Unsupervised Learning: K-Means & Agglomerative Clustering

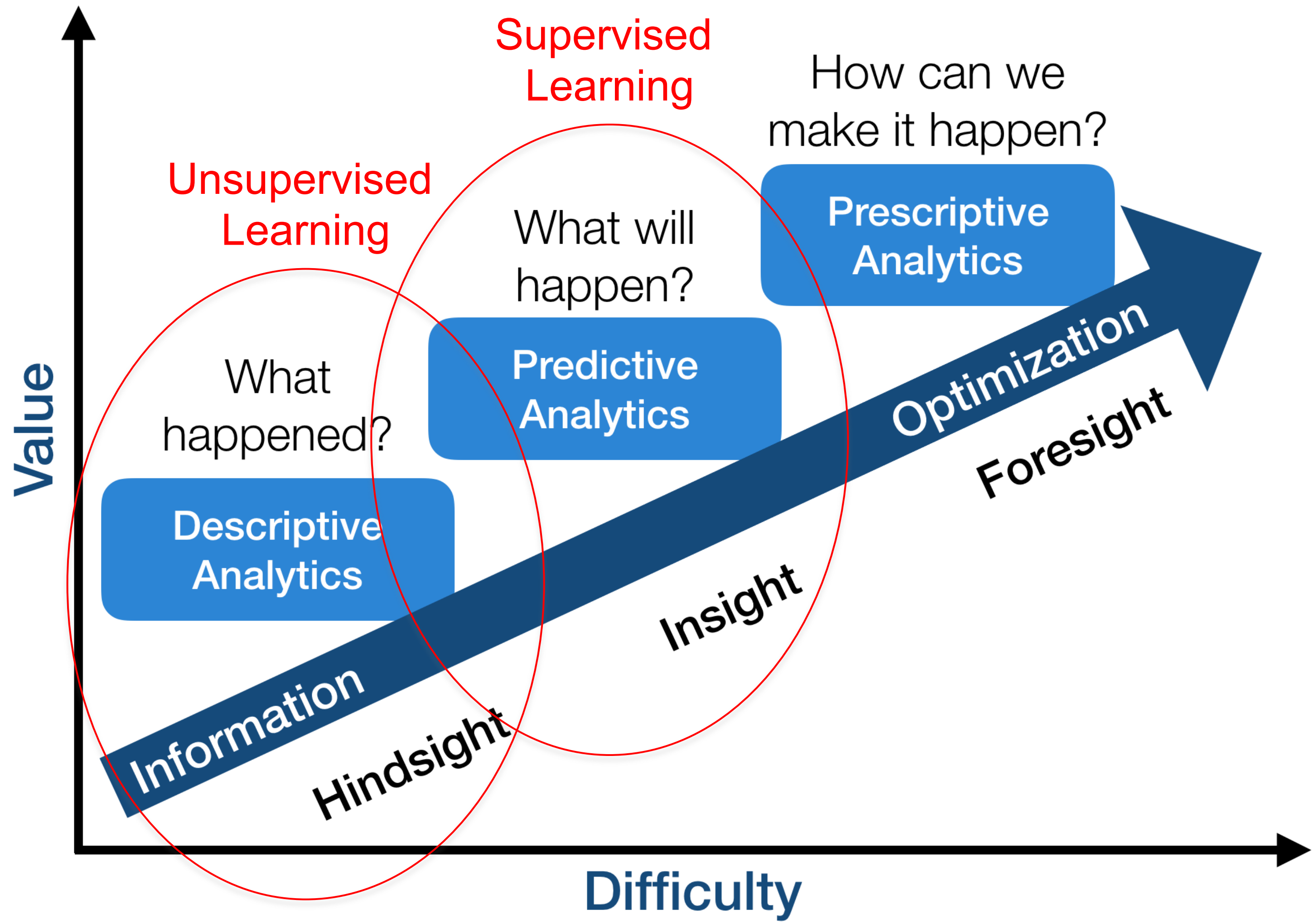
These slides were assembled by Byron Boots, with only minor modifications from Eric Eaton's slides and grateful acknowledgement to the many others who made their course materials freely available online. Feel free to reuse or adapt these slides for your own academic purposes, provided that you include proper attribution.

Types of Learning

	from input ***x***, output:
unsupervised	summary ***z***
supervised	prediction ***y***
reinforcement	action ***a*** to maximize reward ***r***

From Wiggins, Recht

Types of Learning



From Gartner, Recht

Unsupervised Learning

- Supervised learning used labeled data pairs (**x**, y) to learn a function f : X→Y
  - But, what if we don't have labels?

- No labels = **unsupervised learning**

Clustering

Clustering: group together similar points and represent them with a single token

Key Design Choices:

1) What makes two data points similar?

2) How do we compute an overall grouping from pairwise similarities?

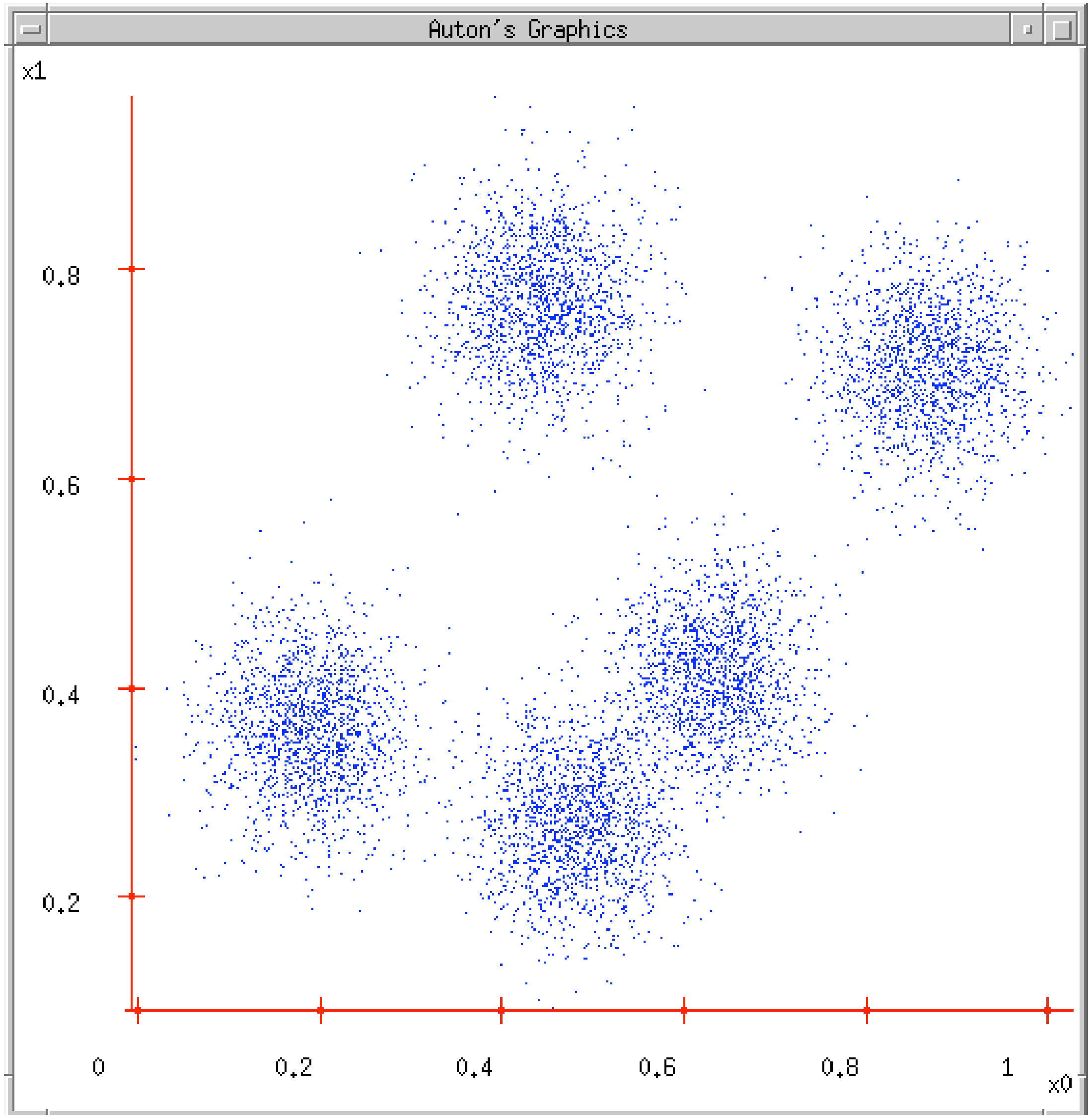
Slide: Derek Hoiem

How might we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters

K-Means Clustering

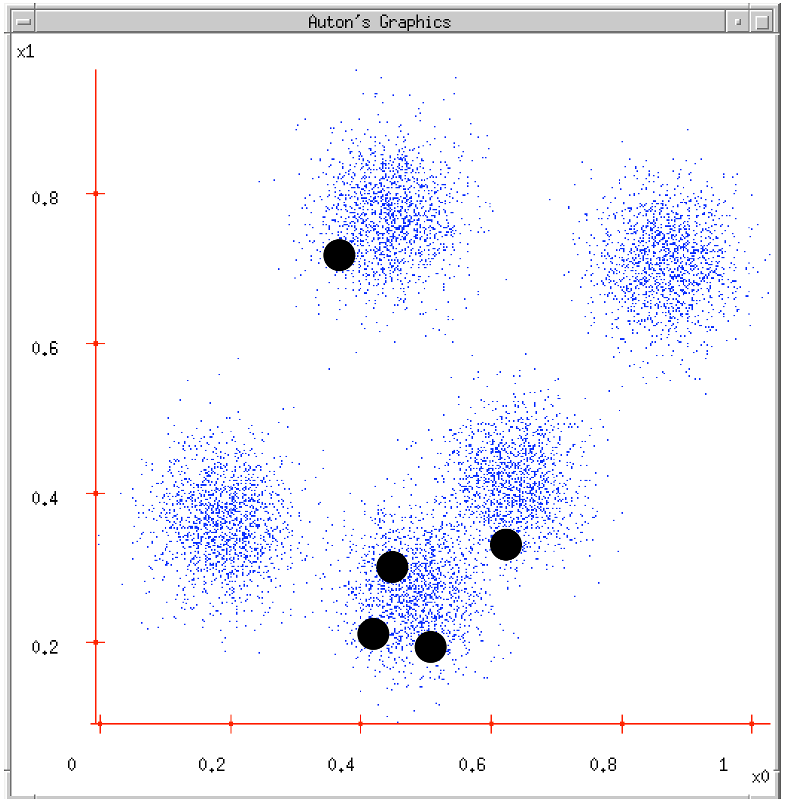
Clustering Data



K-Means Clustering

K-Means (*k* , X)

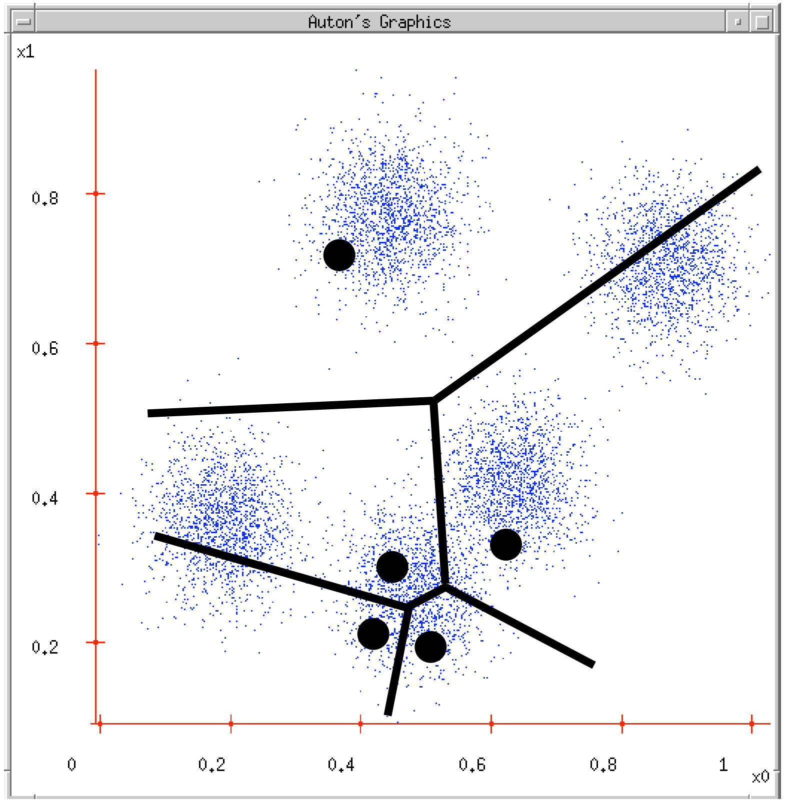
- Randomly choose *k* cluster center locations (centroids)
- Loop until convergence
  - Assign each point to the cluster of the closest centroid
  - Re-estimate the cluster centroids based on the data assigned to each cluster



K-Means Clustering

K-Means (*k* , X)

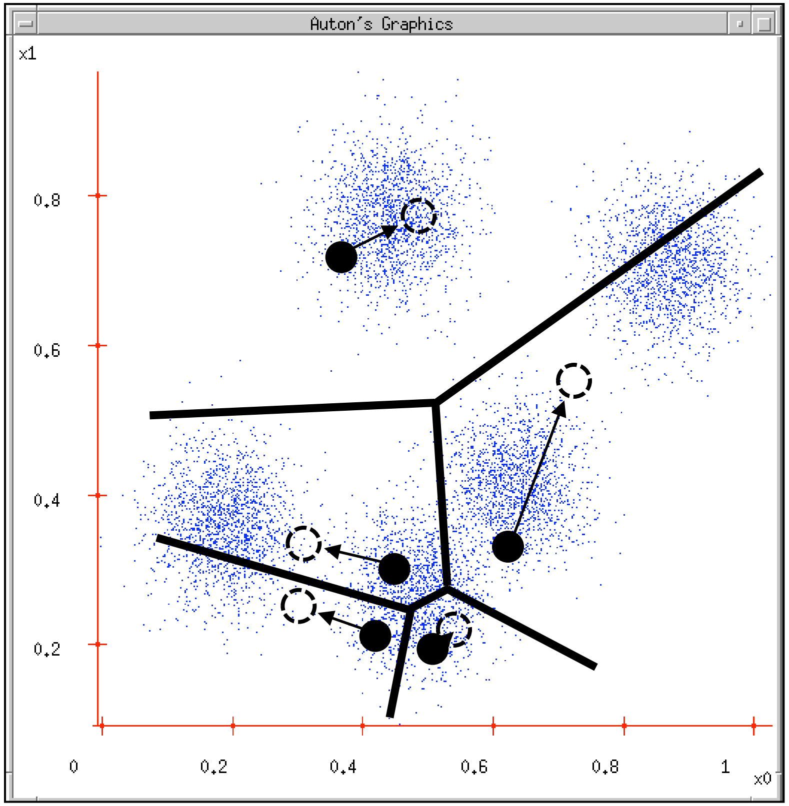
- Randomly choose *k* cluster center locations (centroids)
- Loop until convergence
  - Assign each point to the cluster of the closest centroid
  - Re-estimate the cluster centroids based on the data assigned to each cluster



K-Means Clustering

K-Means (*k* , X)

- Randomly choose *k* cluster center locations (centroids)
- Loop until convergence
  - Assign each point to the cluster of the closest centroid
  - Re-estimate the cluster centroids based on the data assigned to each cluster



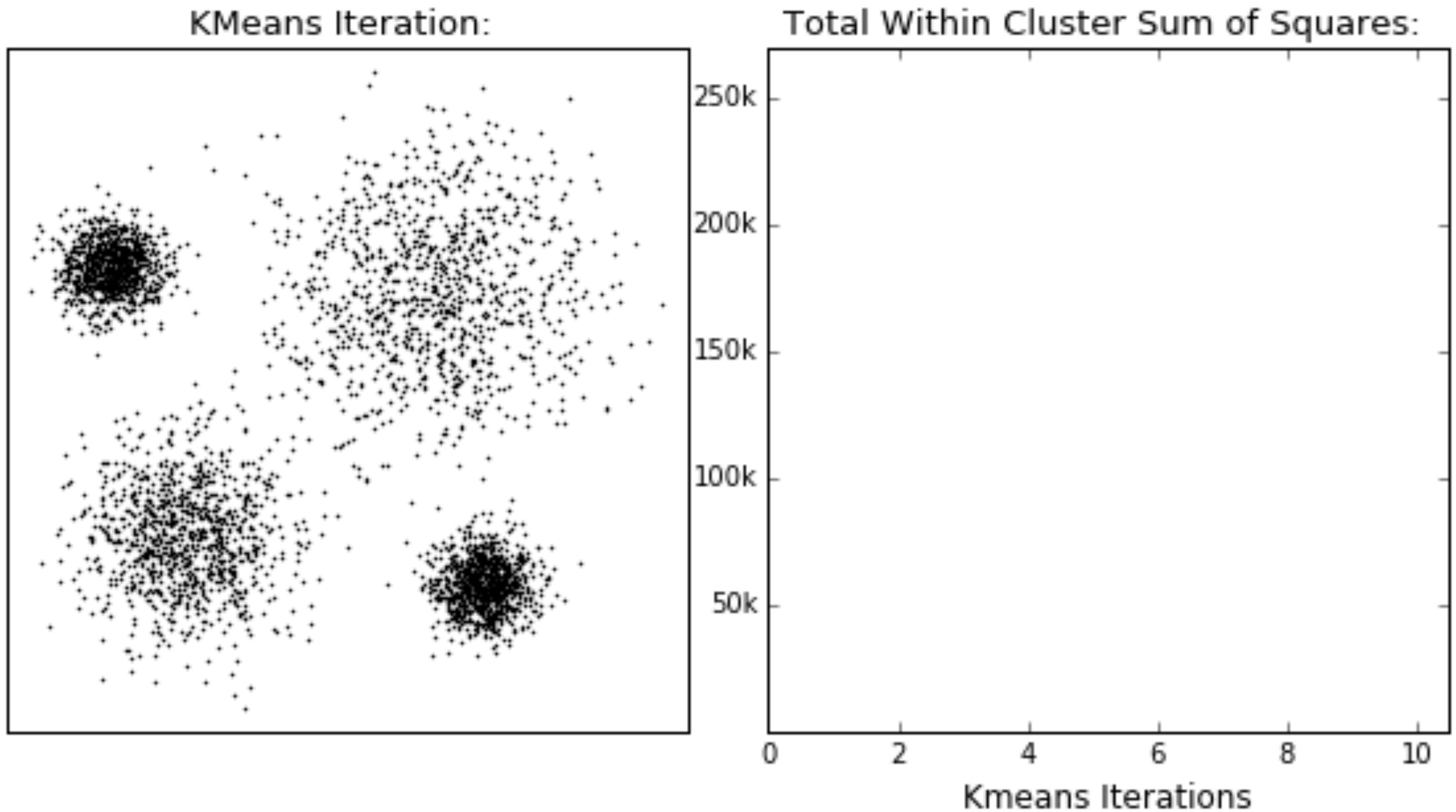
K-Means Objective Function

- K-means finds a local optimum of the following objective function:

$$\arg\min_{\boldsymbol{\mathcal{S}}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in \mathcal{S}_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2$$

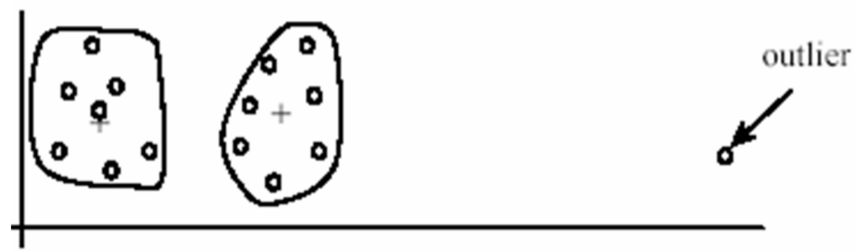
where $\boldsymbol{\mathcal{S}} = \{\mathcal{S}_1, \ldots, \mathcal{S}_k\}$ is a partitioning over $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ s.t. $X = \bigcup_{i=1}^{k} \mathcal{S}_i$ and $\boldsymbol{\mu}_i = \mathrm{mean}(\mathcal{S}_i)$

K-means Demo

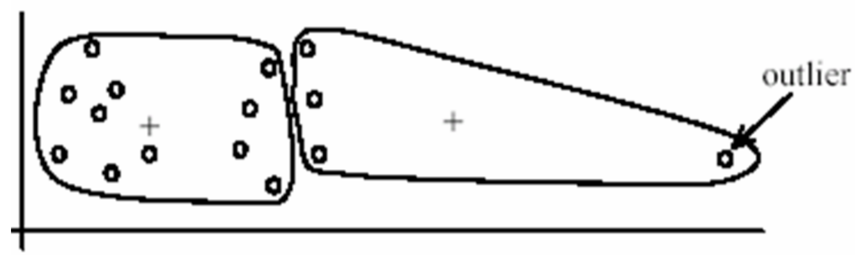

KMeans Iteration:
Total Within Cluster Sum of Squares:
250k
200k
150k
100k
50k
0
2
4
6
8
10
Kmeans Iterations

K-Means pros and cons

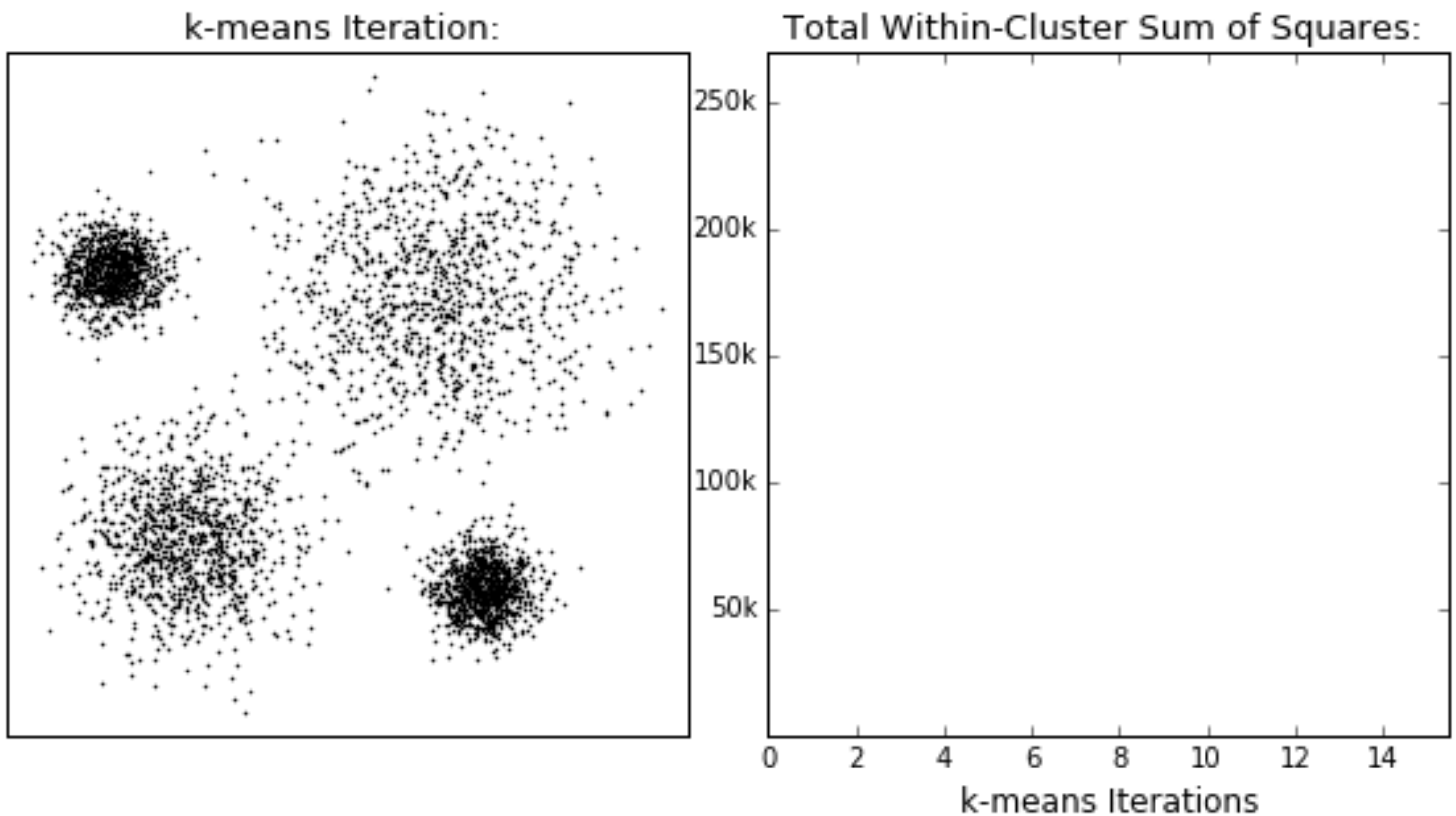
- Pros
  - Finds cluster centers that minimize variance (good representation of data)
  - Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)



(B): Ideal clusters



K-means Demo



k-means Iteration:
Total Within-Cluster Sum of Squares:
250k
200k
150k
100k
50k
0
2
4
6
8
10
12
14
k-means Iterations

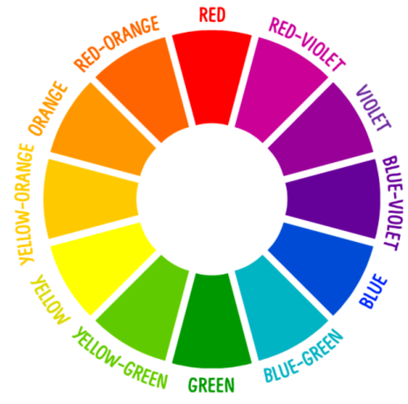
K-Means: initialization

- ***Very*** sensitive to the initial points
  - Do many runs of K-Means, each with different initial centroids
  - Seed the centroids using a better method than randomly choosing the centroids
    - e.g., Farthest-first sampling
- Must manually choose k
  - Learn the optimal k for the clustering
    - Note that this requires a performance measure

K-medoids

- Just like K-means except
  – Represent the cluster with one of its members, rather than the mean of its members
  – Choose the member (data point) that minimizes cluster dissimilarity

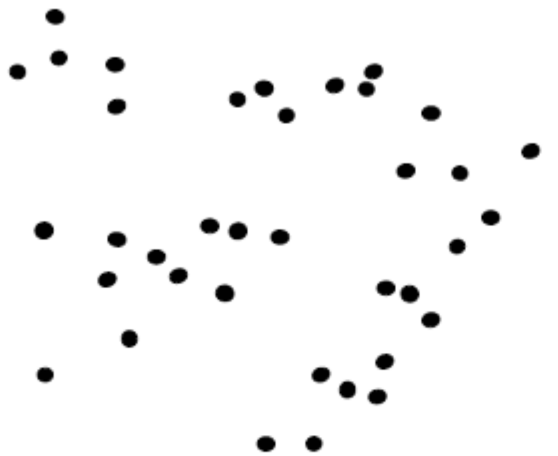
- Applicable when a mean is not meaningful
  – E.g., clustering values of hue



How might we cluster?

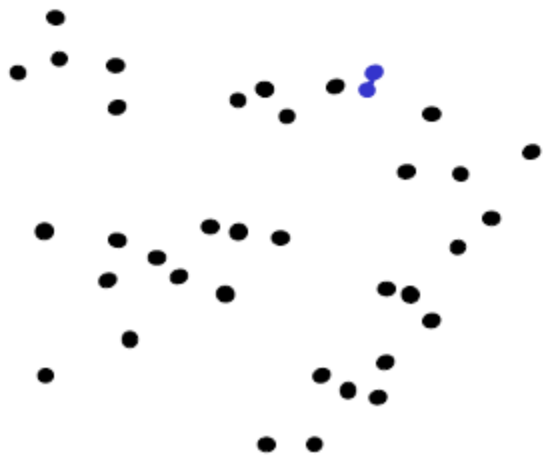
- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters

Agglomerative clustering



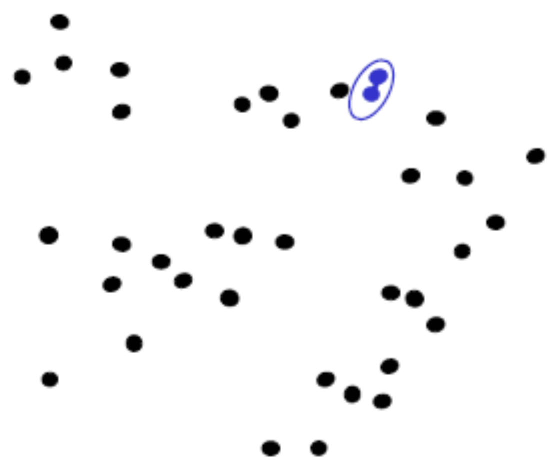
1. Say "Every point is its own cluster"

Agglomerative clustering



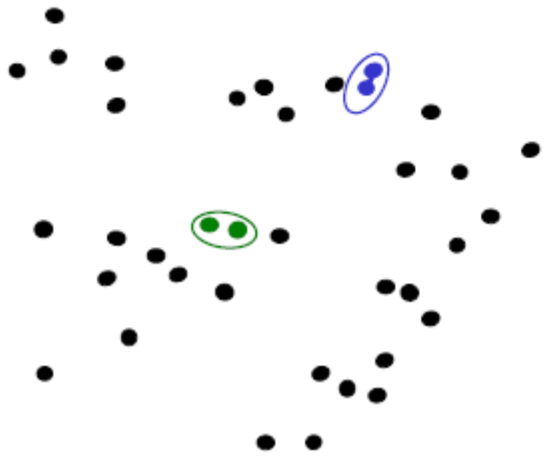
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

Agglomerative clustering



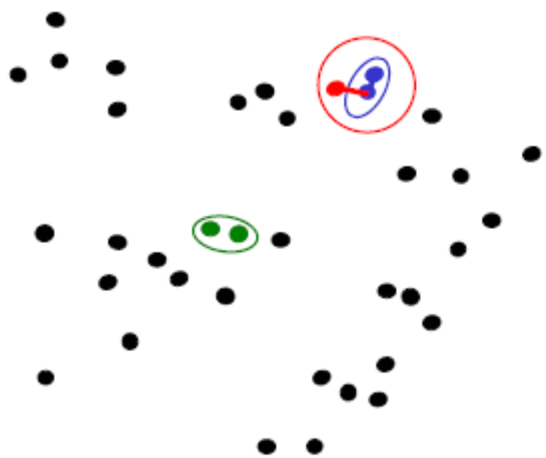
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Agglomerative clustering



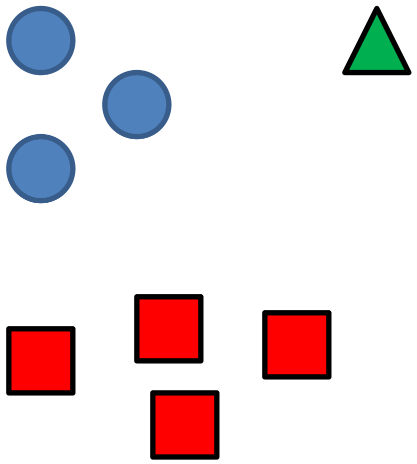
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Agglomerative clustering

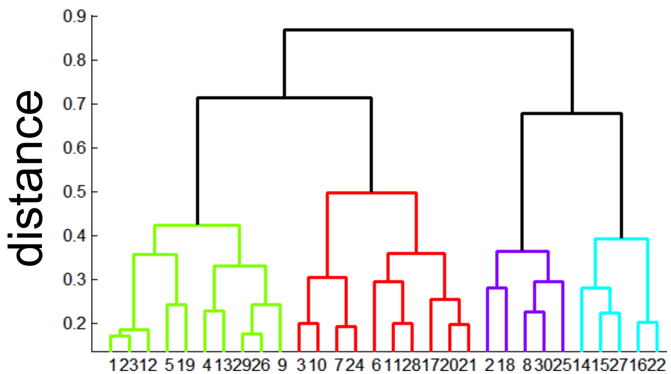


How to define cluster similarity?

- Average distance between points, maximum distance, minimum distance
- Distance between means or medoids

How many clusters?

- Clustering creates a dendrogram (a tree)
- Threshold based on max number of clusters or based on distance between merges



Agglomerative clustering demo