Introduction to Machine Learning and Data Mining Lecture-6: Unsupervised Learning

Prof. Eugene Chang

Today

- Teaching plan update
- Clustering
- Part of slides are based on materials from Prof. Andrew Rosenberg, CUNY, and Prof Jiawei Han, University of Illinois at Urbana-Champaign

Teaching Plan Update

Week	Topic	Reading/Homework
1 (May 15)	Introduction, Python Tutorial	Ch. 1, Ch. 2
2	Math Primer, Python Tools	Ch. 3, Class Notes
3	Decision Trees, SK-learn, Regression	Ch. 12 (HW-1)
4	Bayes Rules, Reviews	Ch. 4
5	Naïve Bayes, Nearest Neighbors	Ch. 7 (HW-2)
6 (Jun 19)	Clustering, Gaussian Mixture Model	Ch. 7, Ch. 14 (HW-3)
7	Model Evaluation, Project Topics	Class Notes
8	HW Solutions, Midterm Review, Project Discussion	Class Notes
9 (July 10)	***Mid-Term Exam***	
10	Support Vector Machine, Project signup	Ch. 8 (HW-4)
11	Ensemble Methods, Project Proposal	Ch. 13
12	Data Mining, Project Updates	Class Notes (HW-5)
13	Data Mining, Project Updates	Class Notes
14	Final review, ***Project Presentation***	Class Notes
15 (Aug 21)	***Final Exam***	

Will release on 06/21

Grades

- Mid-term: 25%
- Final: 30%
- Homework: 20%
- Project: 20% (but scores could vary from 0 – infinity)
- Quiz & attendance: 5%

Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering
- Summary

What is Cluster Analysis?

- Cluster: A collection of data objects
  - similar (or related) to one another within the same group
  - dissimilar (or unrelated) to the objects in other groups
- Cluster analysis (or *clustering*, *data segmentation, …*)
  - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes (i.e., *learning by observations* vs. learning by examples: supervised)
- Typical applications
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithms

Applications of Cluster Analysis

- Data reduction
  - Summarization: Preprocessing for regression, PCA, classification, and association analysis
  - Compression: Image processing: vector quantization
- Hypothesis generation and testing
- Prediction based on groups
  - Cluster & find characteristics/patterns for each group
- Finding K-nearest Neighbors
  - Localizing search to one or a small number of clusters
- Outlier detection: Outliers are often viewed as those “far away” from any cluster

Clustering: Application Examples

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- City-planning: Identifying groups of houses according to their house type, value, and geographical location
- Earth-quake studies: Observed earth quake epicenters should be clustered along continent faults
- Climate: understanding earth climate, find patterns of atmospheric and ocean
- Economic Science: market research

Basic Steps to Develop a Clustering Task

- Feature selection
  - Select info concerning the task of interest
  - Minimal information redundancy
- Proximity measure
  - Similarity of two feature vectors
- Clustering criterion
  - Expressed via a cost function or some rules
- Clustering algorithms
  - Choice of algorithms
- Validation of the results
  - Validation test (also, *clustering tendency* test)
- Interpretation of the results
  - Integration with applications

Quality: What Is Good Clustering?

- A good clustering method will produce high quality clusters
  - high intra-class similarity: cohesive within clusters
  - low inter-class similarity: distinctive between clusters
- The quality of a clustering method depends on
  - the similarity measure used by the method
  - its implementation, and
  - Its ability to discover some or all of the hidden patterns

Measure the Quality of Clustering

- Dissimilarity/Similarity metric
  - Similarity is expressed in terms of a distance function, typically metric: $d(i, j)$
  - The definitions of distance functions are usually rather different for interval-scaled, boolean, categorical, ordinal ratio, and vector variables
  - Weights should be associated with different variables based on applications and data semantics
- Quality of clustering:
  - There is usually a separate "quality" function that measures the "goodness" of a cluster.
  - It is hard to define "similar enough" or "good enough"
    - The answer is typically highly subjective

Considerations for Cluster Analysis

- Partitioning criteria
  - Single level vs. hierarchical partitioning (often, multi-level hierarchical partitioning is desirable)
- Separation of clusters
  - Exclusive (e.g., one customer belongs to only one region) vs. non-exclusive (e.g., one document may belong to more than one class)
- Similarity measure
  - Distance-based (e.g., Euclidian, road network, vector) vs. connectivity-based (e.g., density or contiguity)
- Clustering space
  - Full space (often when low dimensional) vs. subspaces (often in high-dimensional clustering)

Requirements and Challenges

- Scalability
  - Clustering all the data instead of only on samples
- Ability to deal with different types of attributes
  - Numerical, binary, categorical, ordinal, linked, and mixture of these
- Constraint-based clustering
  - User may give inputs on constraints
  - Use domain knowledge to determine input parameters
- Interpretability and usability
- Others
  - Discovery of clusters with arbitrary shape
  - Ability to deal with noisy data
  - Incremental clustering and insensitivity to input order
  - High dimensionality

Major Clustering Approaches

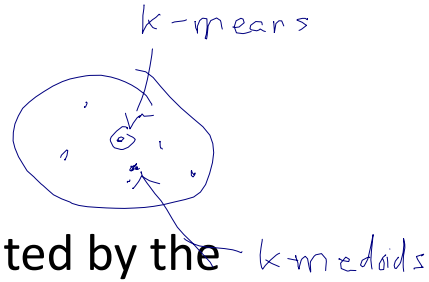
- Partitioning approach:
  - Construct various partitions and then evaluate them by some criterion, e.g., minimizing the sum of square errors
  - Typical methods: k-means, k-medoids, CLARANS
- Hierarchical approach:
  - Create a hierarchical decomposition of the set of data (or objects) using some criterion
  - Typical methods: Diana, Agnes, BIRCH, CAMELEON
- Density-based approach:
  - Based on connectivity and density functions
  - Typical methods: DBSACN, OPTICS, DenClue
- Grid-based approach:
  - based on a multiple-level granularity structure
  - Typical methods: STING, WaveCluster, CLIQUE

Major Clustering Approaches

- Model-based:
  - A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other
  - Typical methods: EM, SOM, COBWEB
- Frequent pattern-based:
  - Based on the analysis of frequent patterns
  - Typical methods: p-Cluster
- User-guided or constraint-based:
  - Clustering by considering user-specified or application-specific constraints
  - Typical methods: COD (obstacles), constrained clustering
- Link-based clustering:
  - Objects are often linked together in various ways
  - Massive links can be used to cluster objects: SimRank, LinkClus

Partitioning Algorithms: Basic Concept

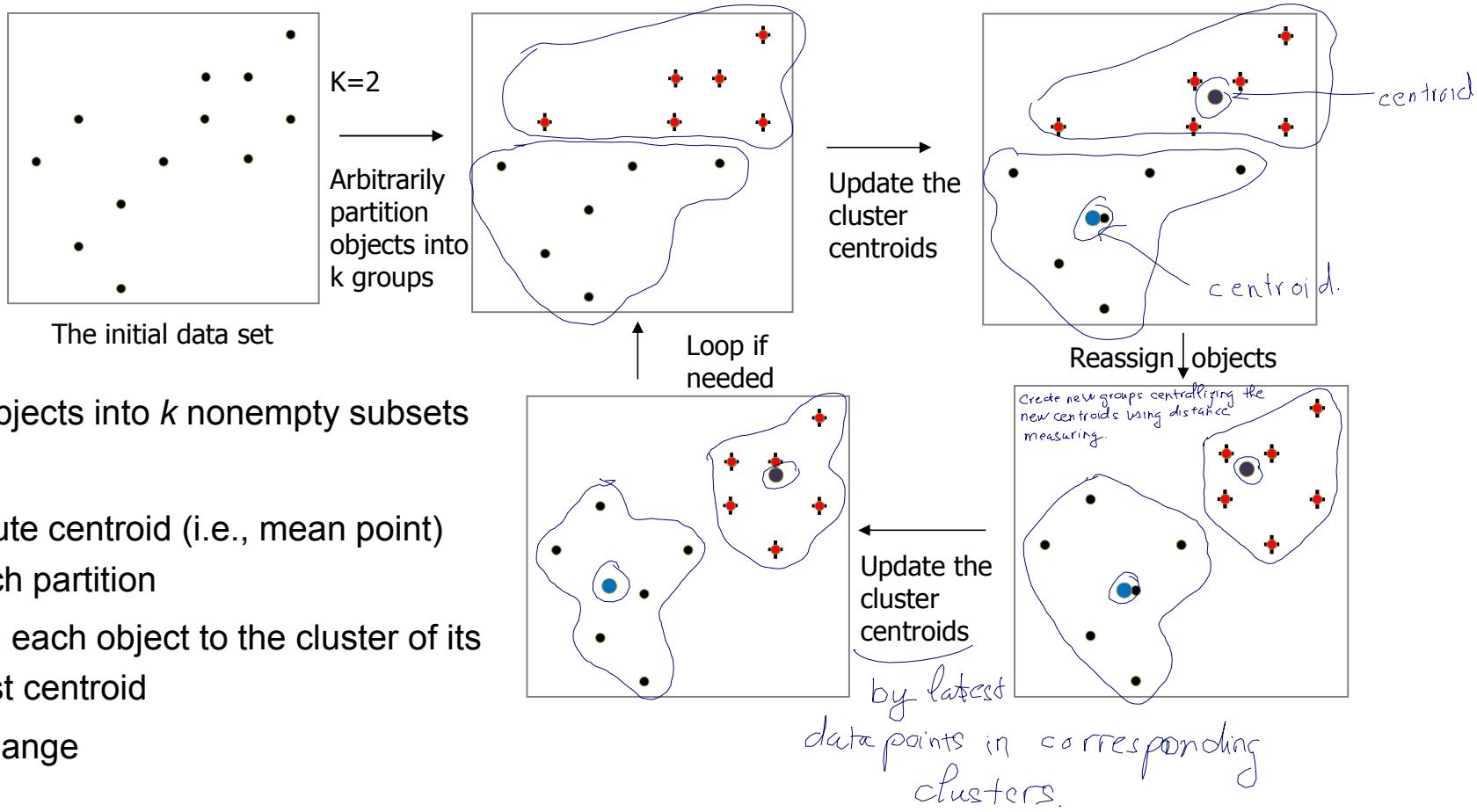
- Partitioning method: Partitioning a database ***D*** of ***n*** objects into a set of ***k*** clusters, such that the sum of squared distances is minimized (where c_i is the centroid or medoid of cluster C_i) $E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (d(p, c_i))^2$
- Given *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67, Lloyd'57/'82): Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster



The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  - Partition objects into *k* nonempty subsets
  - Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
  - Assign each object to the cluster with the nearest seed point
  - Go back to Step 2, stop when the assignment does not change

An Example of *K-Means* Clustering



- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

Comments on the *K-Means* Method

- Strength: *Efficient*: *O*(*tkn*), where *n* is # objects, *k* is # clusters, and *t* is # iterations. Normally, *k*, *t* << *n*.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Weakness
  - Often terminates at a *local optimal*
  - Applicable only to objects in a continuous n-dimensional space
    - Using the k-modes method for categorical data
    - In comparison, k-medoids can be applied to a wide range of data
  - Need to specify *k,* the *number* of clusters, in advance (there are ways to automatically determine the best k (see Hastie et al., 2009)
  - Sensitive to noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

does not work for categorical objects

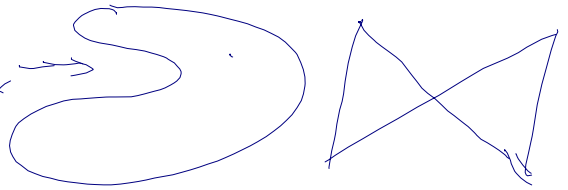
can't use the average method.

center instead of average

can be skewed easily by the outliers.

concave

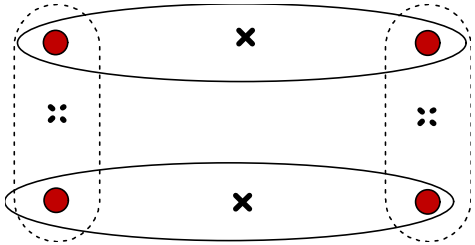
non-convex



k-means : centroid., non categorical.
k-modes : dissimilarity method, categorical data.
k-prototypes : can deal with both categorical & noncategorical data

Variations of the *K-Means* Method

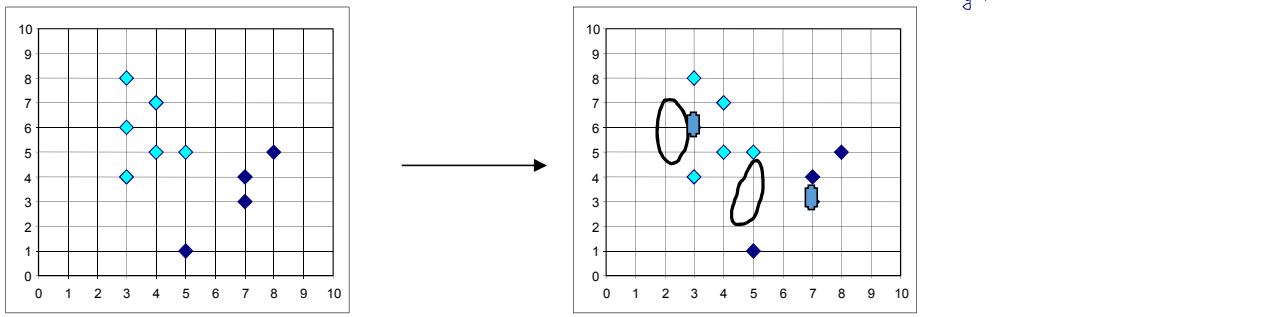
- Most of the variants of the *k-means* which differ in
  - Selection of the initial *k* means
  - Dissimilarity calculations
  - Strategies to calculate cluster means



- Handling categorical data: *k-modes*
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters
  - A mixture of categorical and numerical data: *k-prototype* method

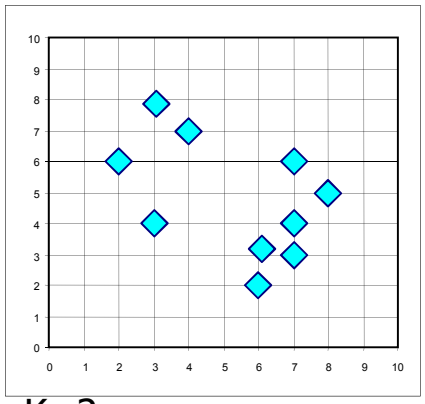
What Is the Problem of the K-Means Method?

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster



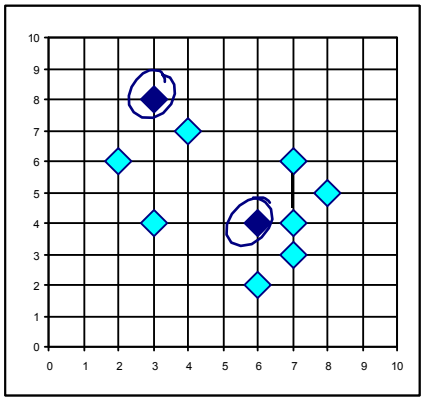
Partitioning Around Medoids

PAM: A Typical K-Medoids Algorithm

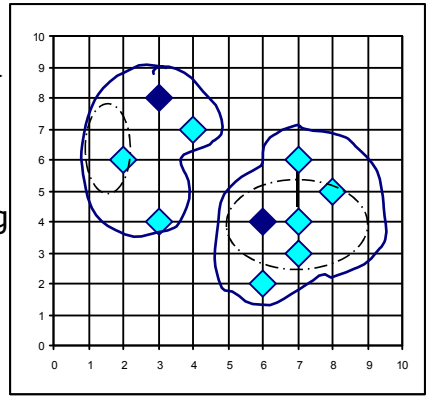


K=2

Arbitrary choose k object as initial medoids

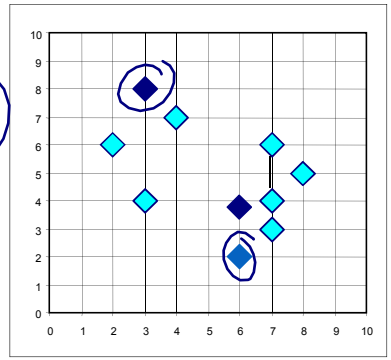


Assign each remaining object to nearest medoids

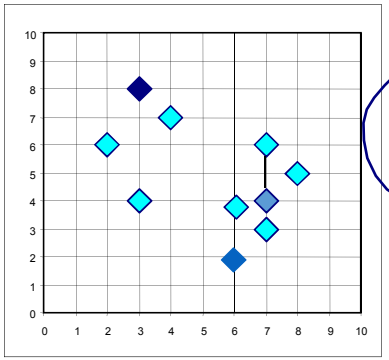


Total Cost = 20

Randomly select a nonmedoid object, O_{ramdom}



Compute total cost of swapping



Total Cost = 26

Swapping O and O_{ramdom}

If quality is improved.

Do loop

Until no change

k-medoid algo v/s k-mean.

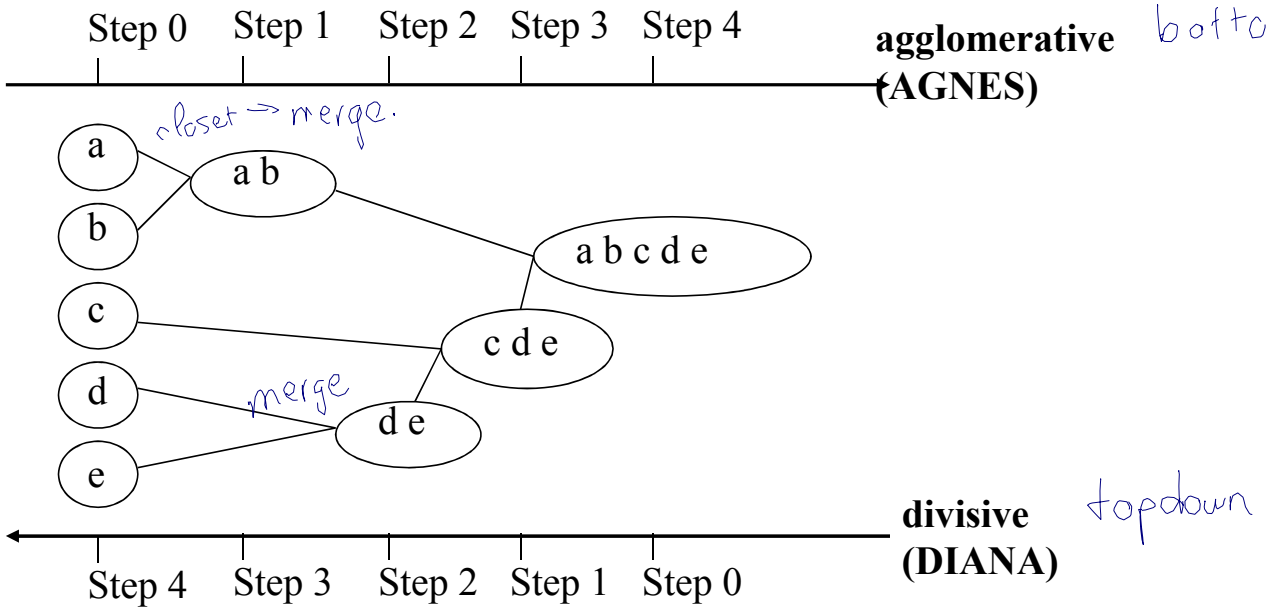
1. Choose k-objects first (instead of choosing k sets first in k-means).
2. Randomly choose next medoid (instead of calculating the next centroids)
   - Calculate the swapping cost, if better then swap

The K-Medoid Clustering Method

- *K-Medoids* Clustering: Find *representative* objects (medoids) in clusters
  - *PAM* (Partitioning Around Medoids, Kaufmann & Rousseeuw 1987)
    - Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
    - *PAM* works effectively for small data sets, but does not scale well for large data sets (due to the computational complexity)
- Efficiency improvement on PAM
  - *CLARA* (Kaufmann & Rousseeuw, 1990): PAM on samples
  - *CLARANS* (Ng & Han, 1994): Randomized re-sampling

Hierarchical Clustering

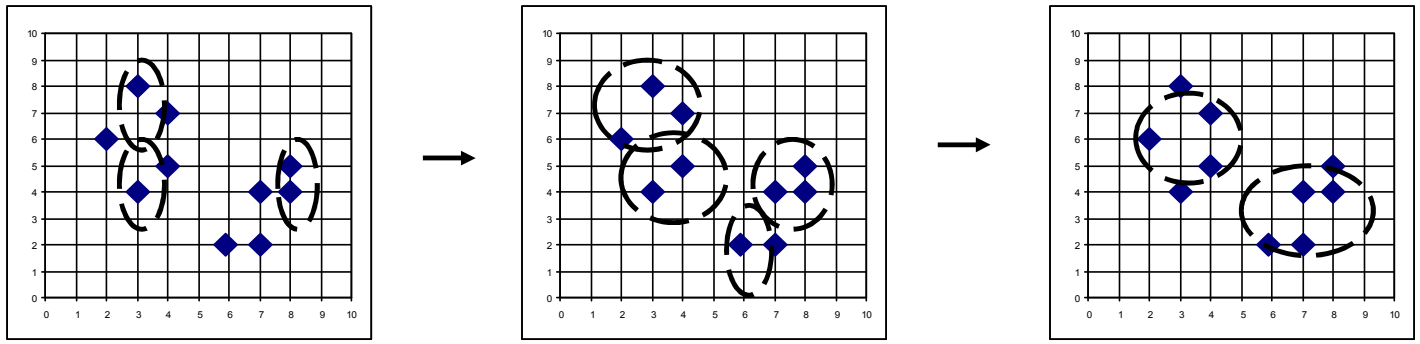
- Use distance matrix as clustering criteria. This method does not require the number of clusters ***k*** as an input, but needs a termination condition ← otherwise all items will combine into 1 cluster (in AGNES) or split into each separate individuals (in DIANA)



bottom up hierarchical cluster method

AGNES (Agglomerative Nesting)

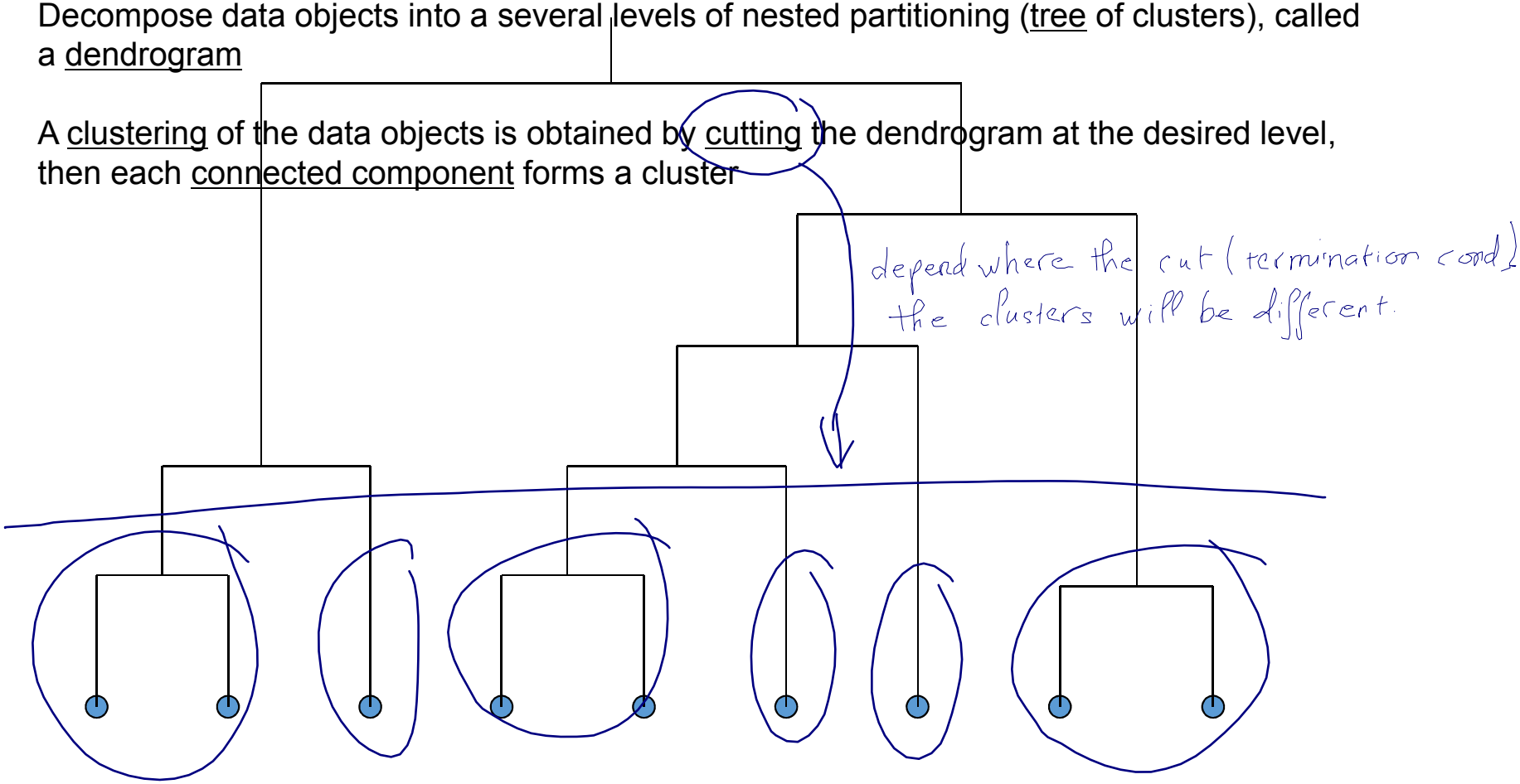
- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical packages, e.g., Splus
- Use the **single-link** method and the dissimilarity matrix
- Merge nodes that have the least dissimilarity ← the most similarity.
- Go on in a non-descending fashion
- Eventually all nodes belong to the same cluster



Dendrogram: Shows How Clusters are Merged

Decompose data objects into a several levels of nested partitioning (tree of clusters), called a dendrogram

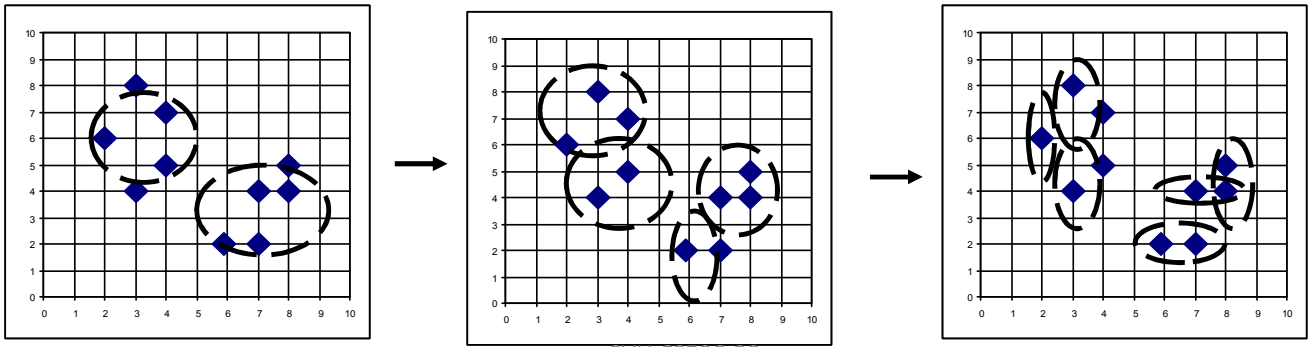
A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster



top down hierachical cluster method

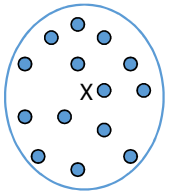
DIANA (Divisive Analysis)

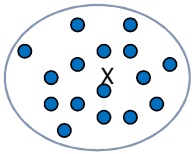
- Introduced in Kaufmann and Rousseeuw (1990)
- Implemented in statistical analysis packages, e.g., Splus
- Inverse order of AGNES
- Eventually each node forms a cluster on its own



5 types of Distance

Distance between Clusters

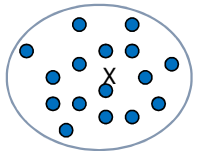




- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $dist(K_i, K_j) = avg(t_{ip}, t_{jq})$
- Centroid: distance between the centroids of two clusters, i.e., $dist(K_i, K_j) = dist(C_i, C_j)$
- Medoid: distance between the medoids of two clusters, i.e., $dist(K_i, K_j) = dist(M_i, M_j)$
  - Medoid: a chosen, centrally located object in the cluster

Formula to calculate.

Centroid, Radius and Diameter of a Cluster (for numerical data sets)



- Centroid: the "middle" of a cluster

$$C_m = \frac{\sum_{i=1}^{N}(t_{ip})}{N}$$

- Radius: square root of average distance from any point of the cluster to its centroid

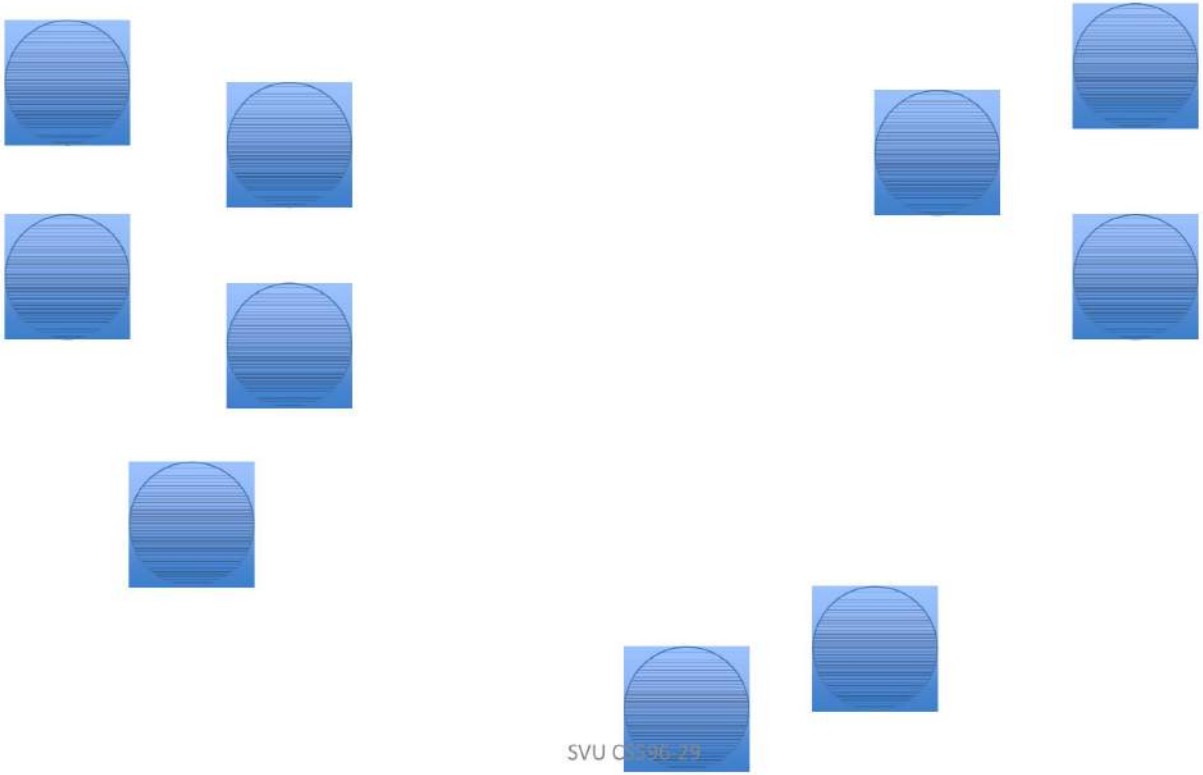
$$R_m = \sqrt{\frac{\sum_{i=1}^{N}(t_{ip} - c_m)^2}{N}}$$

- Diameter: square root of average mean squared distance between all pairs of points in the cluster

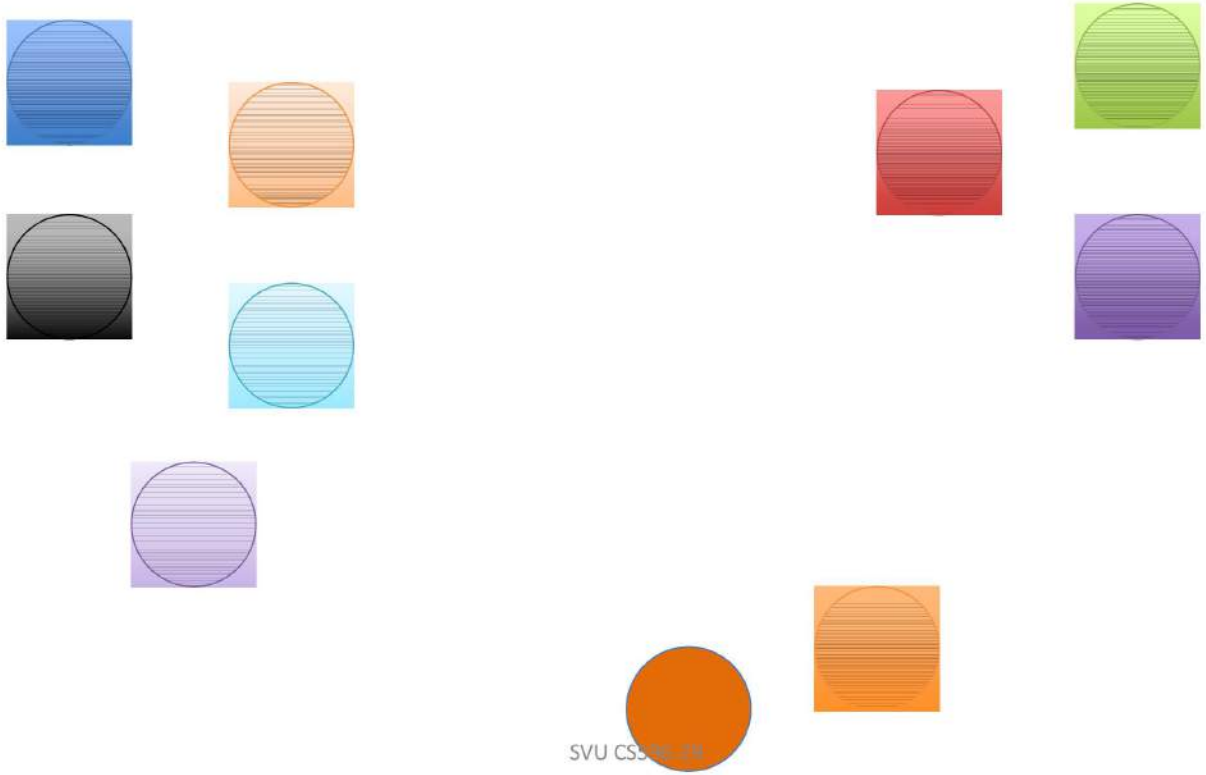
$$D_m = \sqrt{\frac{\sum_{i=1}^{N}\sum_{i=1}^{N}(t_{ip} - t_{iq})^2}{N(N-1)}}$$

Examples

Degenerate Clustering Solutions



Degenerate Clustering Solutions



Two approaches to clustering

- Partitional
  - Divide the space into a fixed number of regions and position their boundaries appropriately
- Hierarchical
  - Either merge or split clusters.

K-Means

- K-Means clustering is a **partitional** clustering algorithm.
  - Identify different partitions of the space for a fixed number of clusters
  - Input: a value for K – the number of clusters
  - Output: K cluster centroids.

K-Means Algorithm

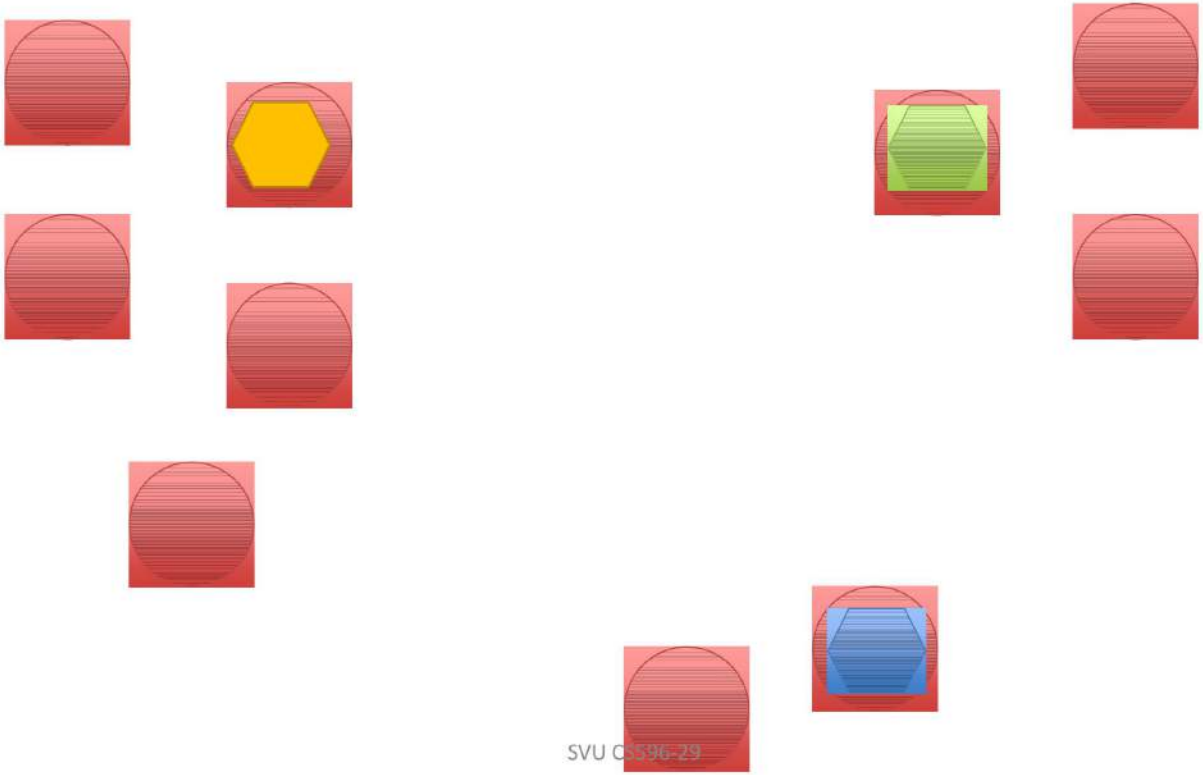
- Given an integer K specifying the number of clusters
- Initialize K cluster centroids
  - Select K points from the data set at random
  - Select K points from the space at random
- For each point in the data set, assign it to the cluster center it is closest to $\underset{C_i}{\operatorname{argmin}}\, d(\vec{x}, C_i)$
- Update each centroid based on the points that are assigned to it
- If any data point has changed clusters, repeat

$$C_i = \frac{1}{|C_i|} \sum_{\vec{x} \in C_i} \vec{x}$$

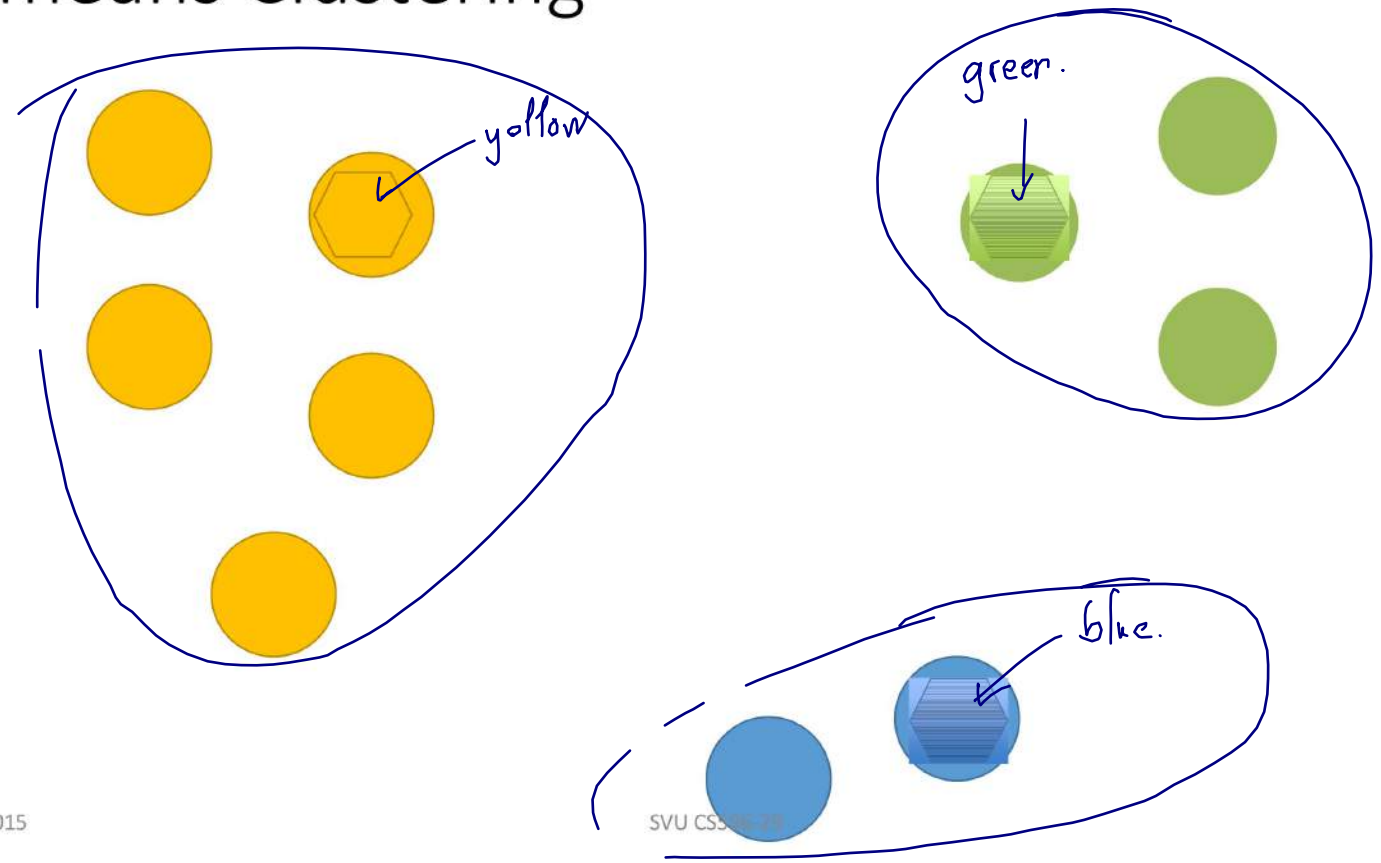
How does K-Means work?

- When an assignment is changed, the **distance** of the data point to its assigned cluster is reduced.
  - IV is lower
- When a cluster centroid is moved, the mean distance from the centroid to its data points is reduced
  - IV is lower.
- At convergence, we have found a local minimum of IV

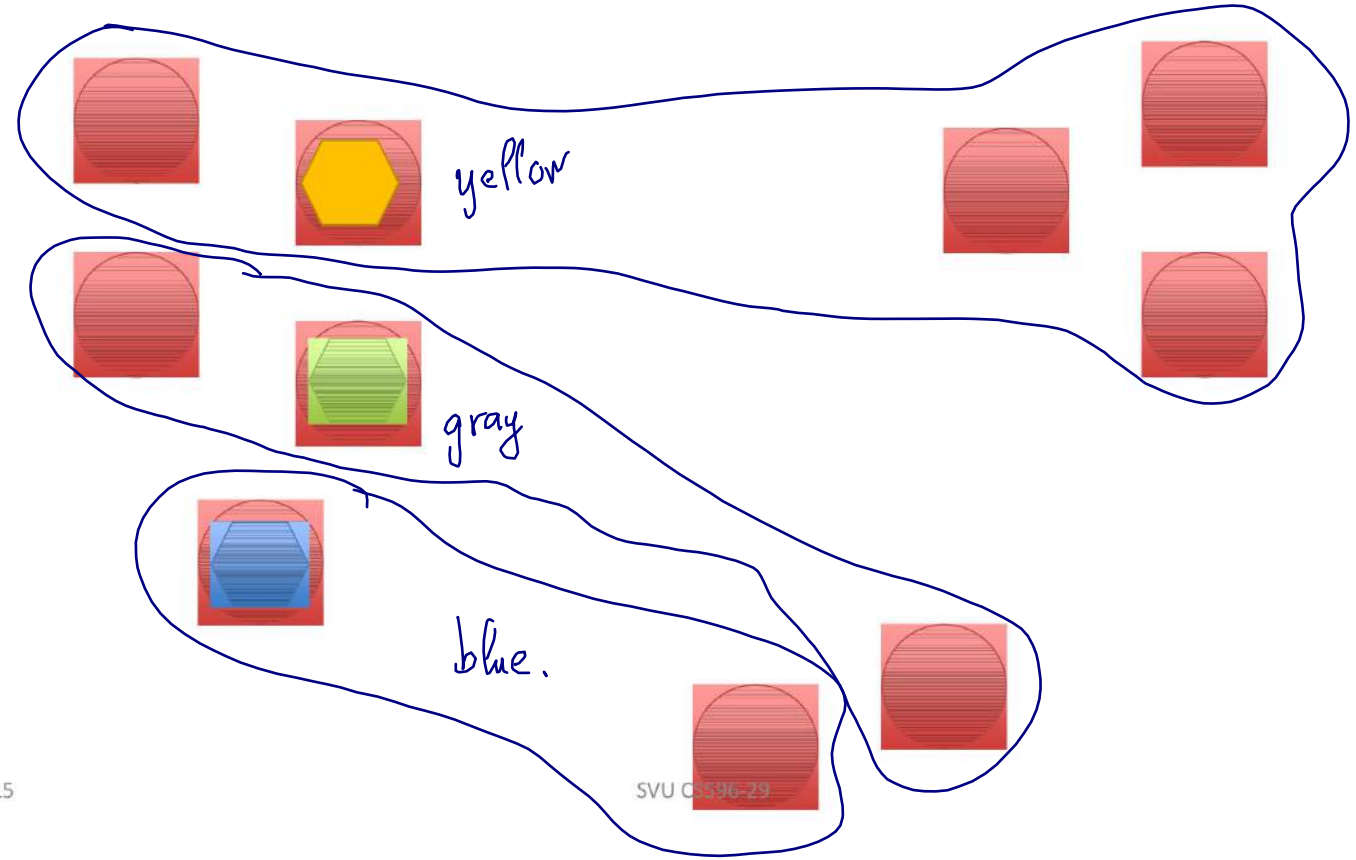
K-means Clustering



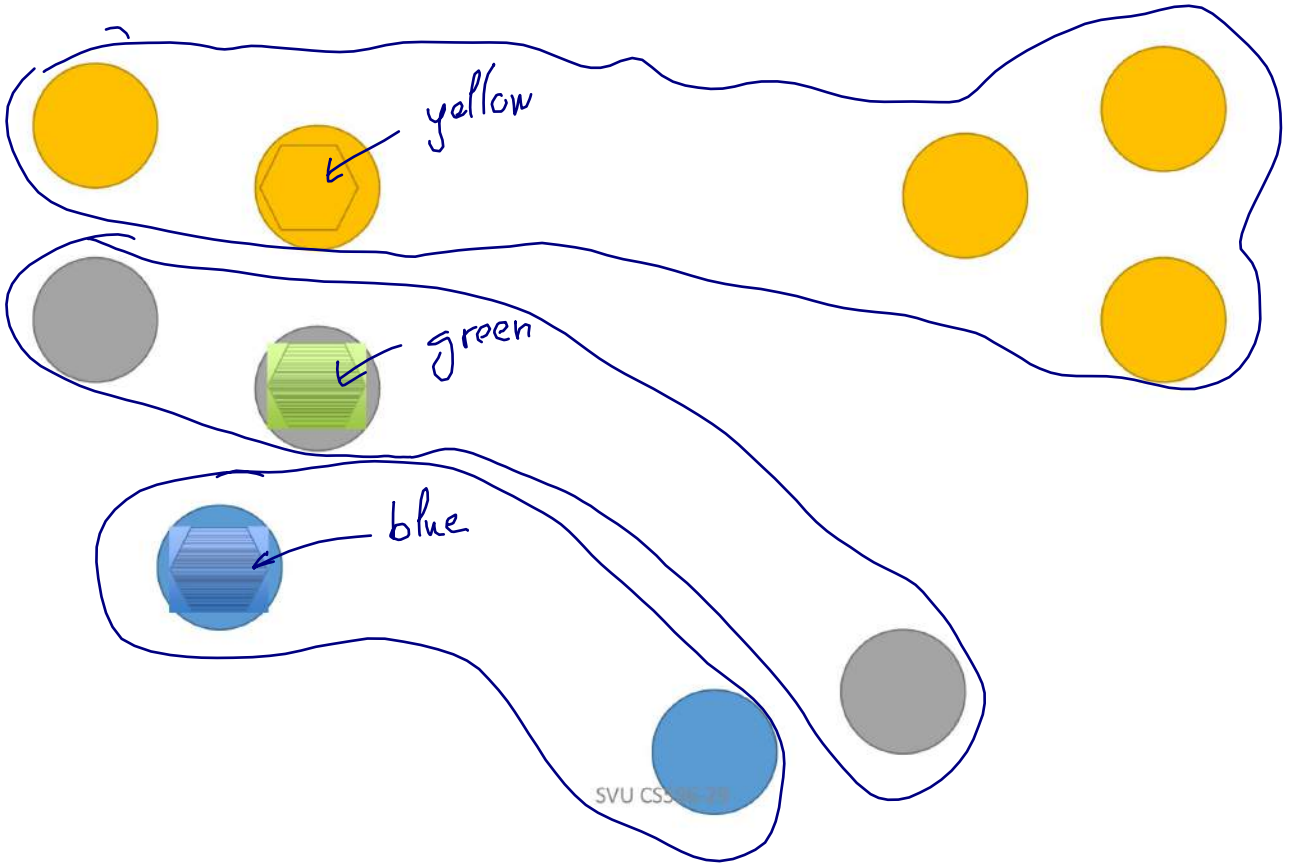
K-means Clustering



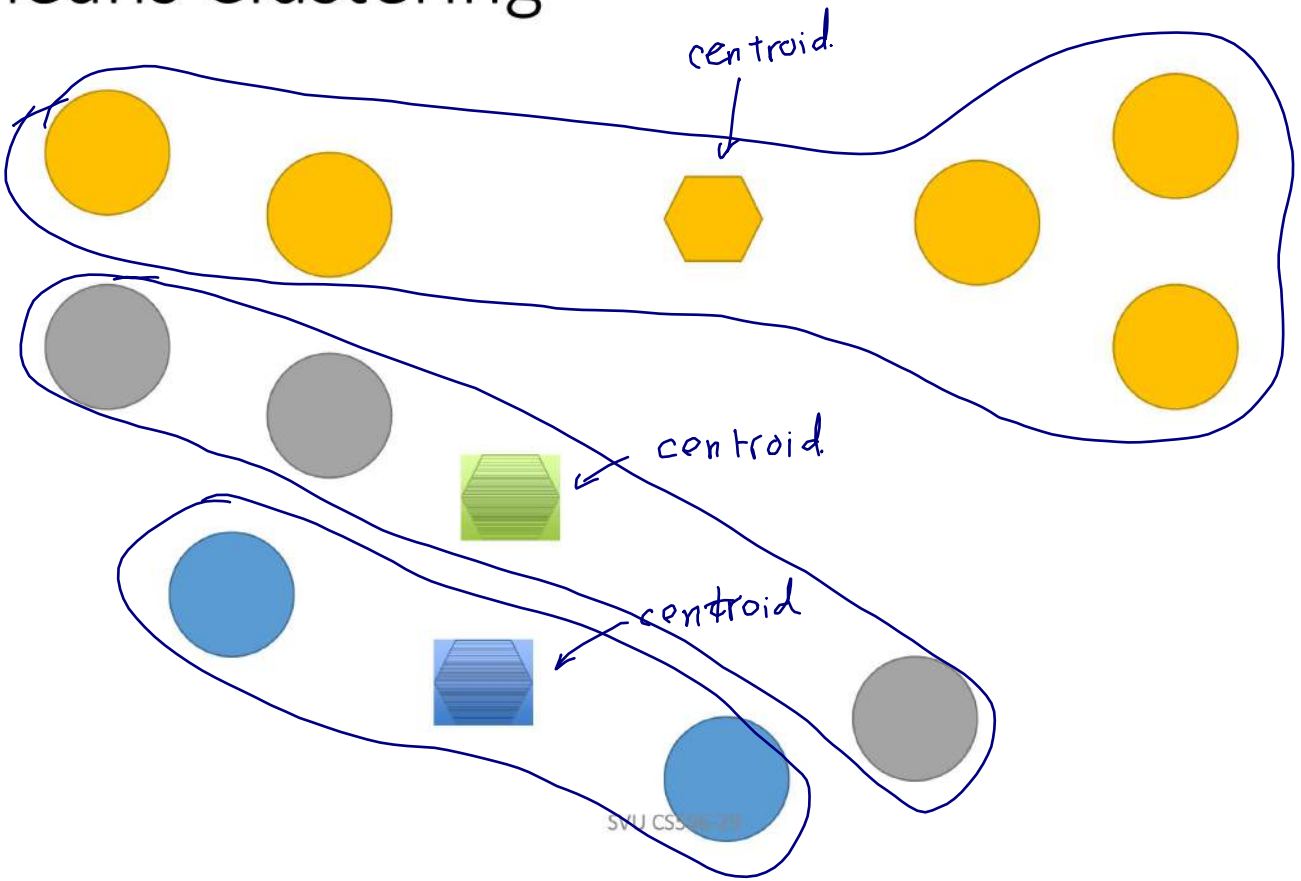
K-means Clustering



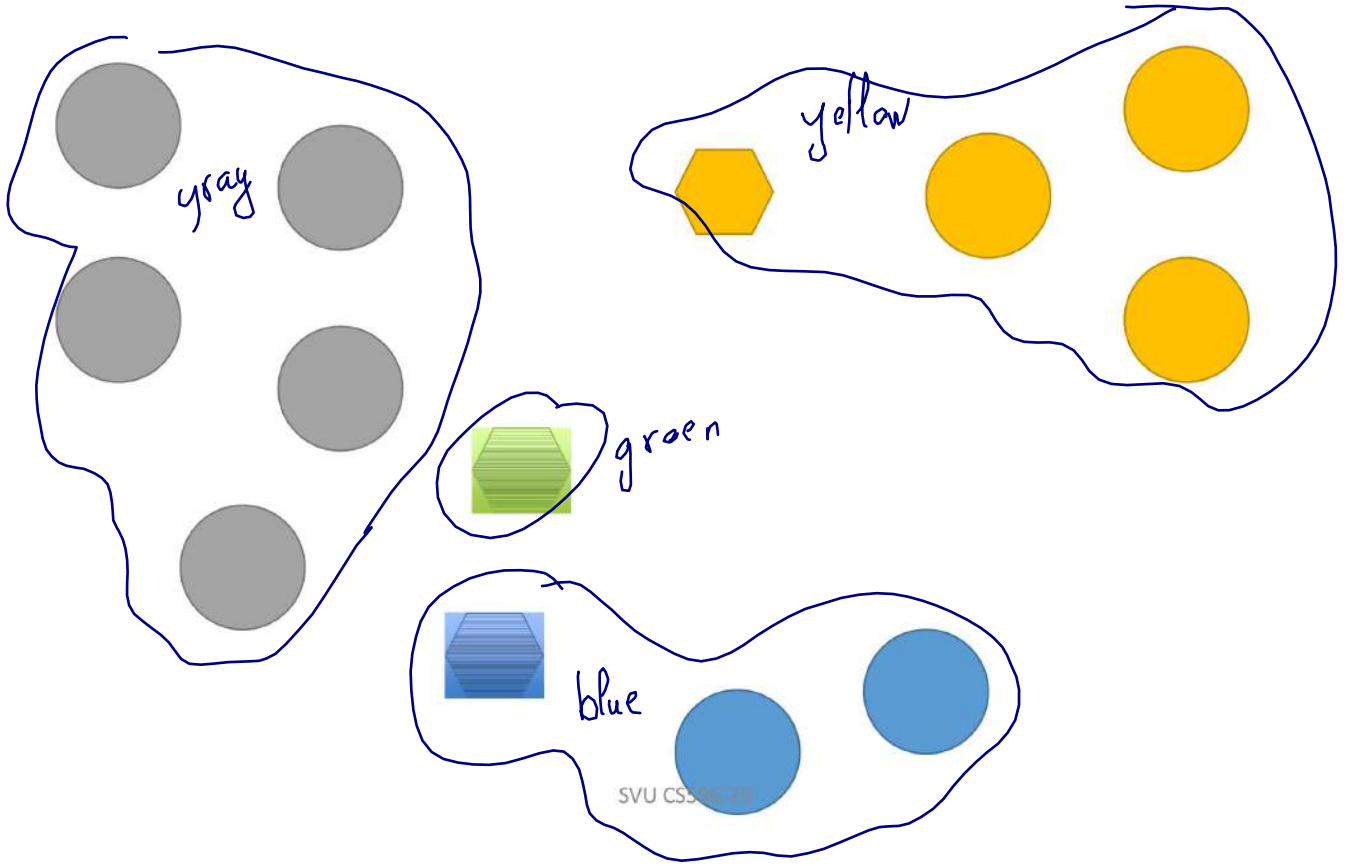
K-means Clustering



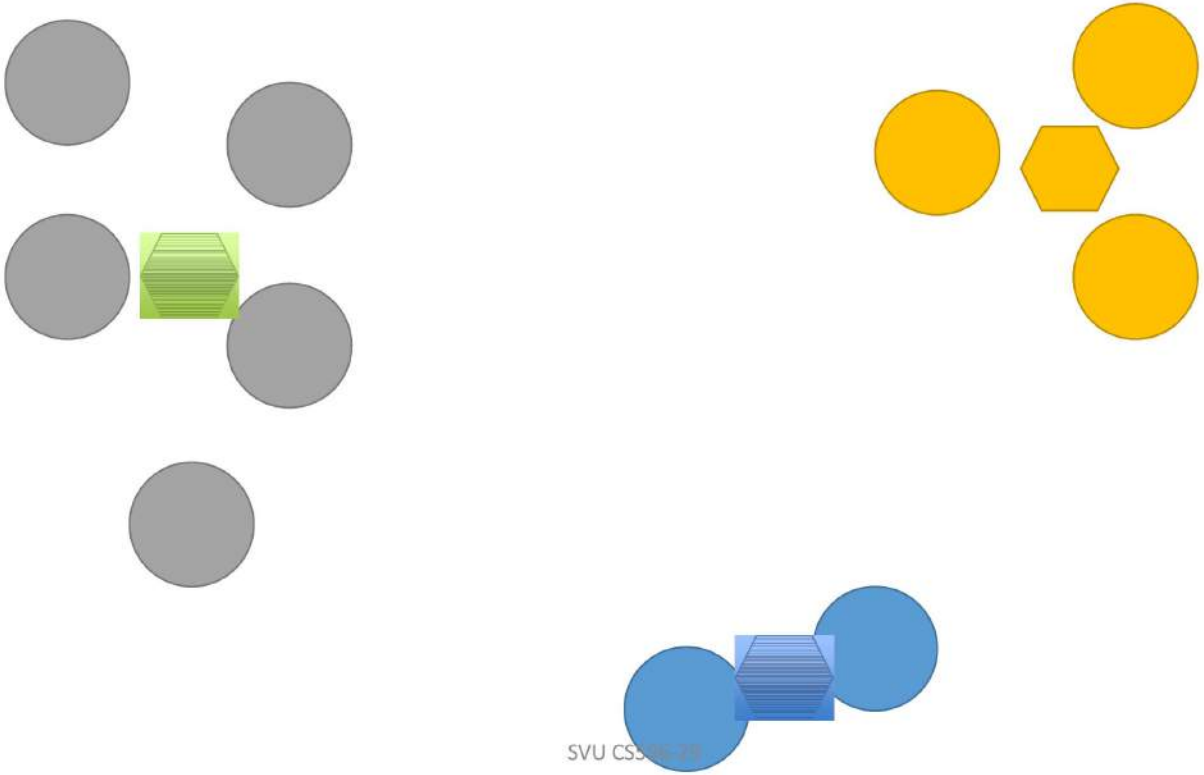
K-means Clustering



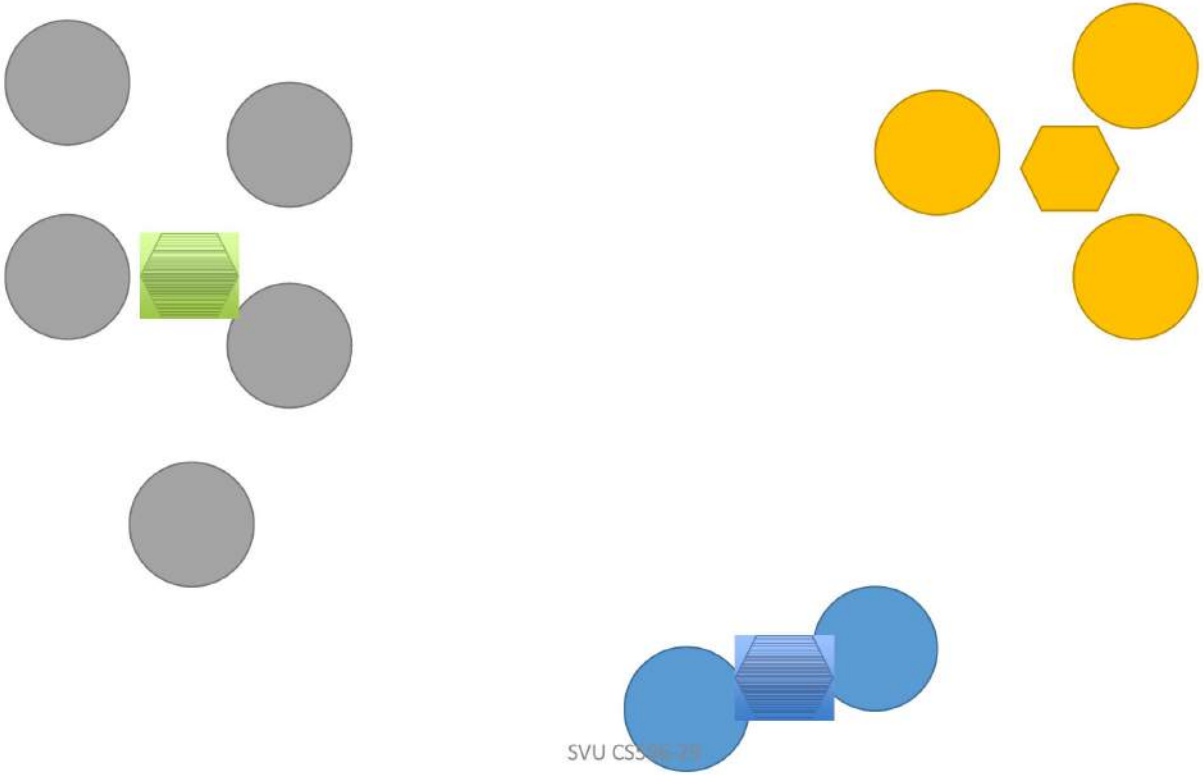
K-means Clustering



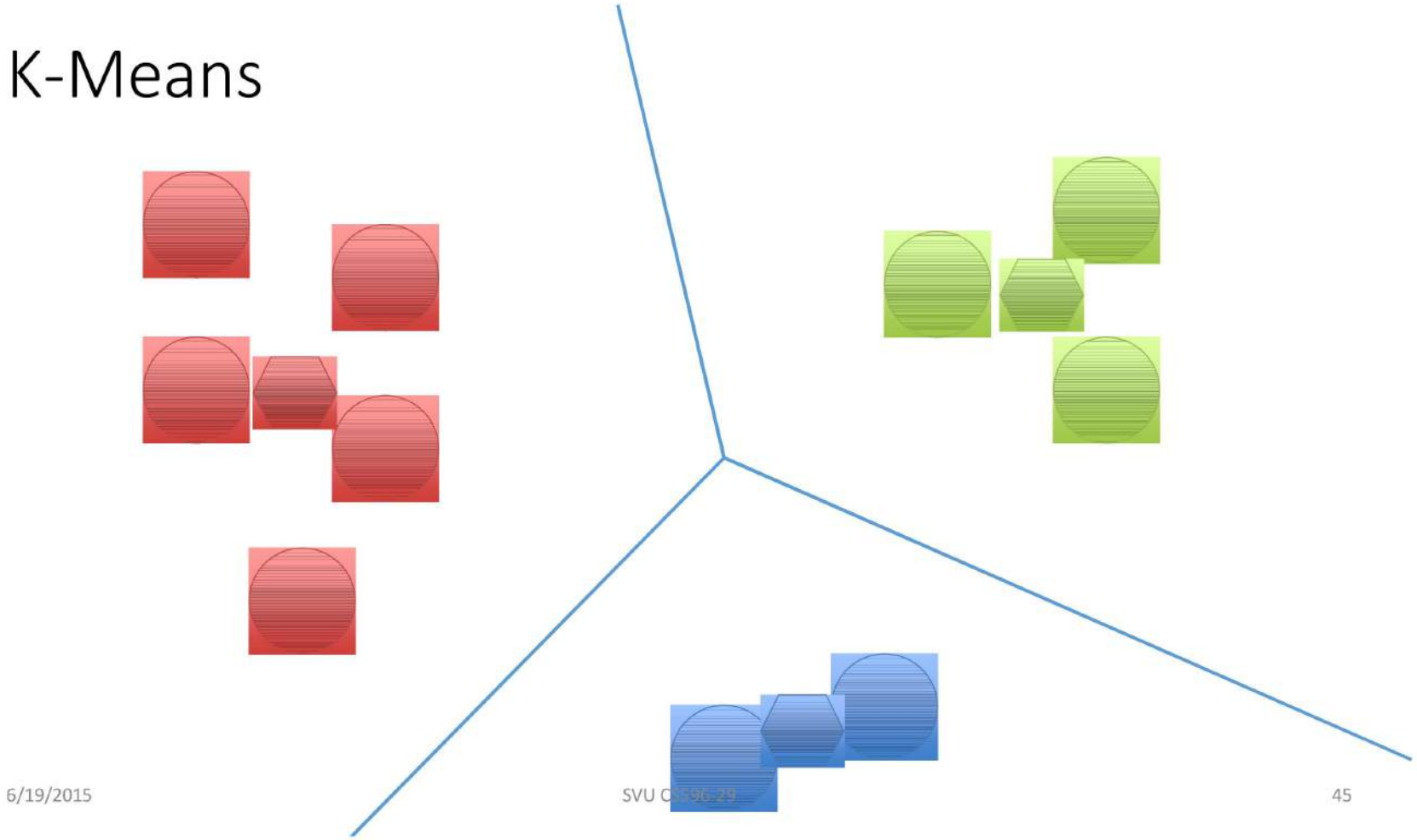
K-means Clustering



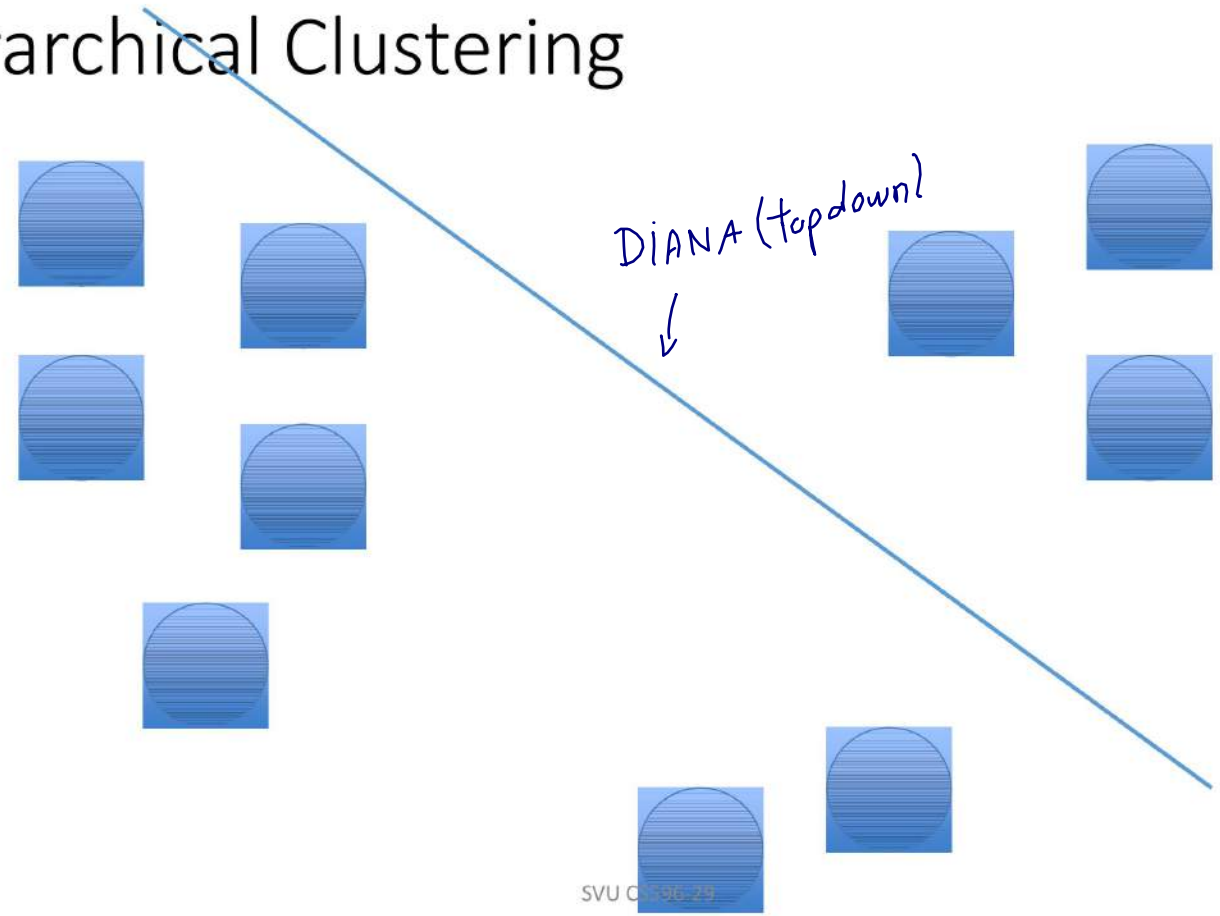
K-means Clustering



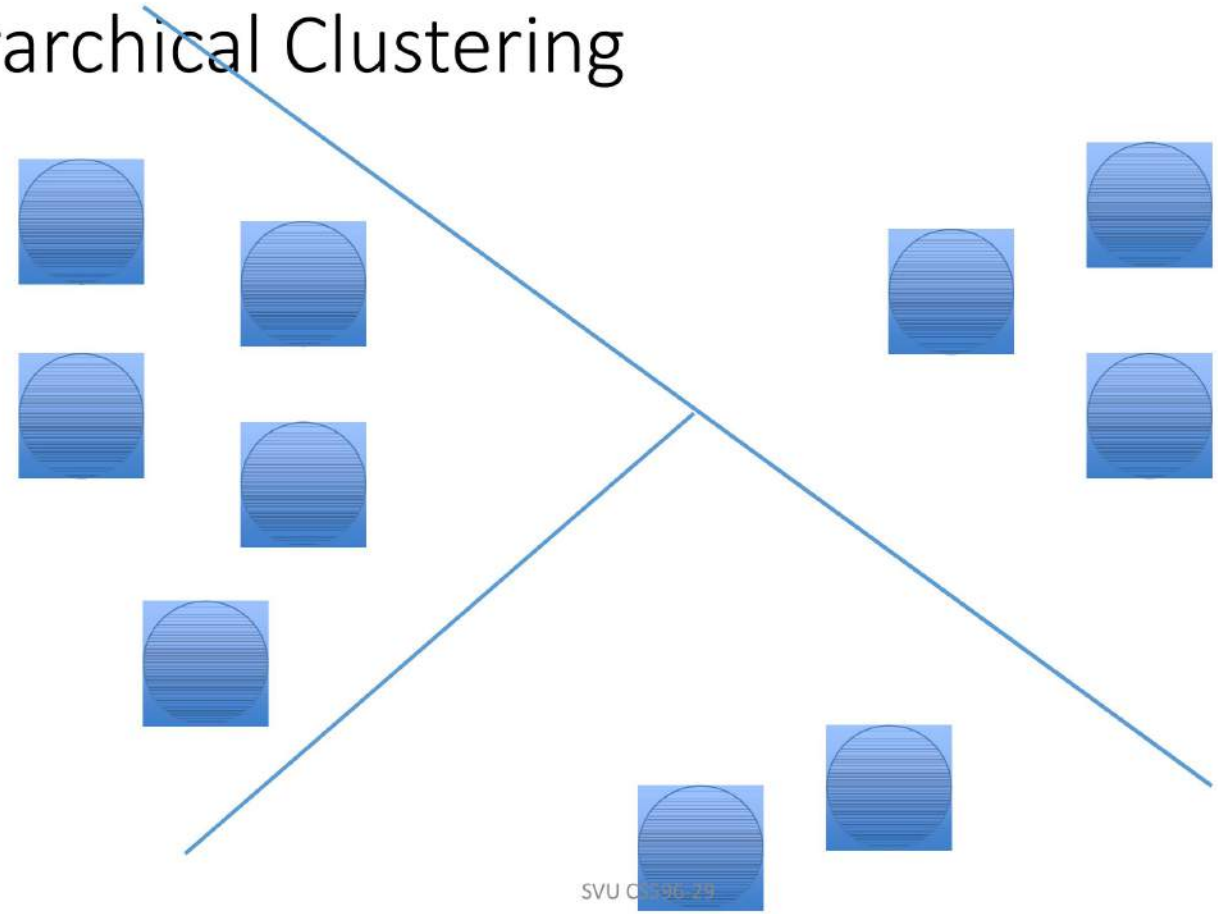
K-Means



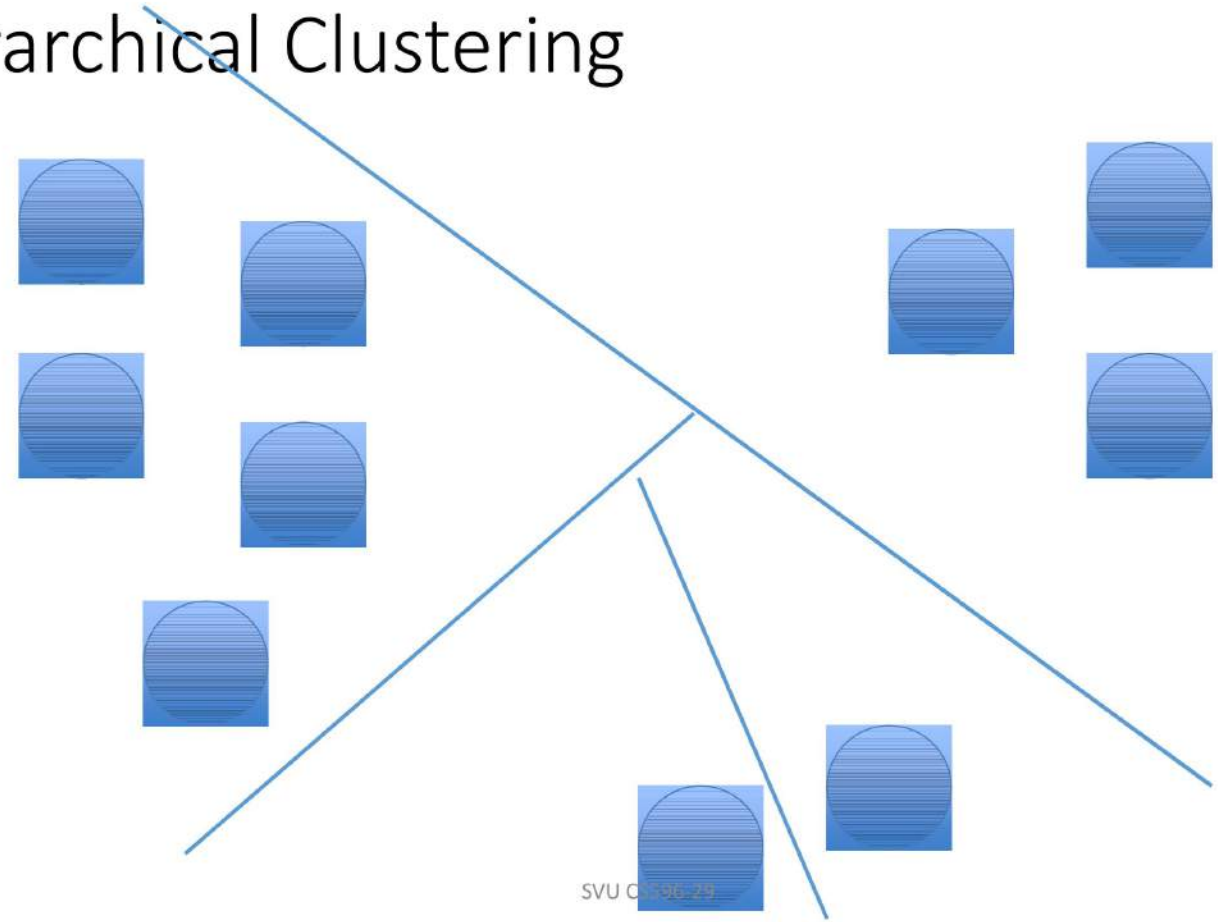
Hierarchical Clustering



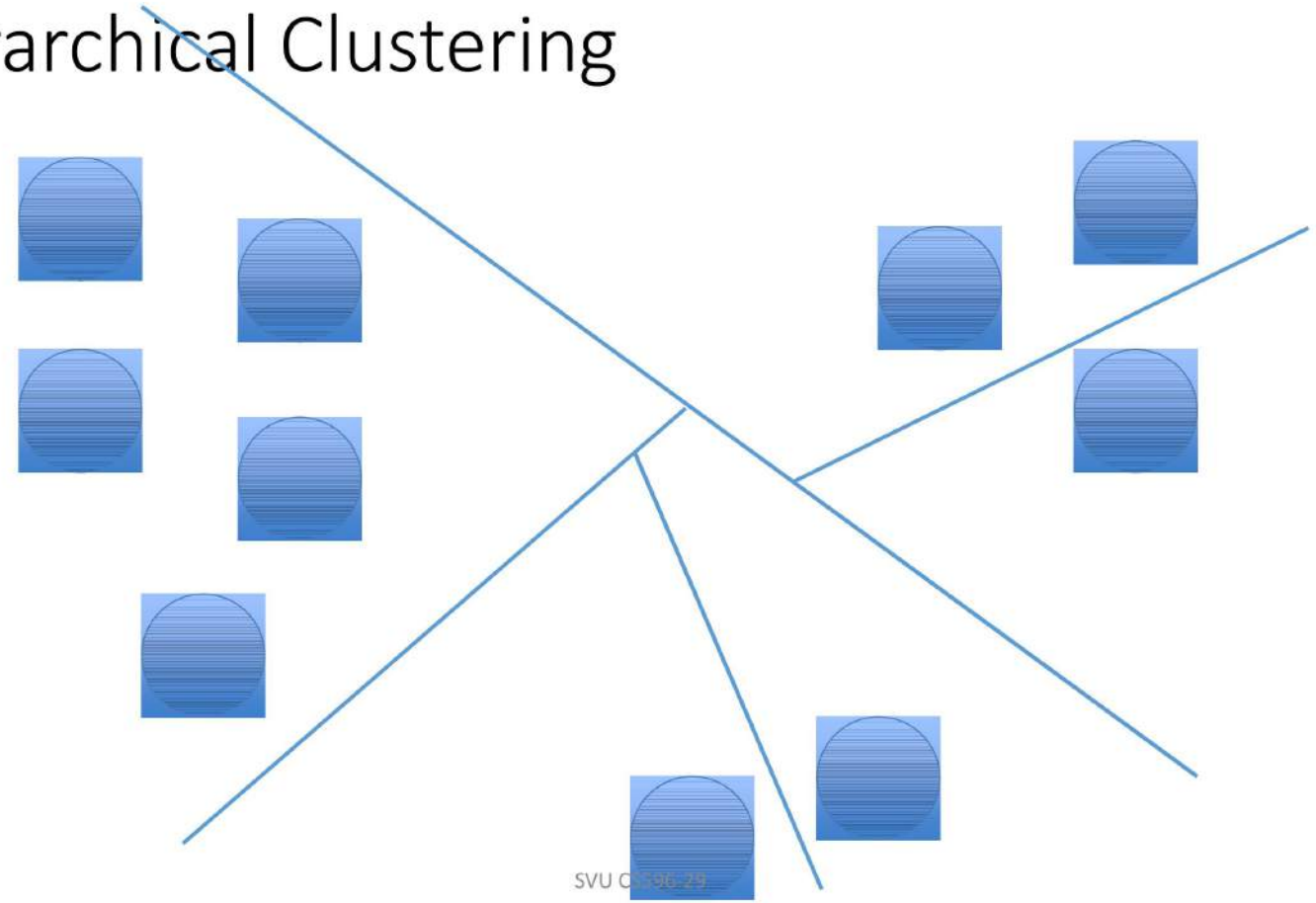
Hierarchical Clustering



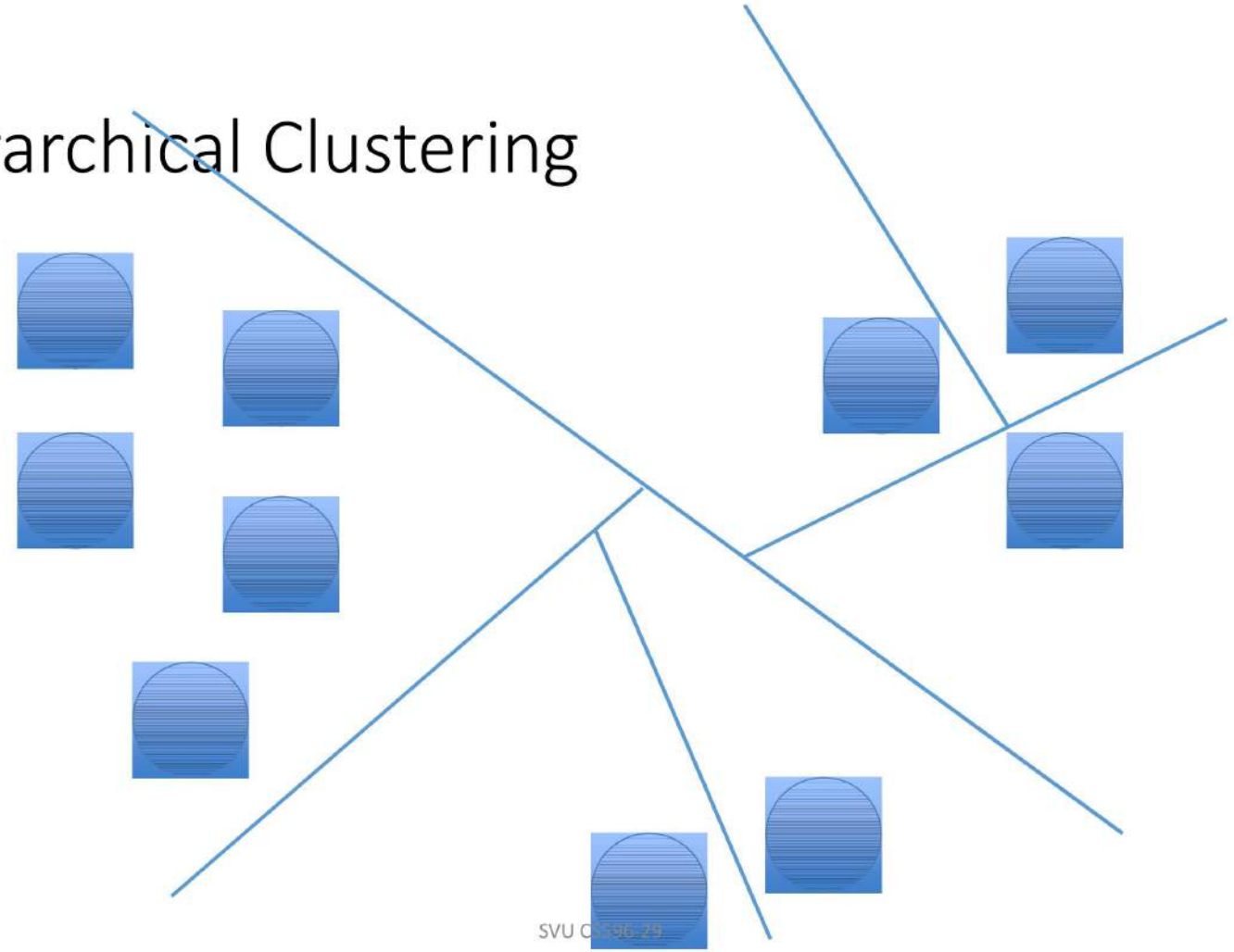
Hierarchical Clustering



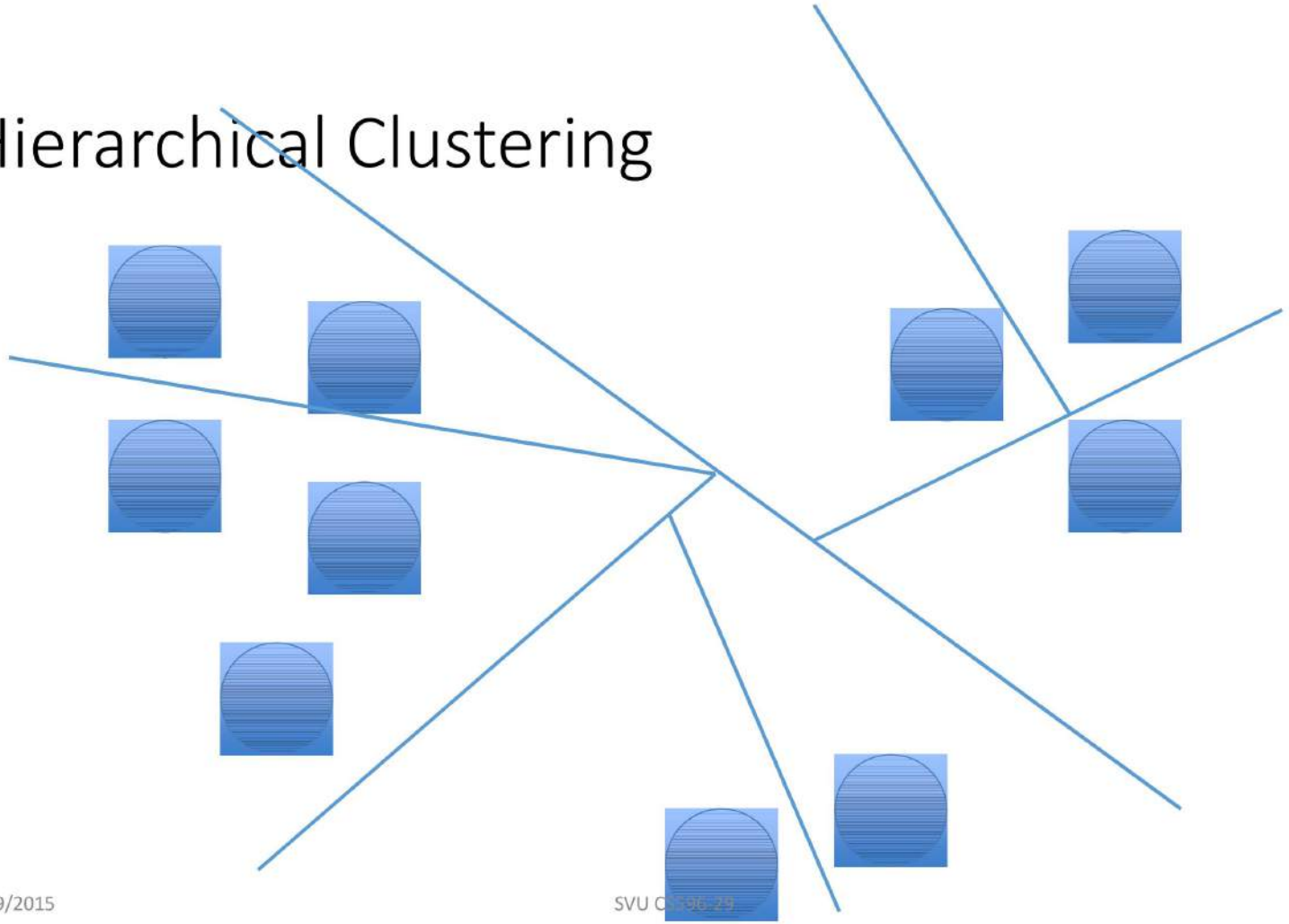
Hierarchical Clustering



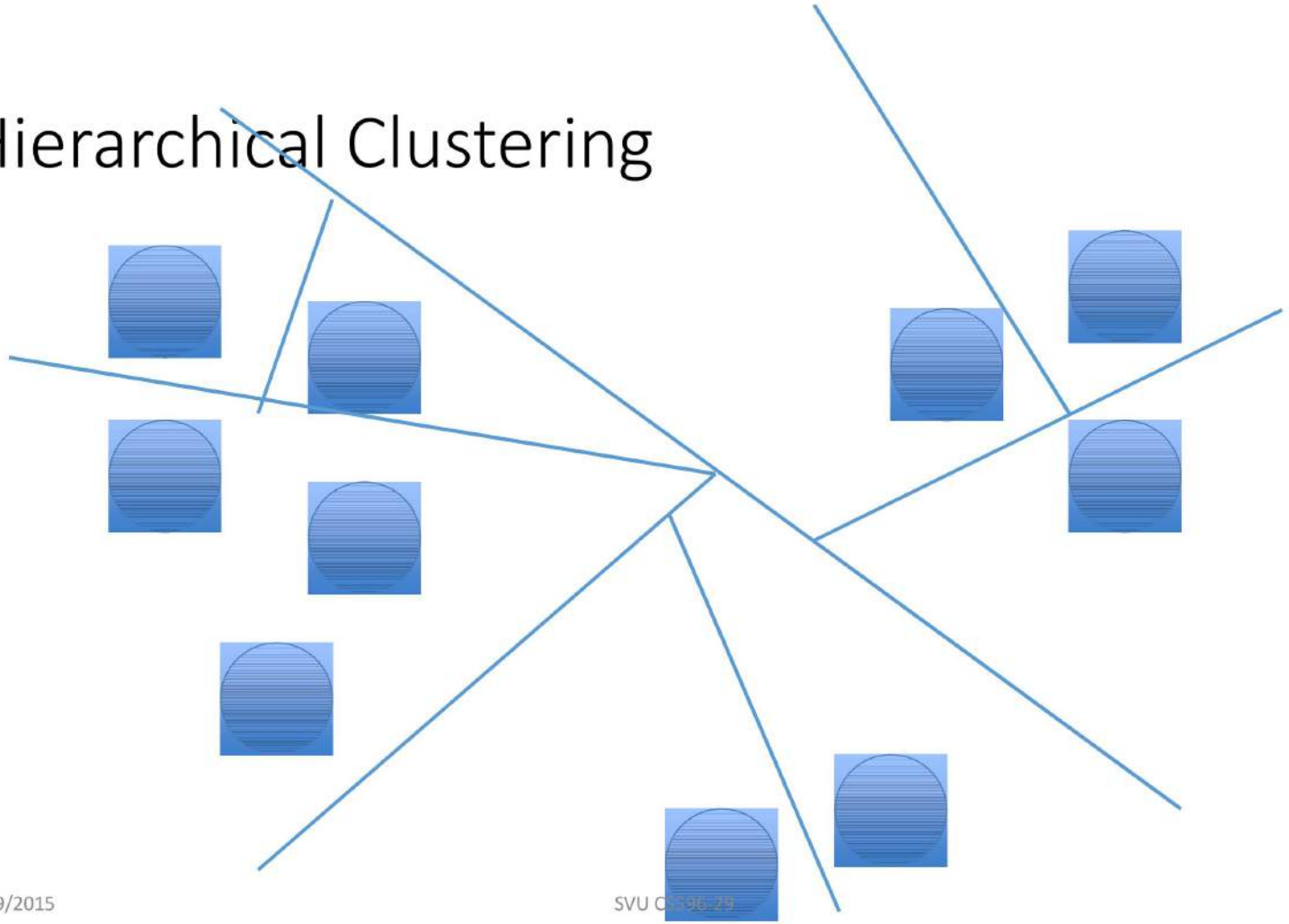
Hierarchical Clustering



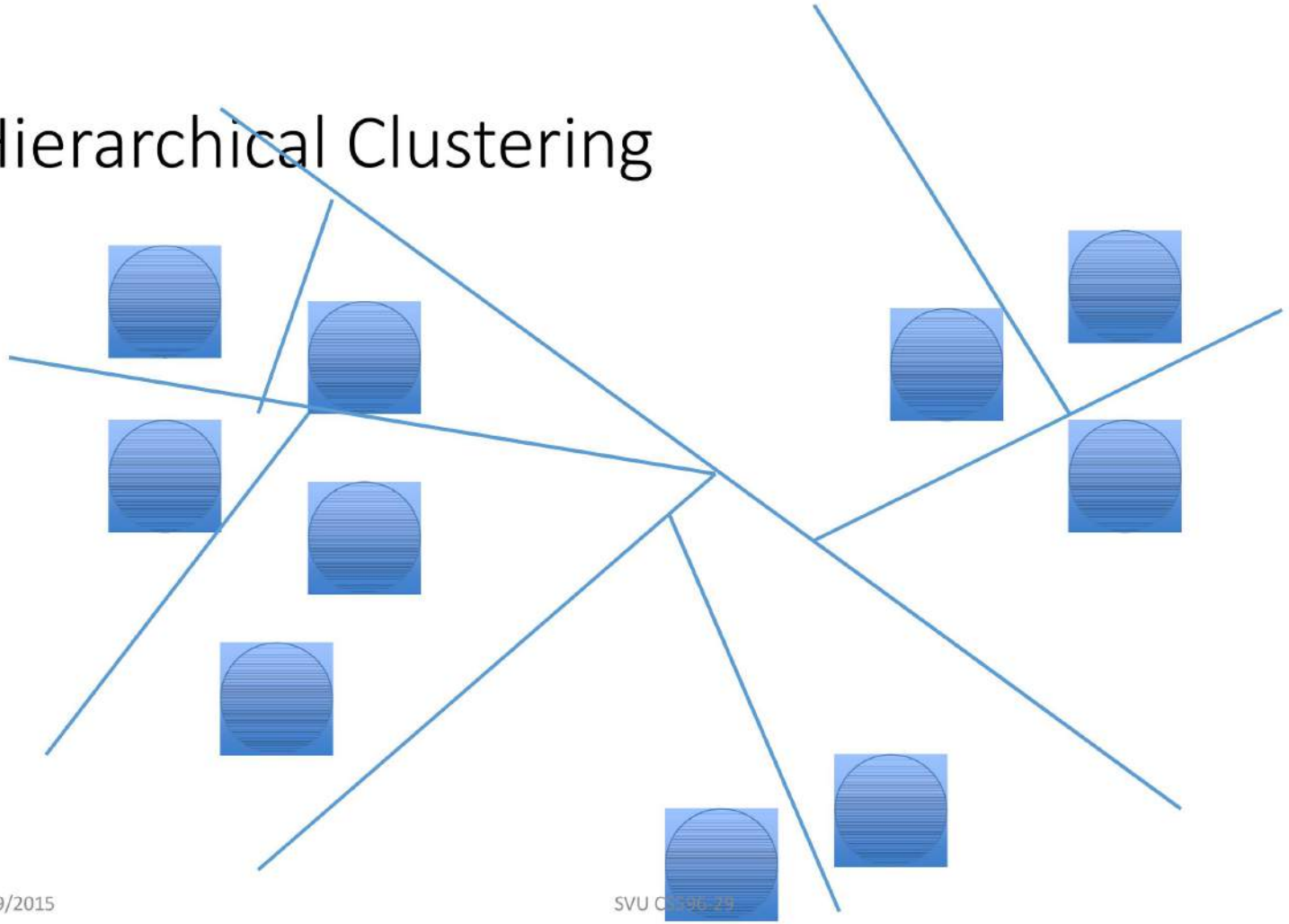
Hierarchical Clustering



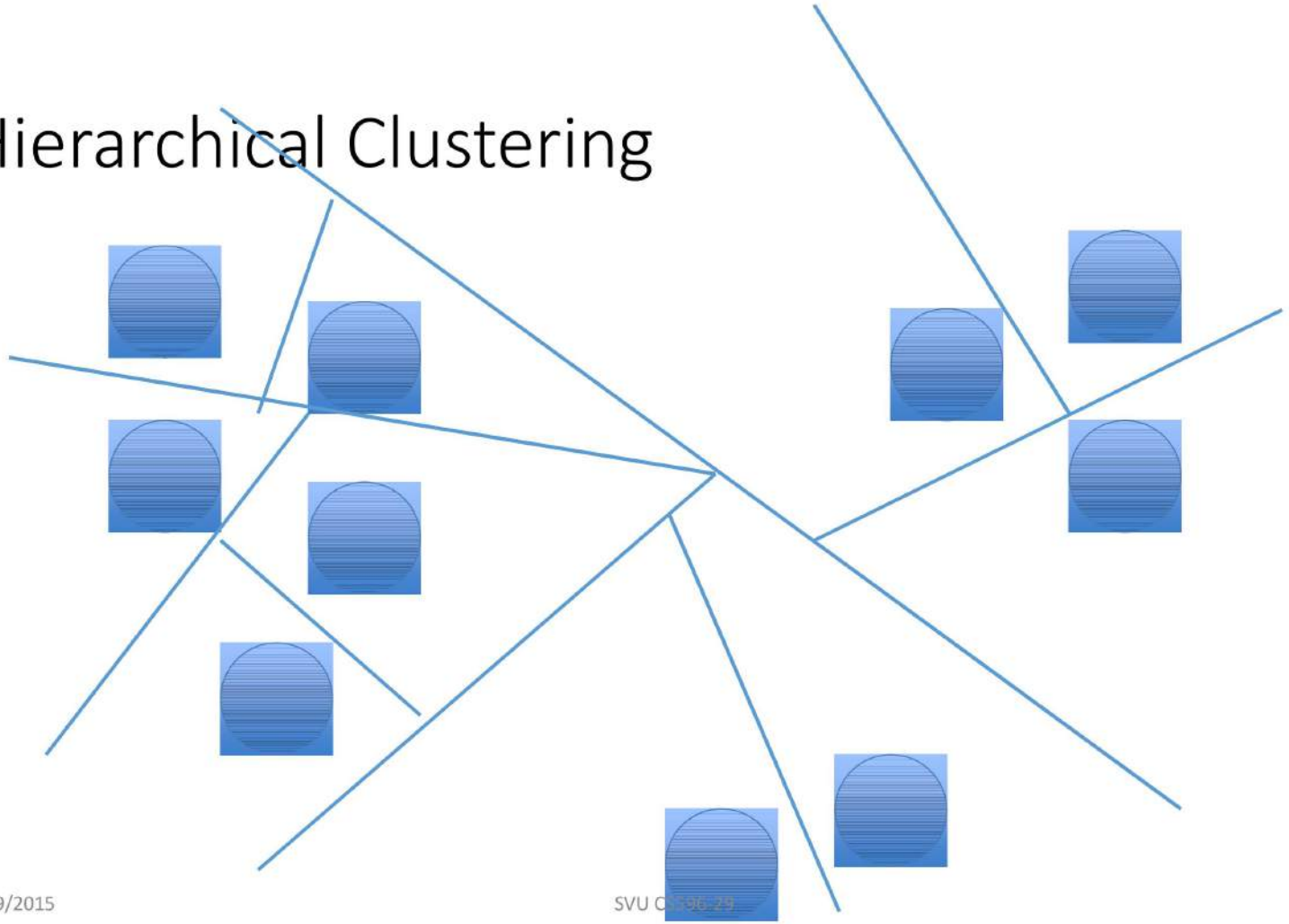
Hierarchical Clustering



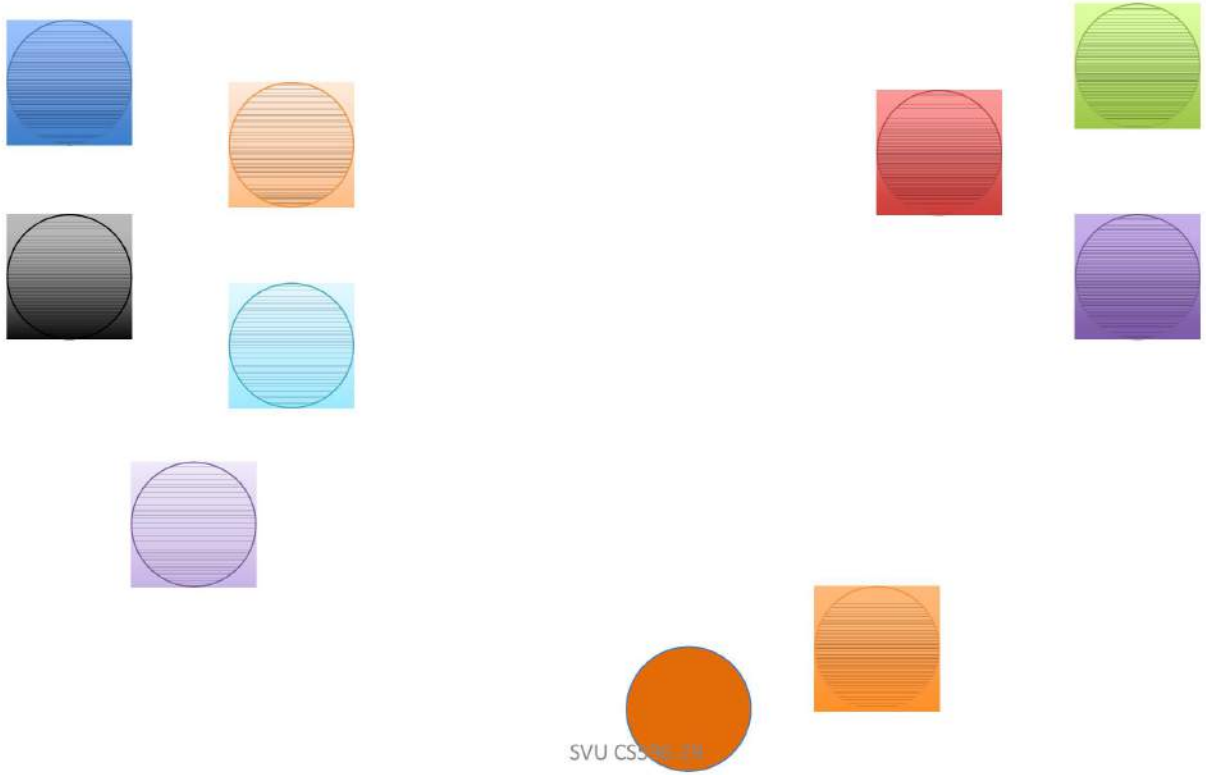
Hierarchical Clustering



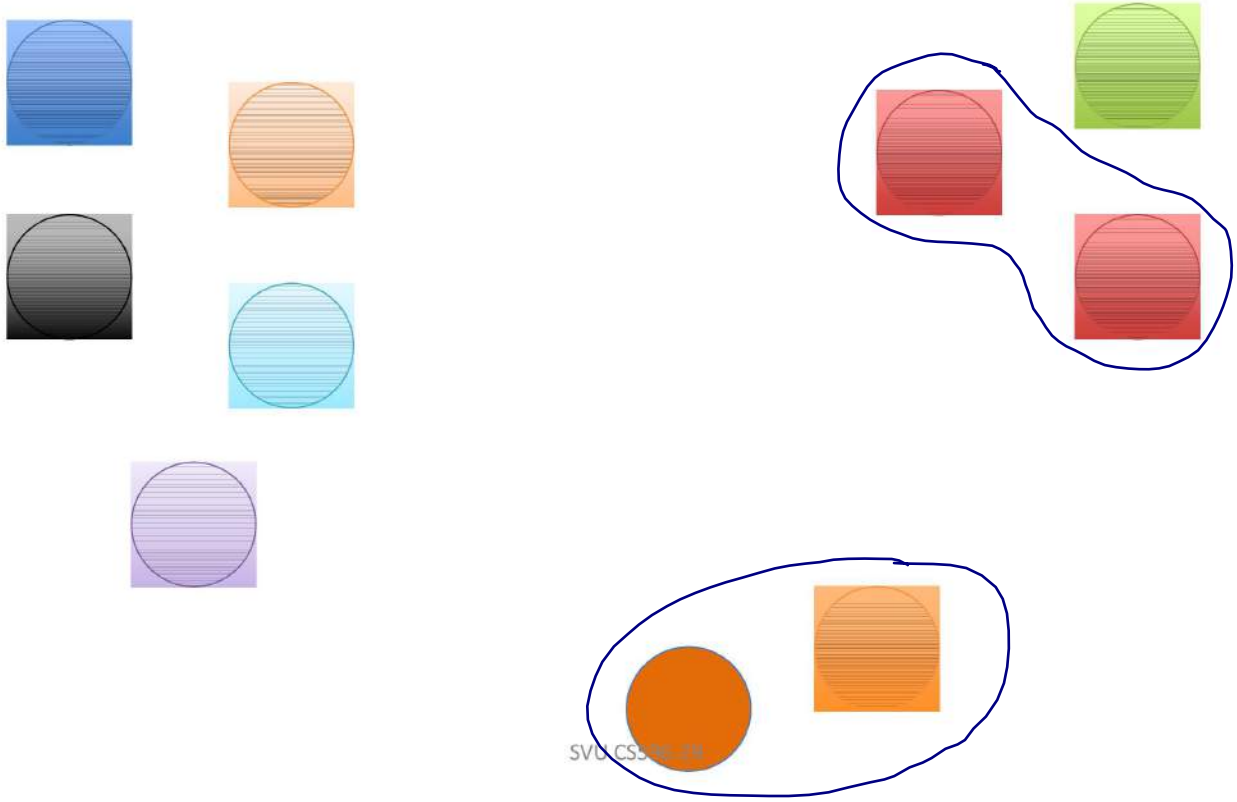
Hierarchical Clustering



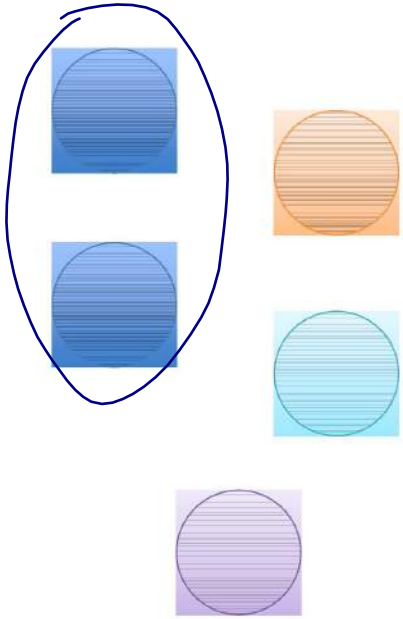
Agglomerative Clustering (AGNES)

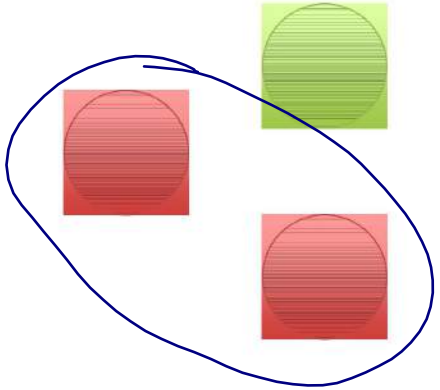


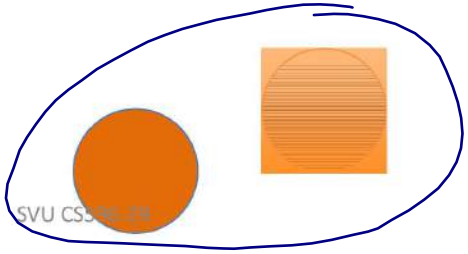
Agglomerative Clustering



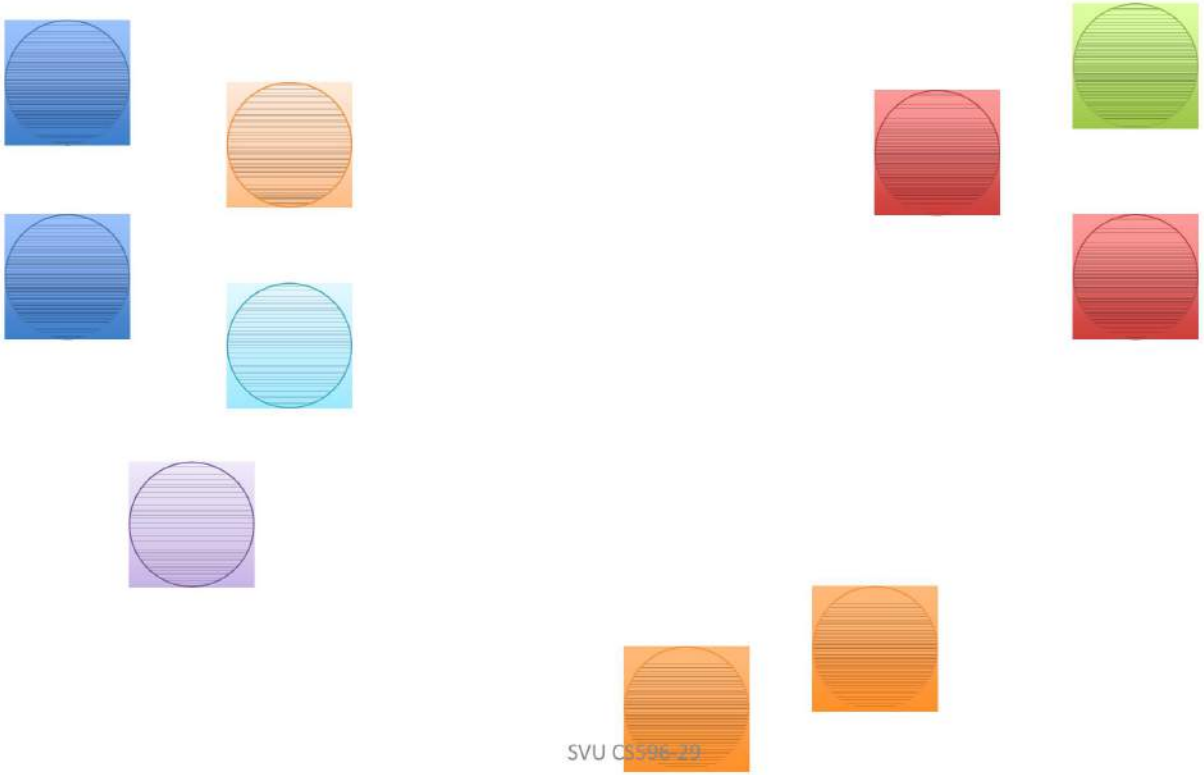
Agglomerative Clustering



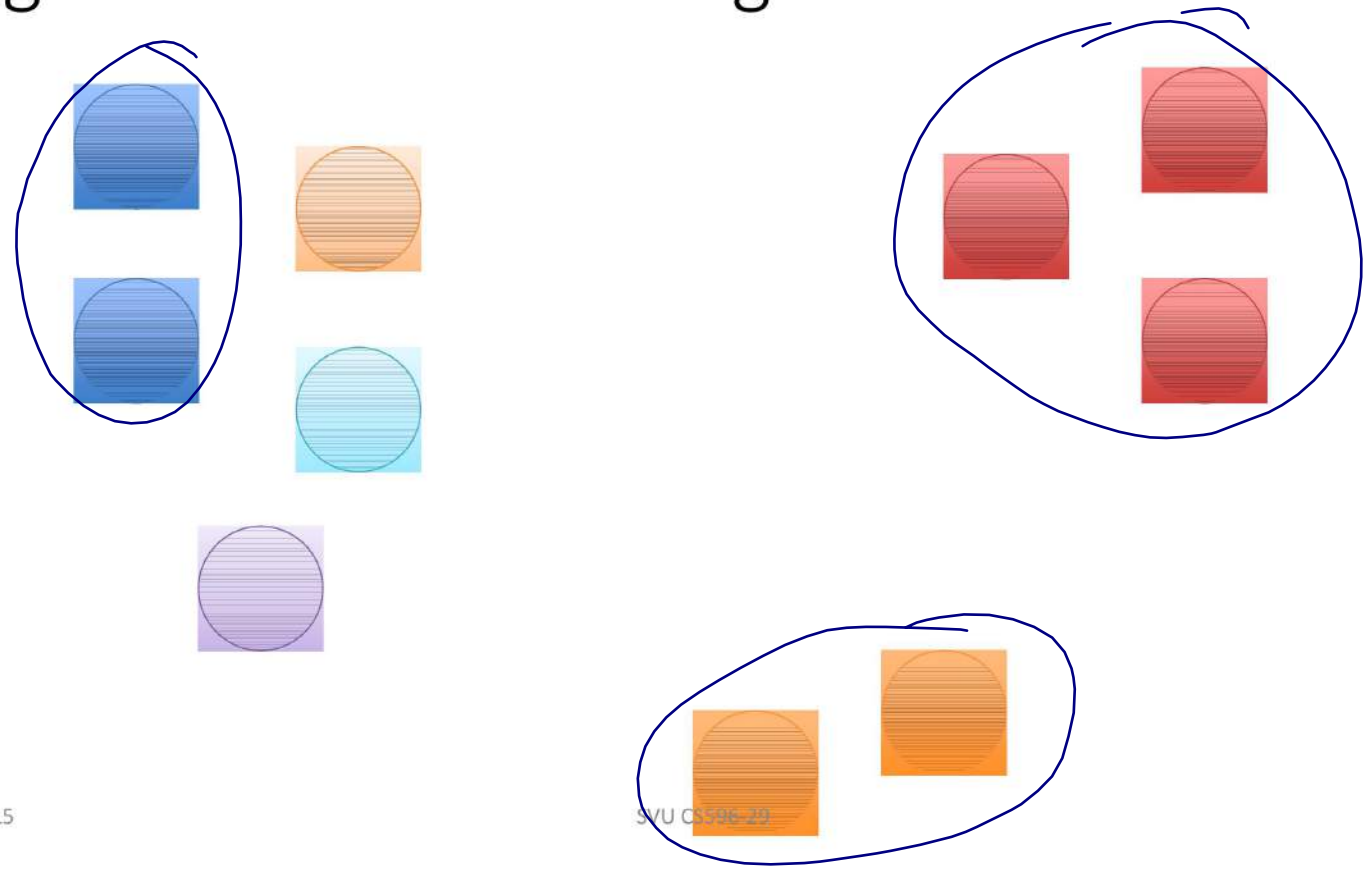




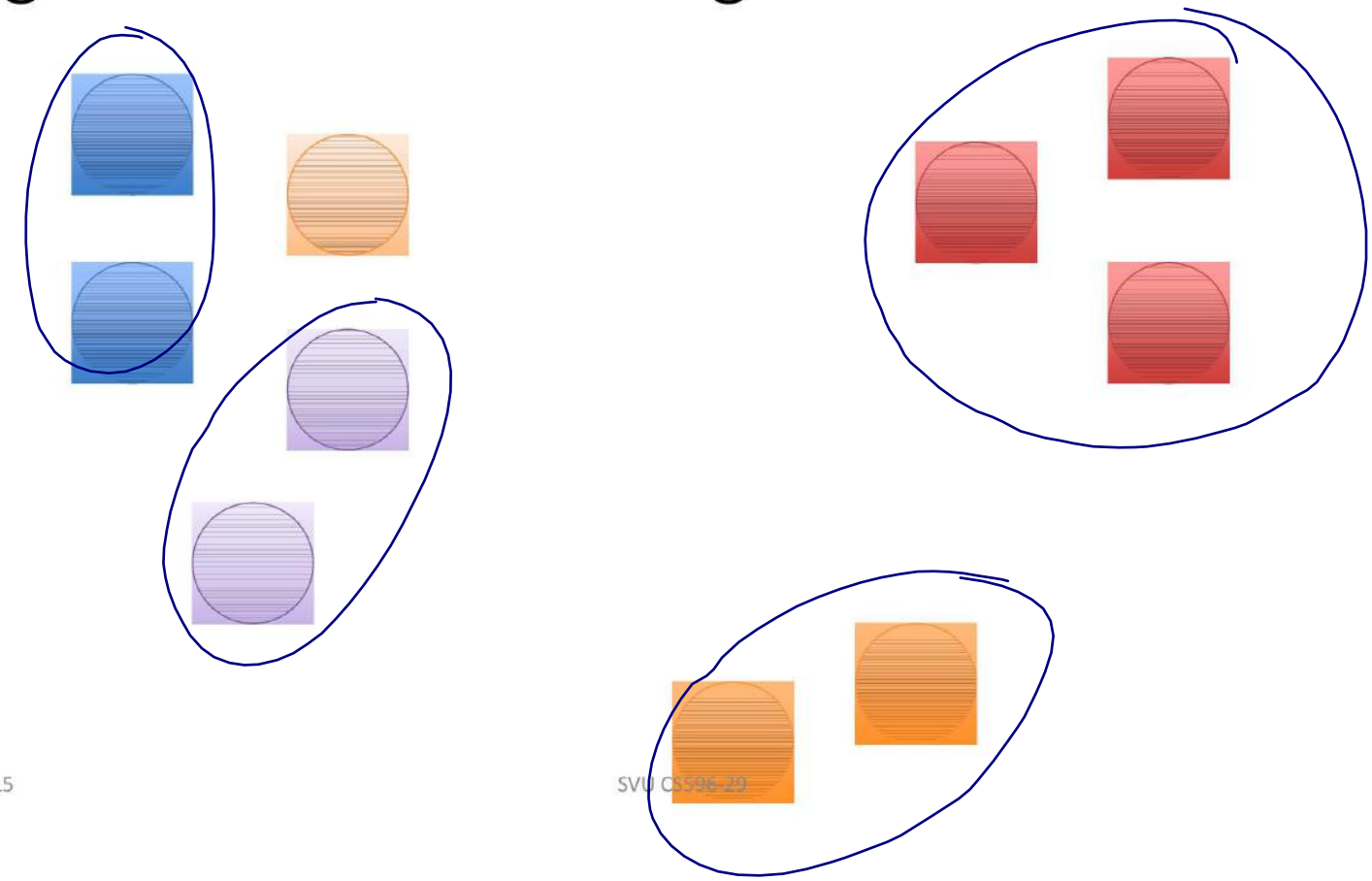
Agglomerative Clustering



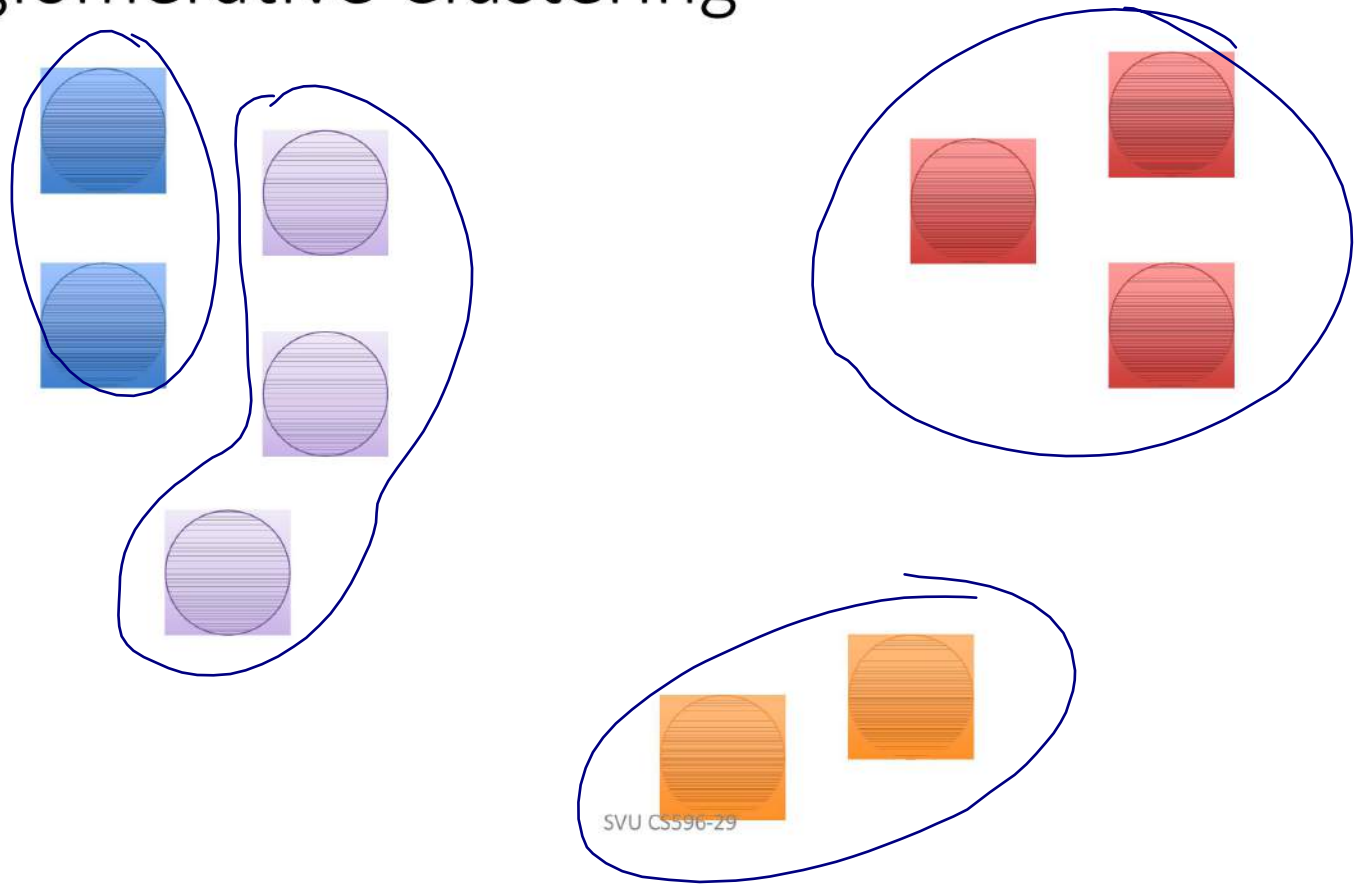
Agglomerative Clustering



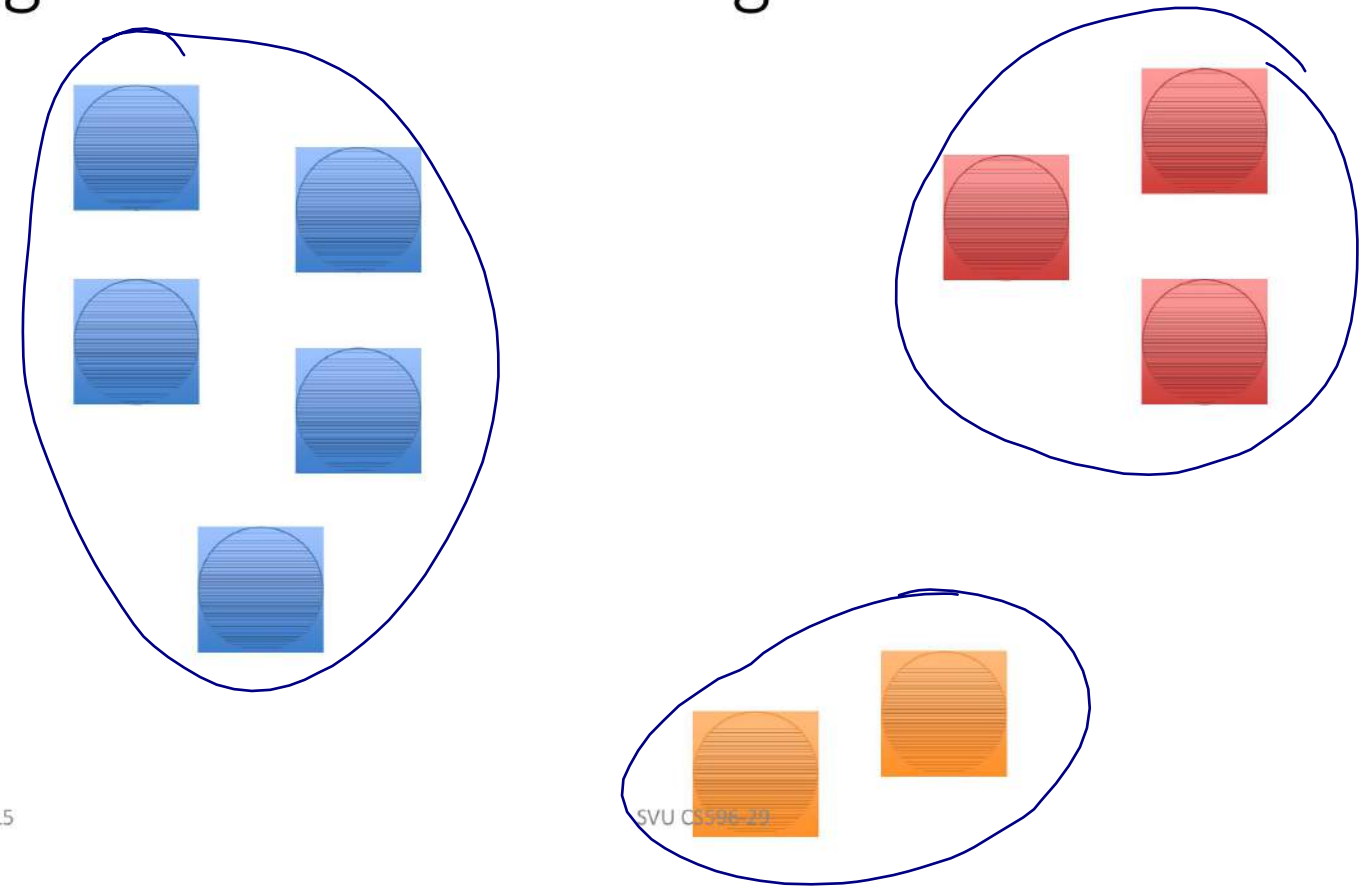
Agglomerative Clustering



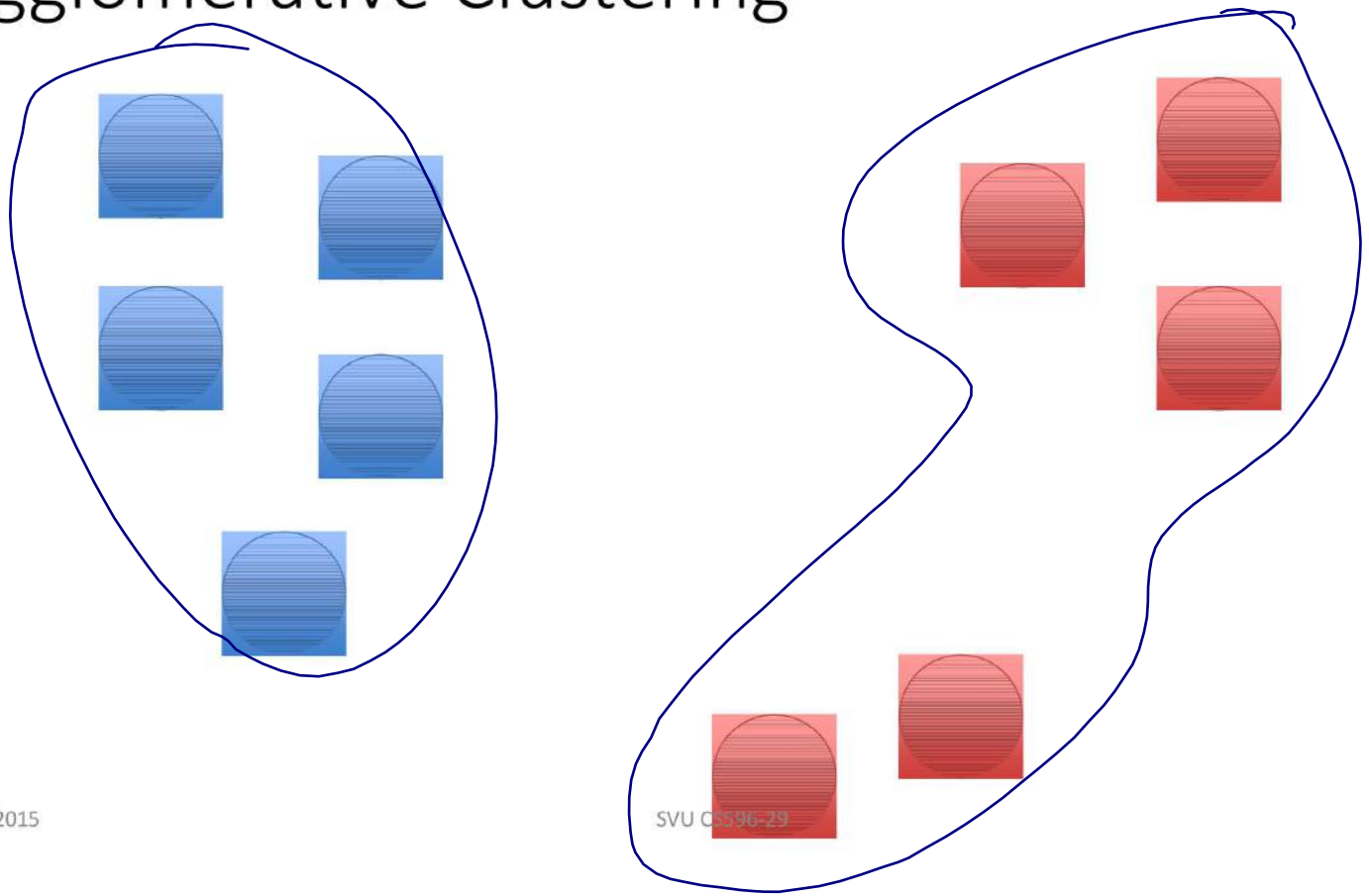
Agglomerative Clustering



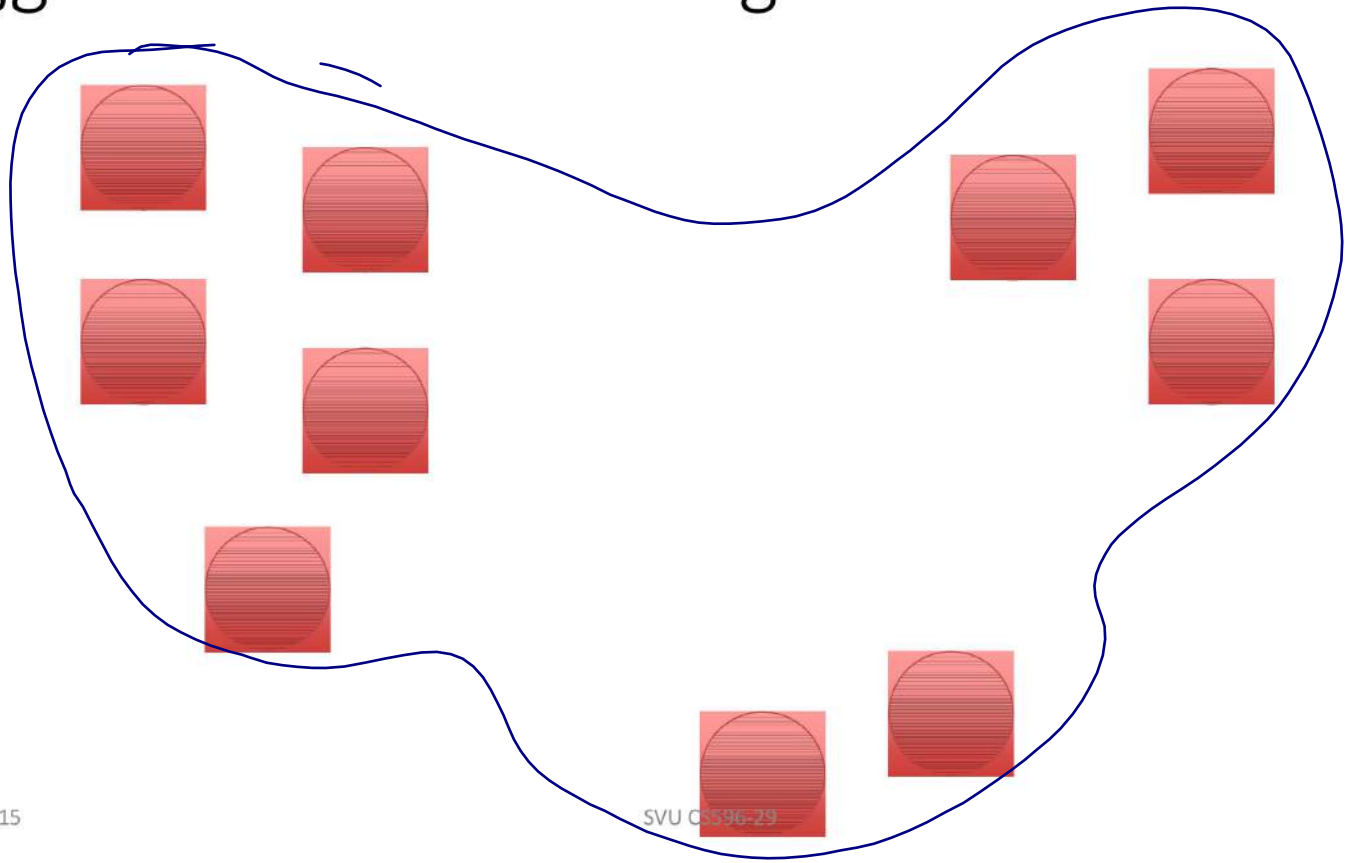
Agglomerative Clustering



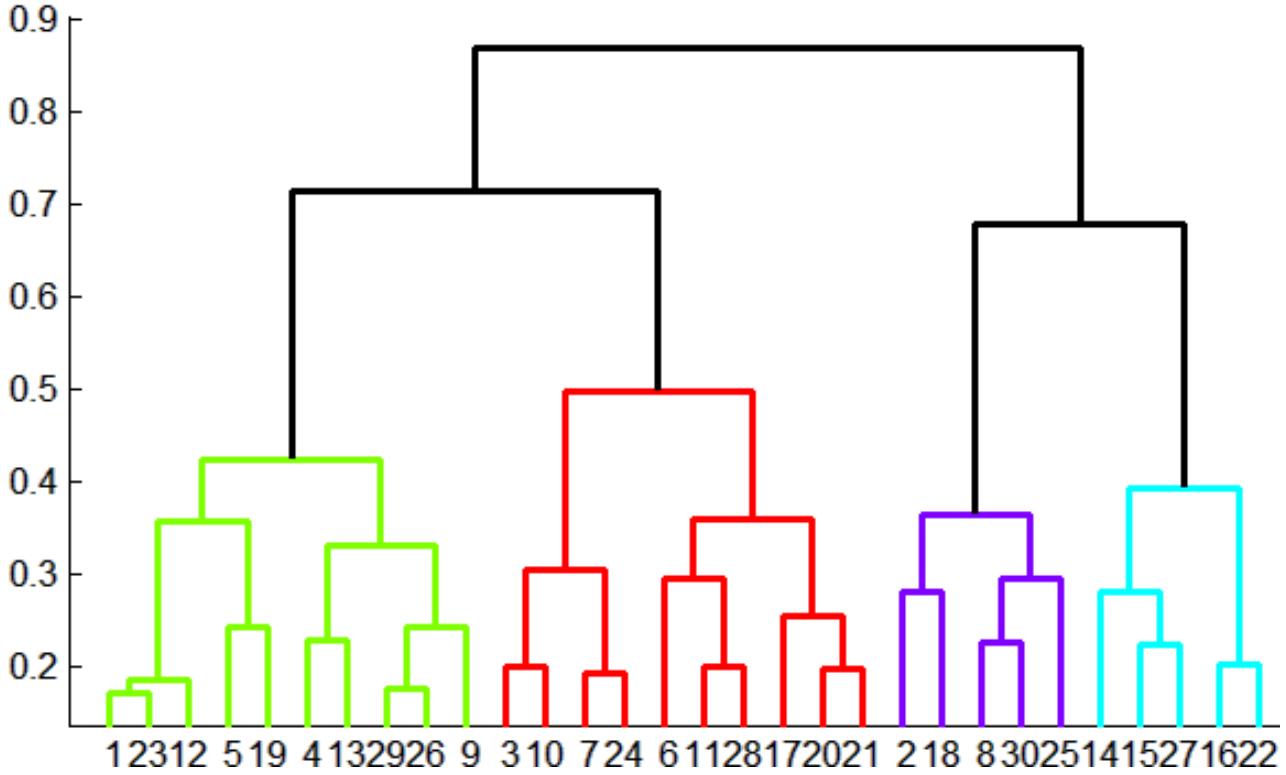
Agglomerative Clustering



Agglomerative Clustering



Dendrogram


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
1 2 3 1 2 5 1 9 4 1 3 2 9 2 6 9 3 1 0 7 2 4 6 1 1 2 8 1 7 2 0 2 1 2 1 8 8 3 0 2 5 1 4 1 5 2 7 1 6 2 2

Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: Basic Concepts
- Partitioning Methods
- Hierarchical Methods
- Density-Based Methods
- Grid-Based Methods
- Evaluation of Clustering 
- Summary

Evaluation of Clustering

Determine the Number of Clusters

- Empirical method
  - # of clusters: $k \approx \sqrt{n/2}$ for a dataset of n points, e.g., n = 200, k = 10
- Elbow method
  - Use the turning point in the curve of sum of within cluster variance w.r.t the # of clusters
- Cross validation method
  - Divide a given data set into *m* parts
  - Use *m* – 1 parts to obtain a clustering model
  - Use the remaining part to test the quality of the clustering
    - E.g., For each point in the test set, find the closest centroid, and use the sum of squared distance between all points in the test set and the closest centroids to measure how well the model fits the test set
  - For any k > 0, repeat it *m* times, compare the overall quality measure w.r.t. different *k's*, and find # of clusters that fits the data the best

Measuring Clustering Quality

- 3 kinds of measures: External, internal and relative
- External: supervised, employ criteria not inherent to the dataset
  - Compare a clustering against prior or expert-specified knowledge (i.e., the ground truth) using certain clustering quality measure
- Internal: unsupervised, criteria derived from data itself
  - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are, e.g., Silhouette coefficient
- Relative: directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm

Cluster Evaluation

- **Intrinsic** Evaluation
  - Measure the compactness of the clusters.
  - or similarity of data points that are assigned to the same cluster
- **Extrinsic** Evaluation
  - Compare the results to some **gold standard** or labeled data.
  - Not covered today.

Intrinsic Evaluation Cluster Variability

- Inter-cluster Variability (IV) ← compare inside.
  - How different are the data points within the same cluster?
- Extra-cluster Variability (EV) ← compare outside
  - How different are the data points in distinct clusters?
- Goal: Minimize IV while maximizing EV

$$\text{Minimize } \frac{IV}{EV} \quad \begin{matrix} \searrow \text{min} \\ \nearrow \text{max} \end{matrix}$$

$$IV = \sum_{C} \sum_{x \in C} d(x, c)$$

$$EV = \frac{1}{N} \sum_{i} \sum_{j} \delta(C(x_i) \neq C(x_j)) d(x_i, x_j)$$

Gaussian Mixture Model (GMM)

K-mean is not an optimal, it only looks at the data

=> Gaussian is better because it involse the behavior of data via generalized function.

The Problem

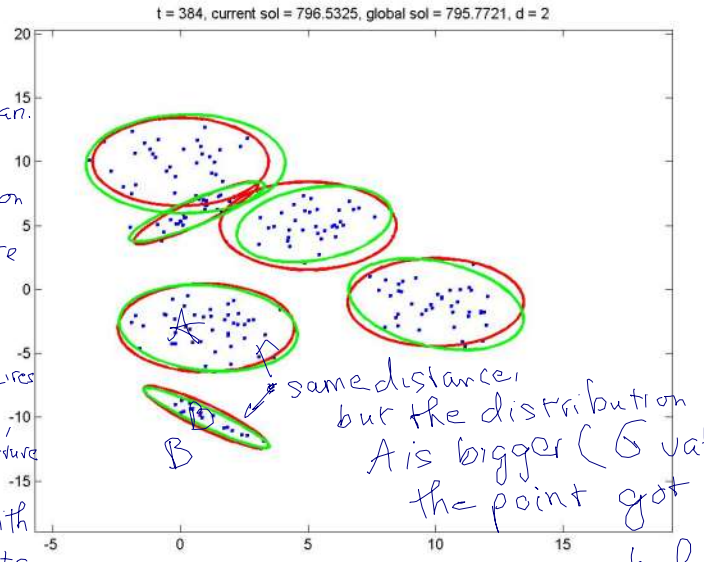
- You have data that you believe is drawn from **n** populations
- You want to identify parameters for each population
- You don't know anything about the populations *a priori*
  - Except you believe that they're Gaussian…

Gaussian Mixture Models

- Rather than identifying clusters by "nearest" centroids (calculate the mean).
- Fit a Set of *k* Gaussians to the data
- Maximum Likelihood over a mixture model

We use Gaussian instead.

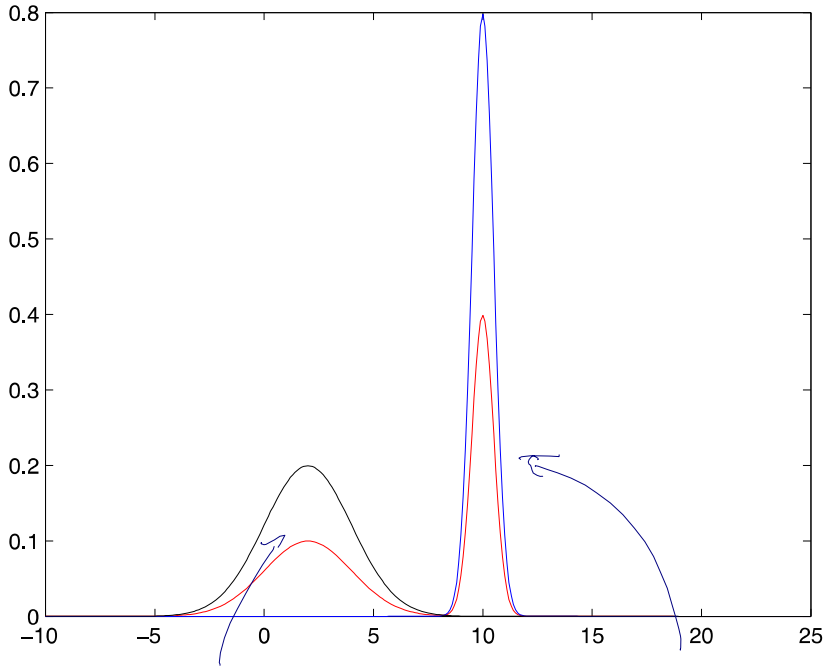
GMM is a probabilistic model
- assumes all datpoints generated from a mixture of a number of Gaussian distribution with unknown parameters.
- implements Expectation-Maximization (EM) algorithm for fitting mixture of Gaussian models.



Pros:
- fastest algo for learning mixtures
- not bias the mean toward zero, or cluster size to specific structure

Cons:
- Singularities: does not converge with insufficient data points.
- number of components: it use ALL components it has access to. Need some held out data.

GMM example



$$f_0(x) = N(x; 2,.2) \qquad f_1(x) = N(x; 10, .5)$$

$$\pi = \begin{bmatrix} .5 & .5 \end{bmatrix}^T$$

Mixture Models

- Formally a Mixture Model is the weighted sum of a number of pdfs where the weights are determined by a distribution, π

$$p(x) = \pi_0 f_0(x) + \pi_1 f_1(x) + \pi_2 f_2(x) + \ldots + \pi_k f_k(x)$$

$$\text{where } \sum_{i=0}^{k} \pi_i = 1 \quad \Rightarrow \quad p(x) = (1-\pi) f_1(x) + \pi f_2(x)$$

$$\boxed{p(x) = \sum_{i=0}^{k} \pi_i f_i(x)}$$

Gaussian Mixture Models = Mixture Model

and $f(x) = \phi_{\theta_j}(x)$

- GMM: the weighted sum of a number of Gaussians where the weights are determined by a distribution, π

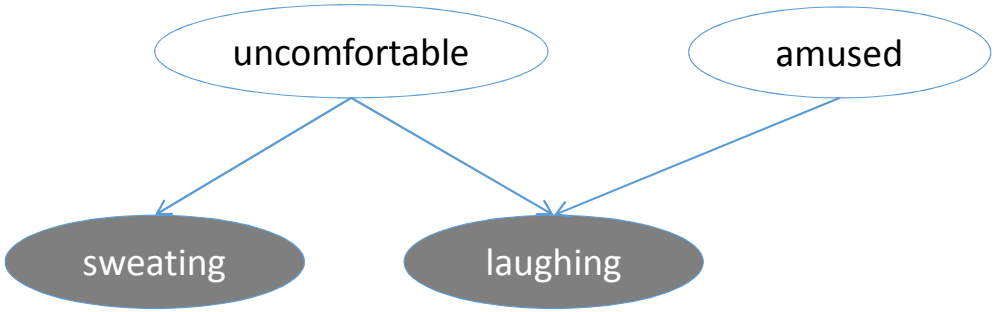
$$p(x) = \pi_0 N(x|\mu_0, \Sigma_0) + \pi_1 N(x|\mu_1, \Sigma_1) + \ldots + \pi_k N(x|\mu_k, \Sigma_k)$$

$$\text{where } \sum_{i=0}^{k} \pi_i = 1$$

$$\boxed{p(x) = \sum_{i=0}^{k} \pi_i N(x|\mu_k, \Sigma_k)}$$

Graphical Models with unobserved variables

- What if you have variables in a Graphical model that are **never** observed?
  - Latent Variables
- Training latent variable models is an unsupervised learning application



- We will train latent variable models using Expectation Maximization

Problem: the GMM does not know which unlabelled points came from which latent component.

Solution: EM, a statistical algorithm, is used to solve this prob.

Loop:
- assumes random components.
- computes for each point a probability of being generated by each component in the model.
- Assign the point
- Tweak the model params

=> converge to local optimum.

Expectation Maximization

- Both the training of GMMs and Graphical Models with latent variables can be accomplished using Expectation Maximization
  - Step 1: Expectation (E-step) ← find the grouping. | Assume random components; Compute probability of each point against the component.
    - Evaluate the "responsibilities" of each cluster with the current parameters
  - Step 2: Maximization (M-step) ← similar to finding the mean in k-mean.
    - Re-estimate parameters using the existing "responsibilities"
- Similar to k-means training. | Assign the point; Tweak the parameters.

Latent Variable Representation

- We can represent a GMM involving a latent variable

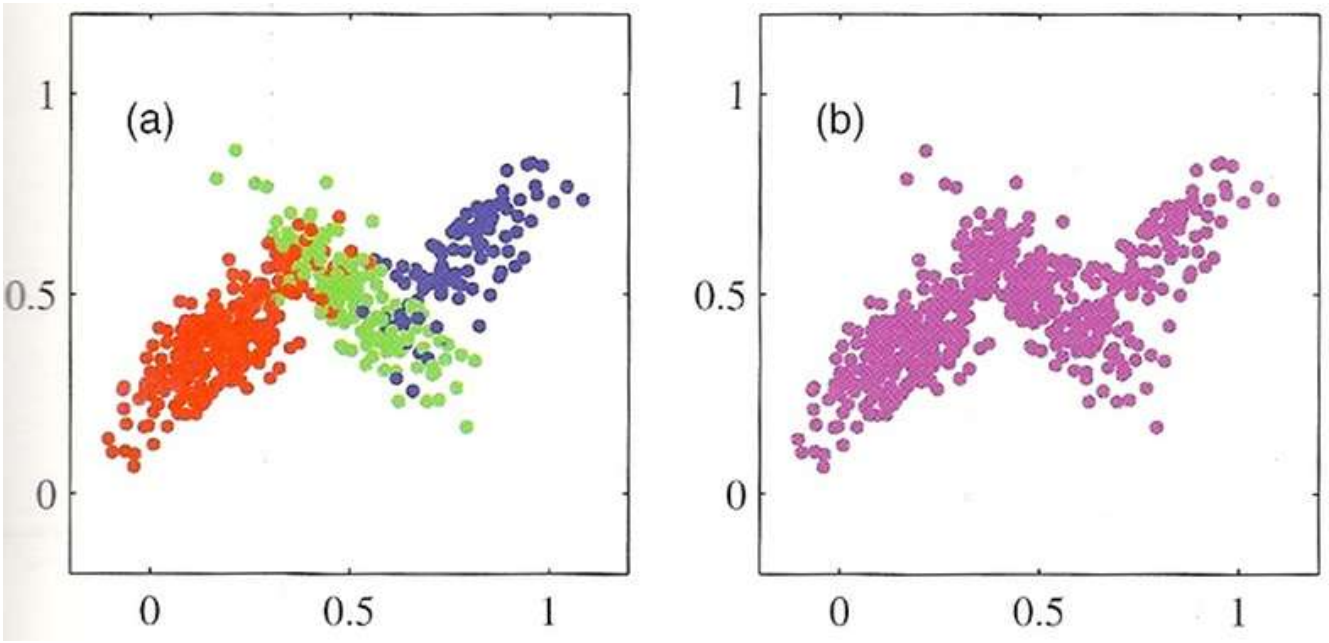
$$p(x) = \sum_{i=0}^{k} \pi_i N(x|\mu_k, \Sigma_k) = \sum_z p(z)p(x|z)$$

$$p(z) = \prod_{k=1}^{K} \pi_k^{z_k} \qquad p(x|z) = \prod_{k=1}^{K} N(x|\mu_k, \Sigma_k)^{z_k}$$

- What does this give us?

TODO: plate notation

GMM data and Latent variables



Last Comment

- We have representations of the joint p(x,z) and the marginal, p(x)…
- The conditional of p(z|x) can be derived using Bayes rule.
  - The **responsibility** that a mixture component takes for explaining an observation x.

$$\tau(z_k) = p(z_k = 1|x) = \frac{p(z_k = 1)p(x|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(x|z_j = 1)}$$

$$= \frac{\pi_k N(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x|\mu_j, \Sigma_j)}$$

Maximum Likelihood over a GMM

- As usual: Identify a likelihood function

$$\ln p(x|\pi,\mu,\Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k) \right\}$$

- And set partials to zero…

Maximum Likelihood of a GMM

- Optimization of means.

$$\ln p(x|\pi,\mu,\Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k) \right\}$$

$$\frac{\partial \ln p(x|\pi,\mu,\Sigma)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_j \pi_j N(x_n|\mu_j,\Sigma_j} \Sigma_k^{-1}(x_k - \mu_k) = 0$$

$$= \sum_{n=1}^{N} \tau(z_{nk}) \Sigma_k^{-1}(x_k - \mu_k) = 0$$

$$\boxed{\mu_k = \frac{\sum_{n=1}^{N} \tau(z_{nk}) x_n}{\sum_{n=1}^{N} \tau(z_{nk})}}$$

Maximum Likelihood of a GMM

- Optimization of covariance

$$\ln p(x|\pi,\mu,\Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n|\mu_k,\Sigma_k) \right\}$$

$$\Sigma_k = \frac{1}{\sum_{n=1}^{N} \tau(z_{nk})} \sum_{n=1}^{N} \tau(z_{nk})(x_k-\mu_k)(x_k-\mu_k)^T$$

- Note the similarity to the regular MLE without **responsibility terms**.

Maximum Likelihood of a GMM

- Optimization of mixing term

$$\ln p(x|\pi,\mu,\Sigma) + \lambda\left(\sum_{k=1}^{K}\pi_k - 1\right)$$

$$0 = \sum_{n=1}^{N}\frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_j \pi_j N(x_n|\mu_j,\Sigma_j} + \lambda$$

$$\boxed{\pi_k = \frac{\sum_{n=1}^{N}\tau(z_n k)}{N}}$$

MLE of a GMM

$$\mu_k = \frac{\sum_{n=1}^{N} \tau(z_{nk}) x_n}{N_k}$$

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \tau(z_{nk})(x_k - \mu_k)(x_k - \mu_k)^T$$

$$\pi_k = \frac{N_k}{N}$$

$$N_k = \sum_{n=1}^{N} \tau(z_n k)$$

EM for GMMs

- Initialize the parameters
  - Evaluate the log likelihood
- Expectation-step: Evaluate the responsibilities
- Maximization-step: Re-estimate Parameters
  - Evaluate the log likelihood
  - Check for convergence

EM for GMMs

- E-step: Evaluate the Responsibilities of model-k to point-n

$$\tau(z_{nk}) = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_j, \Sigma_j)}$$

EM for GMMs

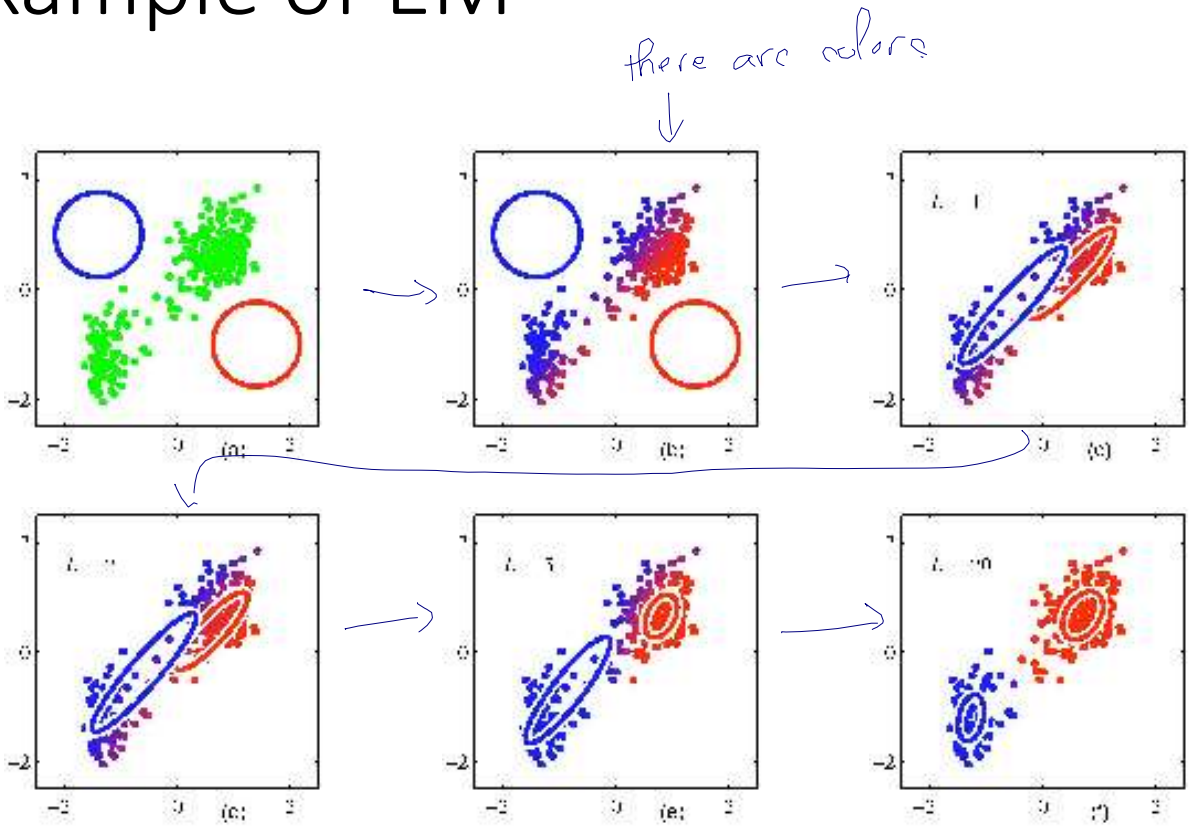
- M-Step: Re-estimate Parameters

$$\mu_k^{new} = \frac{\sum_{n=1}^{N} \tau(z_{nk}) x_n}{N_k}$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \tau(z_{nk})(x_k - \mu_k^{new})(x_k - \mu_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

Visual example of EM

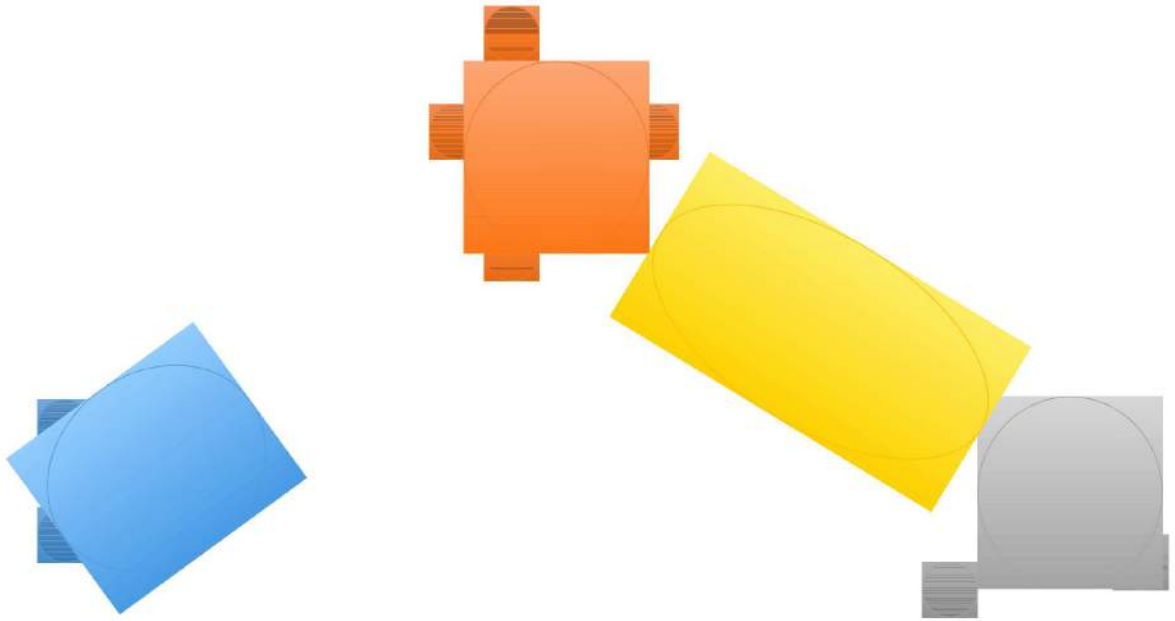


Potential Problems

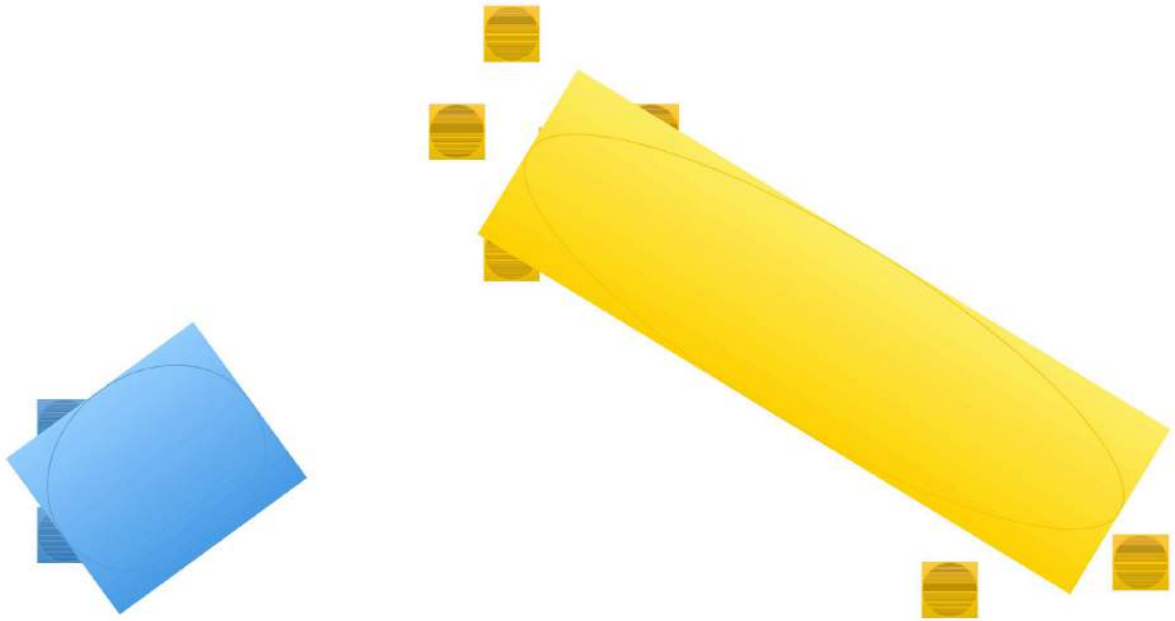
- Incorrect number of Mixture Components

- Singularities

Incorrect Number of Gaussians



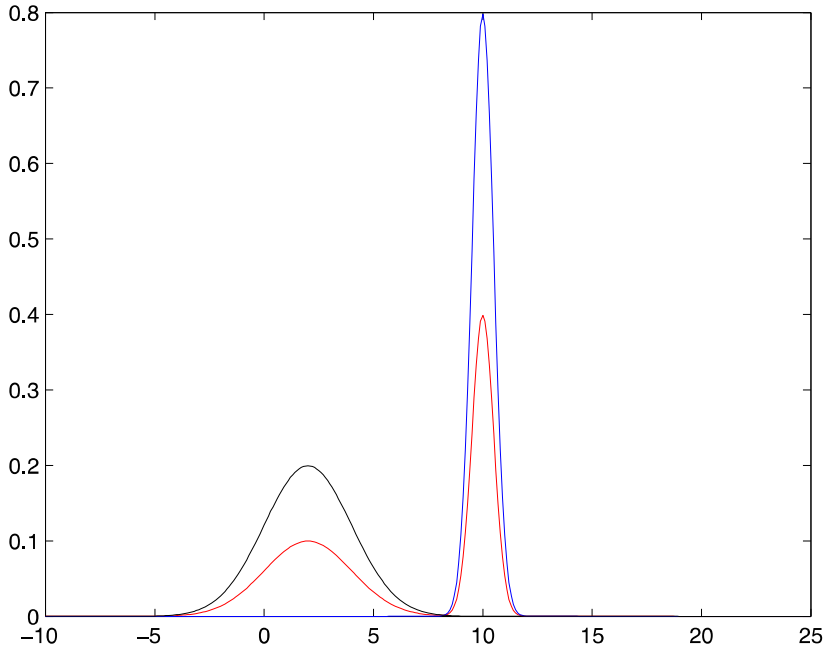
Incorrect Number of Gaussians



Singularities

- A minority of the data can have a disproportionate effect on the model likelihood.
- For example…

GMM example



$$f_0(x) = N(x; 2, 2) \qquad f_1(x) = N(x; 10, .5)$$

$$\pi = \begin{bmatrix} .5 & .5 \end{bmatrix}^T$$

Singularities

- When a mixture component collapses on a given point, the mean becomes the point, and the variance goes to zero.
- Consider the likelihood function as the covariance goes to zero.

- The likelihood approaches infinity.

$$N(x_n|x_n, \sigma^2 I) = \frac{1}{\sqrt{2\pi}} \frac{1}{\sigma_j}$$

$$p(x) = \sum_{i=0}^{k} \pi_i N(x|\mu_k, \Sigma_k)$$

Relationship to K-means

- K-means makes **hard** decisions.
  - Each data point gets assigned to a single cluster.
- GMM/EM makes **soft** decisions.
  - Each data point can yield a posterior p(z|x)
- Soft K-means is a special case of EM.

Soft means as GMM/EM

- Assume equal covariance matrices for every mixture component: $\epsilon \mathbf{I}$
- Likelihood:

$$p(x|\mu_k, \Sigma_k) = \frac{1}{(2\pi\epsilon)^{M/2}} \exp\left\{-\frac{1}{2\epsilon}\|x - \mu_k\|^2\right\}$$

- Responsibilities:

$$\tau(z_{nk}) = \frac{\pi_k \exp\left\{-\|x_n - \mu_k\|^2/2\epsilon\right\}}{\sum_j \pi_j \exp\left\{-\|x_n - \mu_j\|^2/2\epsilon\right\}}$$

- As epsilon approaches zero, the responsibility approaches unity.

Soft K-Means as GMM/EM

- Overall Log likelihood as epsilon approaches zero:

$$\mathbb{E}_z[\ln p(X, Z|\mu, \Sigma, \pi)] \rightarrow -\frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|x_n - \mu_k\|^2 + const.$$

- The expectation of soft k-means is the intercluster variability
- Note: only the means are reestimated in Soft K-means.
  - The covariance matrices are all tied.

General form of EM

- Given a joint distribution over observed and latent variables: $p(X, Z|\theta)$
- Want to maximize: $p(X|\theta)$

1. Initialize parameters θ^{old}
2. E Step: Evaluate: $p(Z|X, \theta^{old})$
3. M-Step: Re-estimate parameters (based on expectation of complete-data log likelihood)

$$\theta^{new} = \text{argmax}_\theta \sum_Z p(Z|X, \theta^{old}) \ln p(X, Z|\theta)$$

4. Check for convergence of params or likelihood

Backup Slides

Extensions to Hierarchical Clustering

- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies)

- Zhang, Ramakrishnan & Livny, SIGMOD'96
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - Phase 1: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - Phase 2: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness:* handles only numeric data, and sensitive to the order of the data record

Clustering Feature Vector in BIRCH

Clustering Feature (CF): ***CF = (N, LS, SS)***

N: **Number of data points**

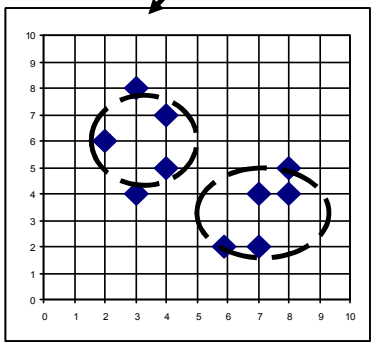
LS: linear sum of N points:

$$\sum_{i=1}^{N} X_i$$

SS: square sum of N points

$$\sum_{i=1}^{N} X_i^2$$

CF = (5, (16,30),(54,190))

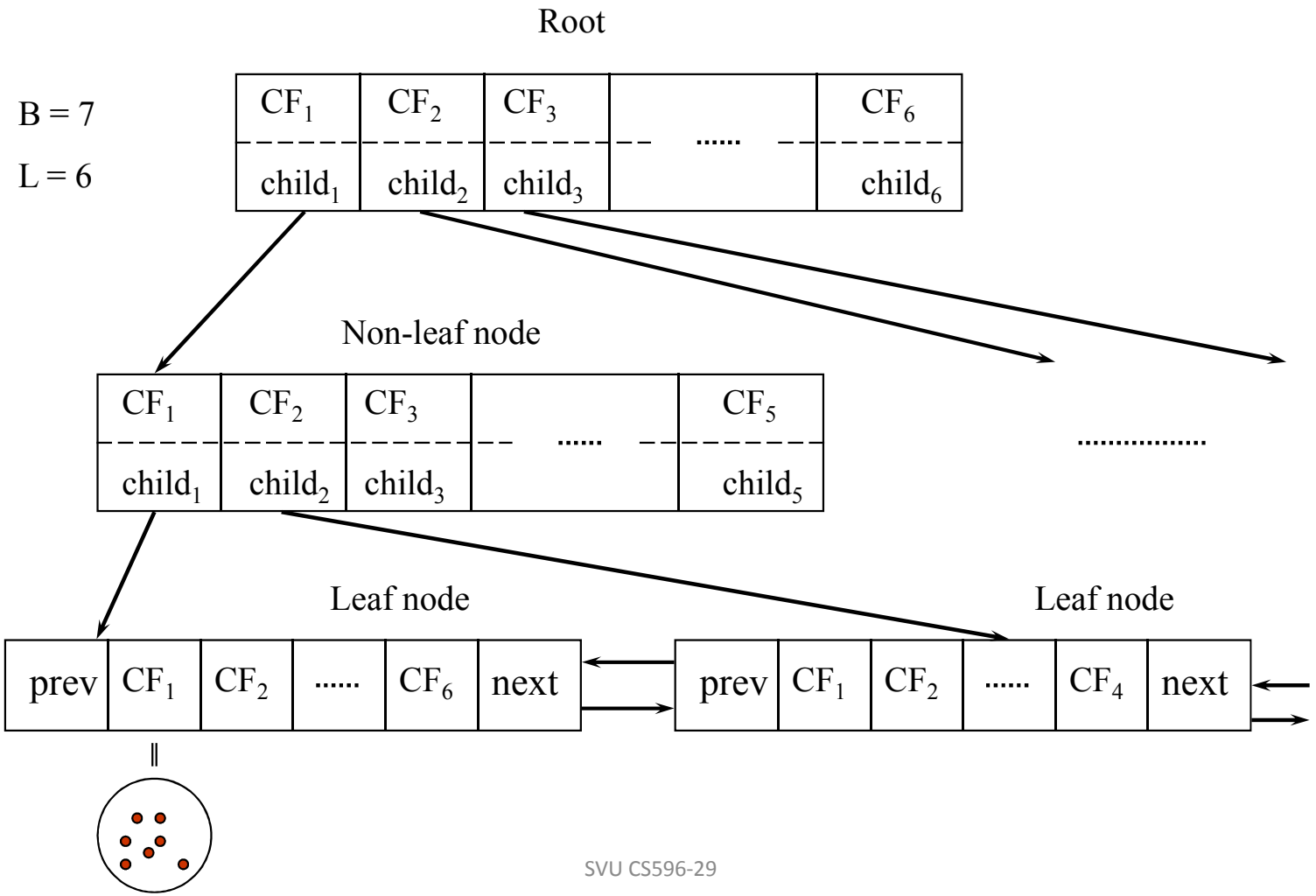


(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

CF-Tree in BIRCH

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently
- A CF tree is a height-balanced tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or "children"
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two parameters
  - Branching factor: max # of children
  - Threshold: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure



The Birch Algorithm

- Cluster Diameter

$$\sqrt{\frac{1}{n(n-1)}\sum (x_i - x_j)^2}$$

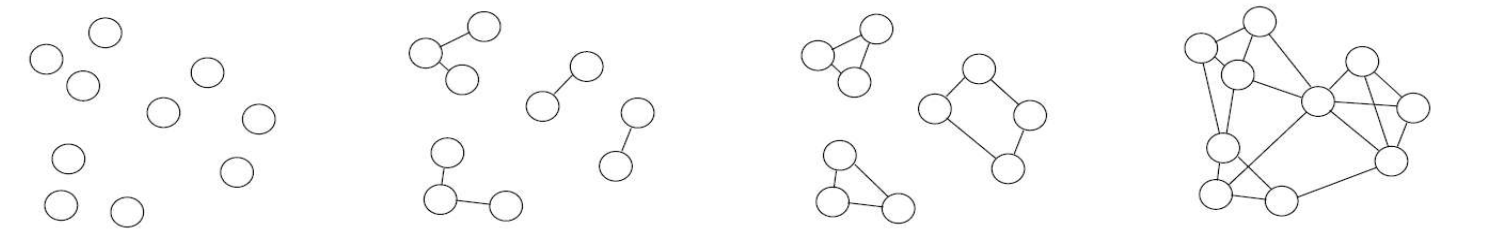
- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If entry diameter > max_diameter, then split leaf, and possibly parents
- Algorithm is O(n)
- Concerns
  - Sensitive to insertion order of data points
  - Since we fix the size of leaf nodes, so clusters may not be so natural
  - Clusters tend to be spherical given the radius and diameter measures

CHAMELEON: Hierarchical Clustering Using Dynamic Modeling

- CHAMELEON: G. Karypis, E. H. Han, and V. Kumar, 1999
- Measures the similarity based on a dynamic model
  - Two clusters are merged only if the *interconnectivity* and *closeness (proximity)* between two clusters are high *relative to* the internal interconnectivity of the clusters and closeness of items within the clusters
- Graph-based, and a two-phase algorithm
  1. Use a graph-partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
  2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

KNN Graphs & Interconnectivity

- k-nearest graphs from an original data in 2D:



(a) Original Data in 2D (b) 1-nearest neighbor graph (c) 2-nearest neighbor graph (d) 3-nearest neighbor graph

- $EC_{\{Ci,Cj\}}$: The absolute inter-connectivity between C_i *and* C_j*: the sum* of the weight of the edges that connect vertices in C_i *to vertices in* C_j
- Internal inter-connectivity of a cluster C_i : *the size of its min-cut bisector* EC_{Ci} *(i.e., the weighted sum* of edges that partition the graph into two roughly equal parts)
- Relative Inter-connectivity (RI):

$$RI(C_i, C_j) = \frac{|EC_{\{C_i,C_j\}}|}{\frac{|EC_{C_i}|+|EC_{C_j}|}{2}}$$

Relative Closeness & Merge of Sub-Clusters

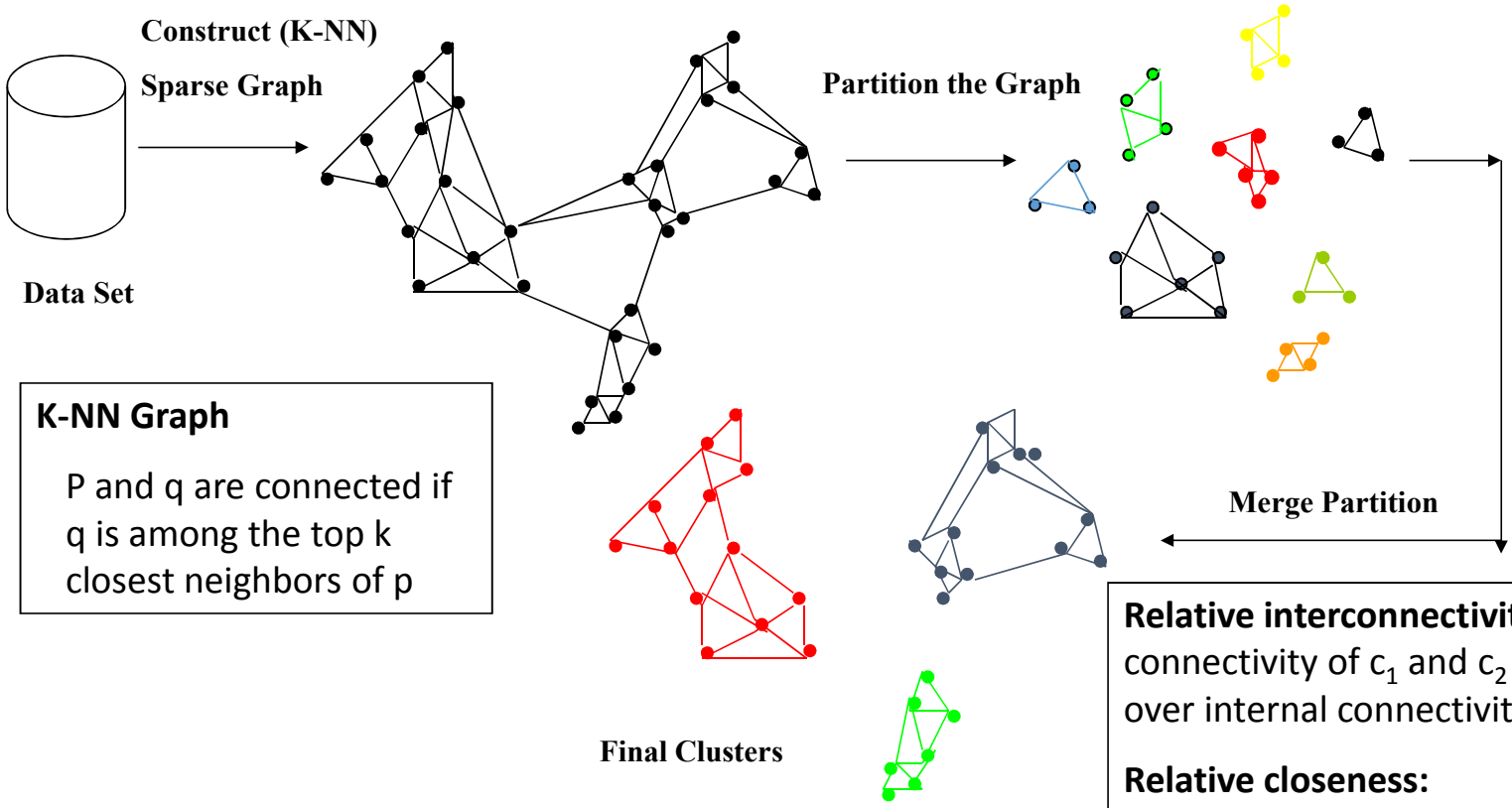
- **Relative closeness** between a pair of clusters C_i *and* C_j *: the absolute closeness* between C_i *and* C_j *normalized w.r.t. the internal closeness of the two clusters* C_i *and* C_j

$$RC(C_i, C_j) = \frac{\overline{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\overline{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i|+|C_j|}\overline{S}_{EC_{C_j}}}$$

  - $\overline{S}_{EC_{C_i}}$ and $\overline{S}_{EC_{C_j}}$ are the average weights of the edges that belong in the min-cut bisector of clusters C_i *and* C_j *,* respectively, and $\overline{S}_{EC_{\{C_i,C_j\}}}$ is the average weight of the edges that connect vertices in C_i *to vertices in* C_j

- **Merge Sub-Clusters:**
  - Merges only those pairs of clusters whose RI and RC are both above some user-specified thresholds
  - Merge those maximizing the function that combines RI and RC

Overall Framework of CHAMELEON



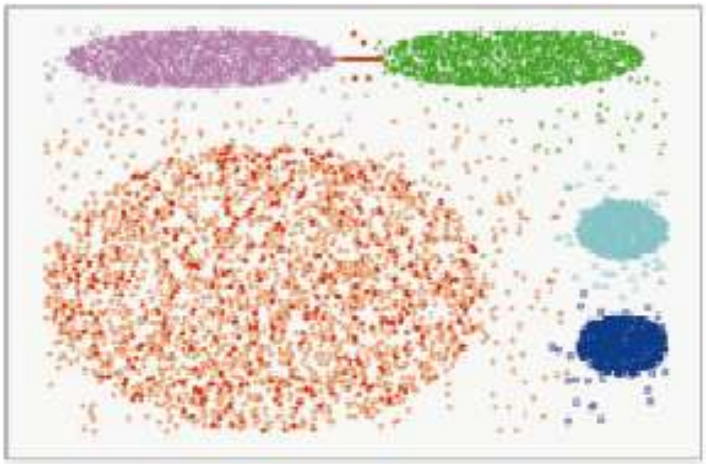
K-NN Graph

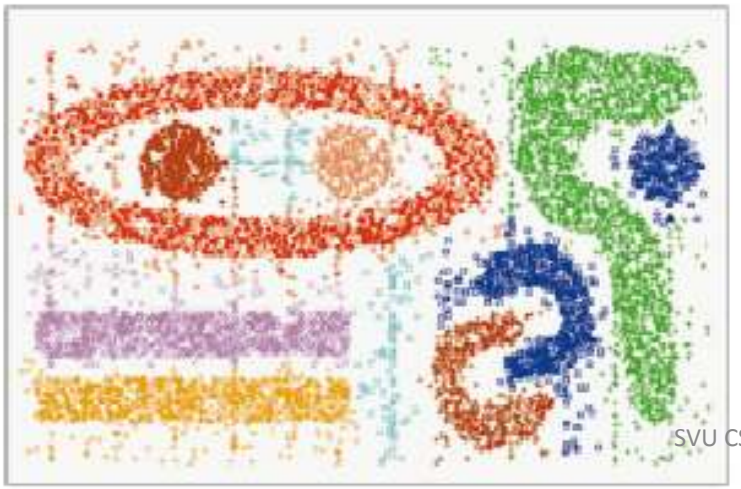
P and q are connected if q is among the top k closest neighbors of p

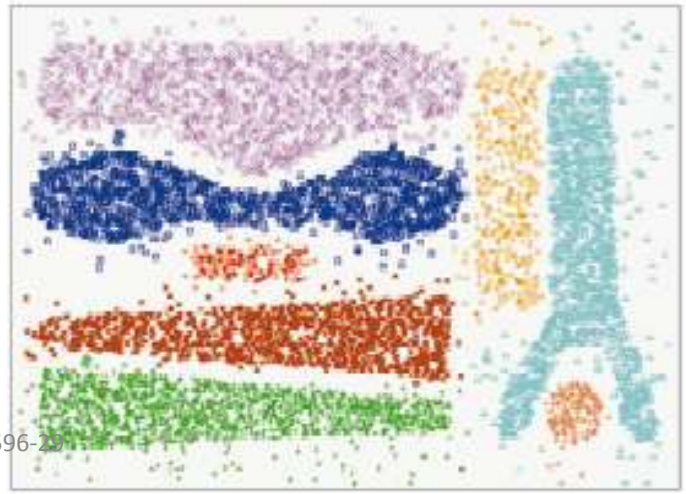
Relative interconnectivity: connectivity of c_1 and c_2 over internal connectivity

Relative closeness: closeness of c_1 and c_2 over internal closeness

CHAMELEON (Clustering Complex Objects)







Probabilistic Hierarchical Clustering

- Algorithmic hierarchical clustering
  - Nontrivial to choose a good distance measure
  - Hard to handle missing attribute values
  - Optimization goal not clear: heuristic, local search
- Probabilistic hierarchical clustering
  - Use probabilistic models to measure distances between clusters
  - Generative model: Regard the set of data objects to be clustered as a sample of the underlying data generation mechanism to be analyzed
  - Easy to understand, same efficiency as algorithmic agglomerative clustering method, can handle partially observed data
- In practice, assume the generative models adopt common distribution functions, e.g., Gaussian or Bernoulli distribution, governed by parameters

Generative Model

- Given a set of 1-D points $X = \{x_1, \ldots, x_n\}$ for clustering analysis & assuming they are generated by a Gaussian distribution:

$$\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- The probability that a point $x_i \in X$ is generated by the model

$$P(x_i|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- The likelihood that X is generated by the model:

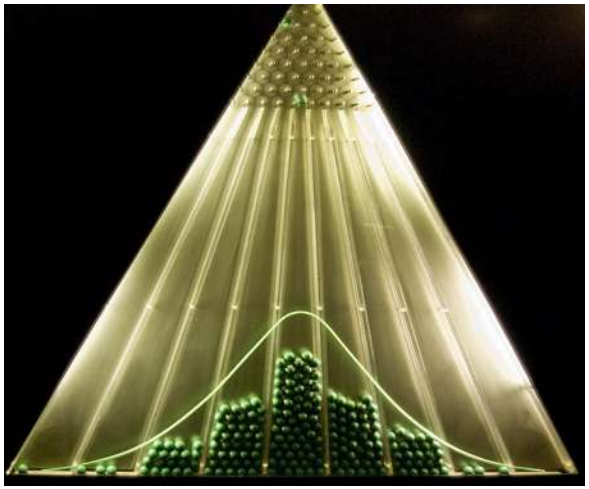
$$L(\mathcal{N}(\mu, \sigma^2) : X) = P(X|\mu, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- The task of learning the generative model: find the parameters μ and σ^2 such that

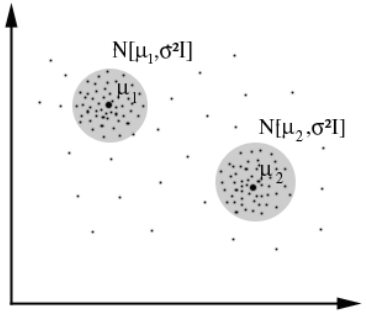
the maximum likelihood

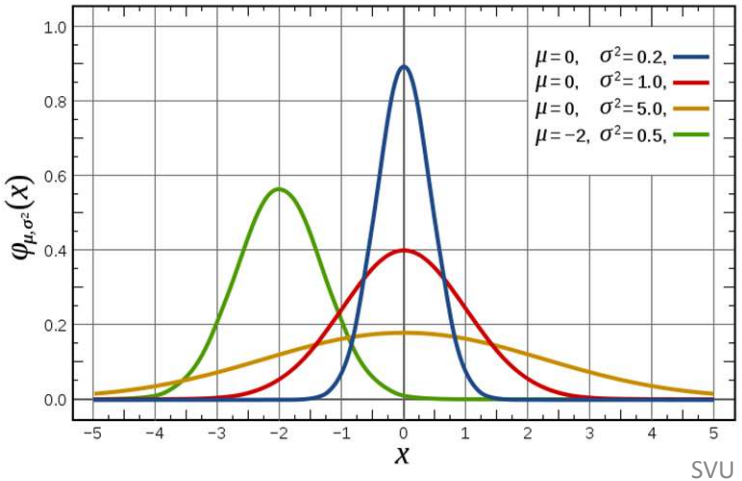
$$\mathcal{N}(\mu_0, \sigma_0^2) = \arg\max\{L(\mathcal{N}(\mu, \sigma^2) : X)\}$$

Gaussian Distribution

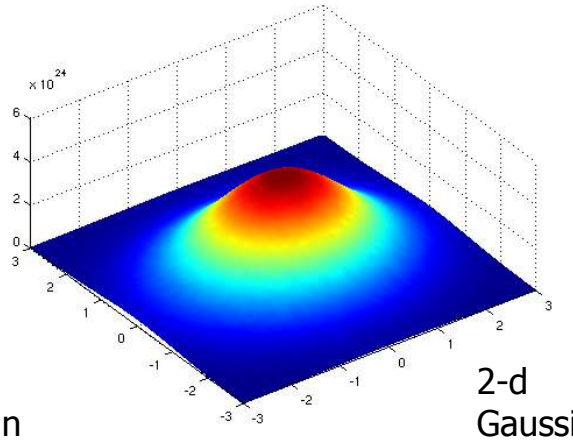


Bean machine: drop ball with pins





1-d Gaussian



2-d Gaussian

From wikipedia and http://home.dei.polimi.it