Python for Data Science

Machine Learning in Python: Clustering

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Articulate the goal of cluster analysis
- Discuss whether cluster analysis is supervised or unsupervised
- List some ways that cluster results can be applied

Cluster Analysis Overview

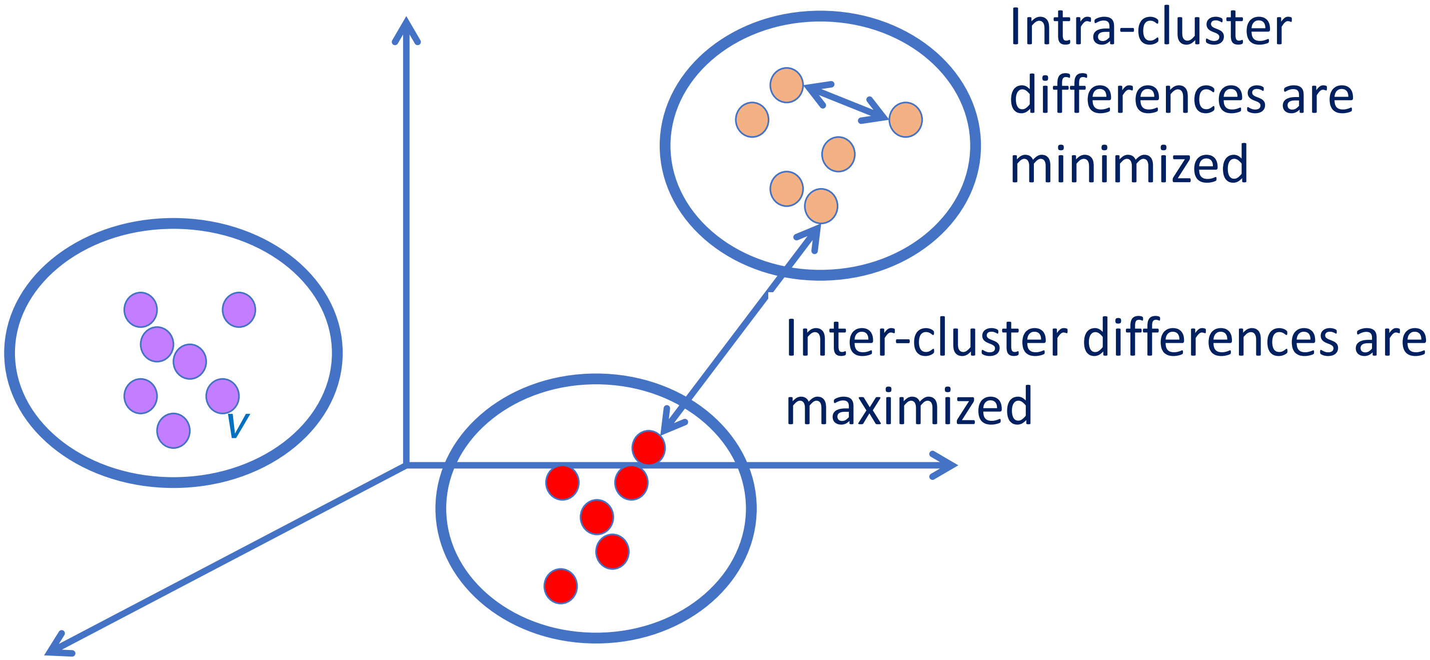
Goal: Organize similar items into groups

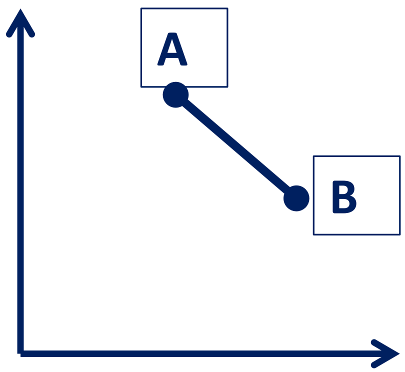
Cluster Analysis Examples

- Segment customer base into groups
- Characterize different weather patterns for a region
- Group news articles into topics
- Discover crime hot spots

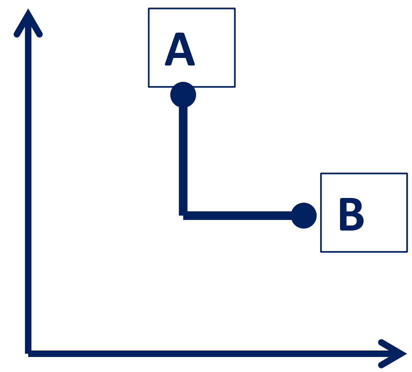
Cluster Analysis

- Divides data into clusters
- Similar items are placed in same cluster

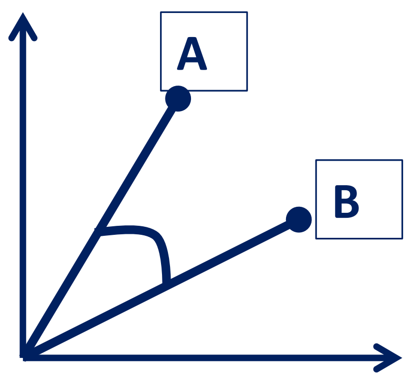




Euclidean Distance

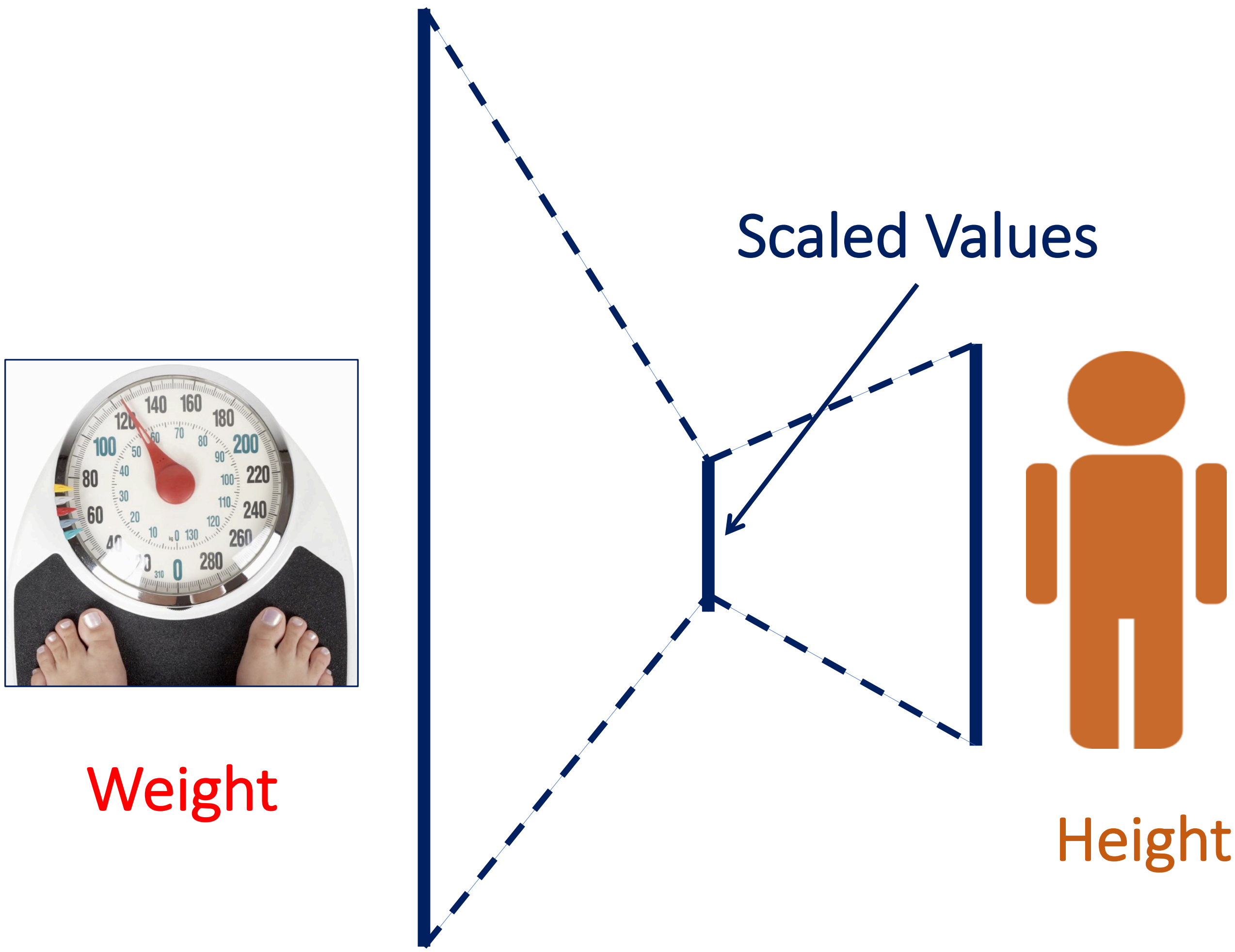


Manhattan Distance



Cosine Similarity

Normalizing Input Variables



Cluster Analysis Notes

Unsupervised

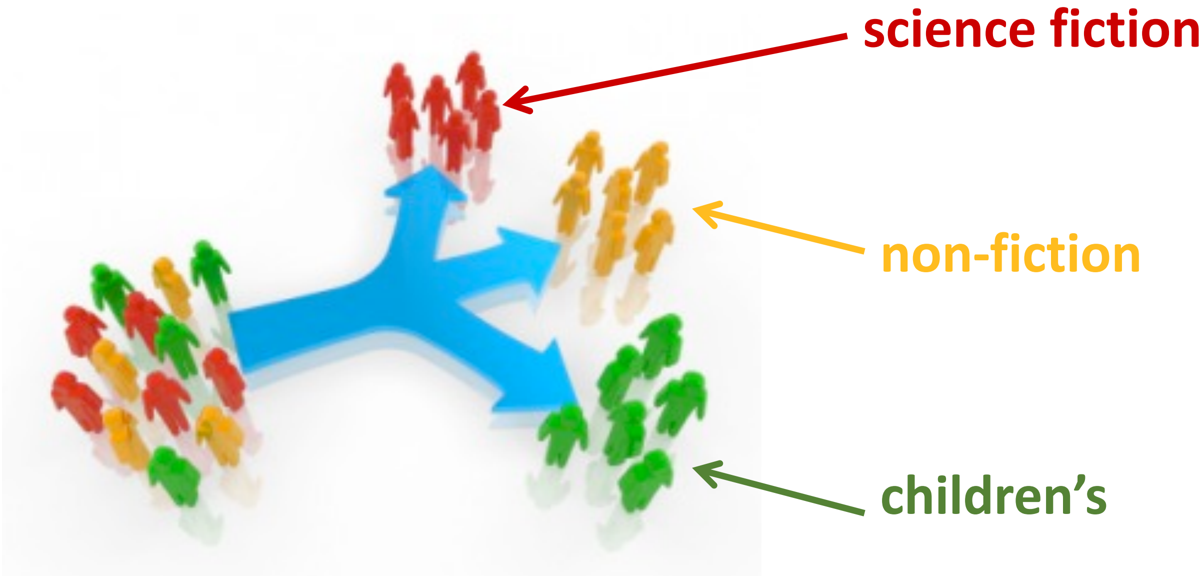
There is no 'correct' clustering

Clusters don't come with labels

Interpretation and analysis required to make sense of clustering results!

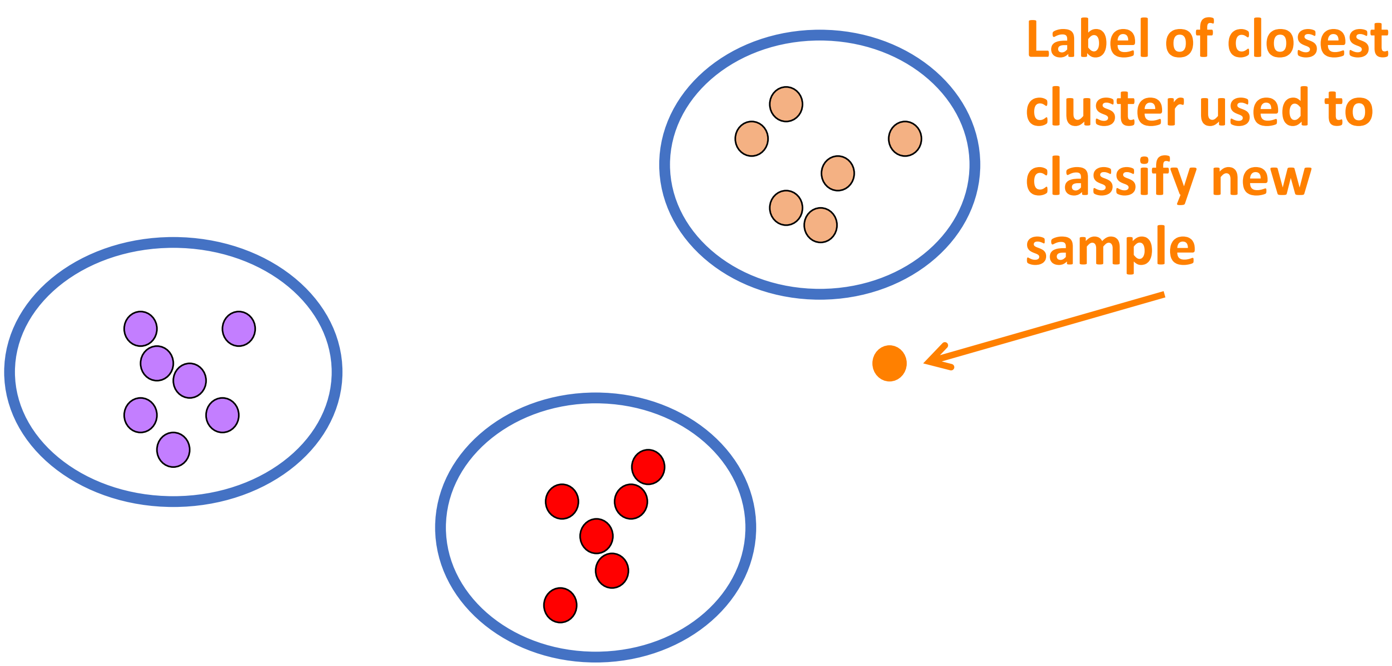
Uses of Cluster Results

- Data segmentation
  - Analysis of each segment can provide insights



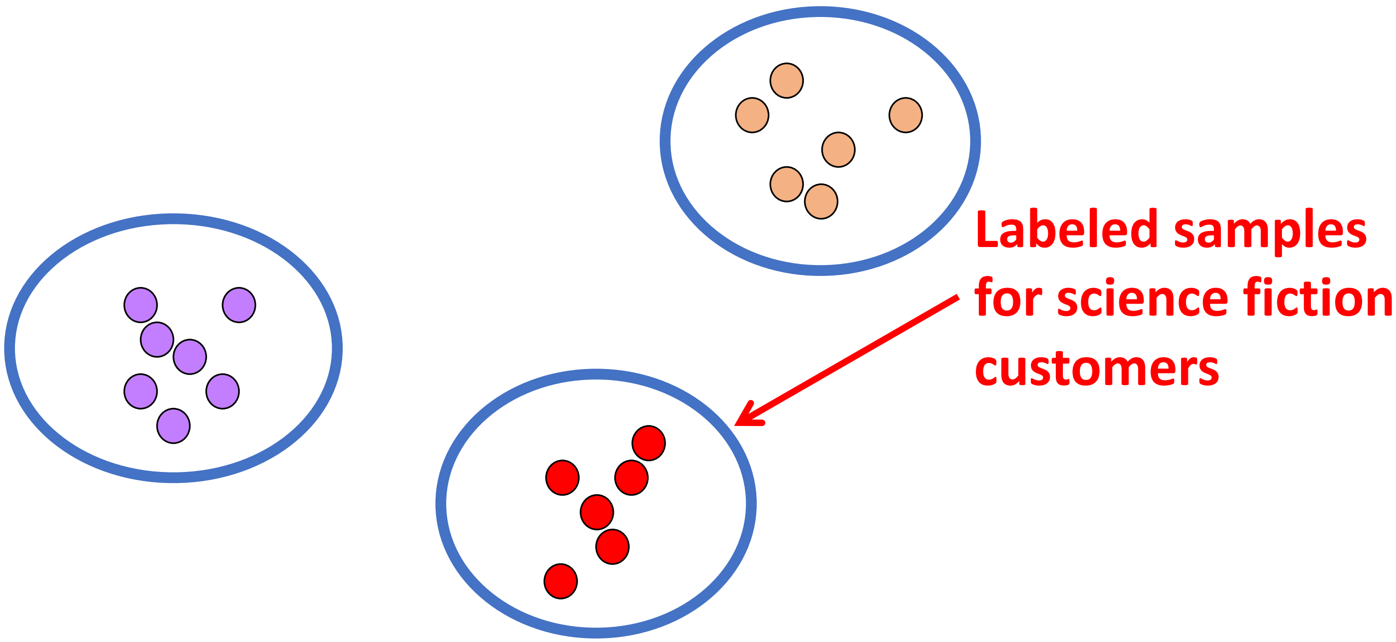
Uses of Cluster Results

- Categories for classifying new data
  - New sample assigned to closest cluster



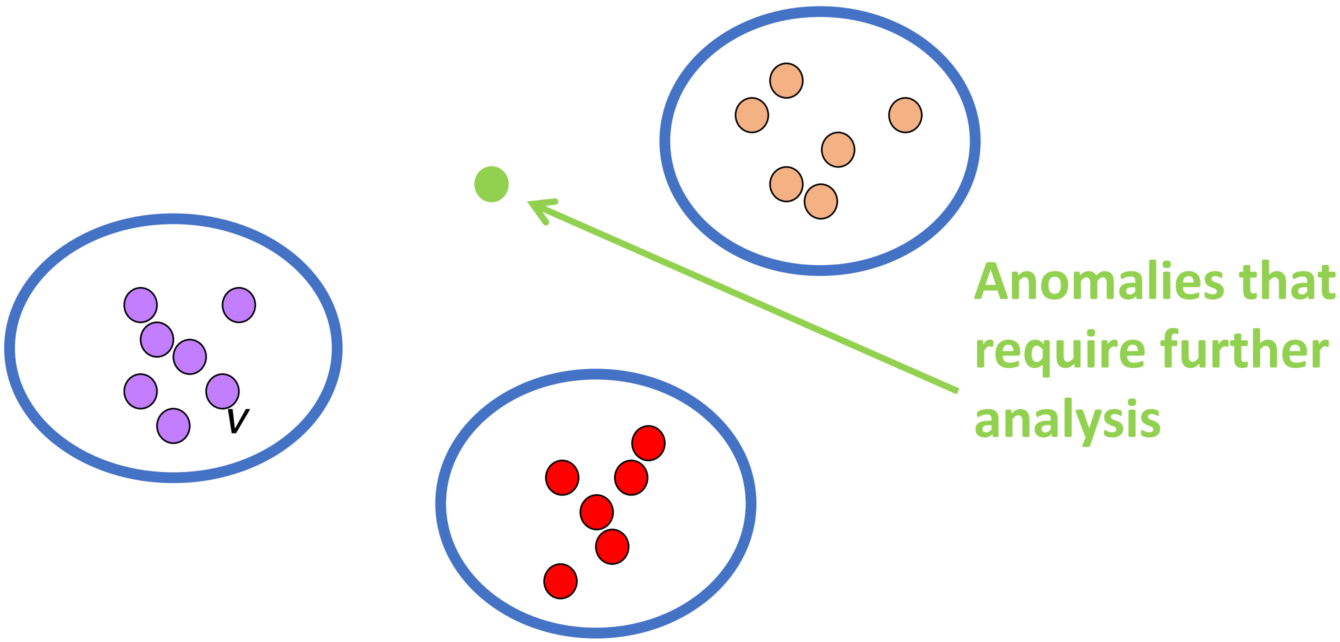
Uses of Cluster Results

- Labeled data for classification
  - Cluster samples used as labeled data

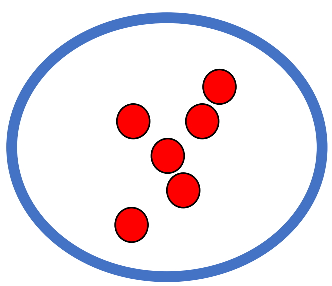


Uses of Cluster Results

- Basis for anomaly detection
  - Cluster outliers are anomalies



- Organize similar items into groups
- Analyzing clusters often leads to useful insights about data
- Clusters require analysis and interpretation



Python for Data Science

Machine Learning in Python: k-Means Clustering

Dr. Ilkay Altintas and Dr. Leo Porter

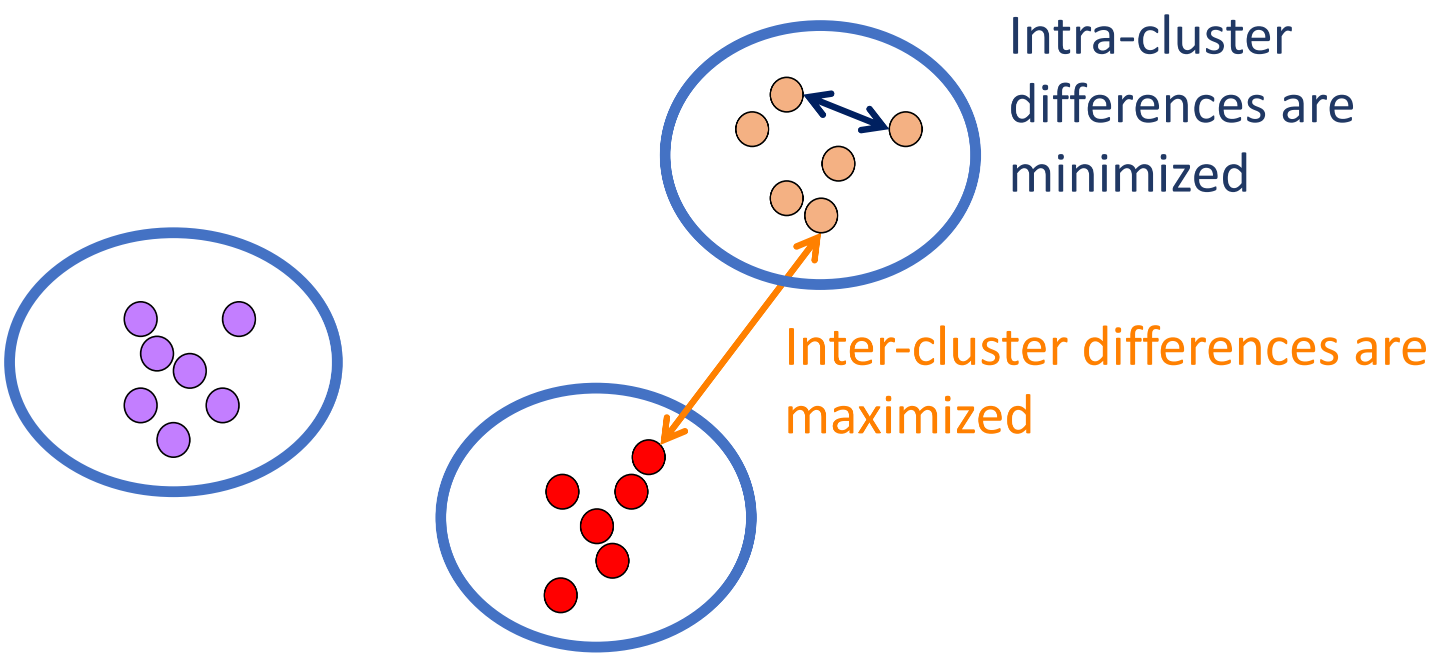
Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Describe the steps in the k-means algorithm
- Explain what the ‘k’ stands for in k-means
- Define cluster centroid

Cluster Analysis

- Divides data into clusters
- Similar items are in same cluster



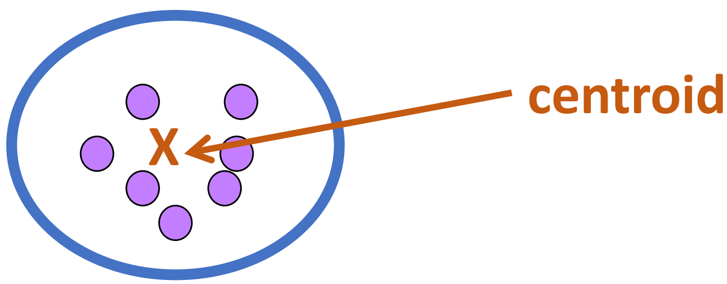
k-Means Algorithm

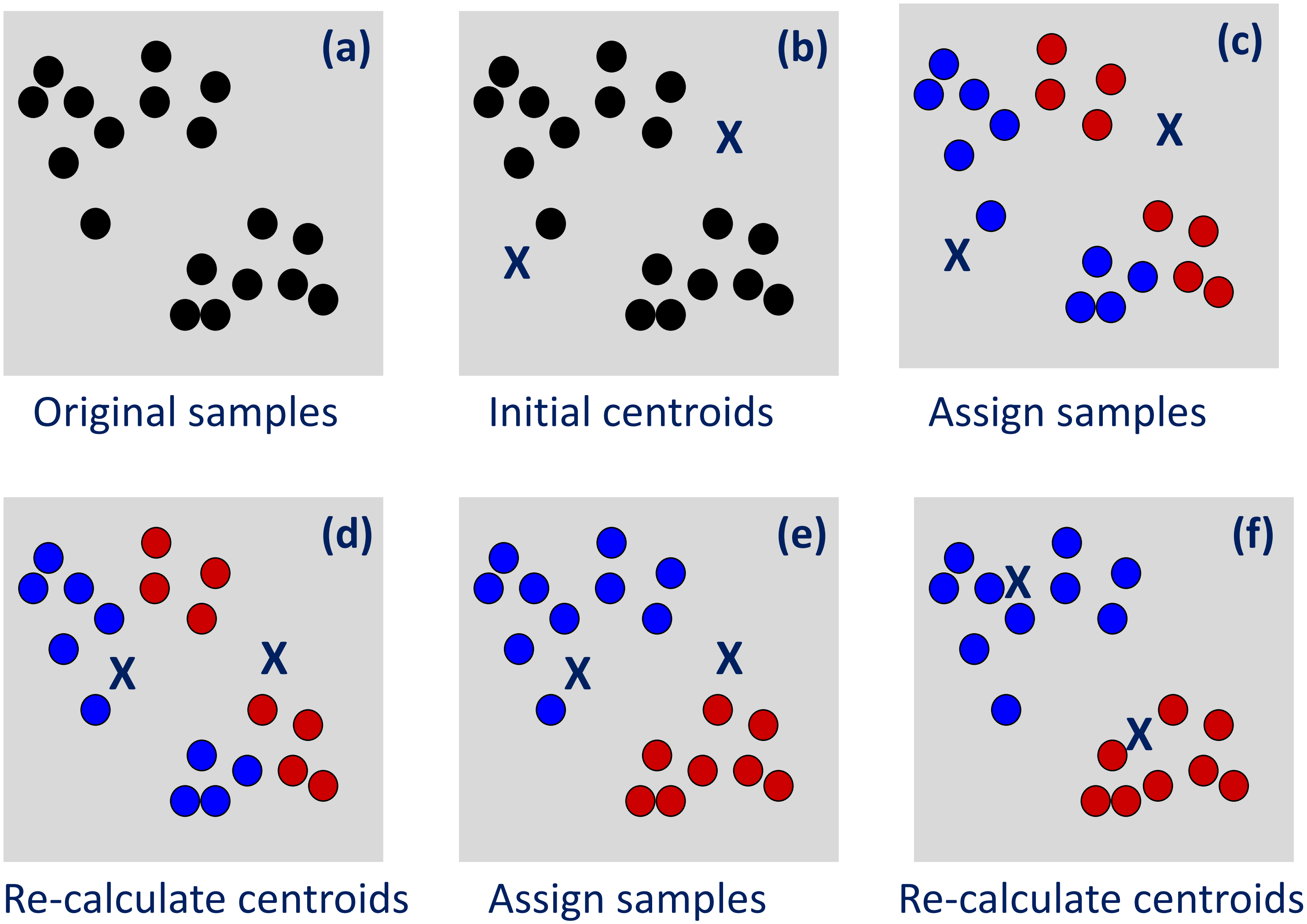
Select k initial centroids (cluster centers)

Repeat

- Assign each sample to closest centroid
- Calculate mean of cluster to determine new centroid

Until some stopping criterion is reached




(a)
Original samples
(b)
X
X
Initial centroids
(c)
X
X
Assign samples
(d)
X
X
Re-calculate centroids
(e)
X
X
Assign samples
(f)
X
X
Re-calculate centroids

Choosing Initial Centroids

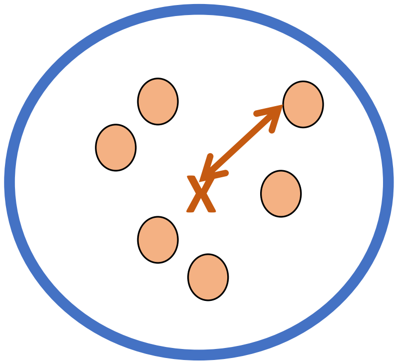
Issue:

Final clusters are sensitive to initial centroids

Solution:

Run k-means multiple times with
different random initial centroids,
and choose best results

Evaluating Cluster Results



error = distance between sample & centroid

squared error = error^2

Sum of squared errors between all samples & centroid

Sum over all clusters

WSSE

Within-Cluster Sum of Squared Error

Using WSSE

$WSSE_1 < WSSE_2$ → WSSE1 is better numerically

Caveats:

- Does not mean that cluster set 1 is more 'correct' than cluster set 2
- Larger values for k will always reduce WSSE

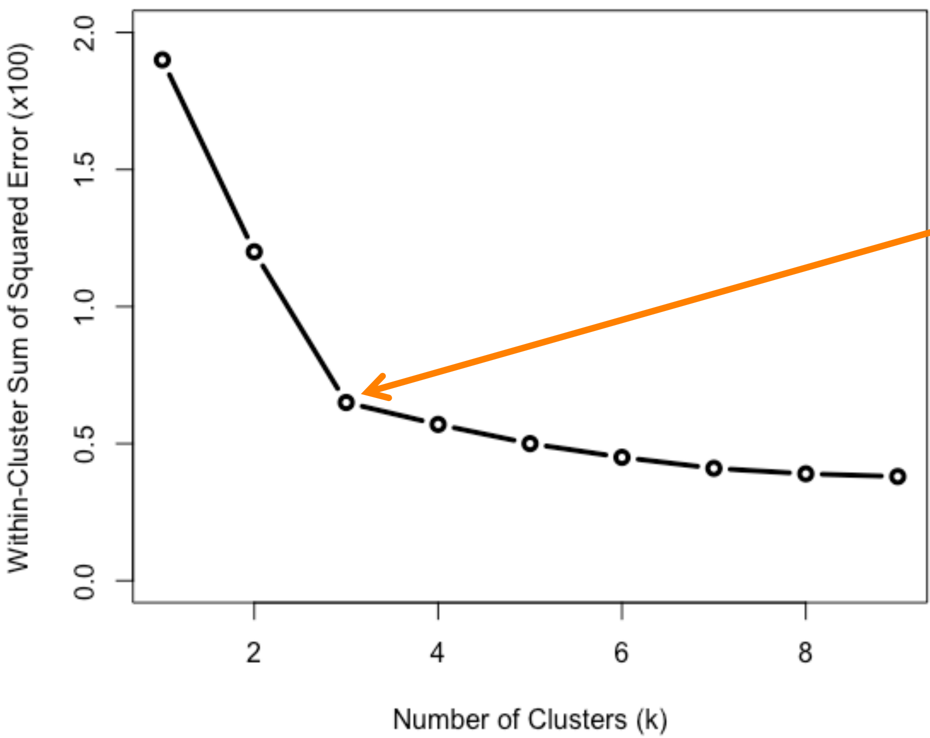
Choosing Value for k

- **Approaches:**

$\boxed{k = ?}$

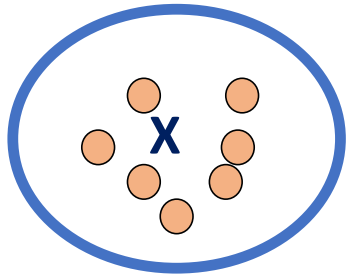
  - Visualization
  - Application-Dependent
  - Data-Driven

Elbow Method for Choosing k



“Elbow” suggests value for k should be 3

Stopping Criteria

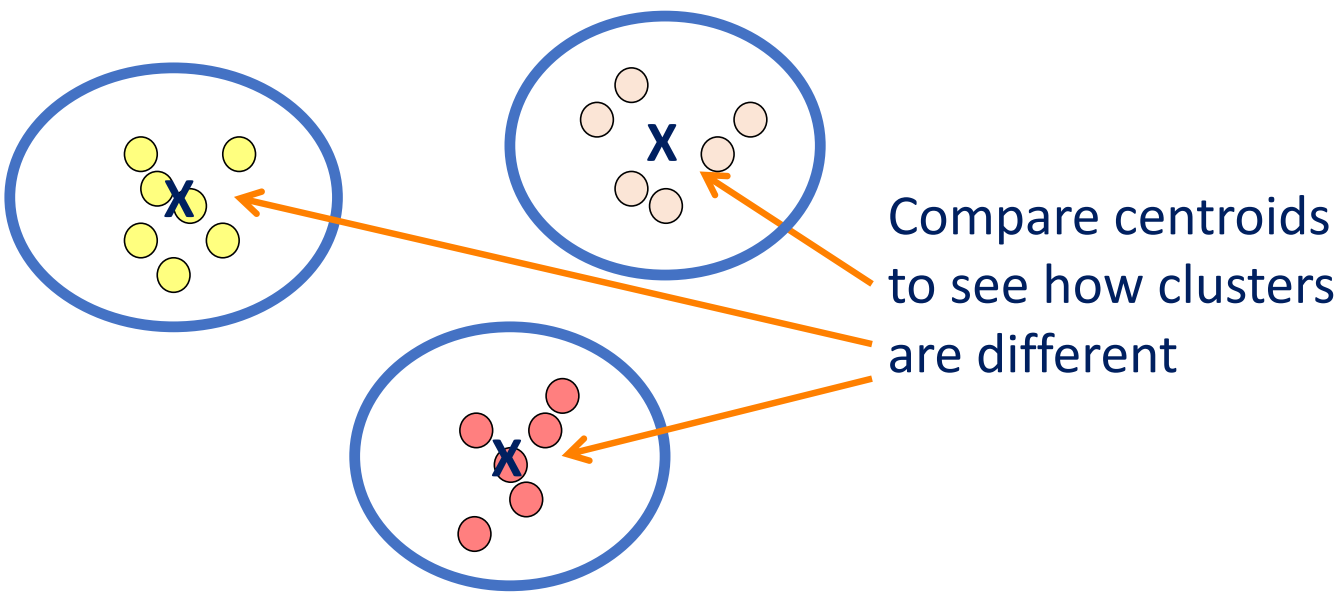


When to stop iterating?

- No changes to centroids
- Number of samples changing clusters is below threshold

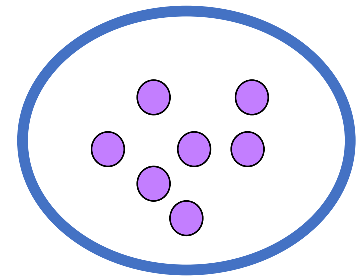
Interpreting Results

- Examine cluster centroids
  - How are clusters different?



Python for Data Science

K-Means Summary



- Classic algorithm for cluster analysis
- Simple to understand and implement and is efficient
- Value of k must be specified
- Final clusters are sensitive to initial centroids