($\vec{\alpha}$ or $\vec{\beta}$) and minimize the objective w.r.t. that block, keeping the other(s) fixed.

while not converged:

$$\vec{\alpha} = \underset{\vec{\alpha}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

$$\vec{\beta} = \underset{\vec{\beta}}{\operatorname{argmin}} J(\vec{\alpha}, \vec{\beta})$$

K-MEANS

K-Means Algorithm (Derivation)

上海科技大学
ShanghaiTech University

Recipe for K-Means Derivation:

1) Define a Model.
2) Choose an objective function.
3) Optimize it!

K-Means Algorithm (Derivation)

上海科技大学 ShanghaiTech University

- Input: unlabeled data $\mathrm{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}, \ \mathbf{x}^{(i)} \in \mathrm{R}^M$
- Goal: Find an assignment of points to clusters
- Model Paramters:
  - cluster centers: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K], \ \mathbf{c}_j \in \mathrm{R}^M$
  - cluster assignments: $\mathbf{z} = [z^{(1)}, z^{(2)}, \ldots, z^{(N)}], \ z^{(i)} \in \{1, \ldots, K\}$
- Decision Rule: assign each point $\mathbf{x}^{(i)}$ to its nearest cluster center $\mathbf{c}_j$

K-Means Algorithm (Derivation)

上海科技大学 ShanghaiTech University

- Input: unlabeled data $D = \{\mathbf{x}^{(i)}\}_{i=1}^N, \ \mathbf{x}^{(i)} \in \mathbb{R}^M$
- Goal: Find an assignment of points to clusters
- Model Paramters:
  - cluster centers: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K], \ \mathbf{c}_j \in \mathbb{R}^M$
  - cluster assignments: $\mathbf{z} = [z^{(1)}, z^{(2)}, \ldots, z^{(N)}], \ z^{(i)} \in \{1, \ldots, K\}$
- Decision Rule: assign each point $\mathbf{x}^{(i)}$ to its nearest cluster center $\mathbf{c}_j$
- Objective:

$$\hat{\mathbf{C}} = \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{N} \boxed{\min_j ||\mathbf{x}^{(i)} - \mathbf{c}_j||_2^2}$$

Question: In English, what is this quantity?

Answer:

K-Means Algorithm (Derivation)

上海科技大学 ShanghaiTech University

- Input: unlabeled data $D = \{\mathbf{x}^{(i)}\}_{i=1}^{N}, \ \mathbf{x}^{(i)} \in \mathrm{R}^M$
- Goal: Find an assignment of points to clusters
- Model Paramters:
  - cluster centers: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K], \ \mathbf{c}_j \in \mathrm{R}^M$
  - cluster assignments: $\mathbf{z} = [z^{(1)}, z^{(2)}, \ldots, z^{(N)}], \ z^{(i)} \in \{1, \ldots, K\}$
- Decision Rule: assign each point $\mathbf{x}^{(i)}$ to its nearest cluster center $\mathbf{c}_j$
- Objective:

$$\begin{aligned}\hat{\mathbf{C}} &= \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{N} \min_{j} ||\mathbf{x}^{(i)} - \mathbf{c}_j||_2^2 \\ &= \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{N} \min_{z^{(i)}} ||\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}||_2^2\end{aligned}$$

K-Means Algorithm (Derivation)

上海科技大学 ShanghaiTech University

- Input: unlabeled data $D = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$, $\mathbf{x}^{(i)} \in R^M$
- Goal: Find an assignment of points to clusters
- Model Paramters:
  - cluster centers: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K]$, $\mathbf{c}_j \in R^M$
  - cluster assignments: $\mathbf{z} = [z^{(1)}, z^{(2)}, \ldots, z^{(N)}]$, $z^{(i)} \in \{1, \ldots, K\}$
- Decision Rule: assign each point $\mathbf{x}^{(i)}$ to its nearest cluster center $\mathbf{c}_j$
- Objective:

$$\hat{\mathbf{C}} = \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{N} \min_j ||\mathbf{x}^{(i)} - \mathbf{c}_j||_2^2$$

$$= \underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{N} \min_{z^{(i)}} ||\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}||_2^2$$

$$\hat{\mathbf{C}}, \hat{\mathbf{z}} = \underset{\mathbf{C},\mathbf{z}}{\operatorname{argmin}} \sum_{i=1}^{N} ||\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}||_2^2$$

K-Means Algorithm (Derivation)

上海科技大学 ShanghaiTech University

- Input: unlabeled data $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}, \ \mathbf{x}^{(i)} \in \mathbb{R}^M$
- Goal: Find an assignment of points to clusters
- Model Paramters:
  - cluster centers: $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K], \ \mathbf{c}_j \in \mathbb{R}^M$
  - cluster assignments: $\mathbf{z} = [z^{(1)}, z^{(2)}, \ldots, z^{(N)}], \ z^{(i)} \in \{1, \ldots, K\}$
- Decision Rule: assign each point $\mathbf{x}^{(i)}$ to its nearest cluster center $\mathbf{c}_j$
- Objective:

$$\hat{\mathbf{C}} = \operatorname*{argmin}_{\mathbf{C}} \sum_{i=1}^{N} \min_{j} ||\mathbf{x}^{(i)} - \mathbf{c}_j||_2^2$$

$$= \operatorname*{argmin}_{\mathbf{C}} \sum_{i=1}^{N} \min_{z^{(i)}} ||\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}||_2^2$$

$$\hat{\mathbf{C}}, \hat{\mathbf{z}} = \operatorname*{argmin}_{\mathbf{C}, \mathbf{z}} \sum_{i=1}^{N} ||\mathbf{x}^{(i)} - \mathbf{c}_{z^{(i)}}||_2^2$$

$$= \boxed{\operatorname*{argmin}_{\mathbf{C}, \mathbf{z}} J(\mathbf{C}, \mathbf{z})}$$

Now apply
Block Coordinate Descent!

K-Means Algorithm

1) **Given** unlabeled feature vectors
   $D = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$
2) **Initialize** cluster centers $c = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
3) **Repeat** until convergence:
   a) $\mathbf{z} \leftarrow \text{argmin}_{\mathbf{z}} J(\mathbf{C}, \mathbf{z})$
      (pick each *cluster assignment* to minimize distance)
   b) $\mathbf{C} \leftarrow \text{argmin}_{\mathbf{C}} J(\mathbf{C}, \mathbf{z})$
      (pick each *cluster center* to minimize distance)

This is an application of
Block Coordinate Descent!
The only remaining step is to figure out
what the argmins boil down to…

K-Means Algorithm

上海科技大学 ShanghaiTech University

1) **Given** unlabeled feature vectors
   $D = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$
2) **Initialize** cluster centers $c = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
3) **Repeat** until convergence:
   a) for i in $\{1, \ldots, N\}$
      $z^{(i)} \leftarrow \mathbf{argmin}_j \, (\| \mathbf{x}^{(i)} - \mathbf{c}_j \|_2)^2$
   b) for j in $\{1, \ldots, K\}$
      $\mathbf{c}_j \leftarrow \mathbf{argmin} \sum_{i:z^{(i)}=j} (\| \mathbf{x}^{(i)} - \mathbf{c}_j \|_2)^2$

The minimization over cluster assignments decomposes, so that we can find each $z^{(i)}$ independently of the others

Likewise, the minimization over cluster centers decomposes, so we can find each $\mathbf{c}_j$ independently

K-Means Algorithm

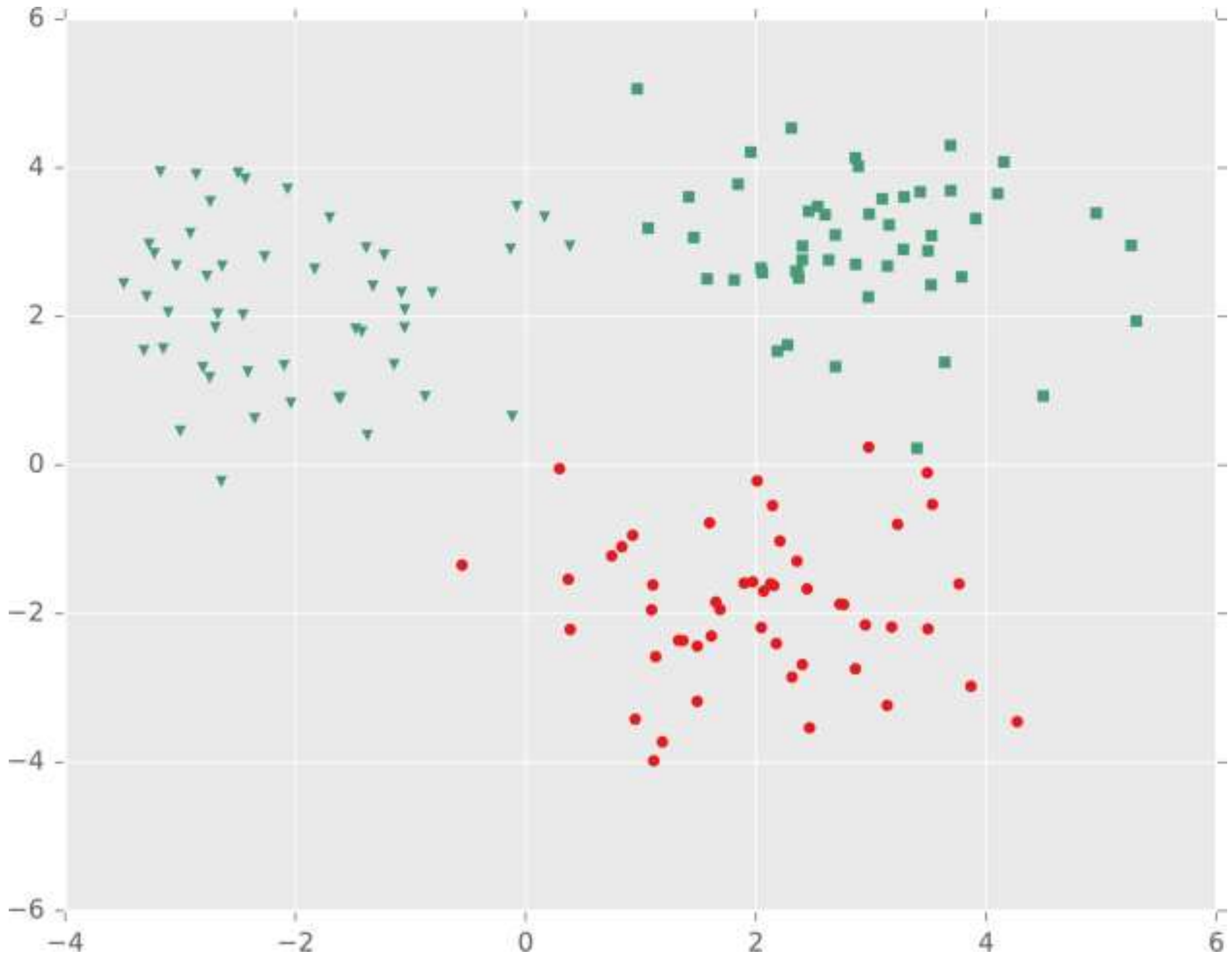
上海科技大学 ShanghaiTech University

1) **Given** unlabeled feature vectors
   $D = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$
2) **Initialize** cluster centers $c = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
3) **Repeat** until convergence:
   a) for i in $\{1, \ldots, N\}$
      $z^{(i)} \leftarrow$ **index** j of cluster center **nearest** to $\mathbf{x}^{(i)}$
   b) for j in $\{1, \ldots, K\}$
      $\mathbf{c}_j \leftarrow$ **mean** of **all** points assigned to cluster j

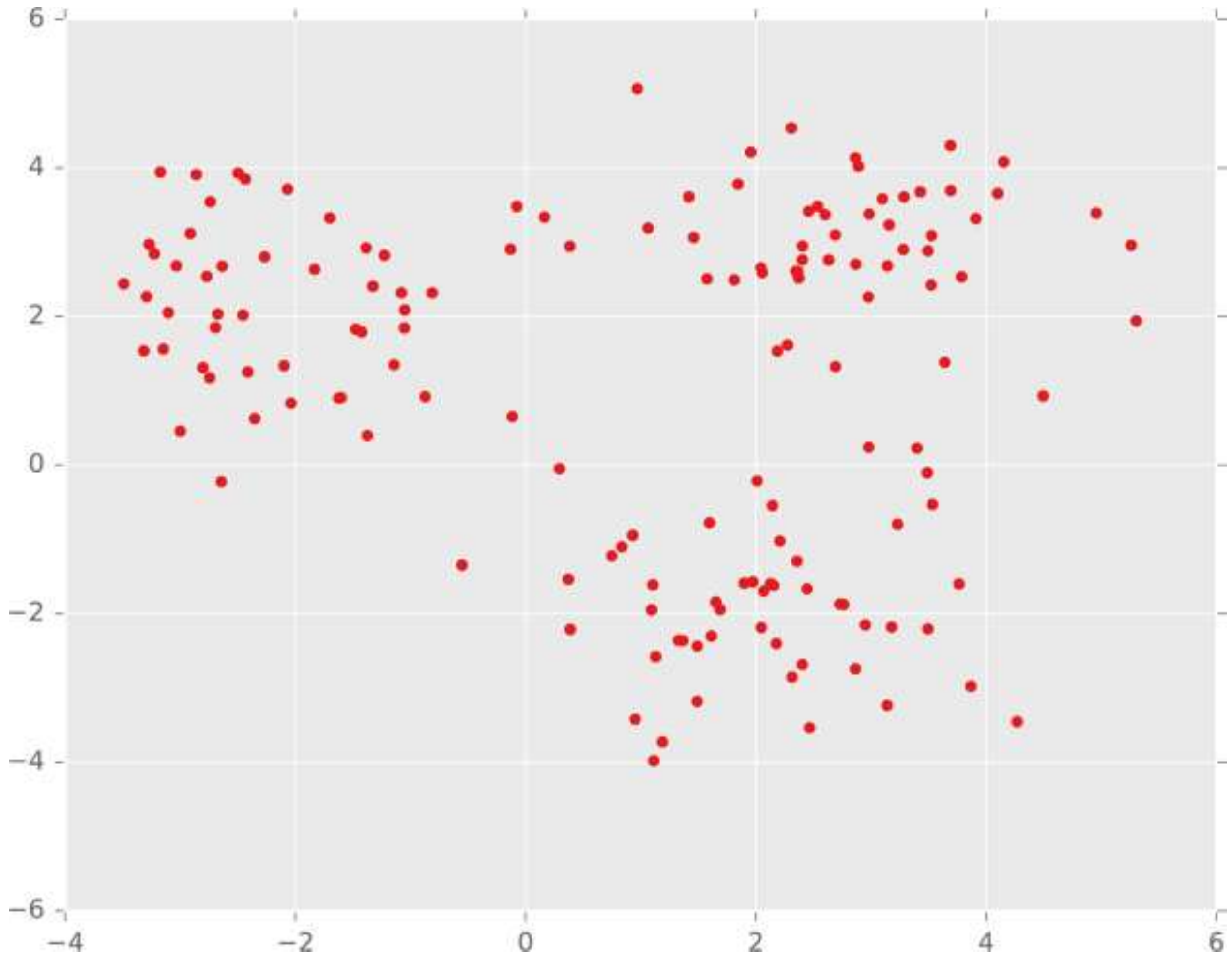
K=3 cluster centers

K-MEANS EXAMPLE

Example: K-Means

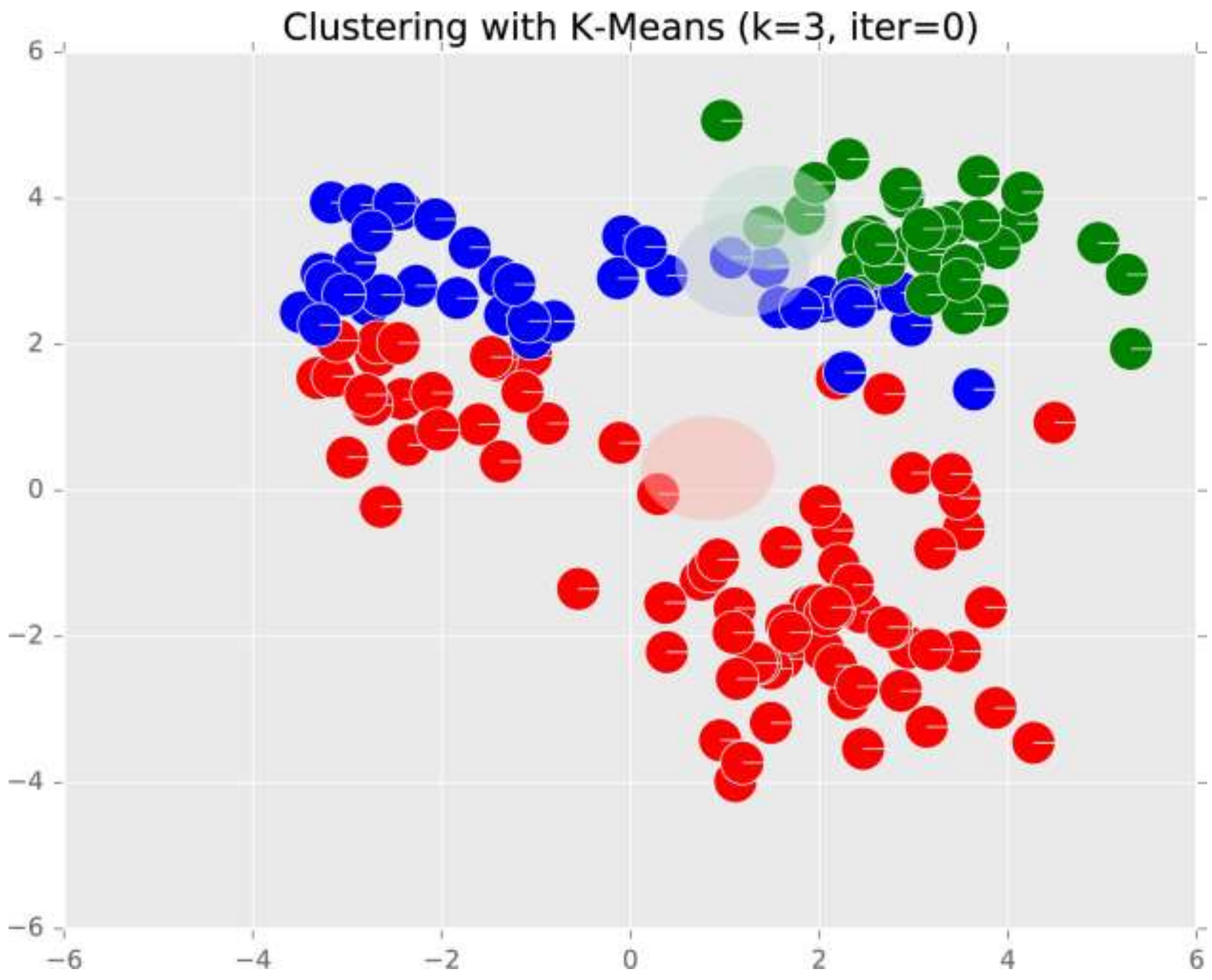


Example: K-Means

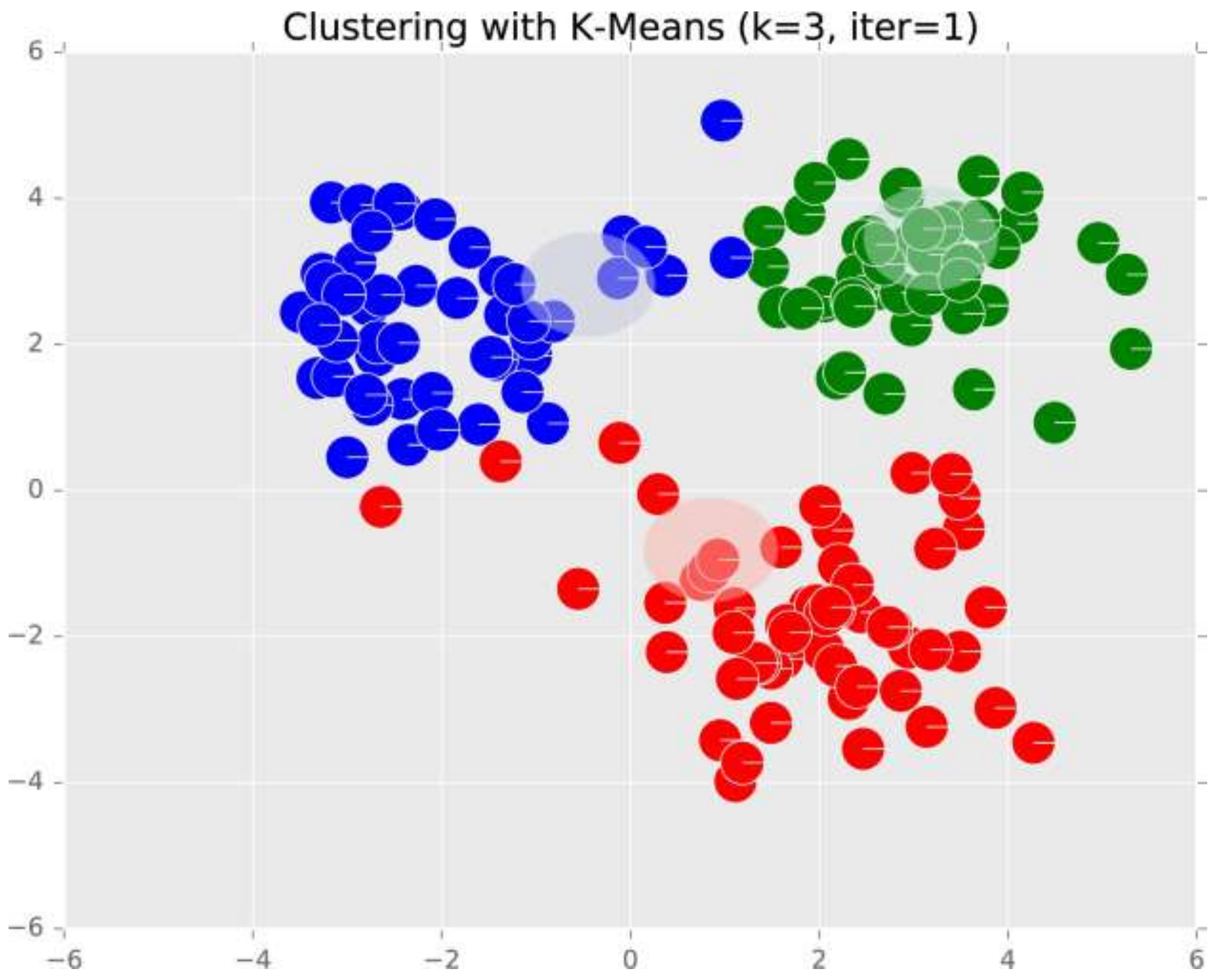


Example: K-Means

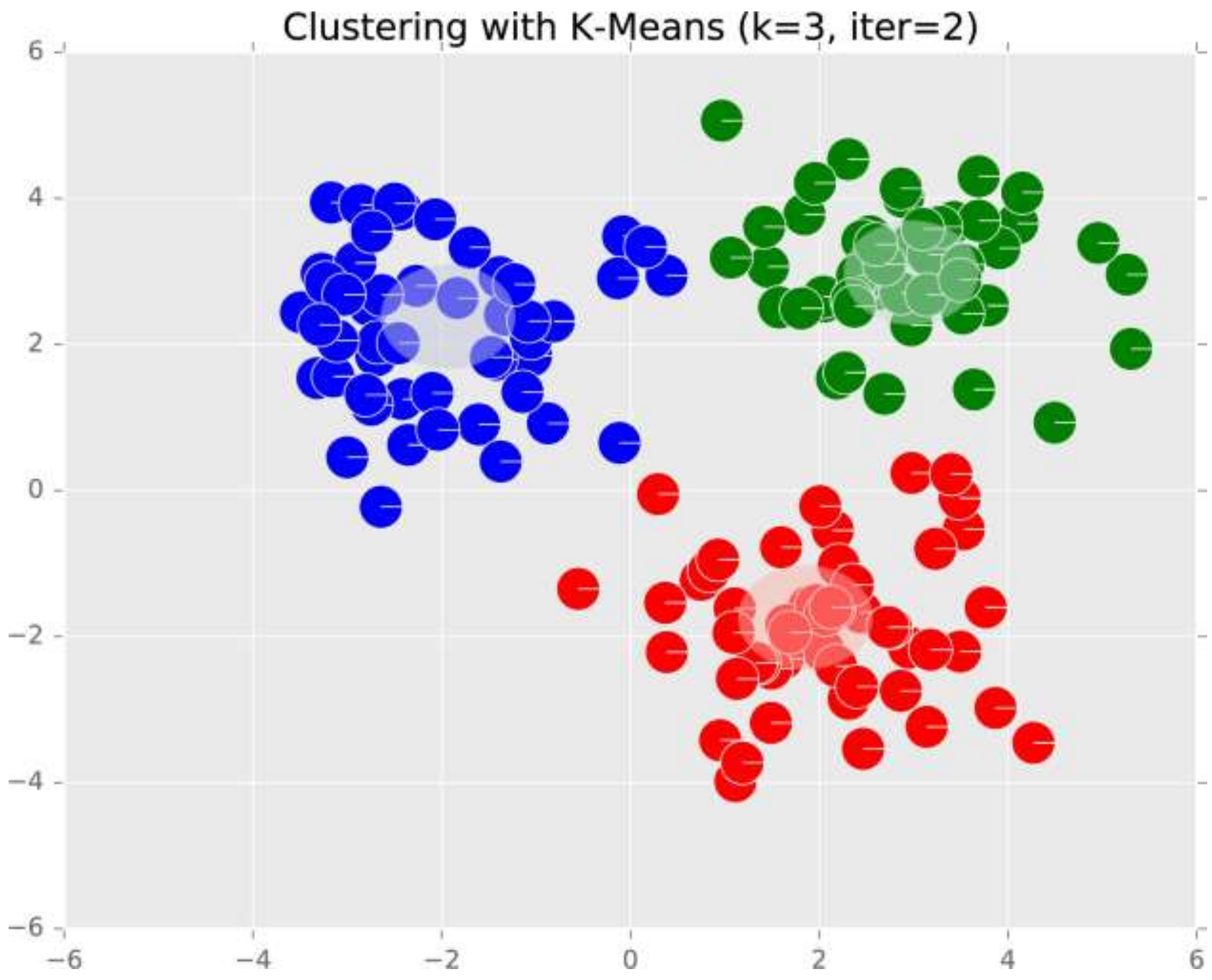
上海科技大学 ShanghaiTech University



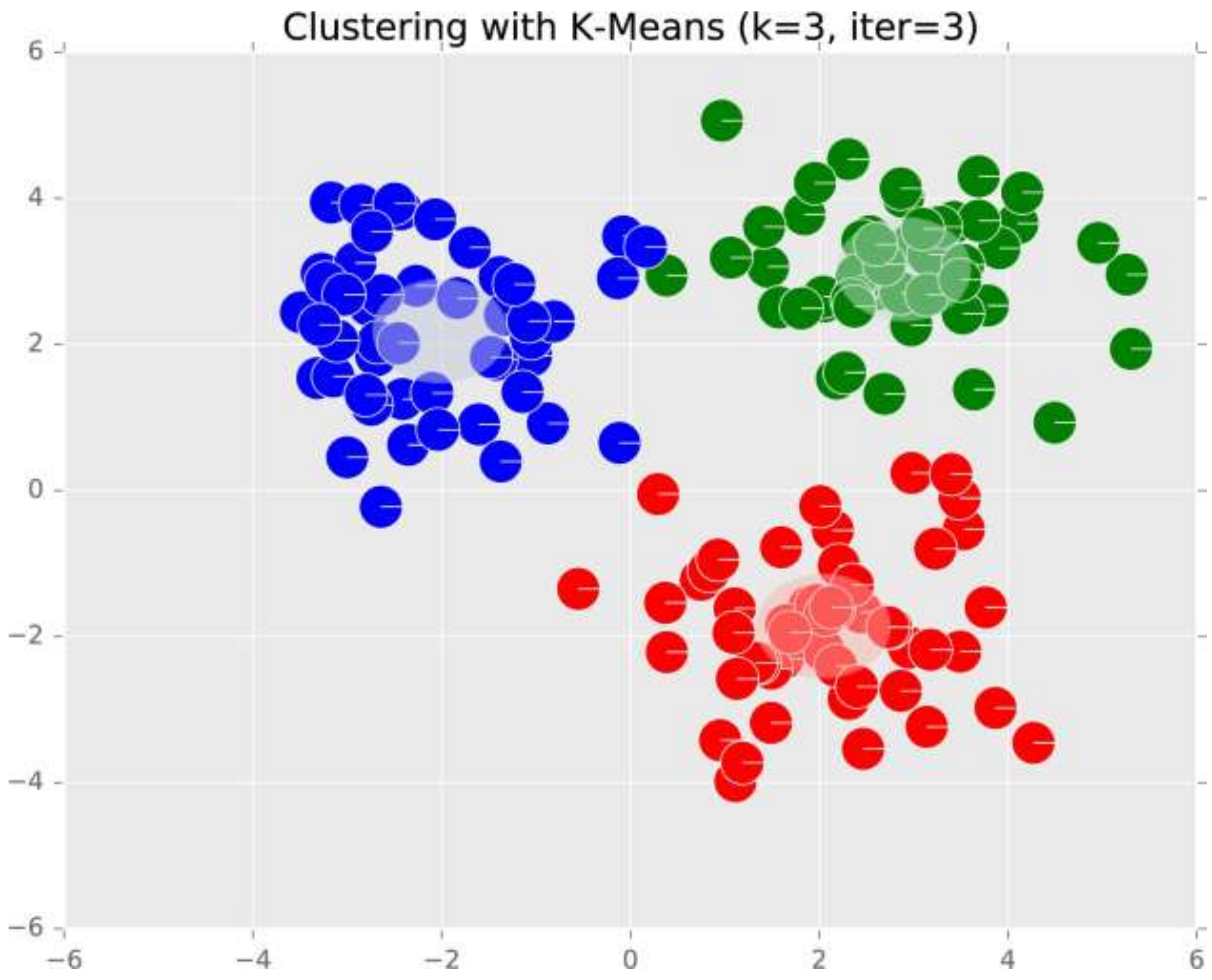
Example: K-Means



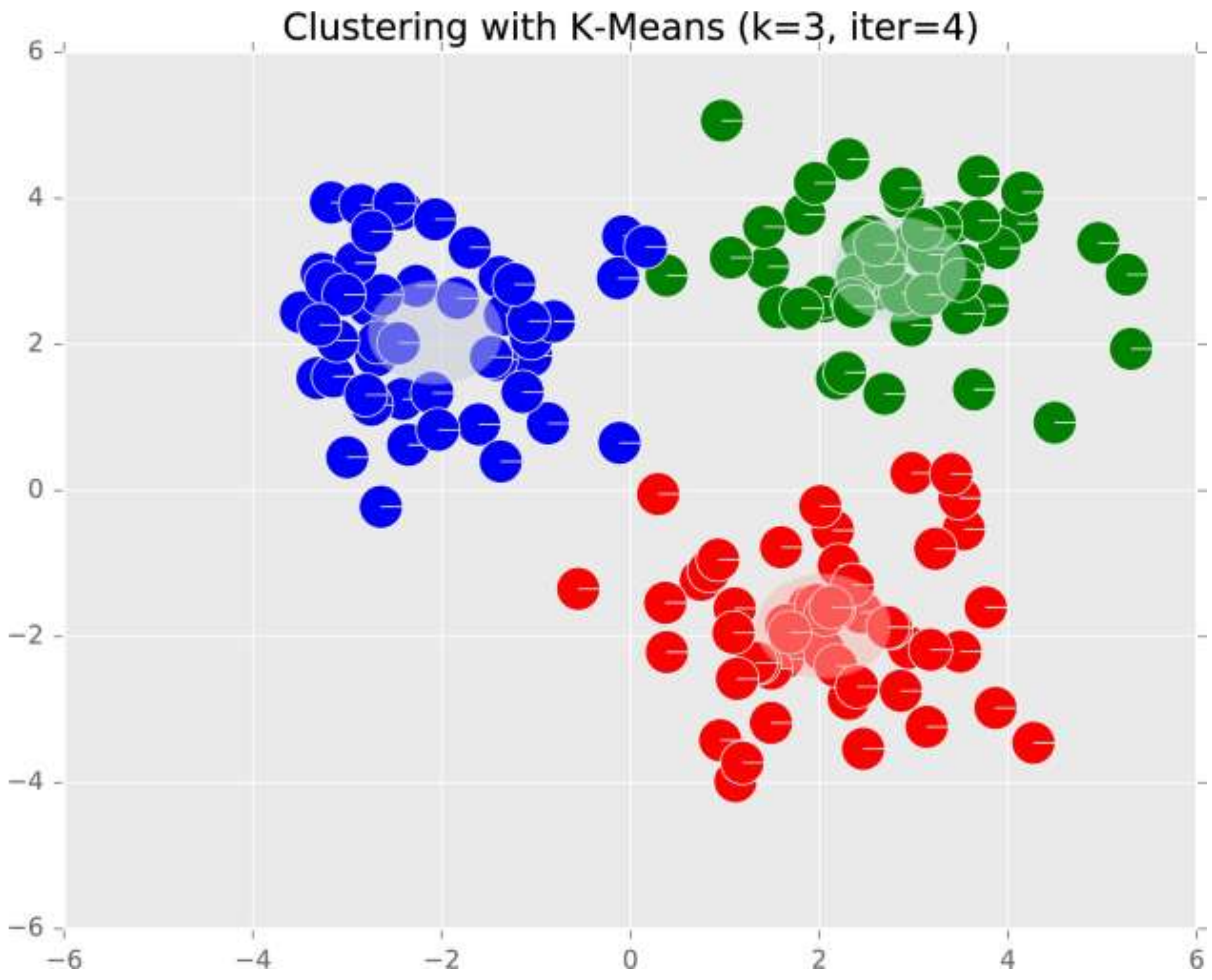
Example: K-Means



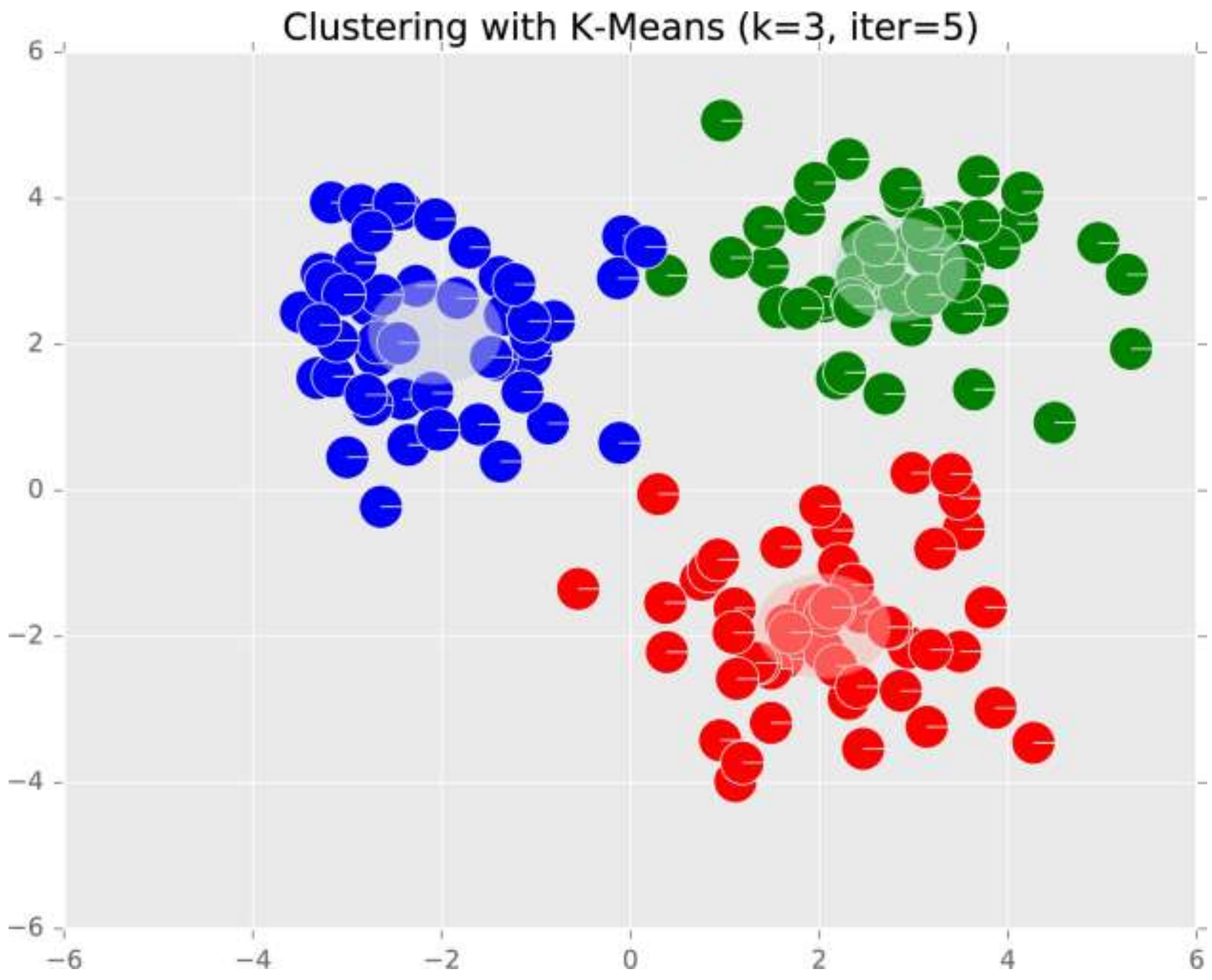
Example: K-Means



Example: K-Means



Example: K-Means

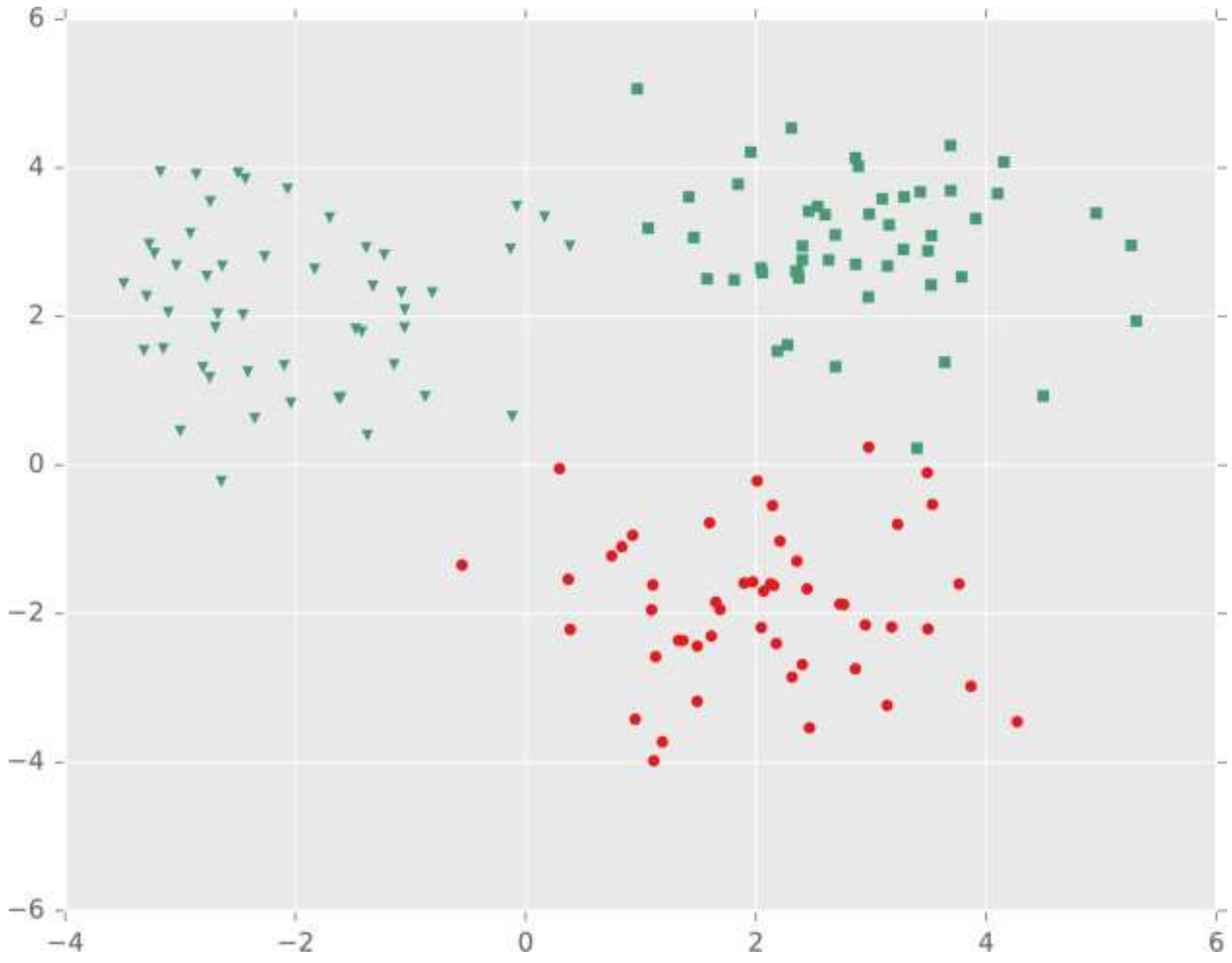


K=2 cluster centers

K-MEANS EXAMPLE

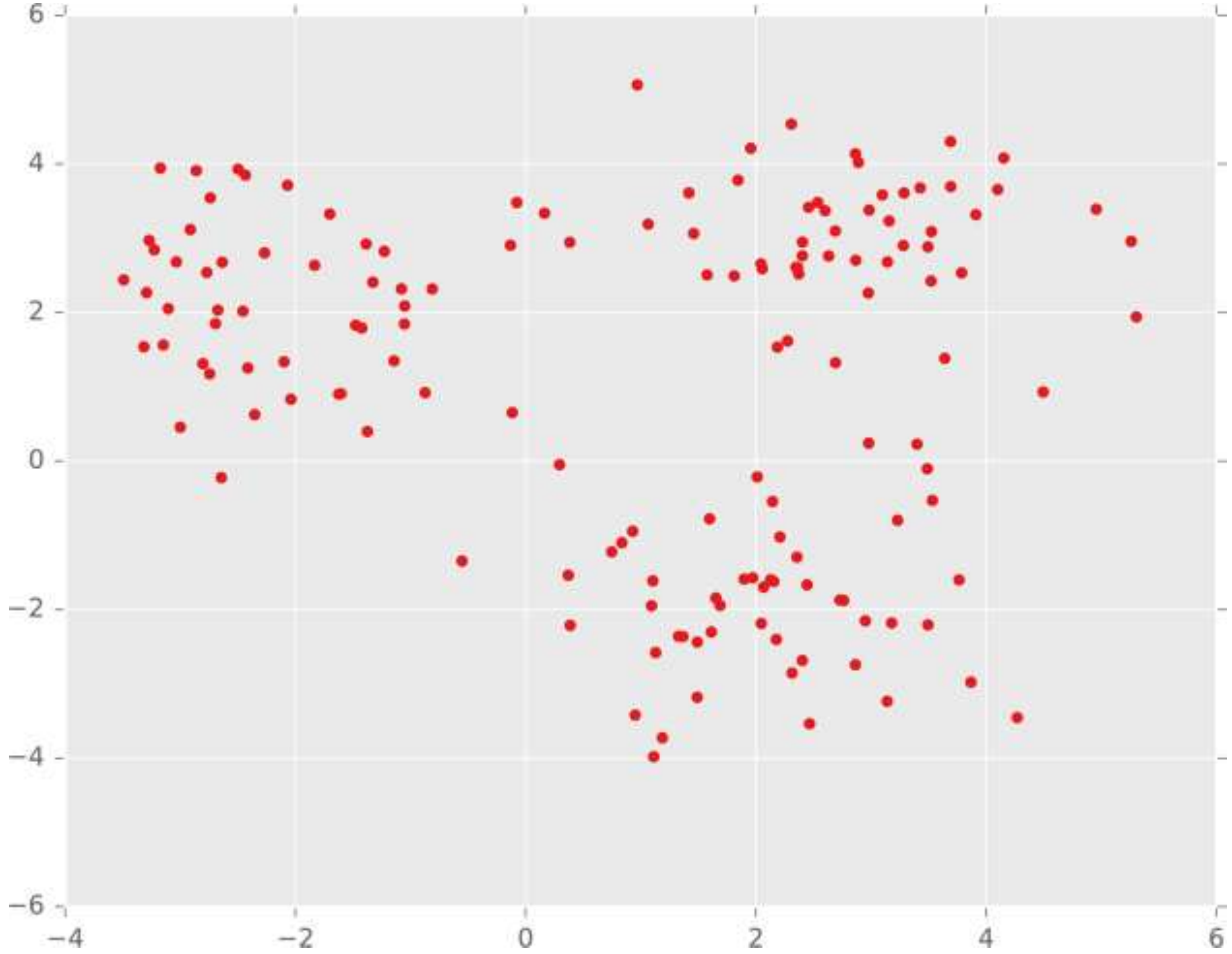
Example: K-Means

上海科技大学
ShanghaiTech University

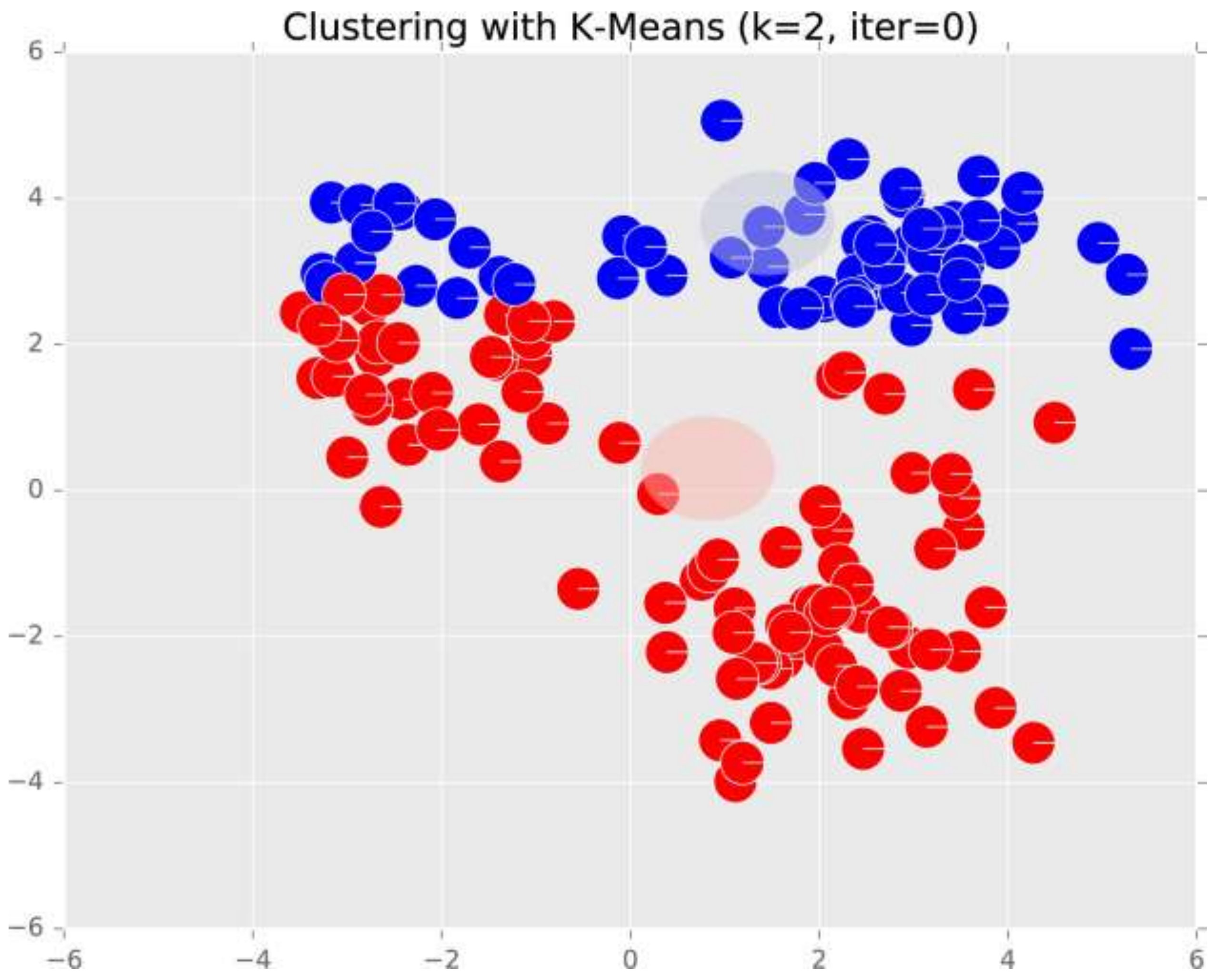


Example: K-Means

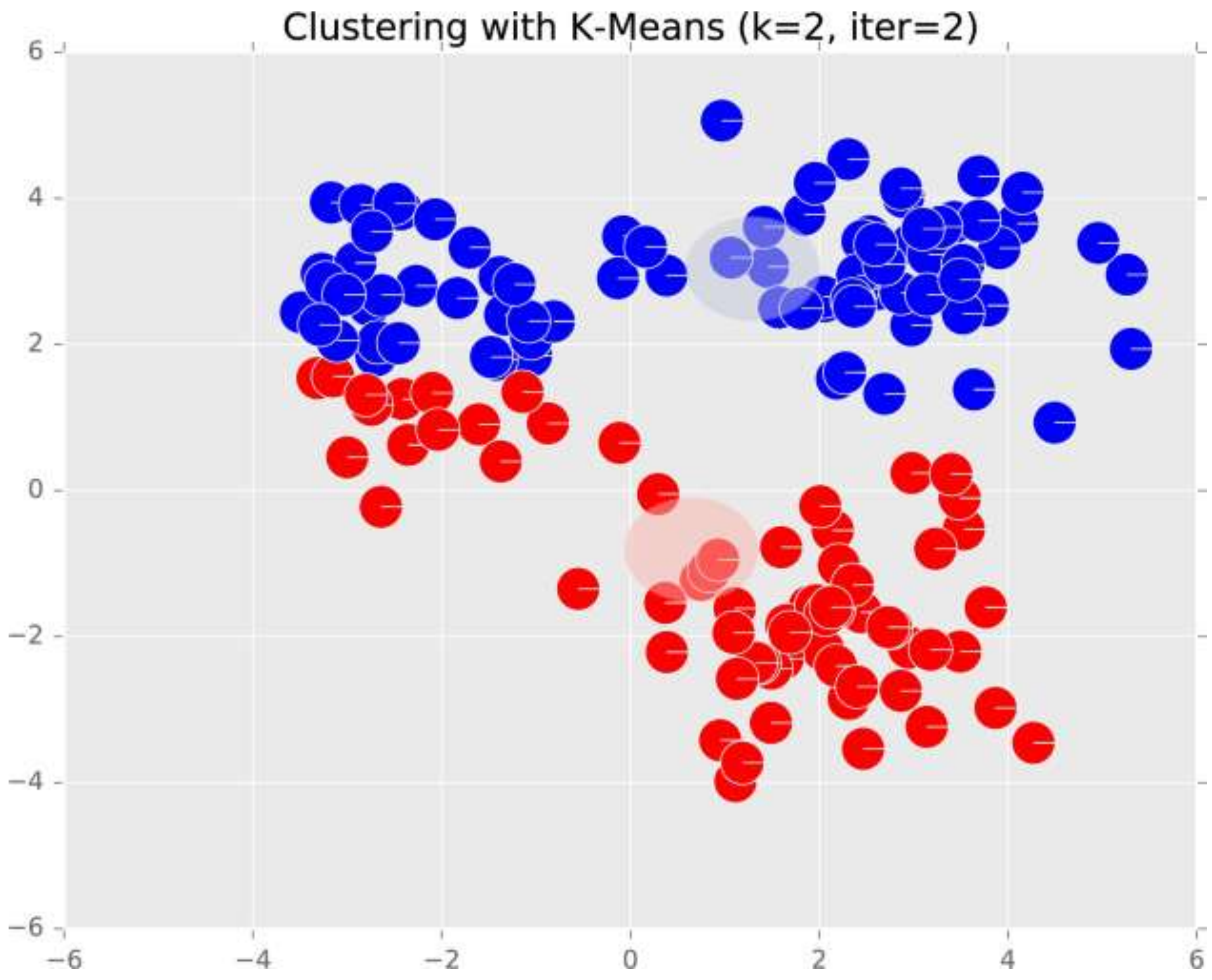
上海科技大学 ShanghaiTech University



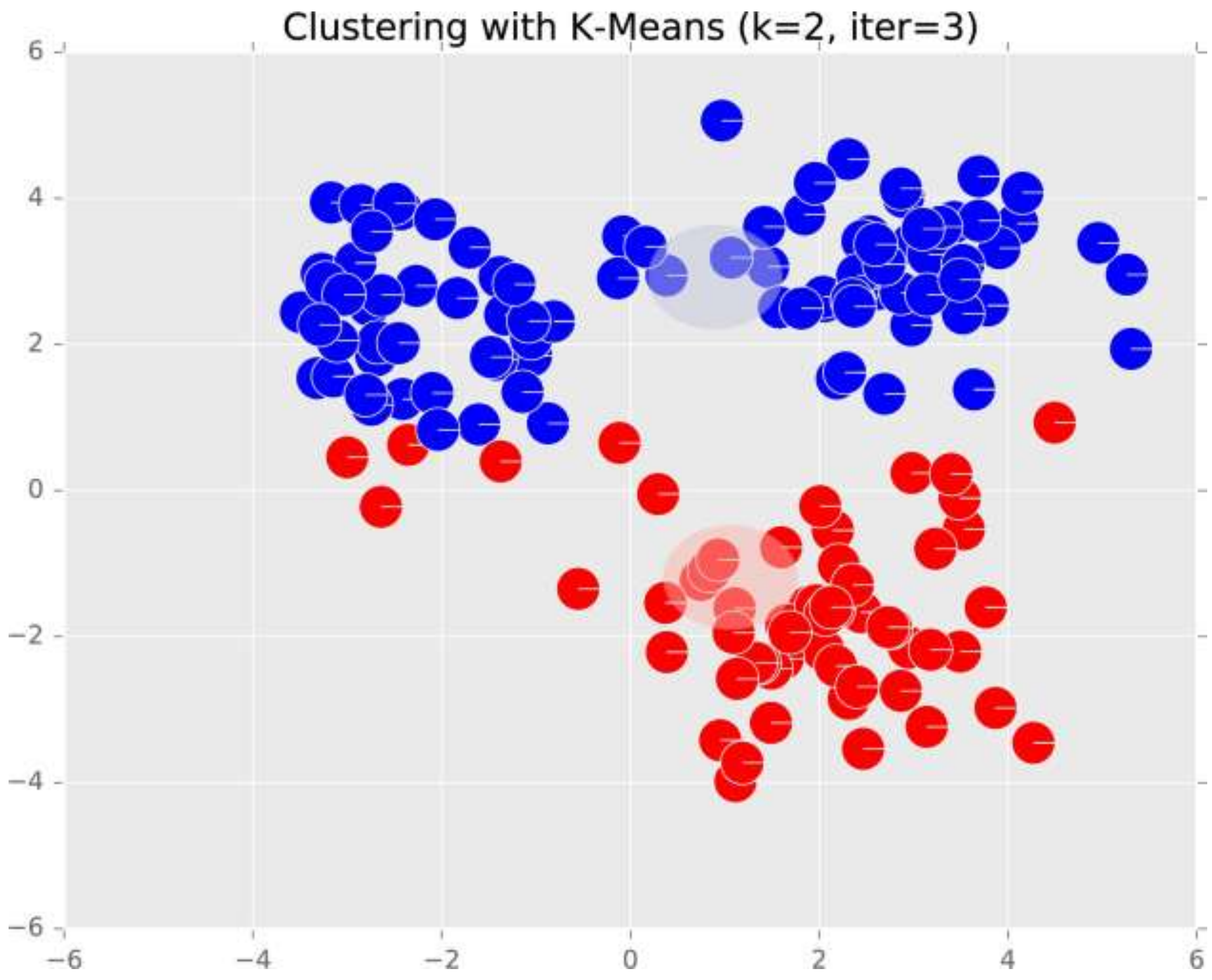
Example: K-Means



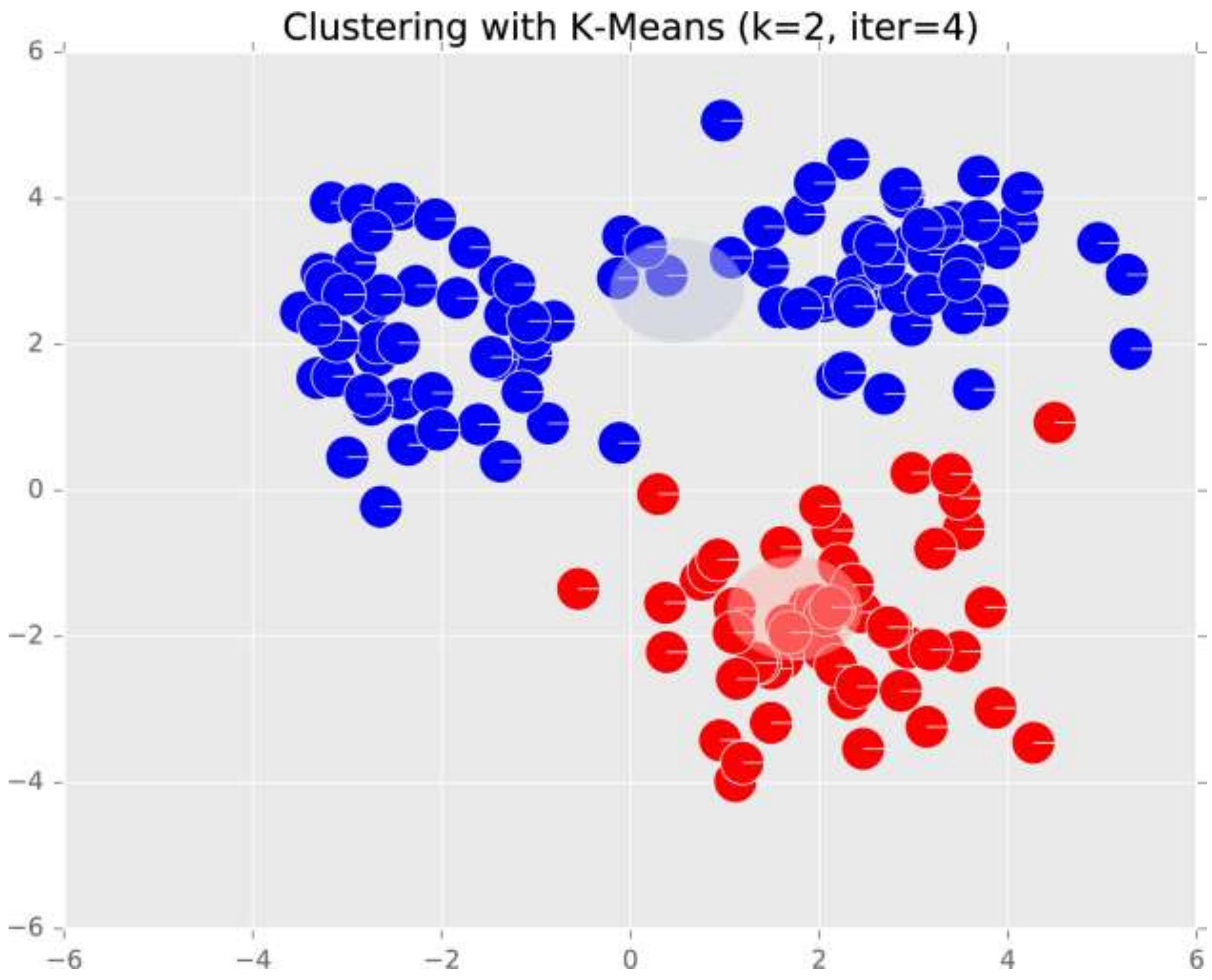
Example: K-Means



Example: K-Means

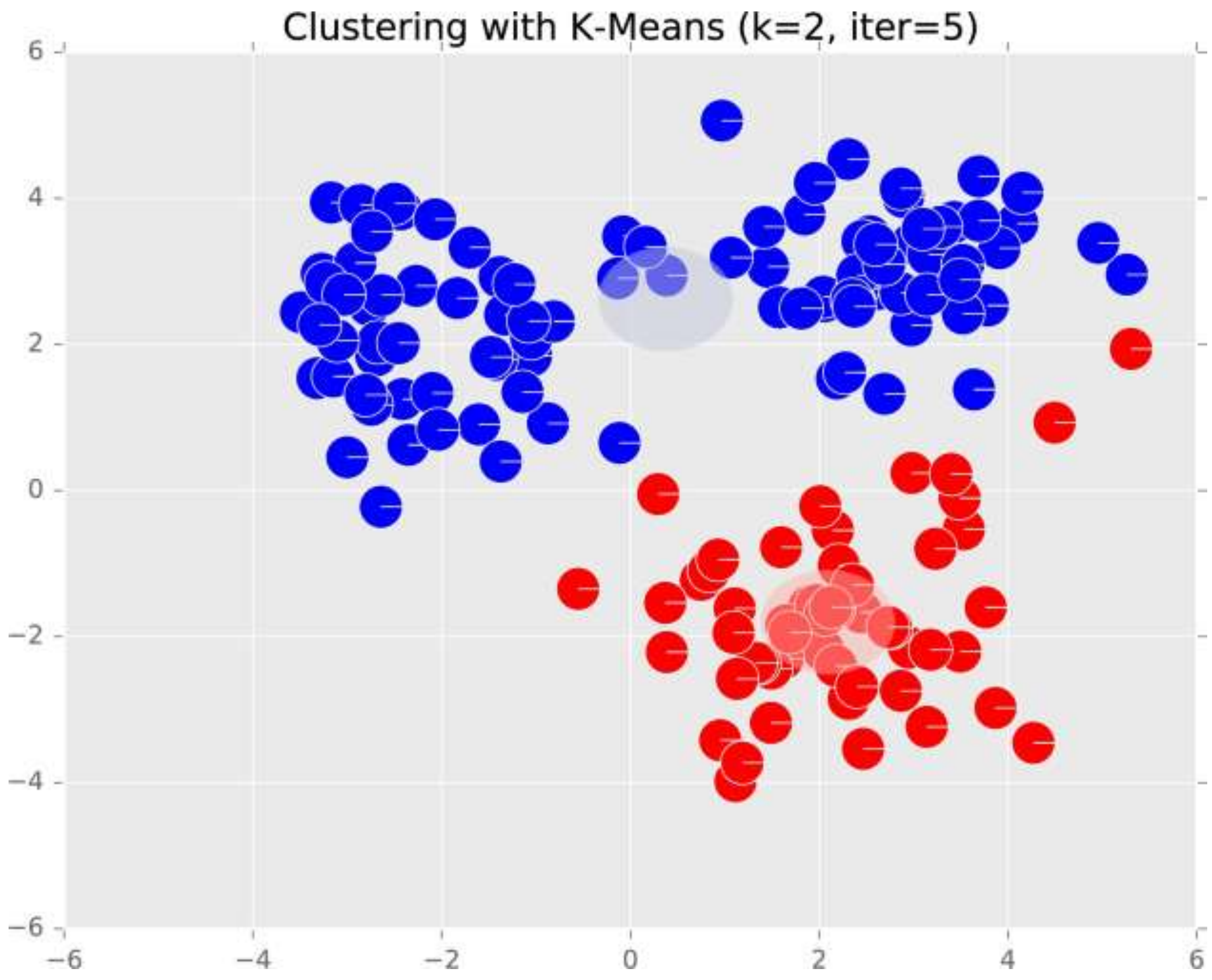


Example: K-Means

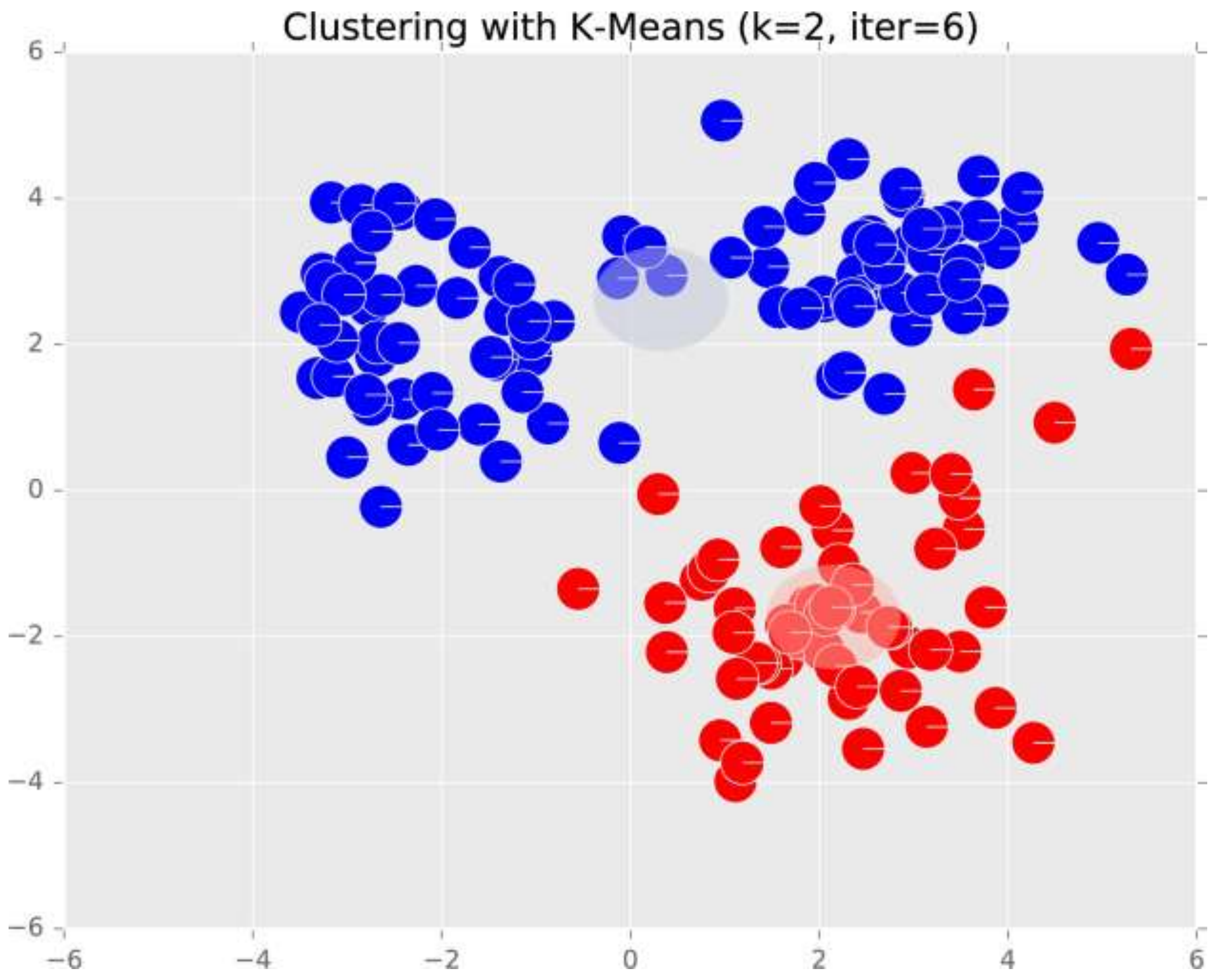


Example: K-Means

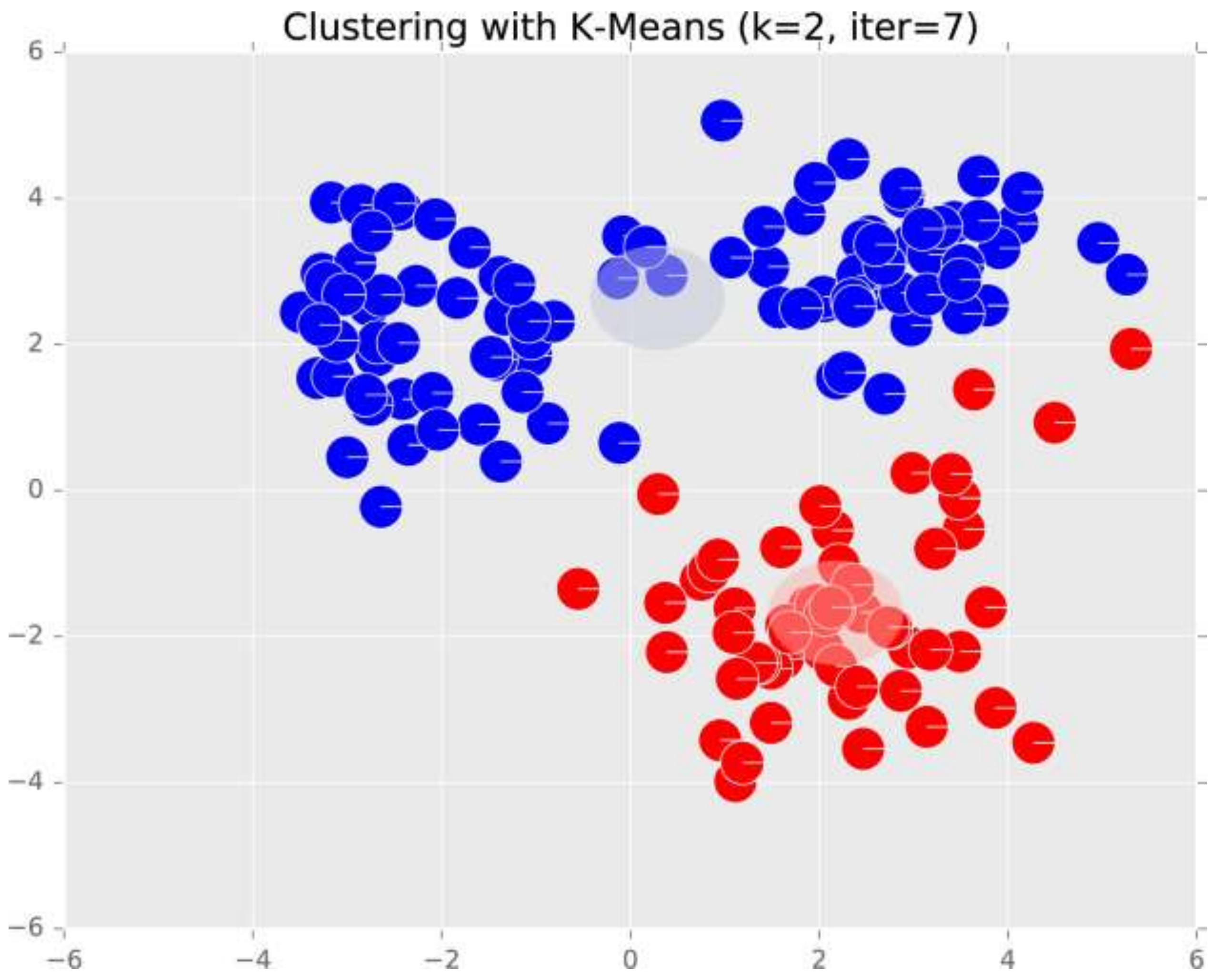
上海科技大学 ShanghaiTech University



Example: K-Means



Example: K-Means



INITIALIZING K-MEANS

Initialization of K-Means

K-Means Algorithm

1) **Given** unlabeled feature vectors
   $D = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\}$
2) **Initialize** cluster centers $c = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$
3) **Repeat** until nv
   a) for i in $\{1, \ldots, N\}$
      $z^{(i)} \leftarrow$ **inde**
   b) for j in $\{1, \ldots, K\}$
      $\mathbf{c}_j \leftarrow$ **mea**

Remaining Question:
How should we initialize the cluster centers?

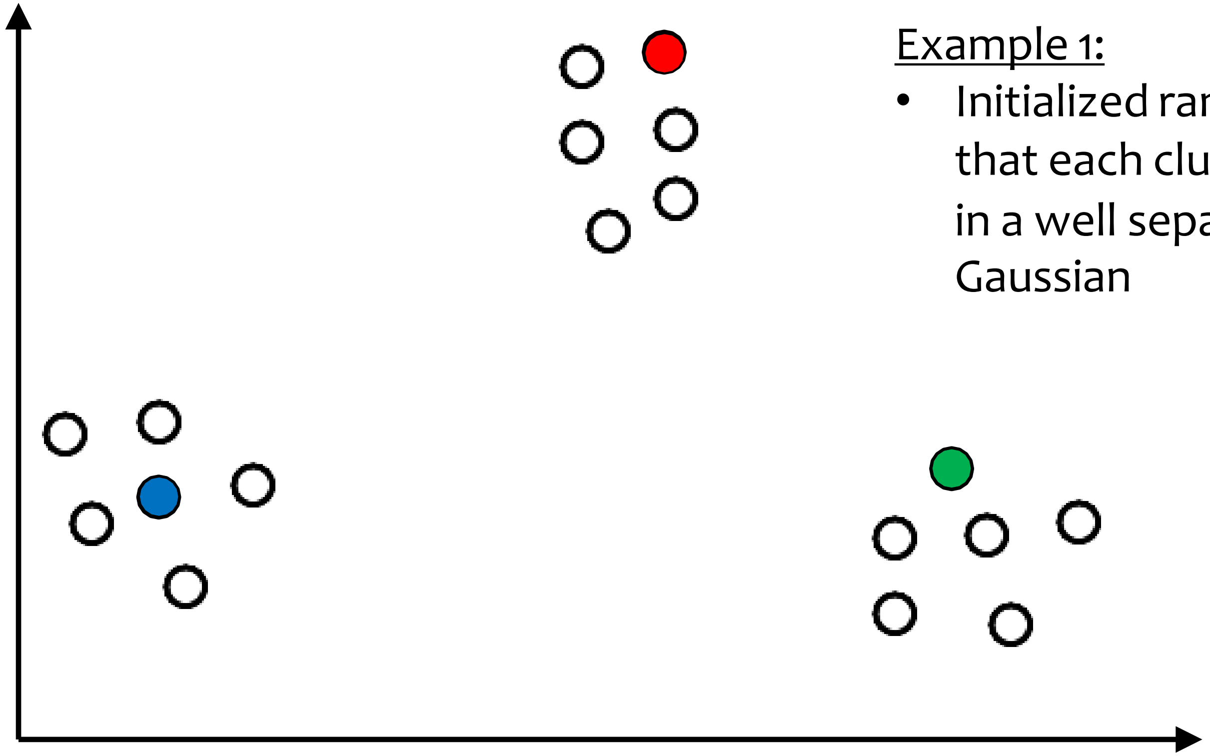
Three Solutions:
1. Random centers (picked from the data points)
2. Furthest point heuristic
3. K-Means++

Initialization for K-Means

上海科技大学 ShanghaiTech University

<u>Algorithm #1: Random Initialization</u>
Select each cluster center uniformly at random from the data points in the training data

<u>Observations:</u>
Even when data comes from well-separated Gaussians…
- …sometimes works great!
- …sometimes get stuck in poor local optima.



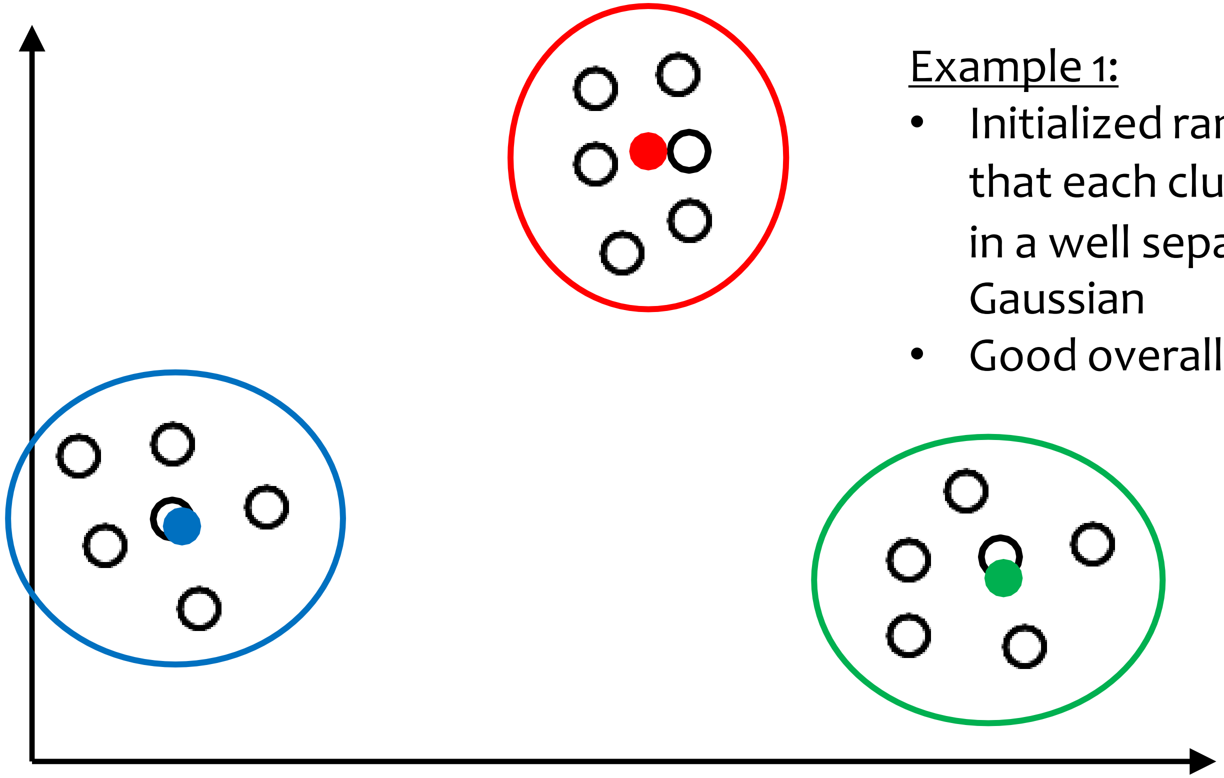
<u>Example 1:</u>
- Initialized randomly such that each cluster center is in a well separated Gaussian

Initialization for K-Means

上海科技大学 ShanghaiTech University

<u>Algorithm #1: Random Initialization</u>
Select each cluster center uniformly at random from the data points in the training data

<u>Observations:</u>
Even when data comes from well-separated Gaussians…
- … sometimes works great!
- … sometimes get stuck in poor local optima.



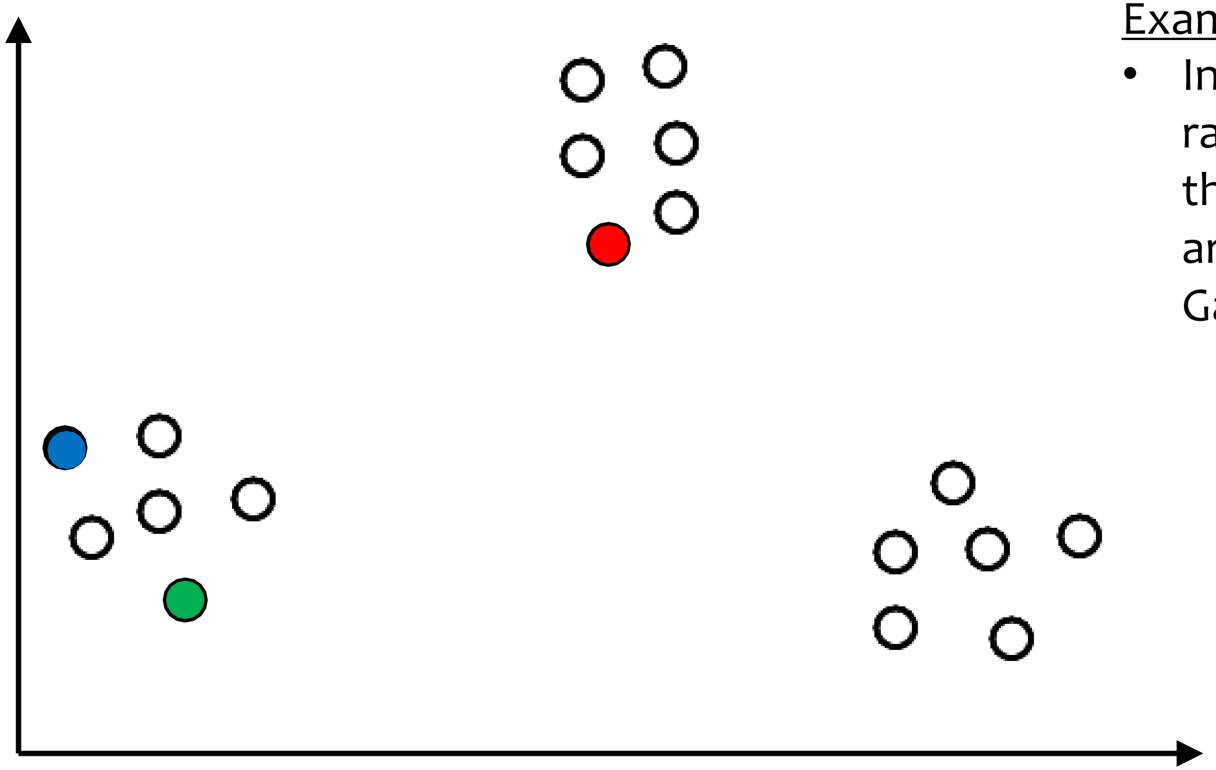
<u>Example 1:</u>
- Initialized randomly such that each cluster center is in a well separated Gaussian
- Good overall performance

Initialization for K-Means

上海科技大学 ShanghaiTech University

<u>Algorithm #1: Random Initialization</u>
Select each cluster center uniformly at random from the data points in the training data

<u>Observations:</u>
Even when data comes from well-separated Gaussians…
- …sometimes works great!
- …sometimes get stuck in poor local optima.



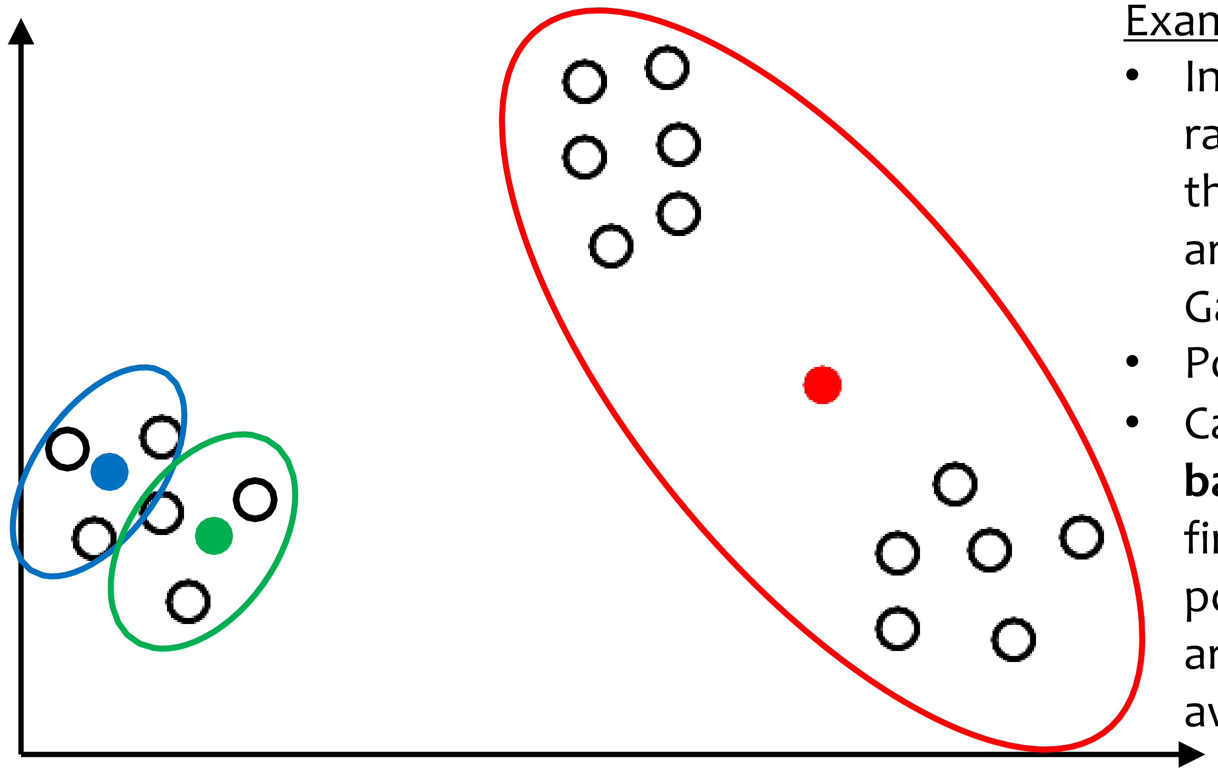
<u>Example 2:</u>
- Initialized randomly such that two centers are in the same Gaussian cluster

Initialization for K-Means

Algorithm #1: Random Initialization
Select each cluster center uniformly at random from the data points in the training data

Observations:
Even when data comes from well-separated Gaussians…

- … sometimes works great!
- … sometimes get stuck in poor local optima.



Example 2:

- Initialized randomly such that two centers are in the same Gaussian cluster
- Poor performance
- Can be **arbitrarily bad** (imagine the final red cluster points moving arbitrarily far away!)

Initialization for K-Means

上海科技大学 ShanghaiTech University

K-Mean Performance (with Random Initialization)

If we do **random initialization**, as k increases, it becomes more likely we won't have perfectly picked one center per Gaussian in our initialization (so K-Means will output a bad solution).

- For k equal-sized Gaussians,

$$\Pr[\text{each initial center is in a different Gaussian}] \approx \frac{k!}{k^k} \approx \frac{1}{e^k}$$

- Becomes unlikely as k gets large.

Slide courtesy of Nina Balcan

Initialization for K-Means

Algorithm #2: Furthest Point Heuristic

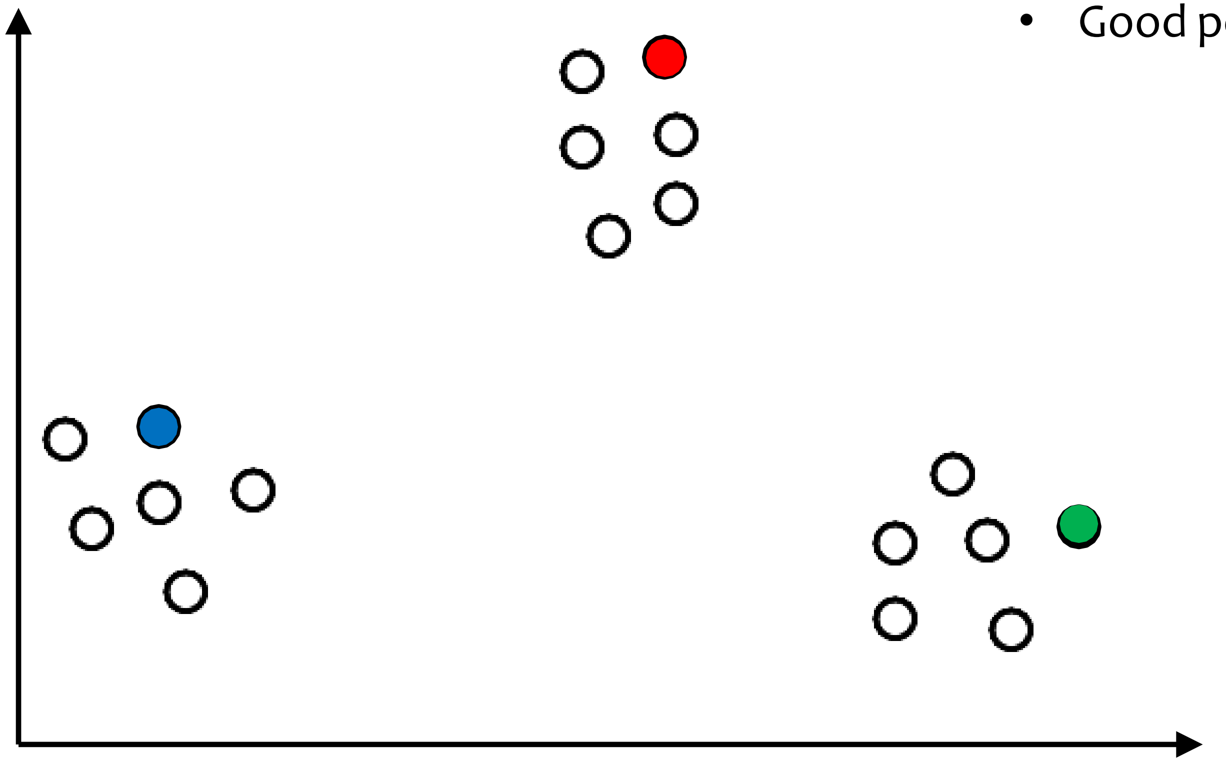
1. Pick the first cluster center c_1 **randomly**
2. Pick each subsequent center c_j so that it is **as far as possible** from the previously chosen centers $c_1, c_2, \ldots, c_{j-1}$

Observations:

- Solves the problem with Gaussian data
- But outliers pose a new problem!

Example 1:

- No outliers
- Good performance



Initialization for K-Means

<u>Algorithm #2: Furthest Point Heuristic</u>

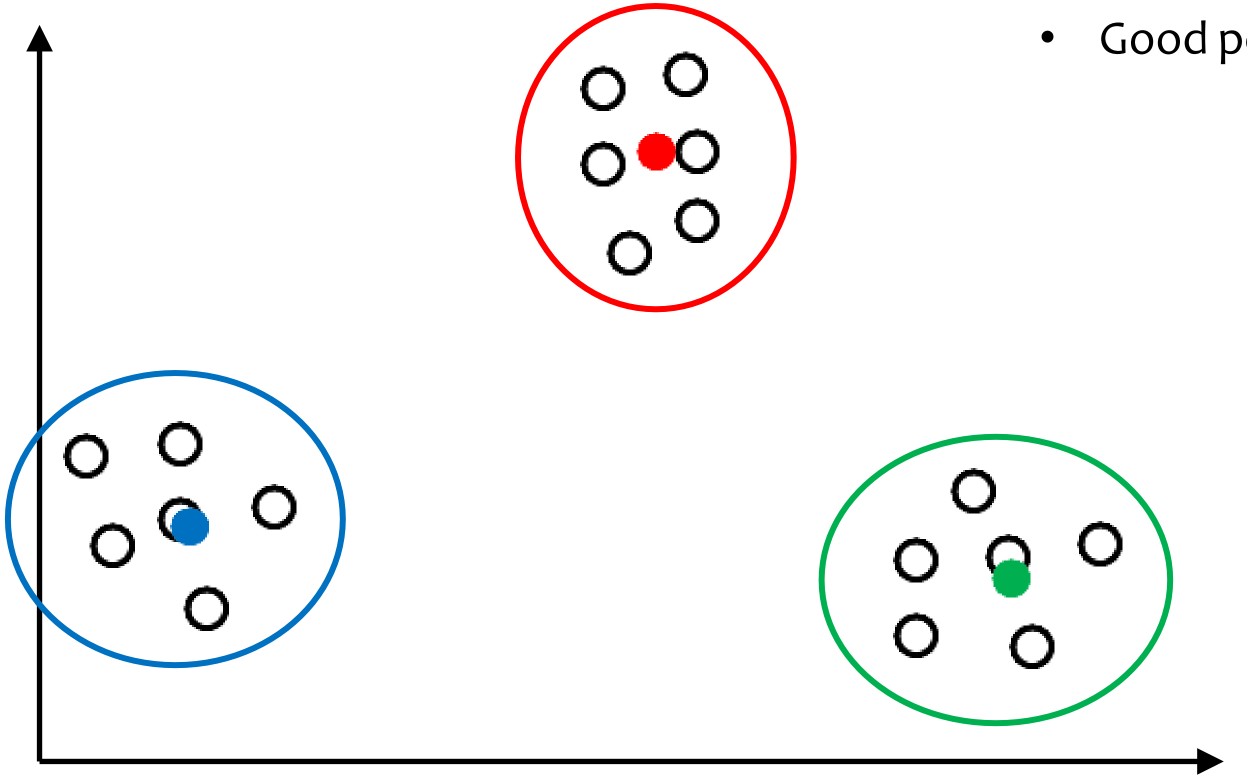
1. Pick the first cluster center $\mathbf{c}_1$ **randomly**
2. Pick each subsequent center $\mathbf{c}_j$ so that it is **as far as possible** from the previously chosen centers $\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{j-1}$

<u>Observations:</u>

- Solves the problem with Gaussian data
- But outliers pose a new problem!

<u>Example 1:</u>

- No outliers
- Good performance



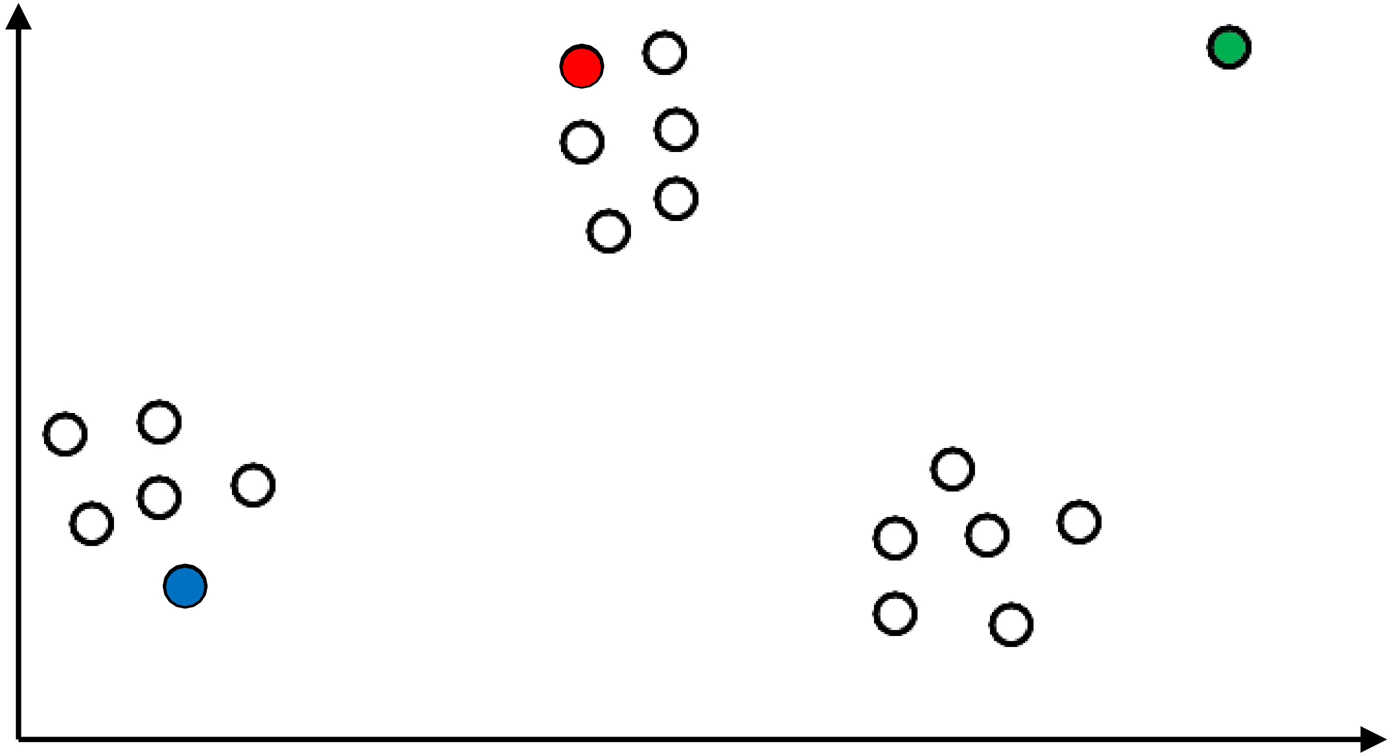
Initialization for K-Means

上海科技大学 ShanghaiTech University

Algorithm #2: Furthest Point Heuristic
1. Pick the first cluster center $\mathbf{c}_1$ **randomly**
2. Pick each subsequent center $\mathbf{c}_j$ so that it is **as far as possible** from the previously chosen centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{j-1}$

Observations:
- Solves the problem with Gaussian data
- But outliers pose a new problem!

Example 2:
- One outlier throws off the algorithm
- Poor performance



Initialization for K-Means

上海科技大学 ShanghaiTech University

<u>Algorithm #2: Furthest Point Heuristic</u>

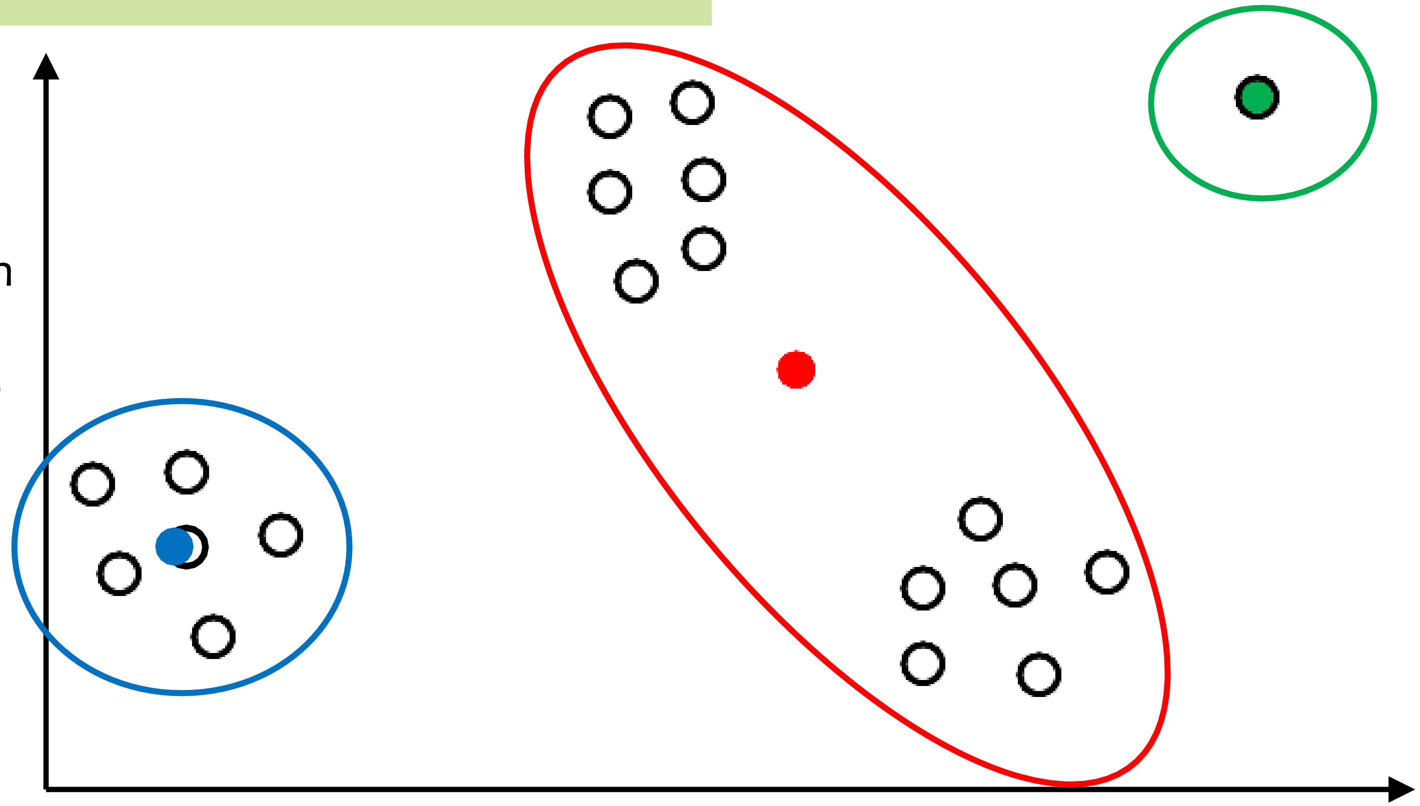
1. Pick the first cluster center $\mathbf{c}_1$ **randomly**
2. Pick each subsequent center $\mathbf{c}_j$ so that it is **as far as possible** from the previously chosen centers $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{j-1}$

<u>Observations:</u>

- Solves the problem with Gaussian data
- But outliers pose a new problem!

<u>Example 2:</u>

- One outlier throws off the algorithm
- Poor performance

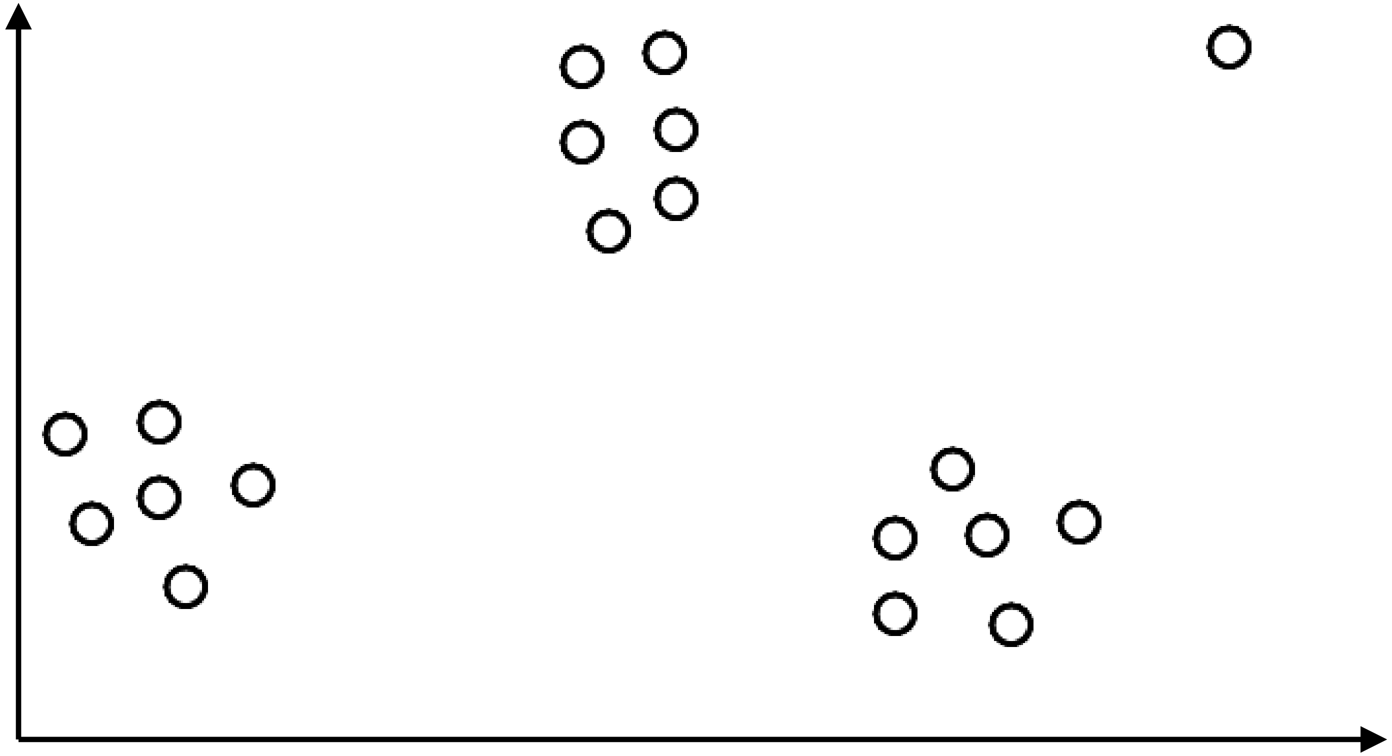


Initialization for K-Means

上海科技大学 ShanghaiTech University

Algorithm #3: K-Means++

- Let $\mathrm{D}(\mathbf{x})$ be the distance between a point x and its nearest center. Chose the next center proportional to $\mathrm{D}^2(\mathbf{x})$.



Initialization for K-M

i	D(x)	$D^2(x)$	$P(c_2 = x^{(i)})$
1	3	9	9/137
2	2	4	4/137
…			
7	4	16	16/137
…			
N	3	9	9/137
	Sum:	137	1.0

Algorithm #3: K-Means++

- Let $\mathrm{D}(\mathbf{x})$ be the distance between a point x and its nearest center. Chose the next center proportional to $\mathrm{D}^2(\mathbf{x})$.

- Choose $\mathbf{c_1}$ at random.
- For $\mathrm{j} = 2, \ldots, \mathrm{K}$
  - Pick $\mathbf{c_j}$ among $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}$ according to the distribution

$$\mathbf{P}(\mathbf{c_j} = \mathbf{x}^{(\mathbf{i})}) \propto \mathbf{min}_{\mathbf{j}'<\mathbf{j}} \left\|\mathbf{x}^{(\mathbf{i})} - \mathbf{c}_{\mathbf{j}'}\right\|^{\mathbf{2}} \quad \mathrm{D}^2(\mathbf{x}^{\mathbf{i}})$$

Theorem: K-Means++ always attains an $\mathrm{O}(\log \mathrm{k})$ approximation to optimal K-Means solution in expectation.

Slide adapted from Nina Balcan

Initialization for K-M

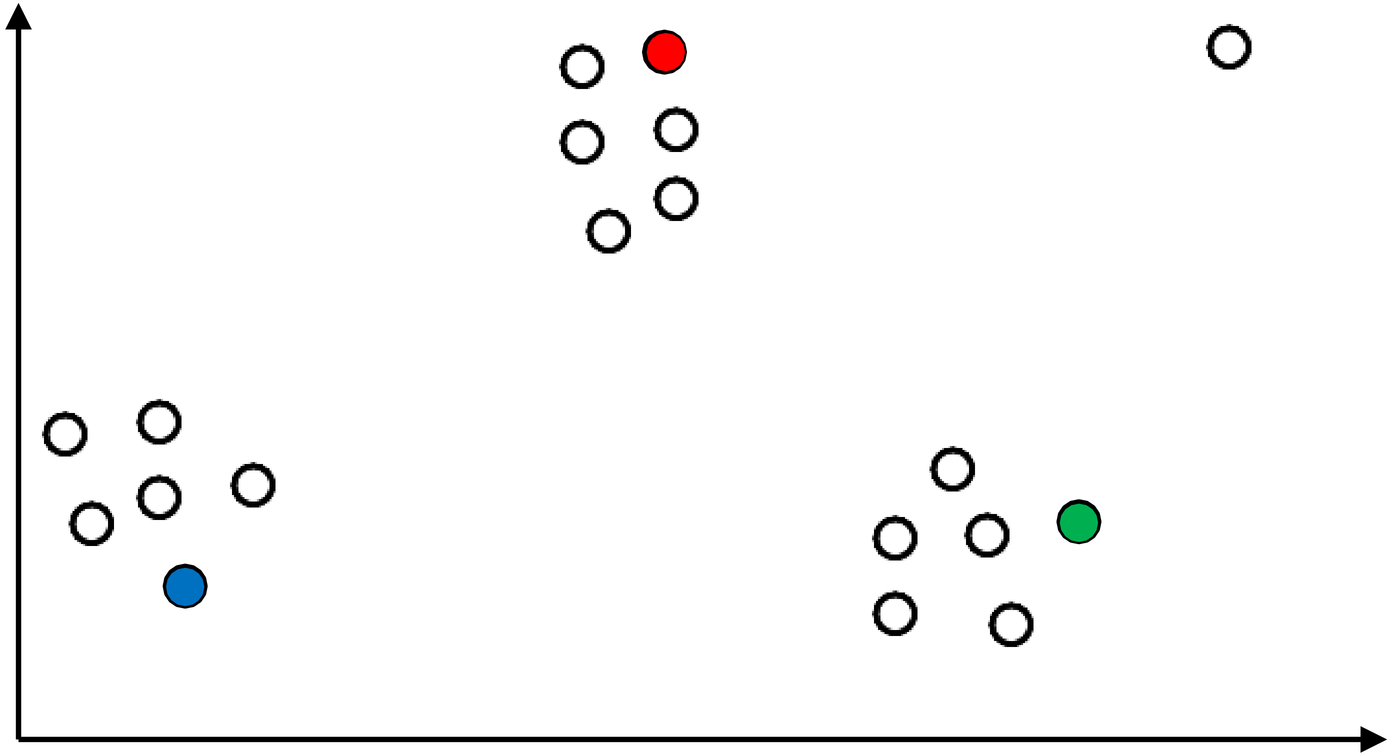
i	D(x)	D²(x)	$P(c_2 = x^{(i)})$
1	3	9	9/137
2	2	4	4/137
…			
7	4	16	16/137
…			
N	3	9	9/137
	Sum:	137	1.0

Algorithm #3: K-Means++

- Let $\mathrm{D}(\mathbf{x})$ be the distance between a point x and its nearest center. Chose the next center proportional to $\mathrm{D}^2(\mathbf{x})$.

Example 1:

- One outlier
- Good performance



Initialization for K-M

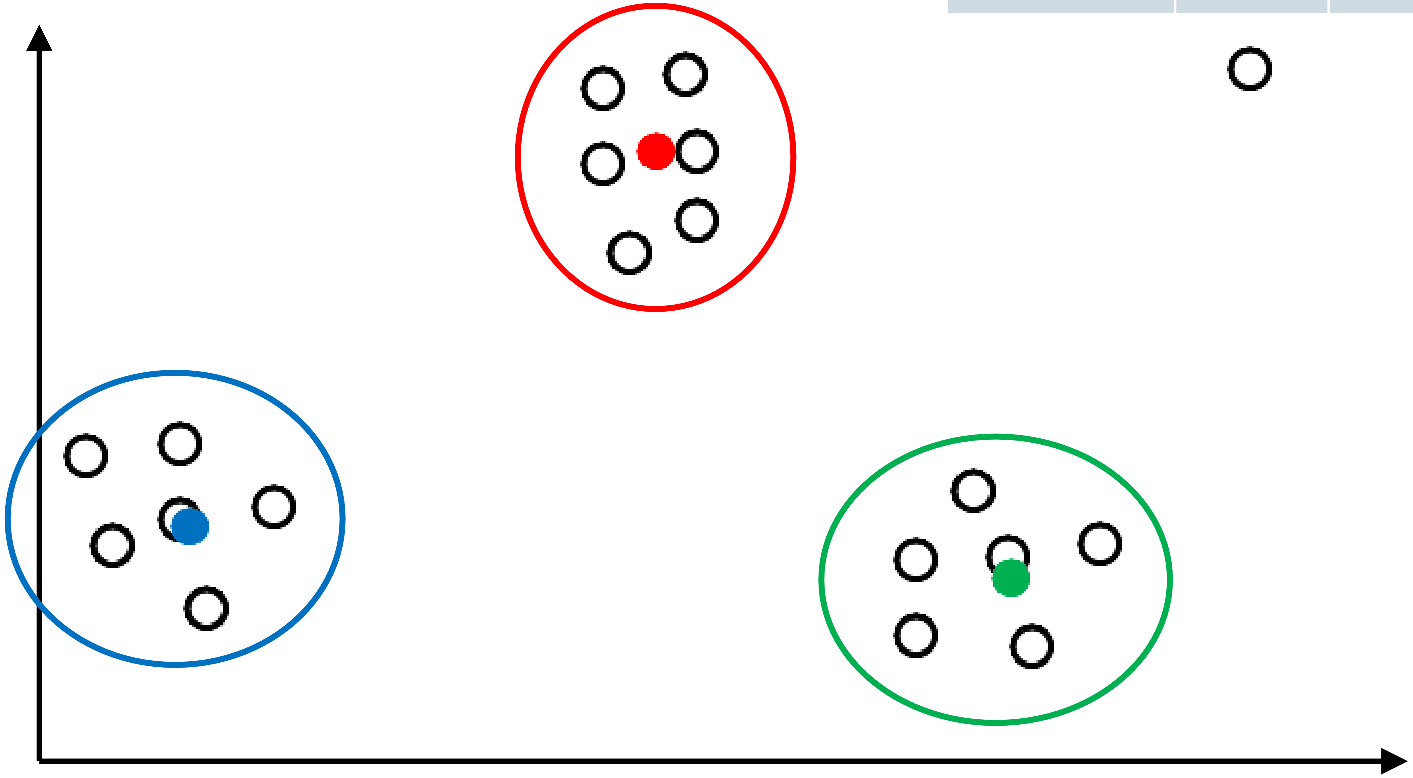
i	D(x)	$D^2(x)$	$P(c_2 = x^{(i)})$
1	3	9	9/137
2	2	4	4/137
…			
7	4	16	16/137
…			
N	3	9	9/137
	Sum:	137	1.0

Algorithm #3: K-Means++

- Let $\mathrm{D}(\mathbf{x})$ be the distance between a point x and its nearest center. Chose the next center proportional to $\mathrm{D}^2(\mathbf{x})$.

Example 1:

- One outlier
- Good performance



Initialization for K-Means

Algorithm #3: K-Means++

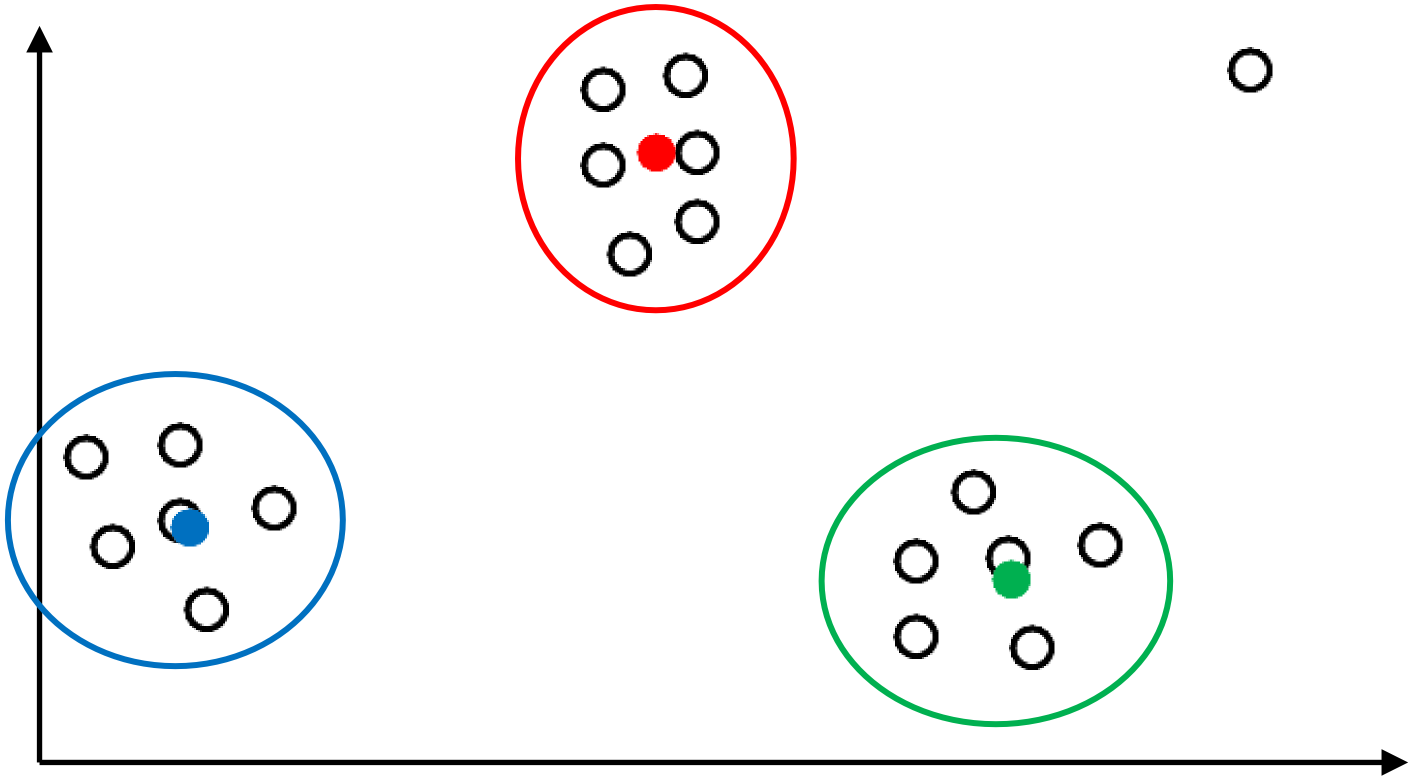
- Let $\mathrm{D}(\mathbf{x})$ be the distance between a point x and its nearest center. Chose the next center proportional to $\mathrm{D}^2(\mathbf{x})$.

Observations:

- Interpolates between random and farthest point initialization
- Solves the problem with Gaussian data
- **And** solves the outlier problem

Example 1:

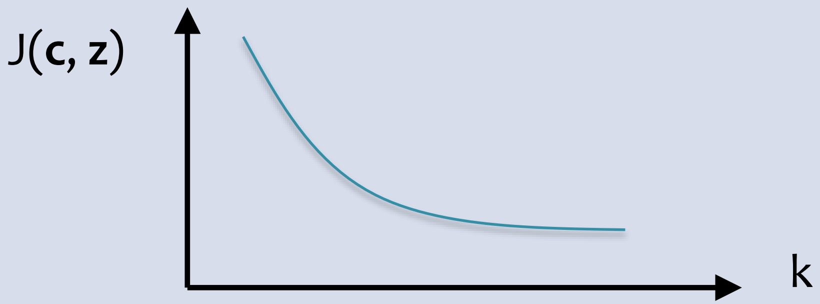
- One outlier
- Good performance



Q&A

Q: In k-Means, since we don't have a validation set, how do we pick k?

A: Look at the training objective function as a function of k and pick the value at the "elbo" of the curve.



Q: What if our random initialization for k-Means gives us poor performance?

A: Do **random restarts**: that is, run k-means from scratch, say, 10 times and pick the run that gives the lowest training objective function value.

The objective function is **nonconvex**, so we're just looking for the best local minimum.

Learning Objectives

K-Means

You should be able to…

1. Distinguish between coordinate descent and block coordinate descent
2. Define an objective function that gives rise to a "good" clustering
3. Apply block coordinate descent to an objective function preferring each point to be close to its nearest objective function to obtain the K-Means algorithm
4. Implement the K-Means algorithm
5. Connect the non-convexity of the K-Means objective function with the (possibly) poor performance of random initialization