MECS 6616

Lecture 2: Clustering

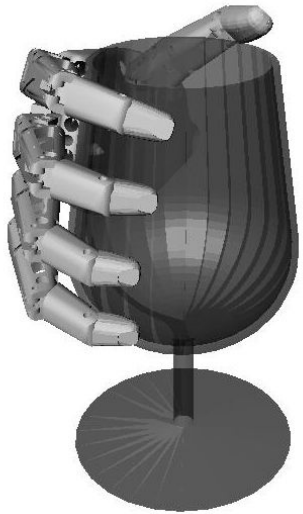
Spring 2020

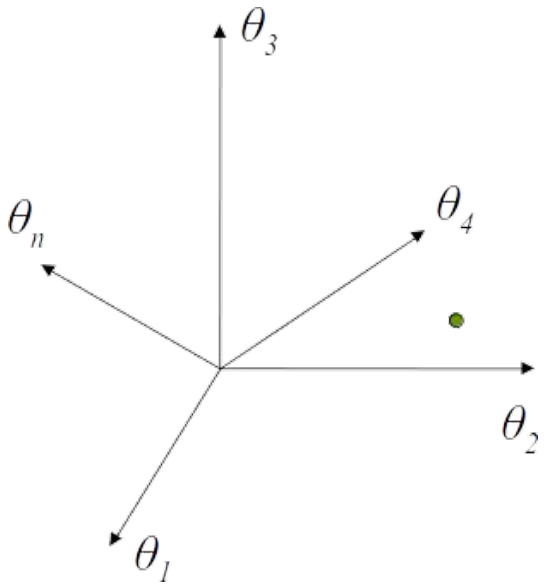
Matei Ciocarlie

Primary sources: [Muller and Guido, Introduction to Machine Learning with Python]
[Hastie et al., The Elements of Statistical Learning]

Unsupervised Learning

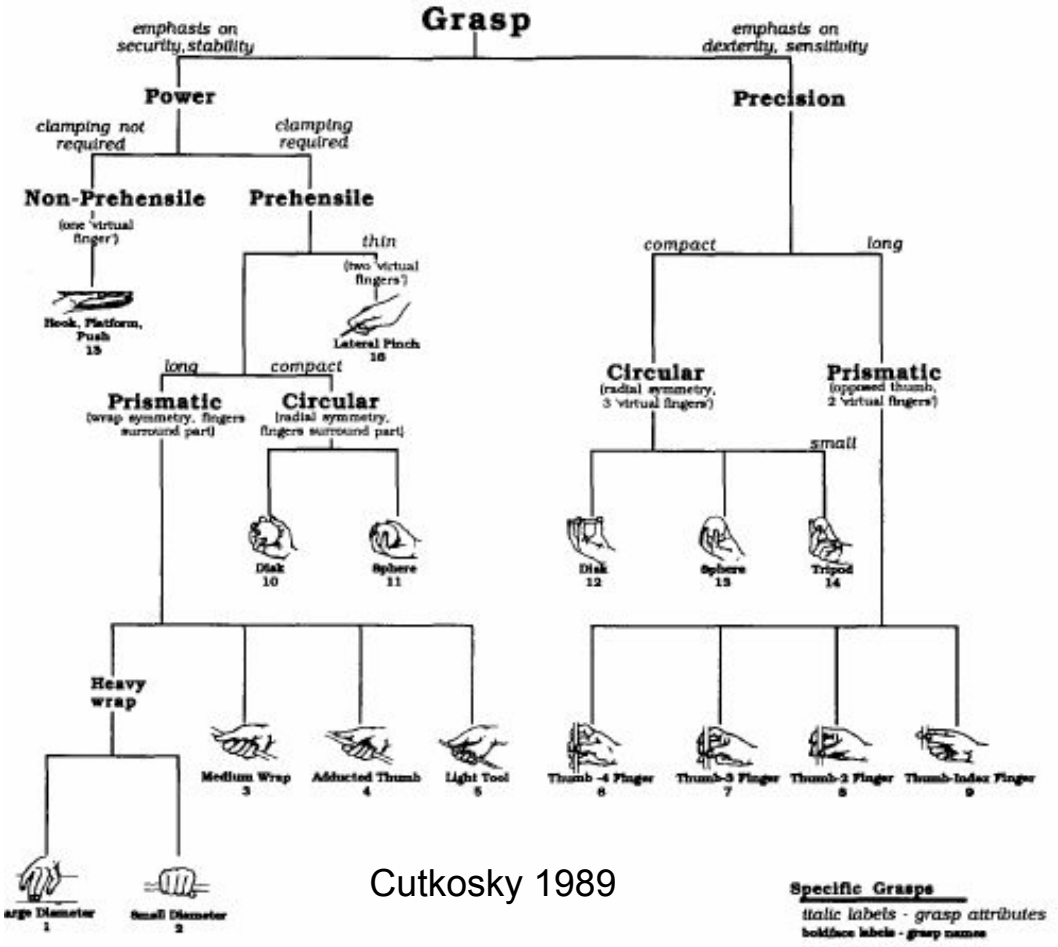
- Finding structure in data
- Example: dexterous hand grasps
  - each grasp is a point in a very high-dimensional space
  - is there some structure in the **useful parts** of this space?





Unsupervised Learning

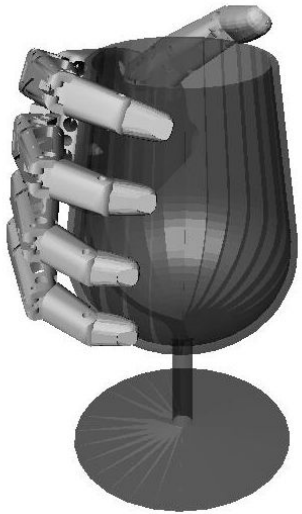
- Finding structure in data
- Grasp taxonomy: long standing idea based on intuition
- What does it mean formally?
  - if you plot where most useful grasps fall in the 20-dimensional space, most of them will be grouped around a few **clusters**



Cutkosky 1989

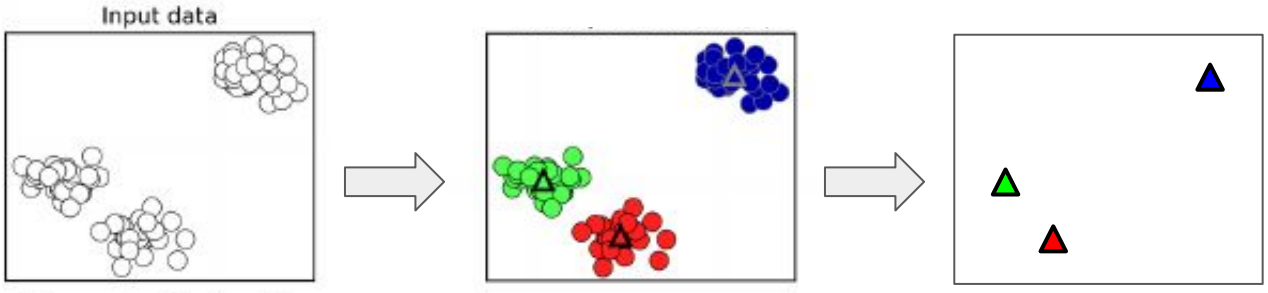
Data Representation

- The high-dimensional vector $\mathbf{x} = [x_1, x_2, \ldots, x_d] \subset \Re^d$
- In this specific case:
  - d ~= 20 (for the human hand)
  - $x_i = q_i$ = angle of the i-th joint of the hand
  - each grasp is a point in a 20-dimensional space
- Our data is usually a high-dimensional point cloud
- Unfortunately, high-dimensional spaces can not be visualized
  - 2D toy examples
  - projections of high-dimensional spaces into fewer dimensions



Clustering

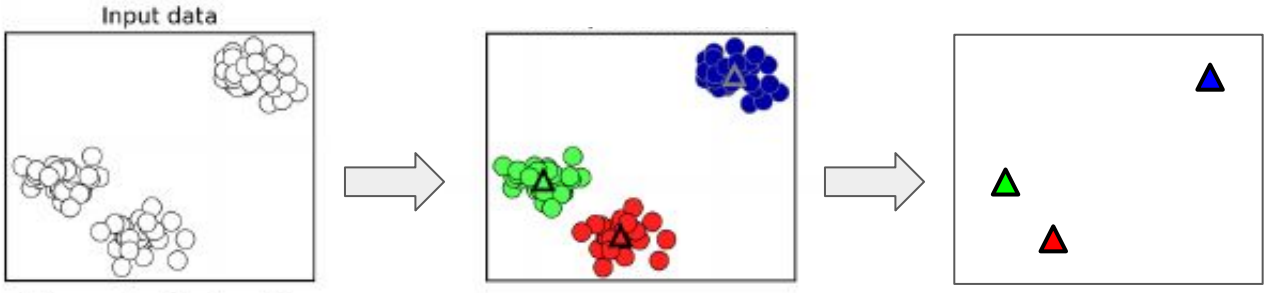
- Given a (potentially large) dataset $\mathbf{S} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, is it the case that many of these points tend to cluster together?
- Why answer this question?
  - reveals something about the mechanism that generated that data
  - we can simplify our reasoning about this space: instead of lots of points in many dimensions, I can reason about a few cluster representatives



[Muller and Guido, Introduction to Machine Learning with Python]

Clustering

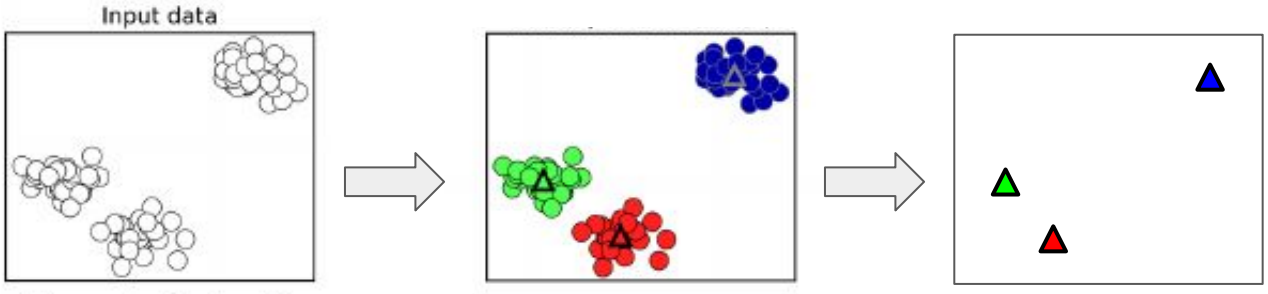
- The “model”:
  - Number of clusters
  - Assignment of the original points to clusters
  - (optional) Some way to choose a cluster “representative”
- The “fitting” procedure
- The “prediction” procedure



[Muller and Guido, Introduction to Machine Learning with Python]

Clustering

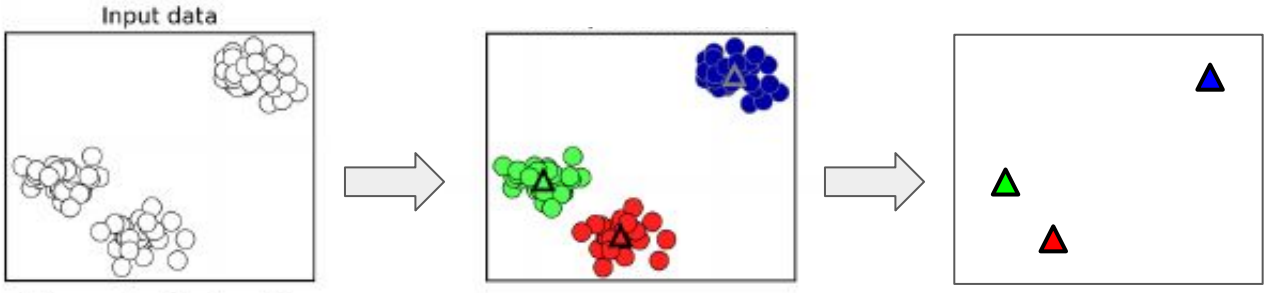
- The “model”
- The “fitting” procedure: choose all of the above such that:
  - points inside the same cluster are “close” to each other
  - points in different clusters are “far” from each other
  - alternatively: points are close to the representative of their cluster
- The “prediction” procedure



[Muller and Guido, Introduction to Machine Learning with Python]

Clustering

- The “model”
- The “fitting” procedure
- The “prediction” procedure:
  - Could be nothing - just gain insight from clustering result
  - Replace original data points with their representative
  - Assign any new point to a cluster (and replace it with its representative)



[Muller and Guido, Introduction to Machine Learning with Python]

Clustering - Formal Definitions

- Clustering “encoder”:
  - assume K clusters, labeled by integer $k \subset \{1, \ldots, \mathrm{K}\}$
  - The encoder assigns a cluster to each point: $\mathrm{C}(i) = k$
- Distance metric $\mathrm{d}(\mathbf{x}_i, \mathbf{x}_j)$
  - often this is Euclidean distance: $\mathrm{d}(\mathbf{x}_i, \mathbf{x}_j) = \sum(x_{i,k} - x_{j,k})^2$
  - but there are many other distance functions!
- Within-cluster point scatter:

$$\mathrm{W}(\mathrm{C}) = \tfrac{1}{2} \sum_{k=1}^{\mathrm{K}} \sum_{\mathrm{C}(i)=\mathrm{k}} \sum_{\mathrm{C}(j)=\mathrm{k}} \mathrm{d}(\mathbf{x}_i, \mathbf{x}_j)$$

Clustering - Formal Definitions

- Within-cluster point scatter:

$$W(C) = \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)=k} d(\mathbf{x}_i, \mathbf{x}_j)$$

- Between-cluster point scatter:

$$B(C) = \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)\neq k} d(\mathbf{x}_i, \mathbf{x}_j)$$

- Goal: minimize within-cluster scatter and maximize between-cluster scatter

Clustering - Formal Definitions

- Total point scatter:

$$T = W(C)+B(C) = \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)=k} d(\mathbf{x}_i, \mathbf{x}_j) + \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)\neq k} d(\mathbf{x}_i, \mathbf{x}_j) =$$

$$= \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \left(\sum_{C(j)=k} d(\mathbf{x}_i, \mathbf{x}_j) + \sum_{C(j)\neq k} d(\mathbf{x}_i, \mathbf{x}_j) \right)$$

$$= \tfrac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{j} d(\mathbf{x}_i, \mathbf{x}_j) =$$

$$= \tfrac{1}{2} \sum_{i} \sum_{j} d(\mathbf{x}_i, \mathbf{x}_j)$$

Clustering - Combinatorics

- How do we find the best cluster assignments?
- Brute force?
  - total number of possible assignments is combinatorial
    - 10 points in 4 clusters: 34,105 possible assignments
    - 19 points in 4 clusters: 10^{10} possible assignments
  - … not an option.
- Clever exact algorithms?
  - often NP-hard (depending on exact problem specification)
  - running time exponential in the size of the problem
- In practice: clever approximate algorithms...

K-means clustering

- Minimize the distance from the points of a cluster to the mean of that cluster.
- NP-hard to solve exactly
- Iterative, greedy approximate algorithm:
    - `Input: N points, number of clusters K`
    - `Create random cluster assignment C`
    - `Repeat:`
        - `Compute the mean of each cluster`
        - `Re-assign each point to the cluster with the closest mean`
    - `Until no point has changed cluster assignment in last iteration`

K-means clustering

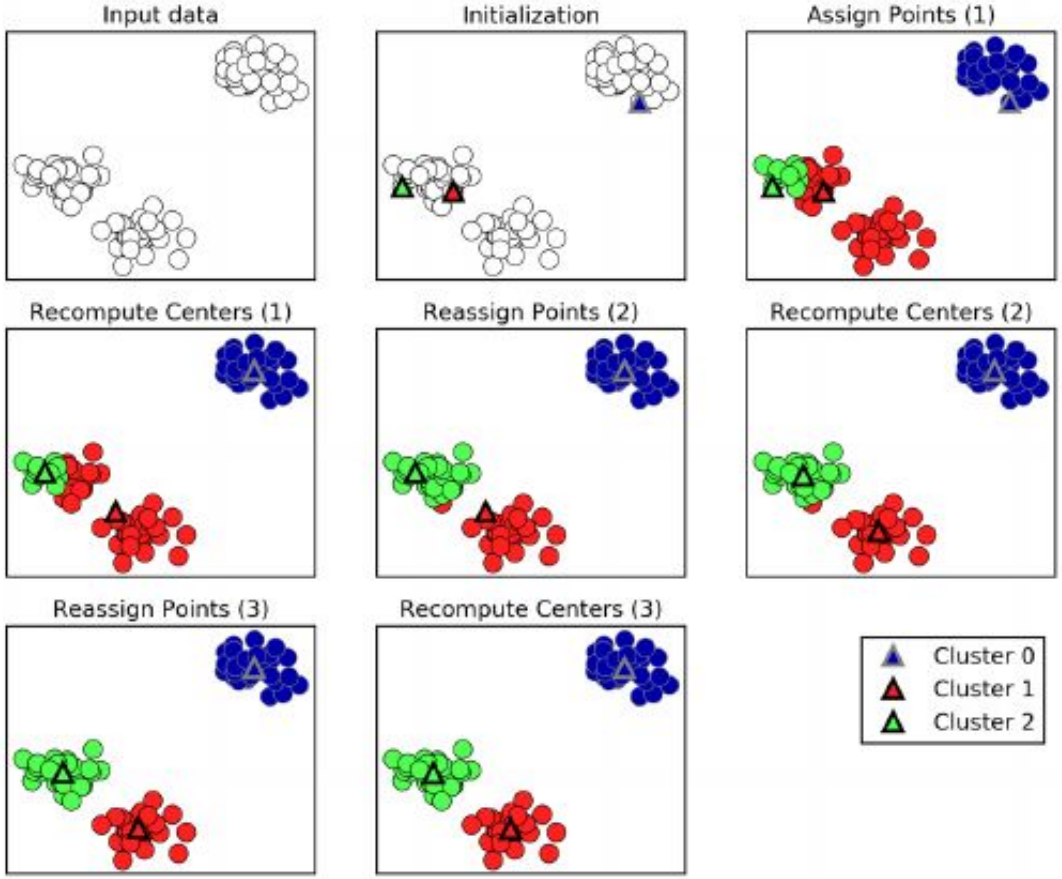


Figure 3-23. Input data and three steps of the k-means algorithm

[Muller and Guido, Introduction to Machine Learning with Python]

K-means clustering

- Implementations
  - scikit-learn k-means
  - scipy cluster vq
- Interfaces
  - what does `.fit(X)` mean?
  - what does `.predict(x)` mean?

K-means Practical Details

- How many clusters are there?

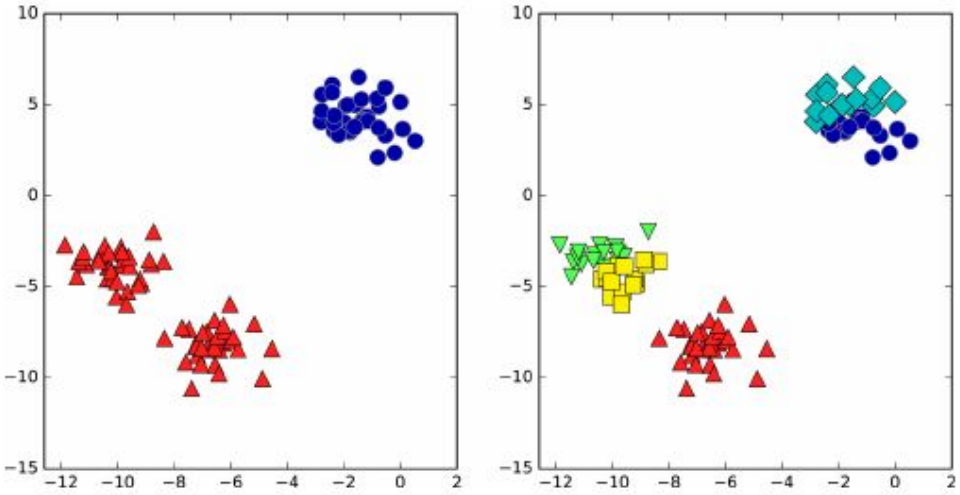


Figure 3-26. Cluster assignments found by k-means using two clusters (left) and five clusters (right)

[Muller and Guido, Introduction to Machine Learning with Python]

K-means Practical Details

- How many clusters are there?

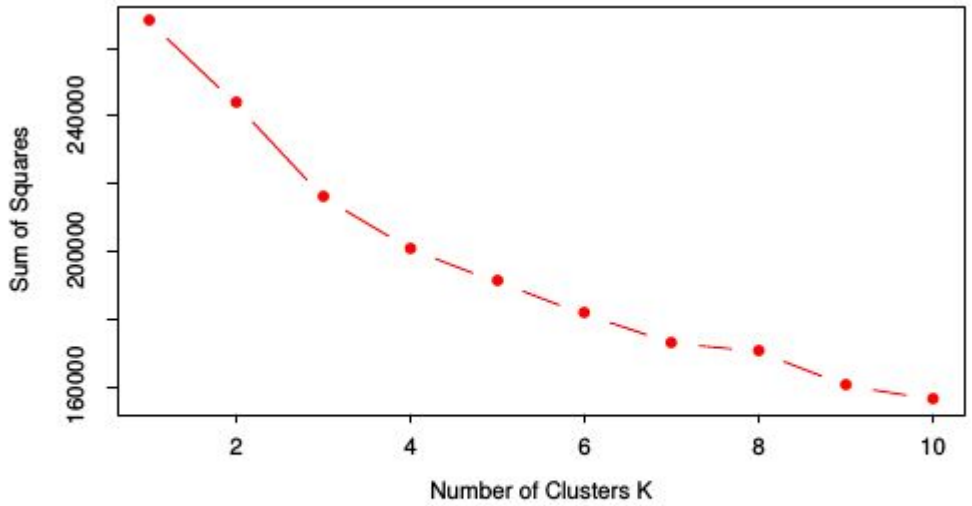


FIGURE 14.8. *Total within-cluster sum of squares for K-means clustering applied to the human tumor microarray data.*

[Hastie et al., The Elements of Statistical Learning]

K-means Practical Details

- All dimensions are treated equally

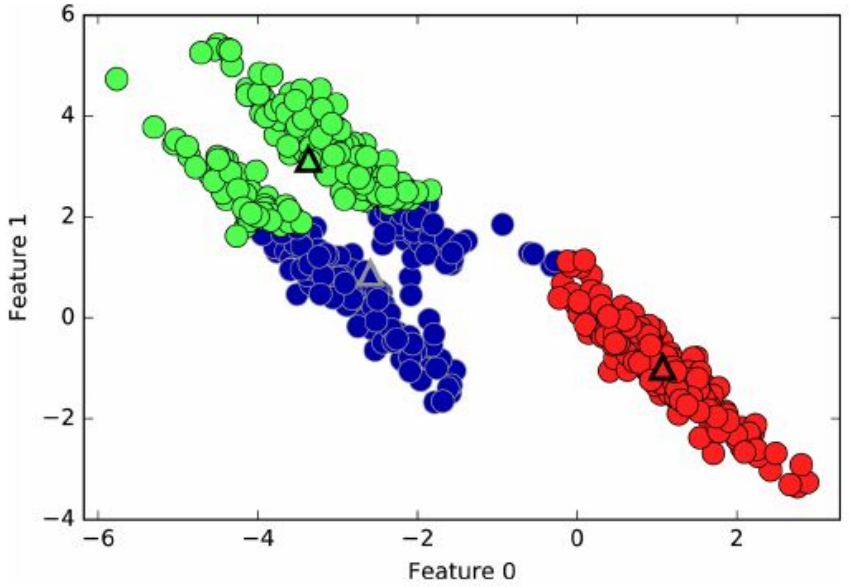


Figure 3-28. k-means fails to identify nonspherical clusters

[Muller and Guido, Introduction to Machine Learning with Python]

K-means Practical Details

- All dimensions are treated equally
- … but standardization can also hurt

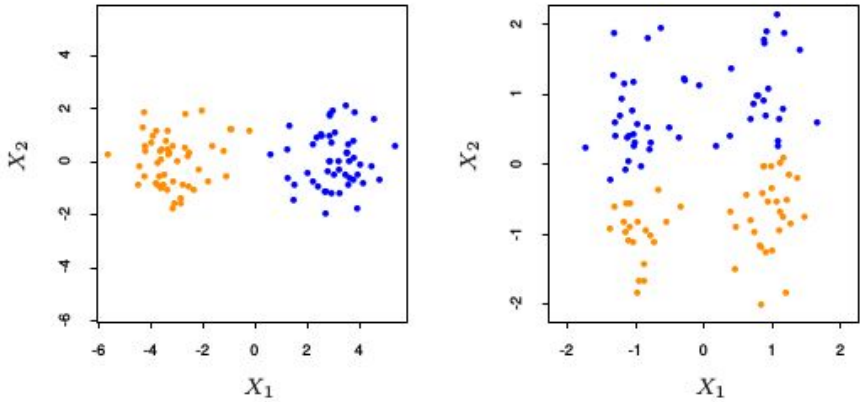


FIGURE 14.5. *Simulated data: on the left, K-means clustering (with K=2) has been applied to the raw data. The two colors indicate the cluster memberships. On the right, the features were first standardized before clustering. This is equivalent to using feature weights* $1/[2 \cdot \mathrm{var}(X_j)]$. *The standardization has obscured the two well-separated groups. Note that each plot uses the same units in the horizontal and vertical axes.*

[Hastie et al., The Elements of Statistical Learning]

K-means Practical Details

- All dimensions are treated equally
- … but standardization can also hurt
- Complex shapes are hard to disentangle

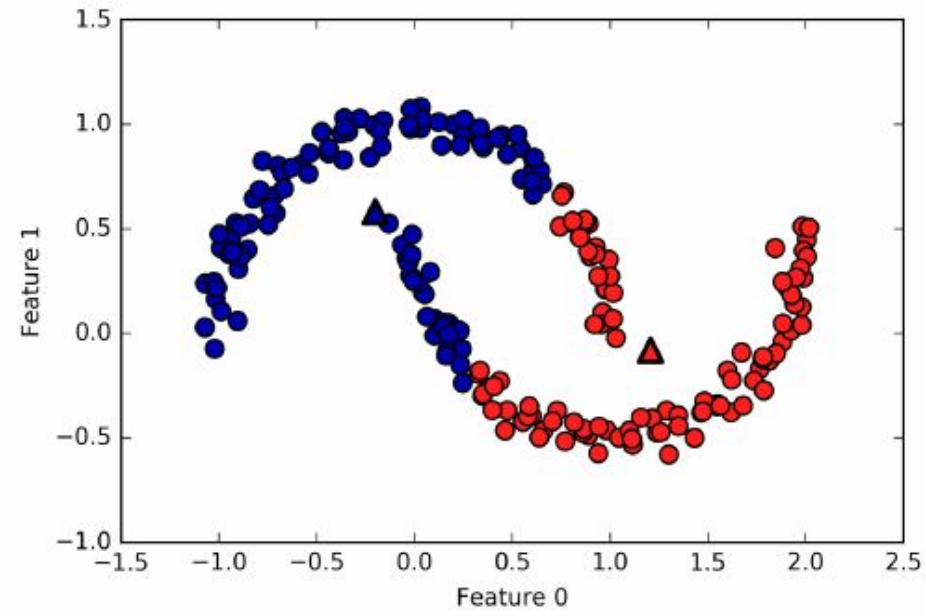
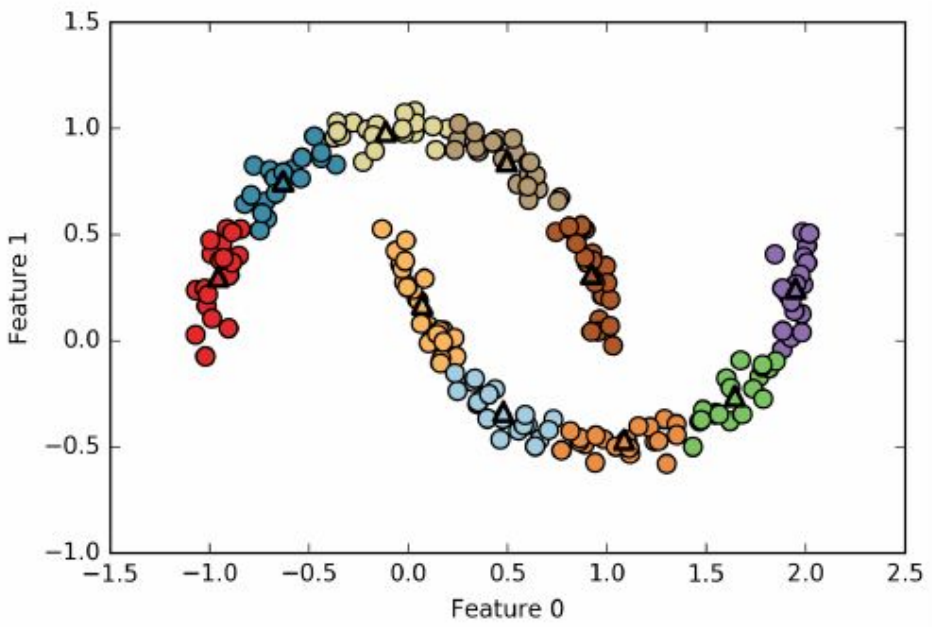


Figure 3-29. k-means fails to identify clusters with complex shapes

[Muller and Guido, Introduction to Machine Learning with Python]

K-means Practical Details

- All dimensions are treated equally
- … but standardization can also hurt
- Complex shapes are hard to disentangle
- … though more clusters can help



[Muller and Guido, Introduction to Machine Learning with Python]

K-means: Different Distance Metrics

- Main K-means loop:
  - `Repeat:`
    - `Compute the mean of each cluster`
    - `Re-assign each point to the cluster with the closest mean`
  - `Until no point has changed cluster assignment in last iteration`

K-means: Different Distance Metrics

- Main K-means loop:
  - `Repeat:`
    - `For each cluster, choose the best representative`
    - `Re-assign each point to closest cluster representative`
  - `Until no point has changed cluster assignment in last iteration`

K-means: Different Distance Metrics

- Main K-means loop:
  - `Repeat:`
    - `For each cluster, choose the best representative`
    - `Re-assign each point to closest cluster representative`
  - `Until no point has changed cluster assignment in last iteration`
- Best representative: minimize distance to all other points

$$\min_{r_k}\left(\sum_{C(i)=k} d(x_i, r_k)\right)$$

K-means: Different Distance Metrics

- Main K-means loop:
  - `Repeat:`
    - `For each cluster, choose the best representative`
    - `Re-assign each point to closest cluster representative`
  - `Until no point has changed cluster assignment in last iteration`
- Best representative: minimize distance to all other points

$$\min_{r_k}\Big(\sum_{C(i)=k} d(x_i, r_k)\Big)$$

- For Euclidean distance: we can prove that the optimal r_k is the mean m_k
  - computable in linear time

$$\sum_{C(i)=k}\sum_{C(j)=k}\|x_i - x_j\|^2 = \sum_{C(i)=k}\|x_i - m_k\|^2$$

K-means: Different Distance Metrics

- Main K-means loop:
  - `Repeat:`
    - `For each cluster, choose the best representative`
    - `Re-assign each point to closest cluster representative`
  - `Until no point has changed cluster assignment in last iteration`
- Best representative: minimize distance to all other points

$$r_k = \min_{j}\Big(\sum_{C(i)=k} d(x_i, x_j) \Big)$$

- For any distance: choose the point in the cluster that does best
  - not necessarily optimal
  - takes quadratic time to compute

Hierarchical Clustering

- Agglomerative (bottom-up)
  - Each point starts as its own cluster
  - Repeat:
    - Merge most “similar” clusters
  - Until desired number of cluster is reached, or no two clusters are “similar enough”
- Divisive (top-down)
  - All points start as one big cluster
  - Repeat:
    - Divide one cluster to produce two new cluster as “dissimilar” as possible
  - Until desired number of cluster is reached, or we can not produce two new clusters that are “dissimilar enough”

Agglomerative Clustering

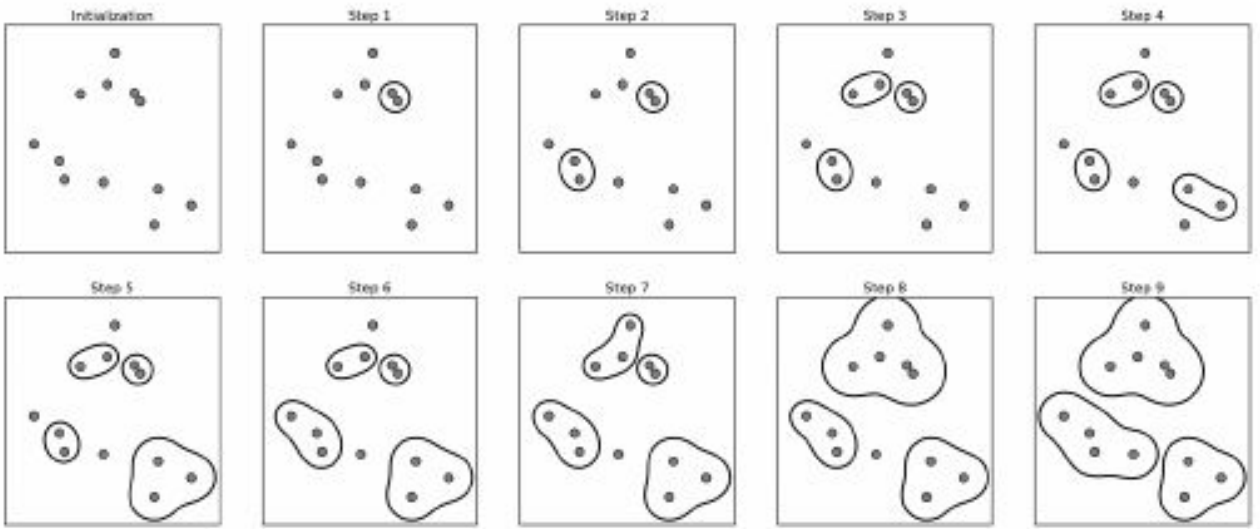


Figure 3-33. Agglomerative clustering iteratively joins the two closest clusters

[Muller and Guido, Introduction to Machine Learning with Python]

Agglomerative Clustering

- What does it mean for two clusters to be “similar”?
  - closest neighbor: $d(G,H) = \min_{\substack{i \subset G \\ j \subset H}} d(\mathbf{x}_i, \mathbf{x}_j)$
  - furthest neighbor: $d(G,H) = \max_{\substack{i \subset G \\ j \subset H}} d(\mathbf{x}_i, \mathbf{x}_j)$
  - average: $d(G,H) = 1/(N_G N_H) \sum_{i \subset G} \sum_{j \subset H} d(\mathbf{x}_i, \mathbf{x}_j)$

Agglomerative Clustering

- Implementations
  - scikit-learn agglomerative clustering
  - scipy cluster vq
- Interfaces
  - there is no `.predict(x)` - why?

Evaluating Clustering with Ground Truth

- Exact labels do not matter (why?)
- What matters is if points that should be together are indeed together
- Example evaluator: scikit-learn adjusted rand index

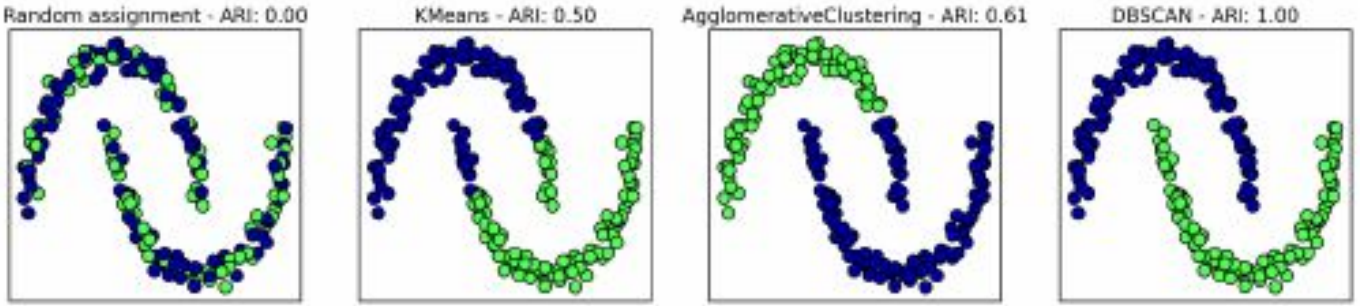


Figure 3-39. Comparing random assignment, k-means, agglomerative clustering, and DBSCAN on the two_moons dataset using the supervised ARI score

[Muller and Guido, Introduction to Machine Learning with Python]

(More) Advanced Topics

Vector Quantization

- Replace each data point with the representative of the cluster it belongs to
- Original:
  - N d-dimensional vectors: $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}] \subset \Re^d$
  - K clusters
  - Total space: Nd
- Quantized data:
  - K d-dimensional vectors (dictionary of cluster representatives)
    - $\mathbf{r}_i = [r_{i,1}, r_{i,2}, \ldots, r_{i,d}] \subset \Re^d$
  - N numbers, or K-dimensional vectors
    - $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}] \rightarrow k \subset \{1..K\}$
    - $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}] \rightarrow [0, 0, 1, \ldots., 0] \subset \Re^K$

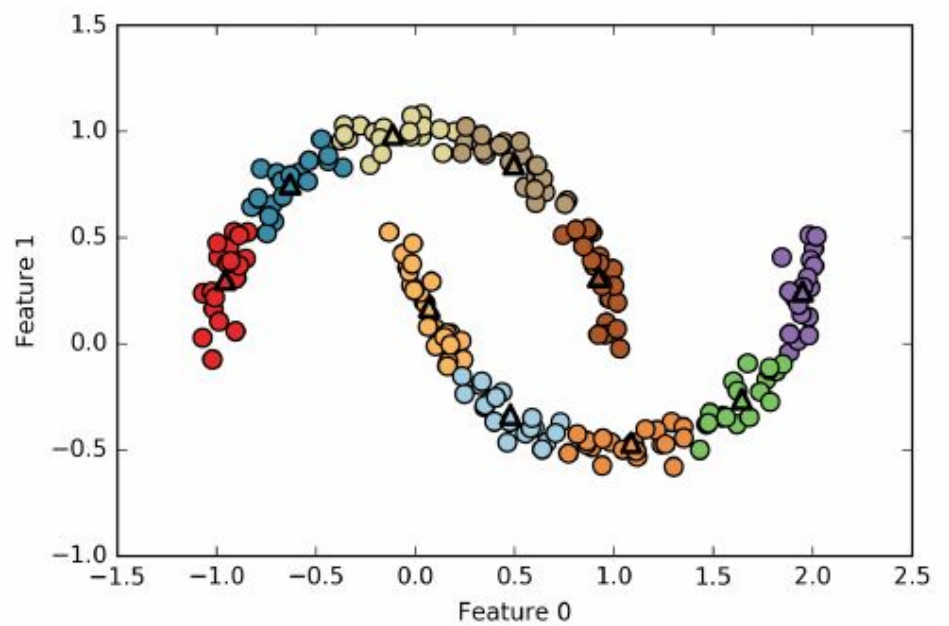
Vector Quantization - Compression

FIGURE 14.9. *Sir Ronald A. Fisher (1890 – 1962) was one of the founders of modern day statistics, to whom we owe maximum-likelihood, sufficiency, and many other fundamental concepts. The image on the left is a 1024×1024 grayscale image at 8 bits per pixel. The center image is the result of 2 × 2 block VQ, using 200 code vectors, with a compression rate of 1.9 bits/pixel. The right image uses only four code vectors, with a compression rate of 0.50 bits/pixel*

[Hastie et al., The Elements of Statistical Learning]

Vector Quantization - Projection to Higher Dims.

- $\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,d}] \rightarrow [0, 0, 1, \ldots., 0] \subset \Re^K$



[Muller and Guido, Introduction to Machine Learning with Python]

K-means and Expectation-Maximization (EM)

- Main K-means loop:
  - `Start: initial (random) cluster assignments`
  - `Repeat:`
    - `For each cluster, choose the best representative`
    - `Re-assign each point to closest cluster representative`
  - `Until no point has changed cluster assignment in last iteration`

- Cluster assignments (responsibilities) are binary, all-or-nothing

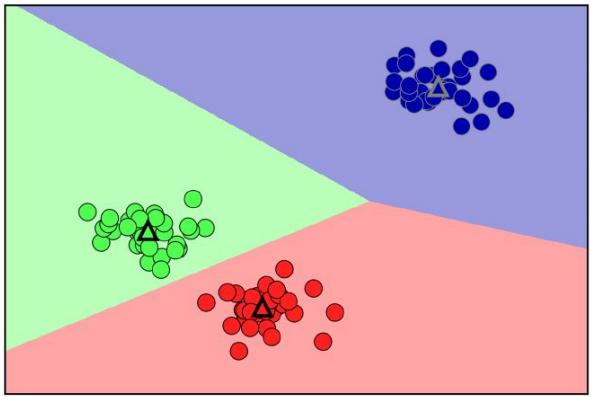
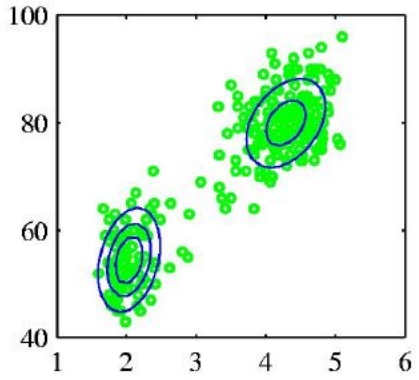


Figure 3-24. Cluster centers and cluster boundaries found by the k-means algorithm

[Muller and Guido, Introduction to Machine Learning with Python]

K-means and Expectation-Maximization (EM)

- What if…
  - Cluster responsibility decreases with distance from representative, but in smooth continuous fashion?
  - E.g.: cluster responsibility modeled as a Gaussian, with mean at the representative, and covariance based on size?



Mixture of two Gaussians

K-means and Expectation-Maximization (EM)

- Main K-means loop:
  - Start: initial (random) cluster assignments
  - Repeat:
    - For each cluster, choose the best representative
    - Re-assign each point to closest cluster representative
  - Until no point has changed cluster assignment in last iteration

- Soft K-means loop:
  - Start: initial (random) clusters, each defined by parameters mean and covariance
  - Repeat:
    - Compute each cluster's responsibility for each point
    - Re-estimate each cluster's parameters (mean and covariance) given computed responsibilities
  - Until we can no longer improve the fit

K-means and Expectation-Maximization (EM)

- Main K-means loop:
  - Start: initial (random) cluster assignments
  - Repeat:
    - For each cluster, choose the best representative
    - Re-assign each point to closest cluster representative
  - Until no point has changed cluster assignment in last iteration

- Soft K-means loop:
  - Start: initial (random) clusters, each defined by parameters mean and covariance
  - Repeat:
    - Compute each cluster's responsibility for each point
    - Re-estimate each cluster's parameters (mean and covariance) given computed responsibilities
  - Until we can no longer improve the fit

K-means and Expectation-Maximization (EM)

- Main K-means loop:
  - Start: initial (random) cluster assignments
  - Repeat:
    - For each cluster, choose the best representative
    - Re-assign each point to closest cluster representative
  - Until no point has changed cluster assignment in last iteration

- Soft K-means loop:
  - Start: initial (random) clusters, each defined by parameters mean and covariance
  - Repeat:
    - EXPECTATION: Compute each cluster's responsibility for each point
    - MAXIMIZATION: Re-estimate each cluster's parameters (mean and covariance) given computed responsibilities
  - Until we can no longer improve the fit

Robotics - What Are We Clustering?

- Hand poses
- Trajectories
- 3D point clouds
- ...

Robotics - What Are We Clusteri

- Hand poses
- Trajectories
- 3D point clouds
- ...

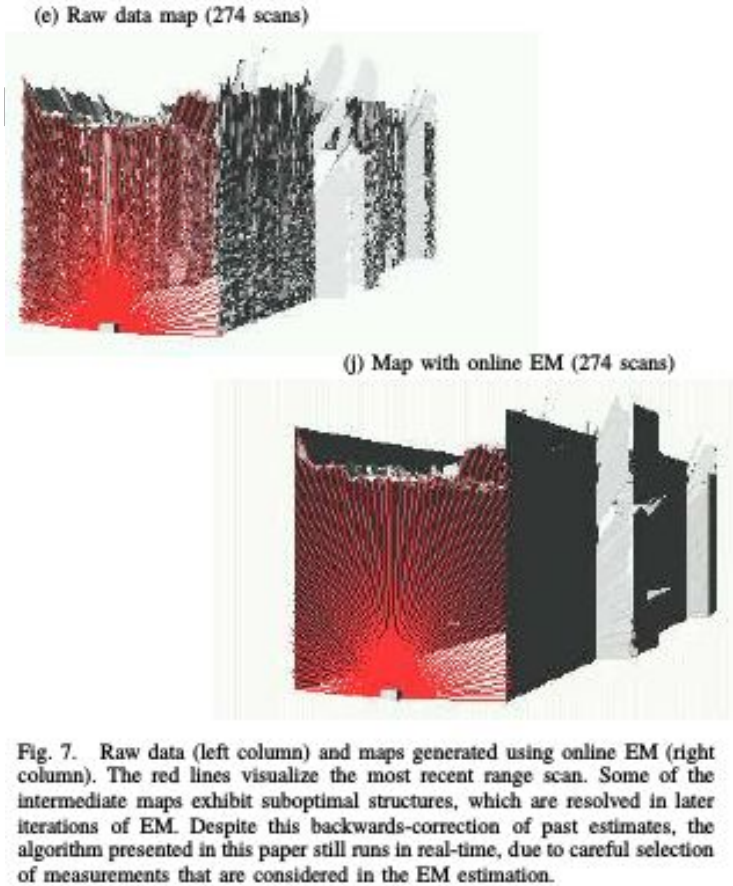


Fig. 7. Raw data (left column) and maps generated using online EM (right column). The red lines visualize the most recent range scan. Some of the intermediate maps exhibit suboptimal structures, which are resolved in later iterations of EM. Despite this backwards-correction of past estimates, the algorithm presented in this paper still runs in real-time, due to careful selection of measurements that are considered in the EM estimation.

[Thrun et al., A Real-Time Expectation Maximization Algorithm for Acquiring Multi-Planar Maps of Indoor Environments with Mobile Robots]