k-Means Clustering

Todd W. Neller
Gettysburg College

Laura E. Brown
Michigan Technological University

Outline

- Unsupervised versus Supervised Learning
- Clustering Problem
- k-Means Clustering Algorithm
- Visual Example
- Worked Example
- Initialization Methods
- Choosing the Number of Clusters *k*
- Variations for non-Euclidean distance metrics
- Summary

Supervised Learning

- **<u>Supervised Learning</u>** – Given training input and output (*x*, *y*) pairs, learn a function approximation mapping *x*'s to *y*'s.
  - **Regression** example: Given (*sepal_width*, *sepal_length*) pairs {(3.5, 5.1), (3, 4.9), (3.1, 4.6), (3.2, 4.7), …}, learn a function f(*sepal_width*) that predicts *sepal_length* well for all *sepal_width* values.
  - **Classification** example: Given (*balance*, *will_default*) pairs {(808, false), (1813, true), (944, true), (1072, false), …}, learn a function f(*balance*) that predicts *will_default* for all *balance* values.

Unsupervised Learning

- **Unsupervised Learning** – Given input data only (no training labels/outputs) learn characteristics of the data's *structure*.
  - **Clustering** example: Given a set of (*neck_size*, *sleeve_length*) pairs representative of a target market, determine a set of clusters that will serve as the basis for shirt size design.
- Supervised vs. Unsupervised Learning
  - Supervised learning: Given input and output, learn approximate mapping from input to output. (The output is the "supervision".)
  - Unsupervised learning: Given input only, output structure of input data.

Clustering Problem

- **Clustering** is grouping a set of objects such that objects in the same group (i.e. cluster) are more similar to each other in some sense than to objects of different groups.
- Our specific clustering problem:
  - Given: a set of n observations $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$, where each observation is a d-dimensional real vector
  - Given: a number of clusters k
  - Compute: a cluster assignment mapping $C(\boldsymbol{x}_i) \in \{1, \dots, k\}$ that minimizes the **within cluster sum of squares (WCSS)**:

$$\sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \boldsymbol{\mu}_{C(\boldsymbol{x}_i)} \right\|^2$$

where **centroid** $\boldsymbol{\mu}_{C(\boldsymbol{x}_i)}$ is the mean of the points in cluster $C(\boldsymbol{x}_i)$

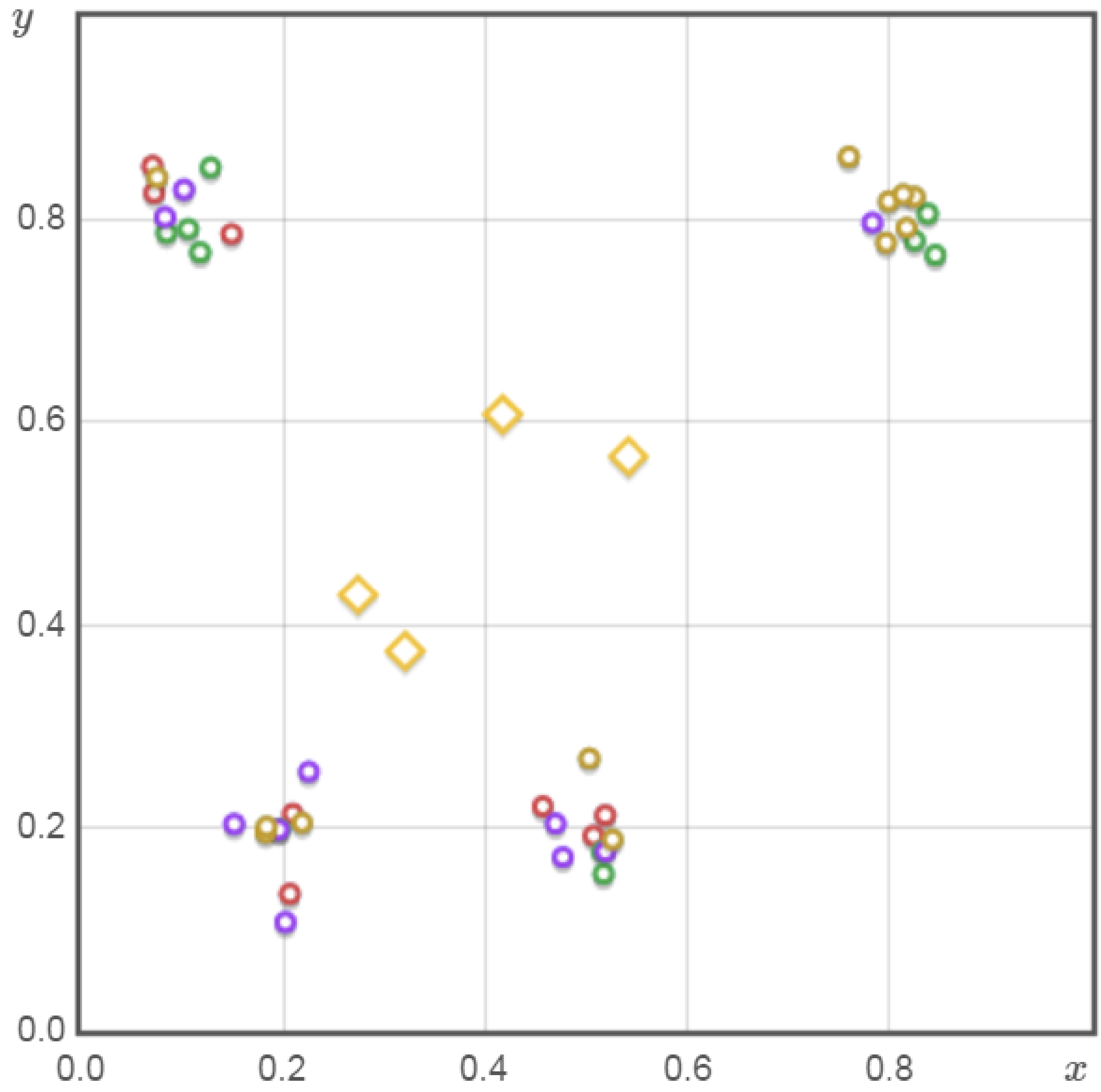
k-Means Clustering Algorithm

- General algorithm:
  - Randomly choose k cluster centroids $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots \boldsymbol{\mu}_k$ and arbitrarily initialize cluster assignment mapping C.
  - While remapping C from each $\boldsymbol{x}_i$ to its closest centroid $\boldsymbol{\mu}_j$ causes a change in C:
    - Recompute $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots \boldsymbol{\mu}_k$ according to the new C
- In order to minimize the WCSS, we alternately:
  - Recompute C to minimize the WCSS holding $\boldsymbol{\mu}_j$ fixed.
  - Recompute $\boldsymbol{\mu}_j$ to minimize the WCSS holding C fixed.

  In minimizing the WCSS, we seek a clustering that minimizes Euclidean distance *variance* within clusters.

Visual Example

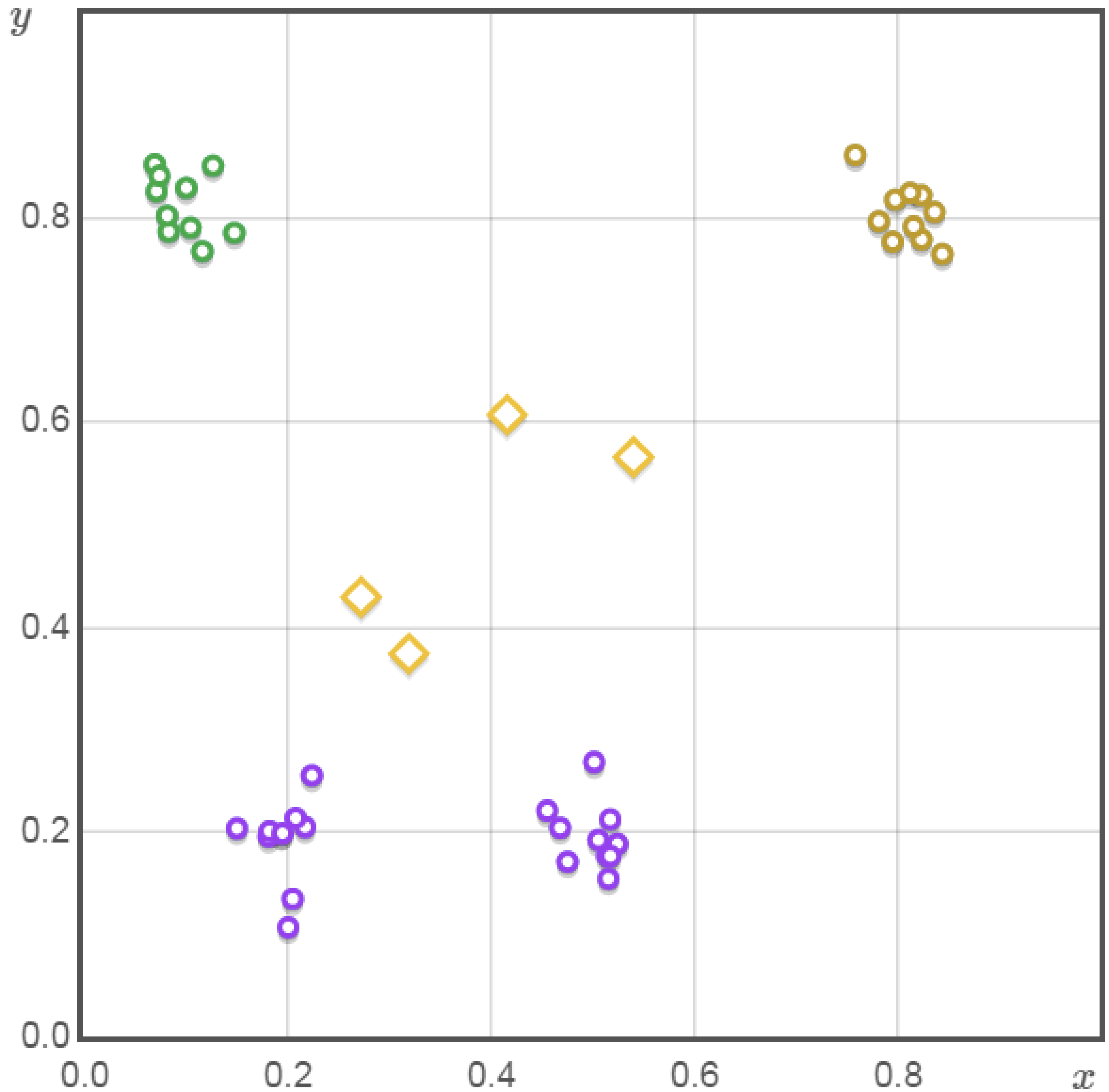
- Circle data points are randomly assigned to clusters (color = cluster).
- Diamond cluster centroids initially assigned to the means of cluster data points.



Screenshots from http://www.onmyphd.com/?p=k-means.clustering. Try it!

Visual Example

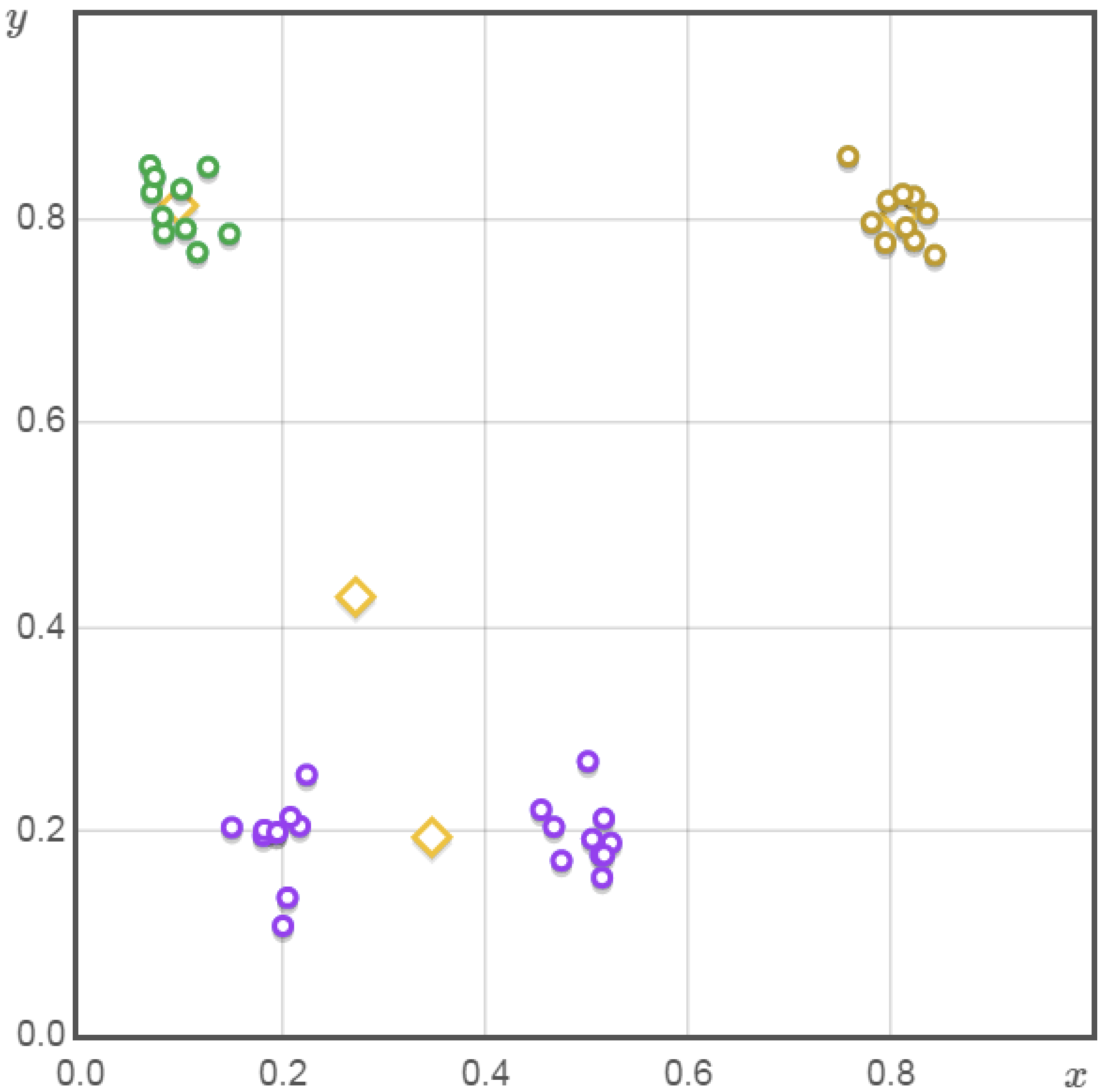
- Circle data points are reassigned to their closest centroid.



Screenshots from http://www.onmyphd.com/?p=k-means.clustering. Try it!

Visual Example

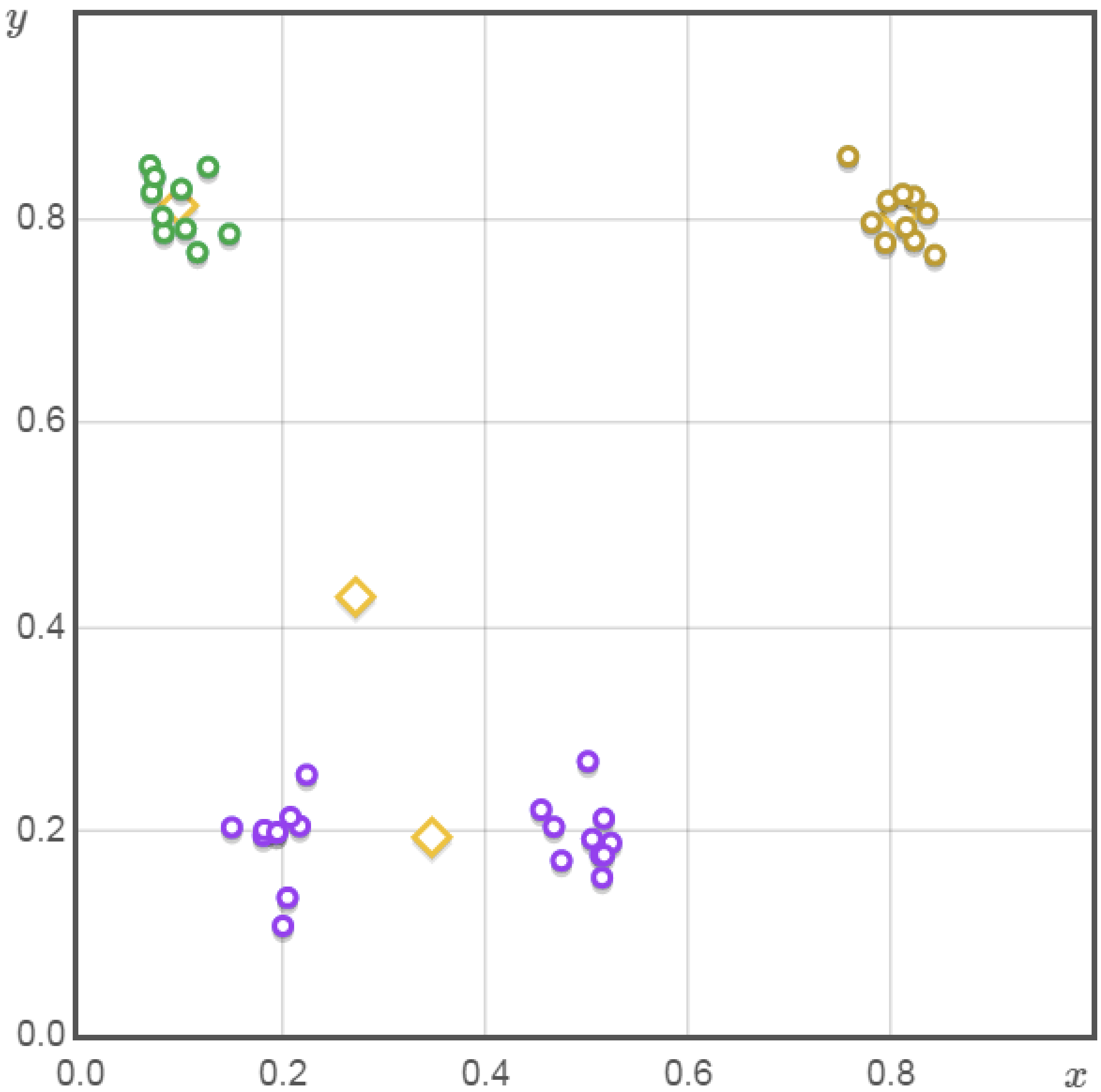
- Diamond cluster centroids are reassigned to the means of cluster data points.
- Note that one cluster centroid no longer has assigned points (red).



Screenshots from http://www.onmyphd.com/?p=k-means.clustering. Try it!

Visual Example

- After this, there is no circle data point cluster reassignment.
- WCSS has been minimized and we terminate.
- However, this is a *local minimum*, not a *global minimum* (one centroid per cluster).



Screenshots from http://www.onmyphd.com/?p=k-means.clustering. Try it!

Worked Example

- Given:
  - n=6 data points with dimensions d=2
  - k=3 clusters
  - Forgy initialization of centroids
- For each data point, compute the new cluster / centroid index to be that of the closest centroid point …

Data

x1	x2
-2	1
-2	3
3	2
5	2
1	-2
1	-4

Cluster / Centroid Index

c
1
1
0
0
0
2

Centroids

index	x1	x2
0	1	-2
1	-2	1
2	1	-4

Worked Example

- For each centroid, compute the new centroid to be the mean of the data points assigned to that cluster / centroid index …

Data

x1	x2
-2	1
-2	3
3	2
5	2
1	-2
1	-4

Cluster / Centroid Index

c
1
1
0
0
0
2

Centroids

index	x1	x2
0	3	.7
1	-2	2
2	1	-4

Worked Example

- For each data point, compute the new cluster / centroid index to be that of the closest centroid point …

Data

x1	x2
-2	1
-2	3
3	2
5	2
1	-2
1	-4

Cluster / Centroid Index

c
1
1
0
0
2
2

Centroids

index	x1	x2
0	3	.7
1	-2	2
2	1	-4

Worked Example

- For each centroid, compute the new centroid to be the mean of the data points assigned to that cluster / centroid index …

Data

x1	x2
-2	1
-2	3
3	2
5	2
1	-2
1	-4

Cluster / Centroid Index

c
1
1
0
0
2
2

Centroids

index	x1	x2
0	4	2
1	-2	2
2	1	-3

Worked Example

- For each data point, compute the new cluster / centroid index to be that of the closest centroid point.
- With no change to the cluster / centroid indices, the algorithm terminates at a local (and in this example *global*) minimum WCSS = $1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 = 6$.

Data

x1	x2
-2	1
-2	3
3	2
5	2
1	-2
1	-4

Cluster / Centroid Index

c
1
1
0
0
2
2

Centroids

index	x1	x2
0	4	2
1	-2	2
2	1	-3

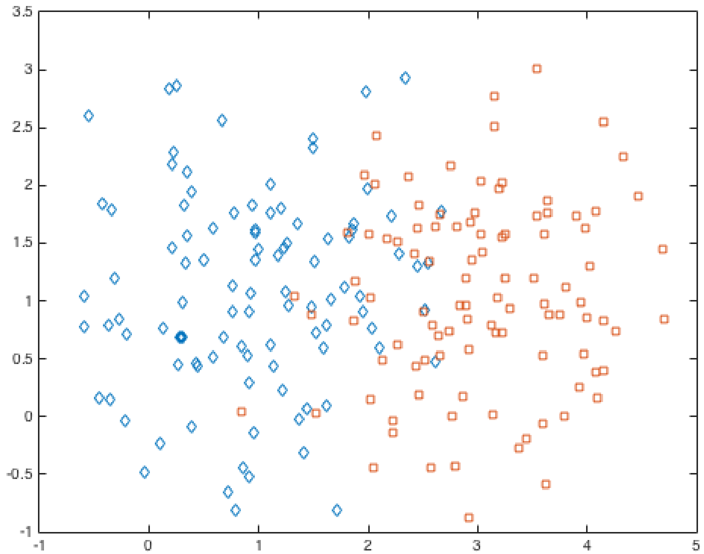
k-Means Clustering Assumptions

- *k*-Means Clustering assumes real-valued data distributed in clusters that are:
  - Separate
  - Roughly hyperspherical (circular in 2D, spherical in 3D) or easily clustered via a Voronoi partition.
  - Similar size
  - Similar density
- Even with these assumptions being met, *k*-Means Clustering is not guaranteed to find the global minimum.

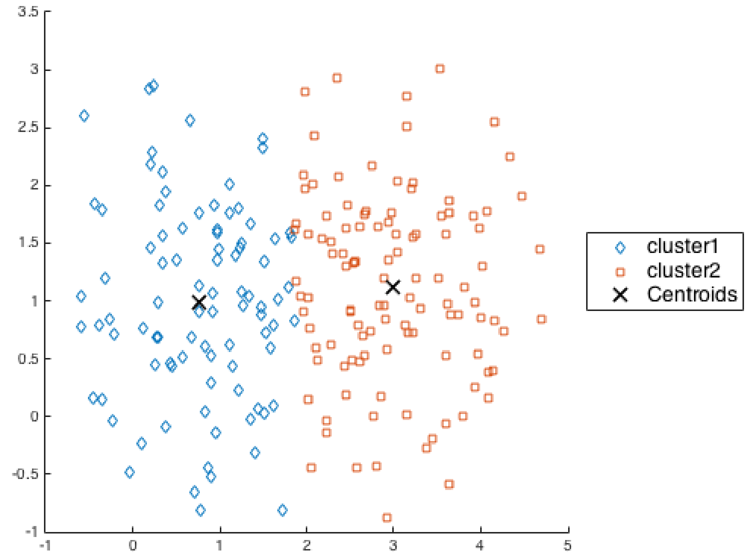
k-Means Limitations

Separate Clusters

Original Data



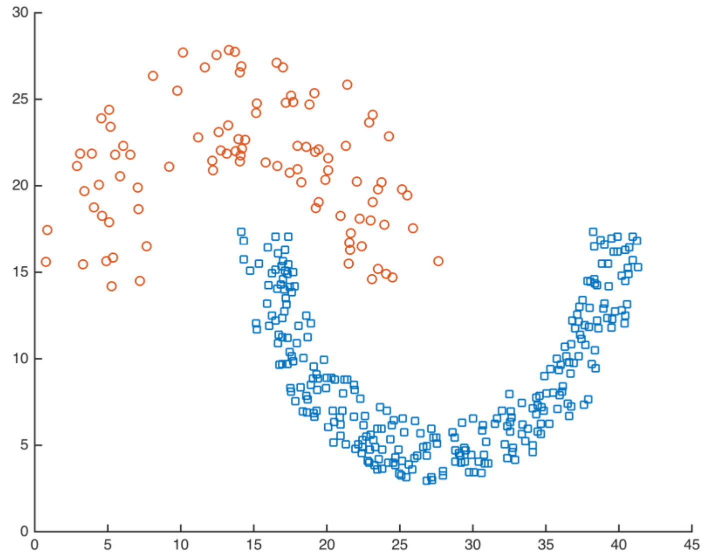
Results of *k*-means Clustering



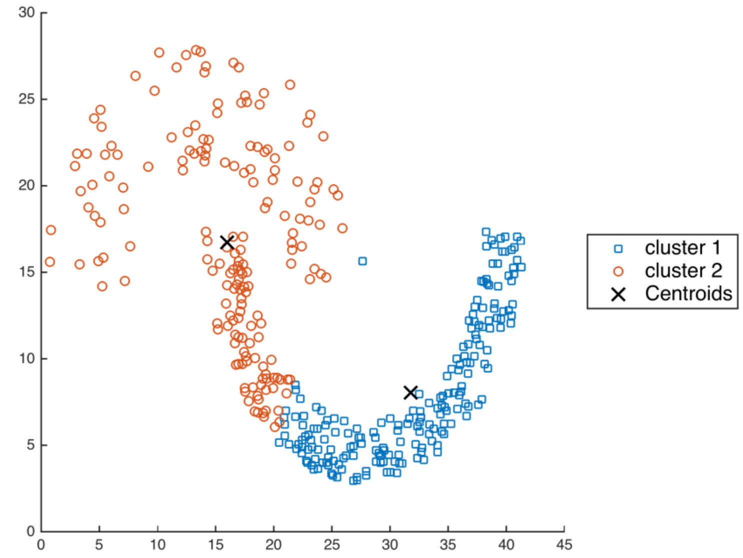
k-Means Limitations

Hyperspherical Clusters

Original Data



Results of *k*-means Clustering

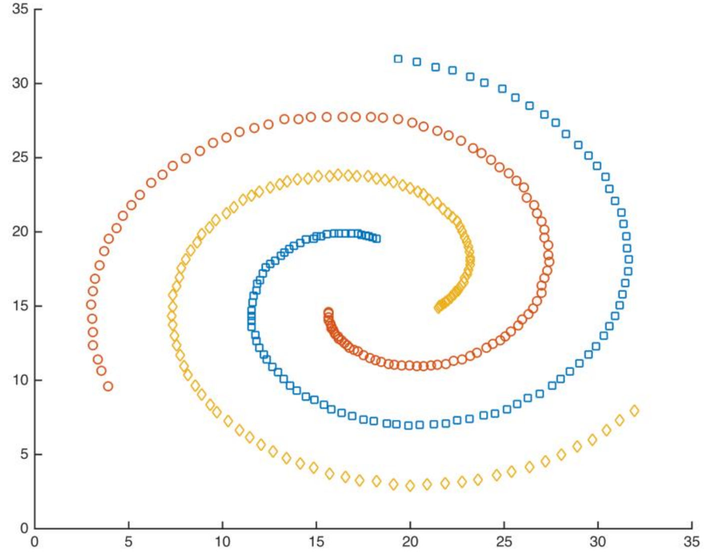


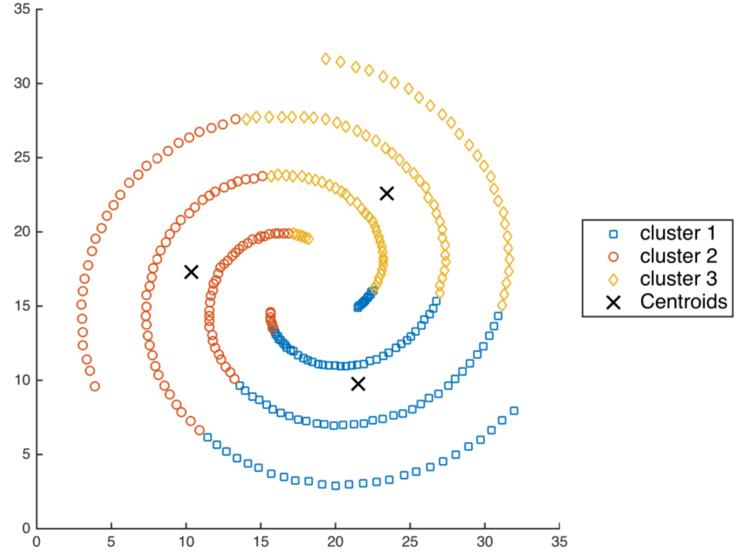
Data available at: http://cs.joensuu.fi/sipu/datasets/

Original data source: Jain, A. and M. Law, Data clustering: A user's dilemma. *LNCS*, 2005.

k-Means Limitations
Hypersphercal Clusters

Original Data | **Results of *k*-means Clustering**



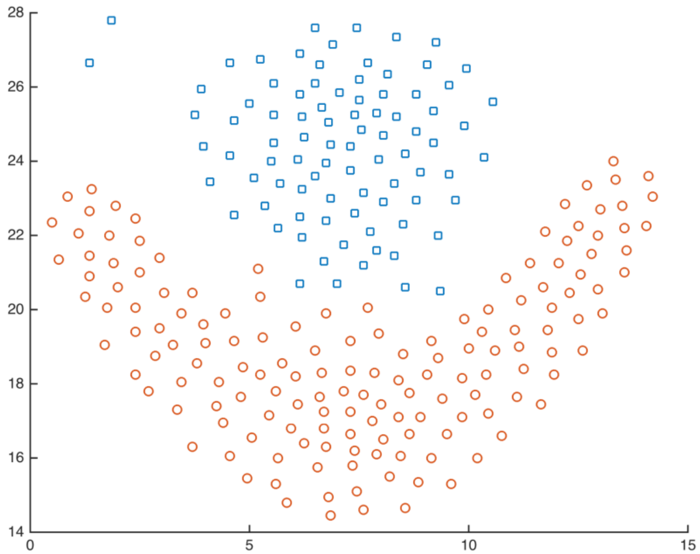


Data available at: http://cs.joensuu.fi/sipu/datasets/
Original data source: Chang, H. and D.Y. Yeung. *Pattern Recognition*, 2008. 41(1): p. 191-203.

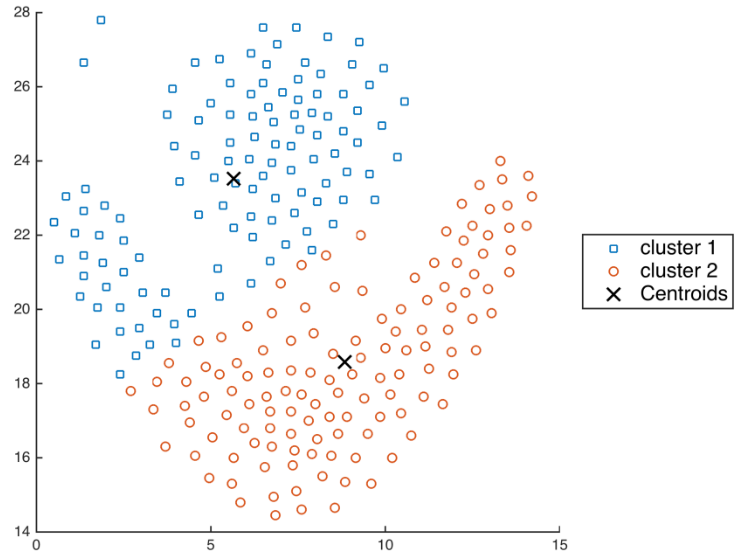
k-Means Limitations

Hypherspherical Clusters

Original Data



Results of *k*-means Clustering



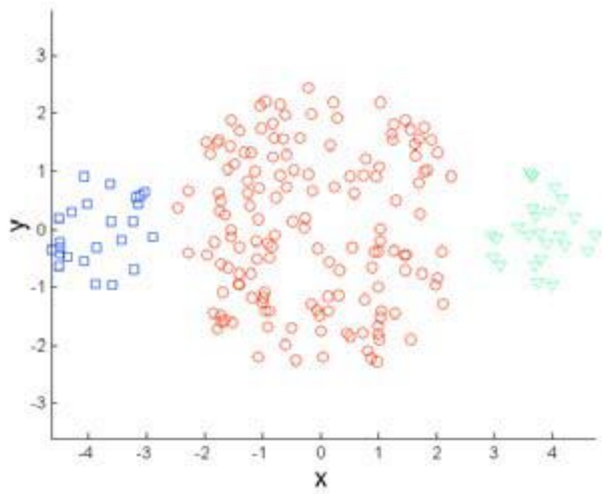
Data available at: http://cs.joensuu.fi/sipu/datasets/

Original data source: Fu, L. and E. Medico. *BMC bioinformatics*, 2007. 8(1): p. 3.

k-Means Limitations

Similar Size Clusters

Original Data



Results of *k*-means Clustering

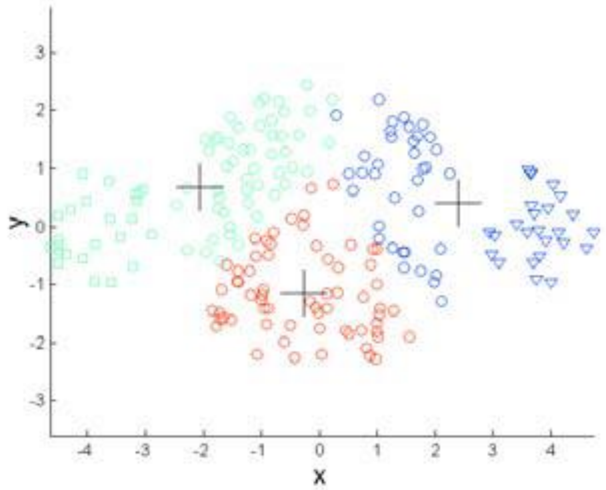
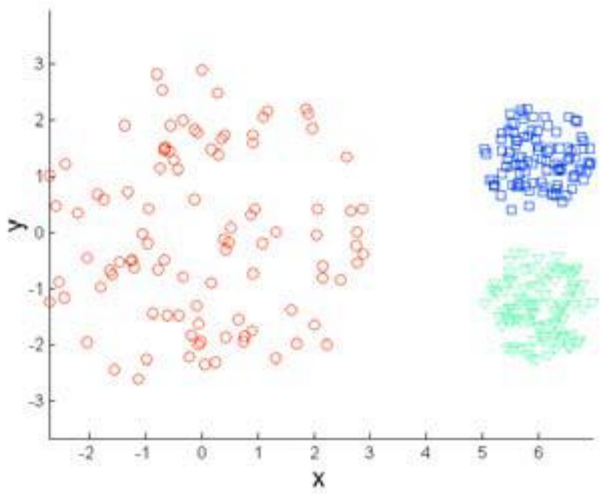


Image source: Tan, Steinbach, and Kumar *Introduction to Data Mining*
http://www-users.cs.umn.edu/~kumar/dmbook/index.php

k-Means Limitations

Similar Density Clusters

Original Data



Results of *k*-means Clustering

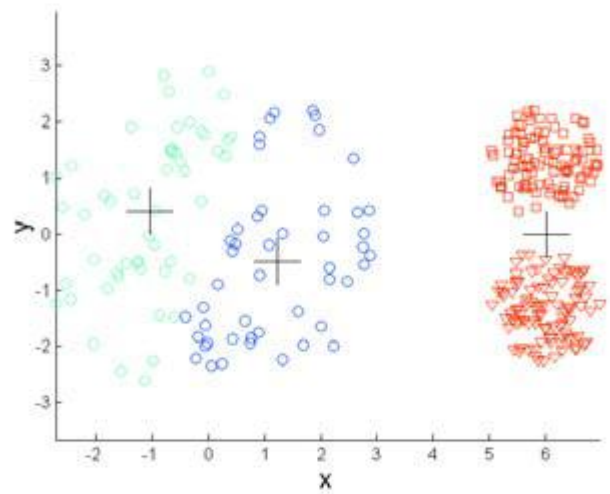


Image source: Tan, Steinbach, and Kumar *Introduction to Data Mining*
http://www-users.cs.umn.edu/~kumar/dmbook/index.php

k-Means Clustering Improvements

- As with many local optimization techniques applied to global optimization problems, it often helps to:
  - apply the approach through multiple separate iterations, and
  - retain the clusters from the iteration with the minimum WCSS.
- Initialization:
  - Random initial cluster assignments create initial centroids clustered about the global mean.
  - Forgy initialization: Choose unique random input data points as the initial centroids. Local (not global) minimum results are still possible. (Try it out.)
  - Distant samples: Choose unique input data points that approximately minimize the sum of inverse square distances between points (e.g. through stochastic local optimization).

Where does the given *k* come from?

- Sometime the number of clusters *k* is determined by the application. Examples:
  - Cluster a given set of (*neck_size*, *sleeve_length*) pairs into *k*=5 clusters to be labeled S/M/L/XL/XXL.
  - Perform color quantization of a 16.7M RGB color space down to a palette of *k*=256 colors.
- Sometimes we need to determine an appropriate value of *k*. How?

Determining the Number of Clusters *k*

- When *k* isn't determined by your application:
  - The Elbow Method:
    - Graph *k* versus the WCSS of iterated *k*-means clustering
    - The WCSS will generally decrease as *k* increases.
    - However, at the most natural *k* one can sometimes see a sharp bend or "elbow" in the graph where there is significant decrease up to that *k* but not much thereafter. Choose that *k*.
  - The Gap Statistic
  - Other methods

The Gap Statistic

- Motivation: We'd like to choose *k* so that clustering achieves the greatest WCSS reduction relative to uniform random data.
- For each candidate k:
  - Compute the log of the best (least) WCSS we can find ($\log(W_k)$).
  - Estimate the expected value $E^*_n\{\log(W_k)\}$ on uniform random data.
    - One method: Generate 100 uniformly distributed data sets of the same size over the same ranges. Perform k-means clustering on each, and compute the log of the WCSS. Average the these log WCSS values.
  - The **gap statistic** for this k would then be $E^*_n\{\log(W_k)\} - \log(W_k)$.
- Select the k that maximizes the gap statistic.
- R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic

Variation: k-Medoids

- Sometimes the Euclidean distance measure is not appropriate (e.g. qualitative data).
- k-Medoids is a k-Means variation that allows a general distance measure $D\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$:
  - Randomly choose k cluster medoids $\boldsymbol{m}_1, \boldsymbol{m}_2, \dots \boldsymbol{m}_k$ from the data set.
  - While remapping C from each $\boldsymbol{x}_i$ to its closest medoid $\boldsymbol{m}_j$ causes a change in C:
    - Recompute each $\boldsymbol{m}_j$ to be the $\boldsymbol{x}_i$ in cluster j that minimizes the total of within-cluster medoid distances $\sum_{i', C(\boldsymbol{x}_{i'})=j} D(\boldsymbol{x}_i, \boldsymbol{x}_{i'})$
- PAM (Partitioning Around Medoids) - as above except when recomputing each $\boldsymbol{m}_j$, replace with *any* non-medoid data set point $\boldsymbol{x}_i$ that minimizes the overall sum of within-cluster medoid distances.

Summary

- **Supervised learning** is given input-output pairs for the task of function approximation.
- **Unsupervised learning** is given input only for the task of finding structure in the data.
- ***k*-Means Clustering** is a simple algorithm for clustering data with separate, hyperspherical clusters of similar size and density.
- **Iterated *k*-Means** helps to find the best global clustering. Local cost minima are possible.

Summary (cont.)

- *k*-Means can be **initialized** with random cluster assignments, a random sample of data points (Forgy), or a distant sample of data points.
- The **number of clusters *k*** is sometimes determined by the application and sometimes via the **Elbow Method**, **Gap Statistic**, etc.
- ***k*-Medoids** is a variation that allows one to handle data with any suitable distance measure (not just Euclidean).