Unsupervised Learning: k-Means and Mixtures of Gaussians

CS6780 – Advanced Machine Learning
Spring 2015

Thorsten Joachims
Cornell University

Reading: Murphy 11.1 – 11.4.2

Supervised Learning vs. Unsupervised Learning

- Supervised Learning
  - Classification: partition examples into groups according to pre-defined categories
  - Regression: assign value to feature vectors
  - Requires labeled data for training
- Unsupervised Learning
  - Clustering: partition examples into groups when no pre-defined categories/classes are available
  - Outlier detection: find unusual events (e.g. hackers)
  - Novelty detection: find changes in data
  - Only instances required, but no labels

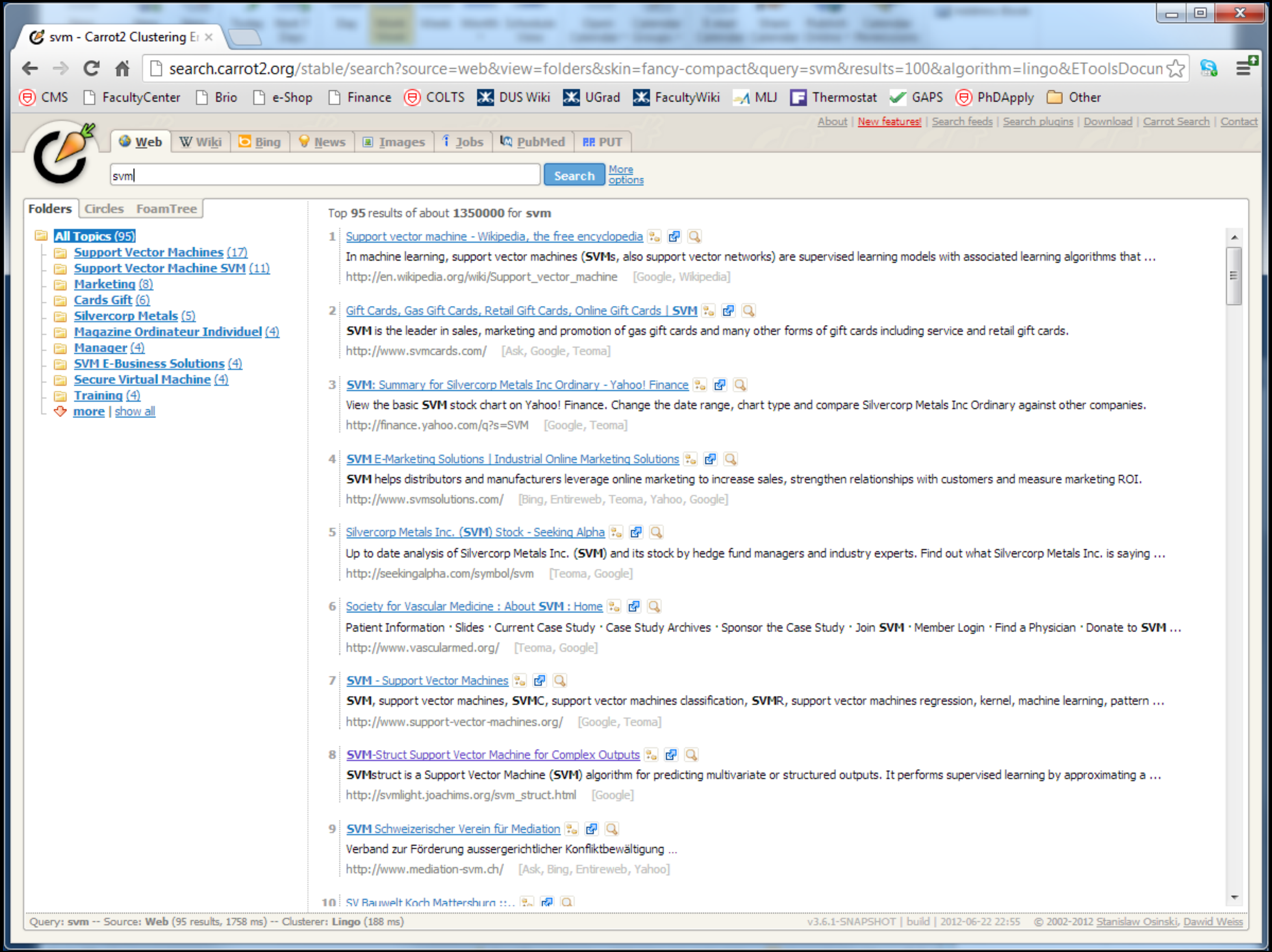
Clustering

- Partition unlabeled examples into disjoint subsets of *clusters*, such that:
  - Examples within a cluster are similar
  - Examples in different clusters are different
- Discover new categories in an *unsupervised* manner (no sample category labels provided).

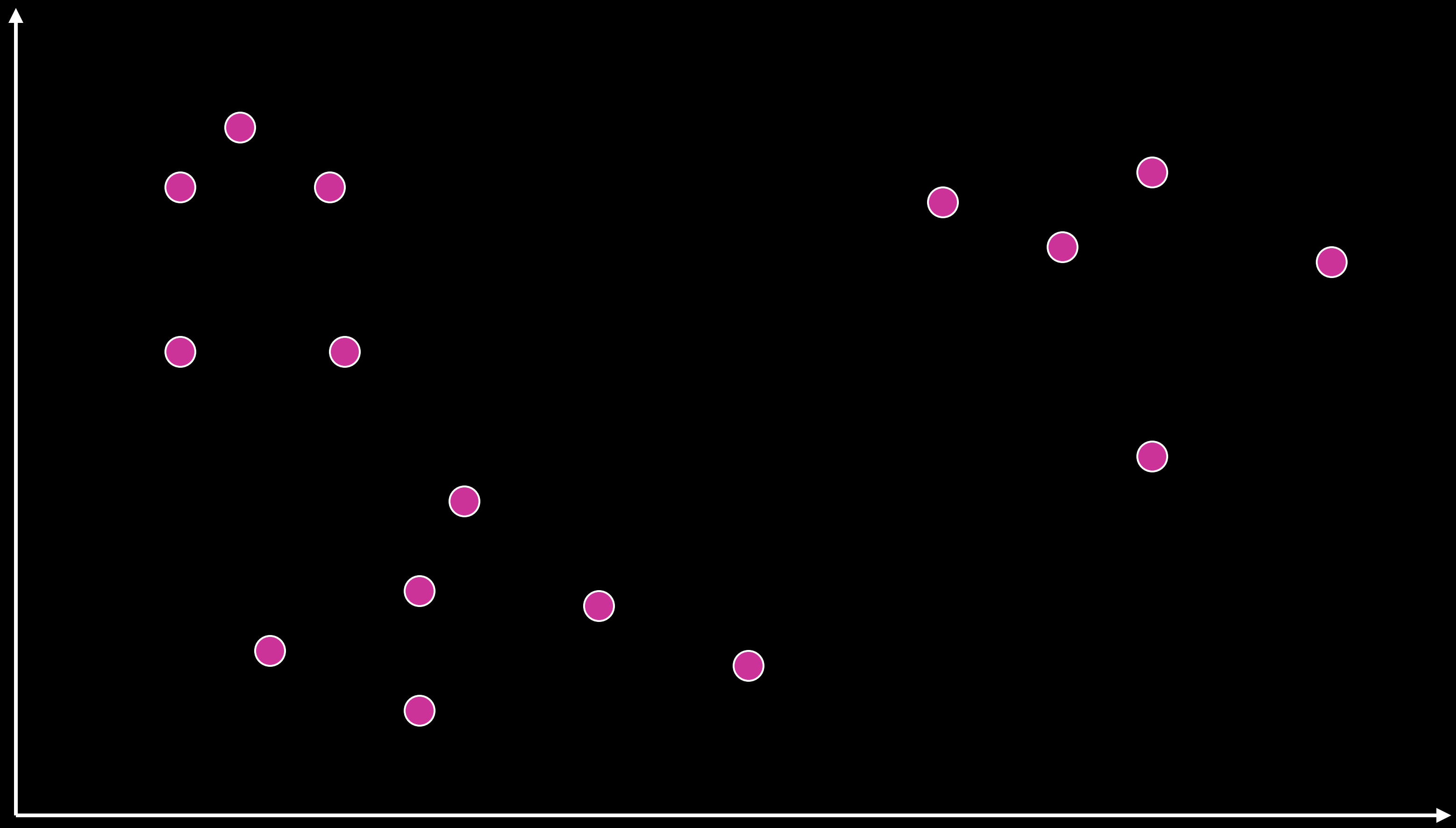
Applications of Clustering

- Exploratory data analysis
- Cluster retrieved documents
  - to present more organized and understandable results to user → "diversified retrieval"
- Detecting near duplicates
  - Entity resolution
    - E.g. "Thorsten Joachims" == "Thorsten B Joachims"
  - Cheating detection
- Automated (or semi-automated) creation of taxonomies
  - e.g. Yahoo, DMOZ
- Compression

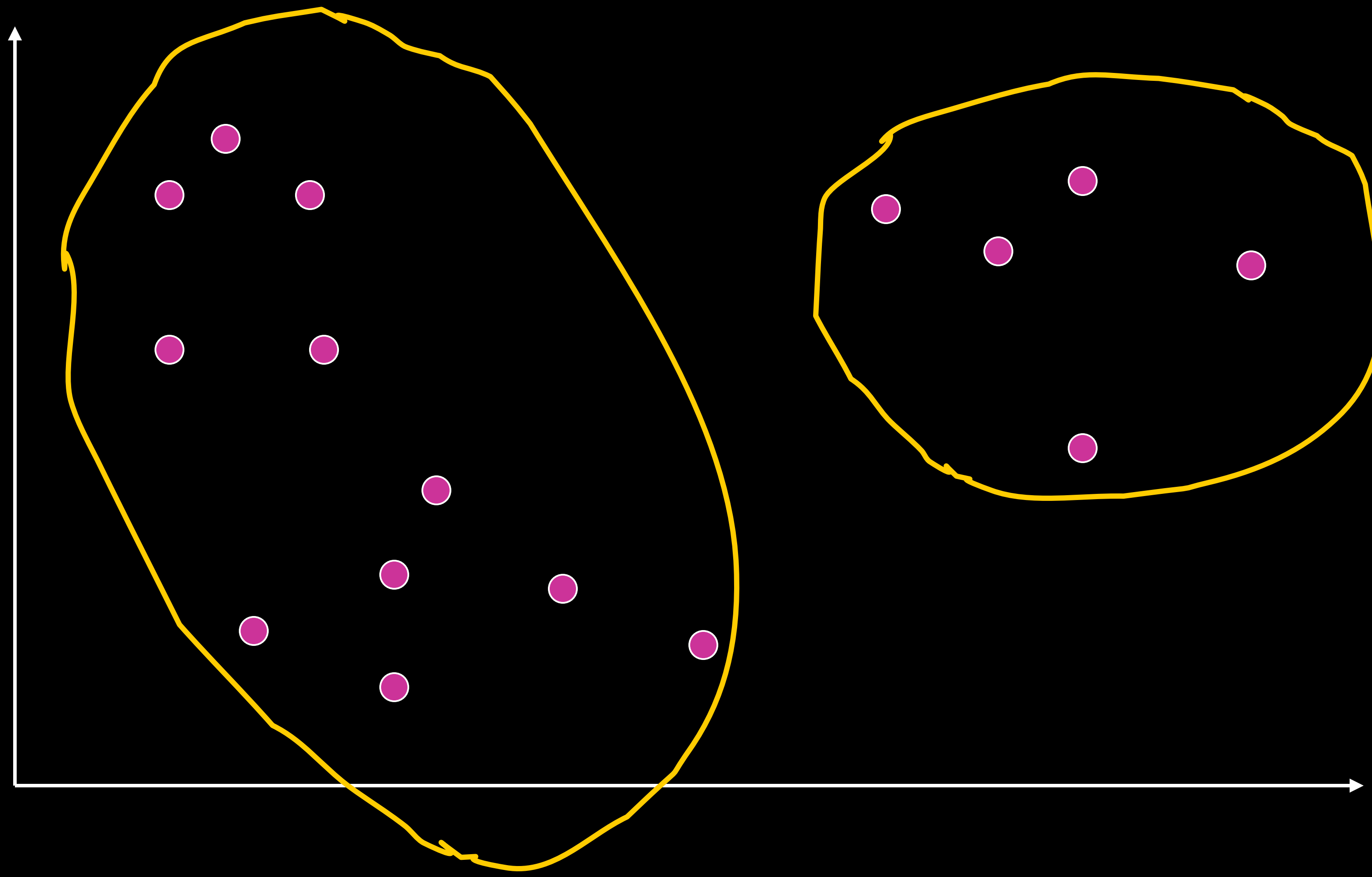
Applications of Clustering



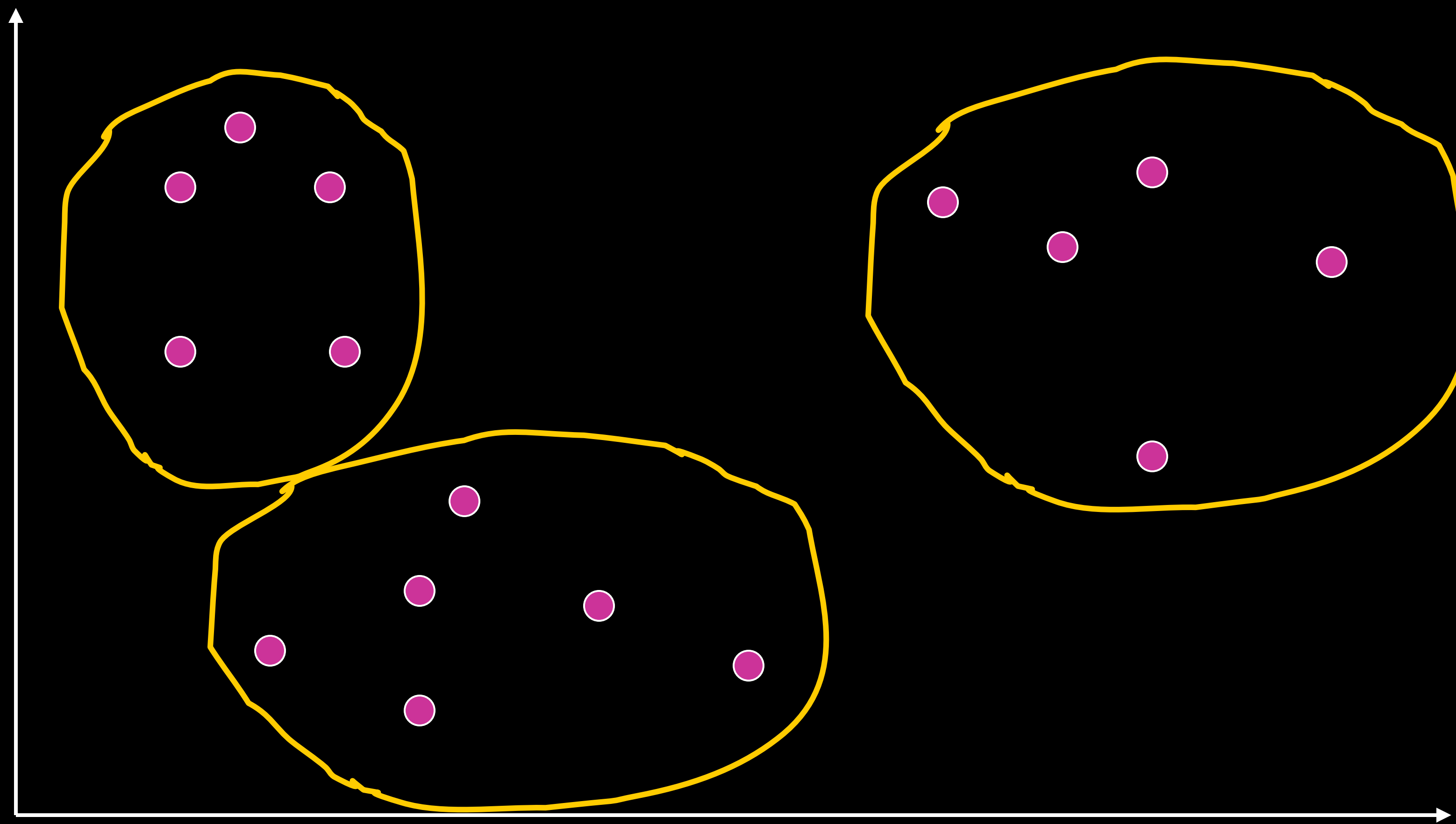
Clustering Example



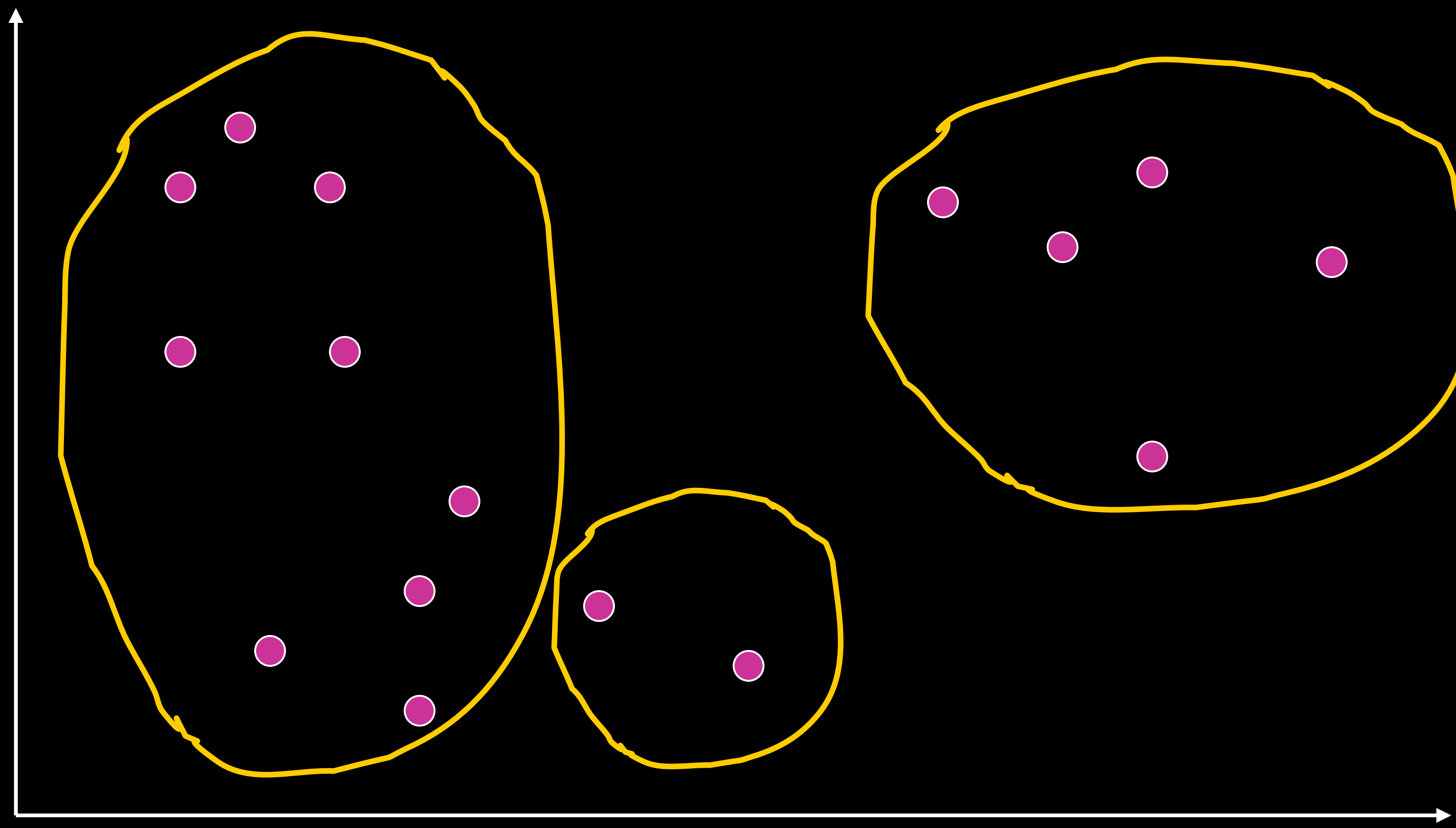
Clustering Example



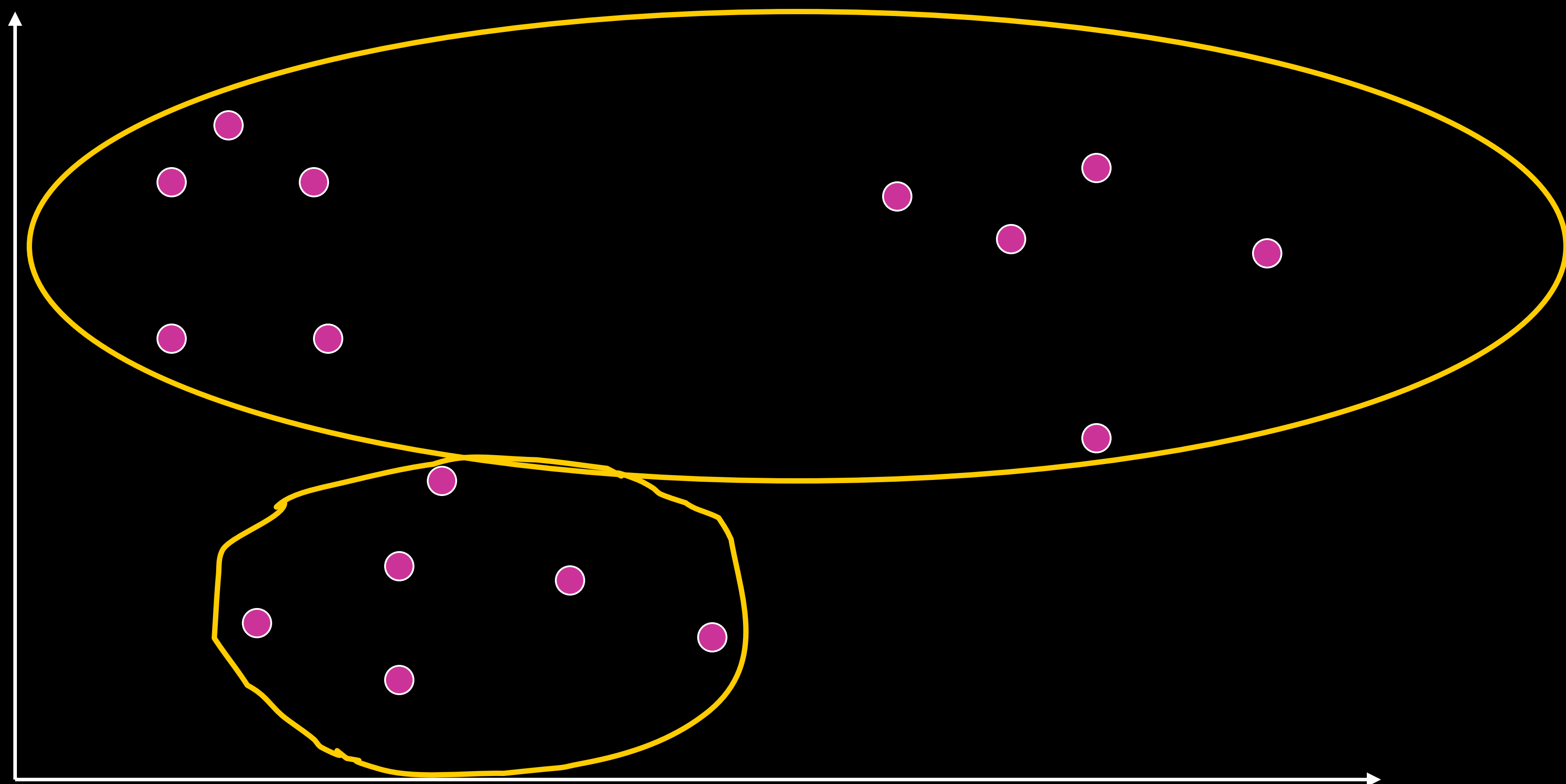
Clustering Example



Clustering Example



Clustering Example



Clustering Criterion

- Evaluation function that assigns a (usually real-valued) value to a clustering
  - Clustering criterion typically function of
    - within-cluster similarity and
    - between-cluster dissimilarity
- Optimization
  - Find clustering that maximizes the criterion
    - Global optimization (often intractable)
    - Greedy search
    - Approximation algorithms

Similarity (Distance) Measures

- Euclidian distance (L_2 norm):

$$L_2(\vec{x}, \vec{x}') = \sqrt{\sum_{i=1}^{N} \left(x_i - x_i'\right)^2}$$

- L_1 norm:

$$L_1(\vec{x}, \vec{x}') = \sqrt{\sum_{i=1}^{N} \left|x_i - x_i'\right|}$$

- Cosine similarity:

$$\cos(\vec{x}, \vec{x}') = \frac{\vec{x} * \vec{x}'}{\|\vec{x}\| \, \|\vec{x}'\|}$$

- Kernels

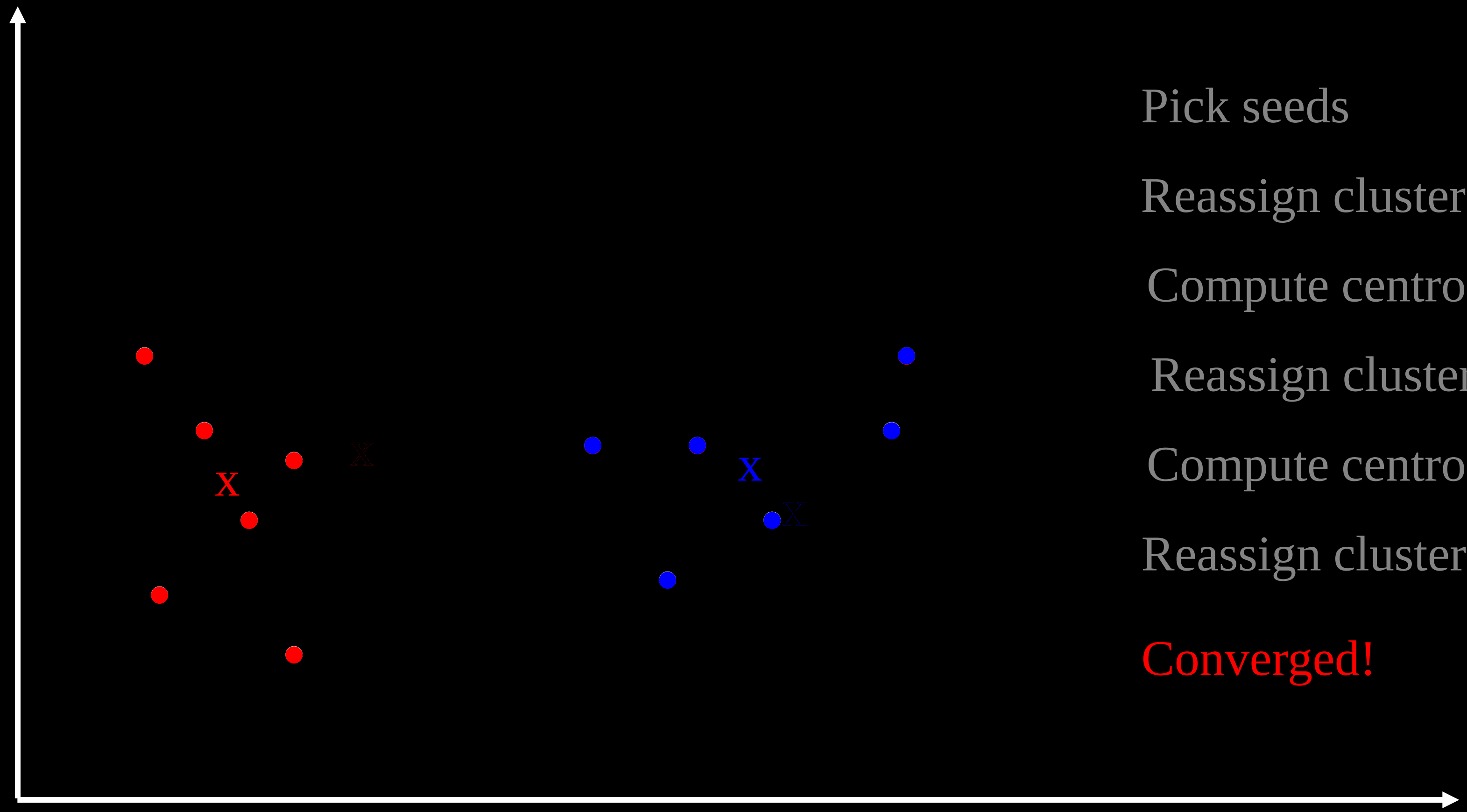
K-Means Algorithm

- Input: k = number of clusters, distance measure d
- Select k random instances $\{s_1, s_2, \ldots s_k\}$ as seeds.
- Until clustering converges or other stopping criterion:
  - For each instance x_i:
    - Assign x_i to the cluster c_j such that $d(x_i, s_j)$ is min.
  - For each cluster c_j *//update the centroid of each cluster*
    - $s_j = \mu(c_j)$

Note: Clusters represented via ***centroids***

$$\vec{\mu}(c) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

K-means Example
(k=2)



Pick seeds

Reassign clusters

Compute centroids

Reassign clusters

Compute centroids

Reassign clusters

Converged!

Time Complexity

- Assume computing distance between two instances is O(N) where N is the dimensionality of the vectors.
- Reassigning clusters for n points: O(kn) distance computations, or O(knN).
- Computing centroids: Each instance gets added once to some centroid: O(nN).
- Assume these two steps are each done once for i iterations: O($iknN$).
- Linear in all relevant factors, assuming a fixed number of iterations.

Buckshot Algorithm

Problem

- Results can vary based on random seed selection, especially for high-dimensional data.
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.

Idea: Combine HAC and K-means clustering.

- First randomly take a sample of instances of size $n^{1/2}$
- Run group-average HAC on this sample
- Use the results of HAC as initial seeds for K-means.
- Overall algorithm is efficient and avoids problems of bad seed selection.

Non-Hierarchical Clustering

- K-means clustering (“hard”)
- Mixtures of Gaussians and training via Expectation maximization Algorithm (“soft”)

Clustering as Prediction

- Setup
  - Learning Task: $P(X)$
  - Training Sample: $S = (\vec{x}_1, \ldots, \vec{x}_n)$
  - Hypothesis Space: $H = \{h_1, \ldots, h_{|H|}\}$ each describes $P(X|h_i)$ where h_i are parameters
  - Goal: learn which $P(X|h_i)$ produces the data
- What to predict?
  - Predict where new points are going to fall

Gaussian Mixtures and EM

- Gaussian Mixture Models
  - Assume

$$P(X = \vec{x}|h_i) = \sum_{j=1}^{k} P(X = \vec{x}|Y = j, h_i)P(Y = j)$$

where $P(X = \vec{x}|Y = j, h) = N\left(X = \vec{x}\middle|\vec{\mu}_j, \Sigma_j\right) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{1}{2\sigma_{ij}^2}(x-\mu_{ij})^2}$

and $h = (\vec{\mu}_1, \dots, \vec{\mu}_k, \Sigma_1, \dots, \Sigma_k)$.

- EM Algorithm
  - Assume $P(Y)$ and k known and $\Sigma_i = 1$.
  - REPEAT
    - $\vec{\mu}_j = \frac{\sum_{i=1}^{n} P\left(Y=j\middle|X=\vec{x}_i,\vec{\mu}_1,\dots,\vec{\mu}_k\right)\vec{x}_i}{\sum_{i=1}^{n} P(Y=j|X=\vec{x}_i,\vec{\mu}_1,\dots,\vec{\mu}_k)}$
    - $P(Y = j|X = \vec{x}_i, \vec{\mu}_1, \dots, \vec{\mu}_k) = \frac{P(X=\vec{x}_i|Y=j,\vec{\mu}_j)P(Y=j)}{\sum_{l=1}^{k} P(X=\vec{x}_i|Y=l,\vec{\mu}_l)P(Y=l)} = \frac{\mathrm{e}^{-0.5\left(\vec{x}_i-\vec{\mu}_j\right)^2} P(Y=j)}{\sum_{l=1}^{k} e^{-0.5\left(\vec{x}_i-\vec{\mu}_l\right)^2} P(Y=l)}$