

SVM Homework

Linear Kernel

Number of SV: 623

Accuracy on training set: 68.15%

Accuracy on test set: 71.28%

Precision/recall on test set: 83.82%/33.84%

Training set:

359 spam

601 not-spam

Test set:

1454 spam

2227 not-spam

Confusion Matrix:

	Predicted class	
	Spam	Not Spam
True class Spam	492	962
Not-Spam	588	1641

SVM Homework

Polynomial Kernel (d=3)

Number of SV: 0, b = 0

Accuracy on training set: 0%

Accuracy on test set: 60.5%

Precision/recall on test set: nan%/0%

Training set:

359 spam

561 not-spam

Test set:

1454 spam

2227 not-spam

Confusion Matrix (test set):

	Predicted class	
	Spam	Not Spam
True class Spam	0	1454
Not-Spam	0	2227

SVM Homework

Radial Basis Function Kernel

Number of SV: 899

Accuracy on training set: 73.37%

Accuracy on test set: 61.29%

Precision/recall on test set: 100%/1.99%

Training set:

359 spam

561 not-spam

Test set:

1454 spam

2227 not-spam

Confusion Matrix (test set):

	Predicted class	
	Spam	Not Spam
True class Spam	29	1425
Not-Spam	0	2227

- Best generalization performance?
- Number of support vectors related to generalization performance?
- Evidence of overfitting?
- Why is precision larger than recall?
- Why is classification accuracy on test data sometimes higher than on training data?

Clustering, part 2

How to evaluate clusters produced by k -means?

Hard to evaluate if we don't already know the correct clusters

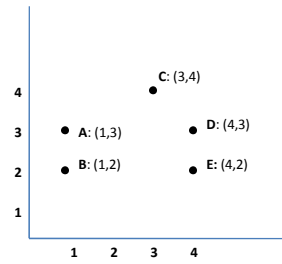
Types of evaluation

- Unsupervised:
 - cluster cohesion (compactness)
 - cluster separation (isolation)

- Supervised
 - How well clusters match externally supplied class labels

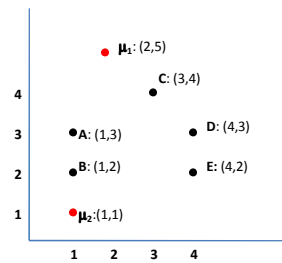
Unsupervised evaluation measures

Example:



Unsupervised evaluation measures

Example:



Unsupervised evaluation measures

Residual Sum of Squares (RSS)

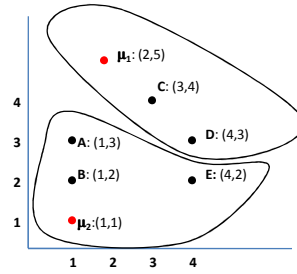
$$RSS = \sum_k \sum_{x \in \text{cluster } k} \text{distance}(x, \mu_k)^2 = 25$$

$$\text{cohesion}(C_i) = \frac{1}{\sum_{x,y \in C_i} (\text{distance}(x,y))} = \begin{cases} .71 \\ .14 \end{cases}$$

Average cohesion = .425

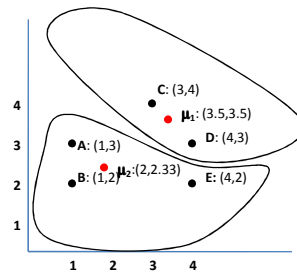
$$\text{separation}(C_i, C_j) = \sum_{x \in C_i, y \in C_j} (\text{distance}(x,y)) = 13.4$$

Example:



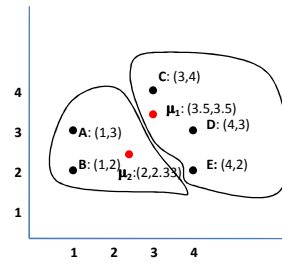
Unsupervised evaluation measures

Example:



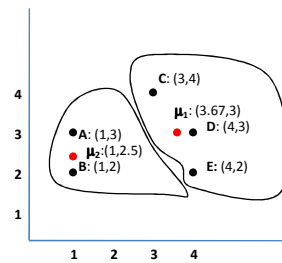
Unsupervised evaluation measures

Example:



Unsupervised evaluation measures

Example:



Unsupervised evaluation measures

Residual Sum of Squares (RSS)

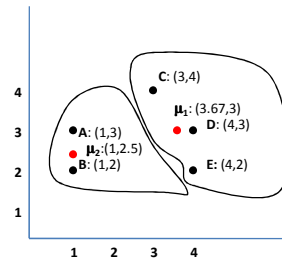
$$RSS = \sum_k \sum_{x \in \text{cluster } k} \text{distance}(x, \mu_k)^2 = 4.1$$

$$\text{cohesion}(C_i) = \frac{1}{\sum_{x,y \in C_i} (\text{distance}(x,y))} = \begin{cases} .22 \\ 1 \end{cases}$$

Average cohesion = .61

$$\text{separation}(C_i, C_j) = \sum_{x \in C_i, y \in C_j} (\text{distance}(x,y)) = 17.4$$

Example:



Supervised evaluation measures

- **Entropy:** Degree to which each cluster consists of objects of a single class.

Let $p_{ij} = m_{ij}/m_i$, where m_i is the number of objects in cluster C_i , m_{ij} is the number of objects of class j in cluster C_i , and L is the number of classes. Thus p_{ij} is the probability that a member of cluster C_i is in class j .

$$\text{entropy}(C_i) = - \sum_{j=1}^L p_{ij} \log_2 p_{ij}$$

Examples:

Assume two classes.



Entropy?



Entropy?

- **Precision:** Fraction of a cluster C_i that consists of objects of a specified class j .

$$precision(C_i, j) = p_{ij}$$

- **Recall:** Fraction of objects of class j that are contained in cluster C_i .

$$recall(C_i, j) = \frac{m_{ij}}{m_j}$$

- **F-measure:** Extent to which a cluster C_i contains *only* objects of a particular class j and *all* objects of that class.

$$f(C_i, j) = \frac{2 \times precision(C_i, j) \times recall(C_i, j)}{precision(C_i, j) + recall(C_i, j)}$$

Example: Assume two clusters, C_i and C_j .

What is $f(C_i, j)$ if $p_{ij} = 0$?



What is $f(C_i, j)$ if $p_{ij} = 1$?



What is $f(C_i, j)$ if one half the instances in cluster i are in class j ?



Finding the K for K-means ("Cluster Cardinality")

- How to decide K ?

Residual Sum of Squares (RSS)

$$RSS = \sum_k \sum_{x \in \text{cluster } k} \text{distance}(\mathbf{x}, \boldsymbol{\mu}_k)^2$$

- Naive guess: use K that minimizes unsupervised evaluation measures

– Problem?

$$\text{cohesion}(C_i) = \frac{1}{\sum_{x, y \in C_i} \text{distance}(\mathbf{x}, \mathbf{y})}$$

- Other ideas?

$$\text{separation}(C_i, C_j) = \sum_{x \in C_i, y \in C_j} \text{distance}(\mathbf{x}, \mathbf{y})$$

Cluster cardinality

- One method: impose a penalty for “model complexity” (i.e., too many clusters):

$$K = \arg \min_k [RSS_{\min}(k) + \lambda k]$$

But, how to determine λ ? Can use information-theoretic arguments

- e.g., “Akaike information criterion” – see textbook – where λ is set to $2Mk$, where M is the dimensionality of the vectors.
- Many other possible methods for determining K

Model-Based Clustering

- “Discriminative” vs. “generative” models
 - Discriminative model: Find a discrimination “surface” that separates classes.
 - Generative model: Assume data was generated by a model and attempt to recover that model.
- K-means as a “generative” model
 - Model is the set of centroids
 - Assumes normal distribution and “spherical” covariance

Gaussian Mixture Models (GMMs): A “soft” version of K -means

K -means:

Assumption: Data is clustered about K centroids, which are the means of the data in the cluster

Goal: Find the centroids that best cluster the data.

Method: Guess the centroids. Then, alternate:

1. Assign each data point to a centroid
2. Update the centroid to be the mean of the new cluster

Stopping criteria:

- Data points don't change clusters
- Centroids don't change location

GMMs:

Assumption: Each data point is generated by sampling from one of K Gaussians, with some probability distribution over the Gaussians.

Goal: Find parameters (μ , σ^2) of these Gaussians and probability distribution over these Gaussians that had the highest probability (“maximum likelihood”) of generating the given data.

Method: Expectation-Maximization (EM)

algorithm: Guess the parameters and probability distribution. Then, alternate:

E step: For each data point, calculate the “responsibility” of each Gaussian, using the current parameter values.

M step: Using the calculated “responsibilities”, reestimate the current parameters.

Stopping criteria: Analogous to K -means

Normal (Gaussian) distribution

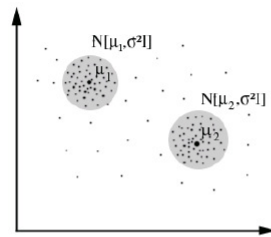
- Univariate:

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

- Multivariate

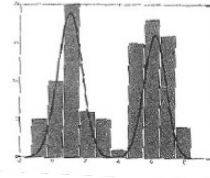
$$N(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Gaussian Mixture Models (GMMs)



To generate data:

- Choose one of the Gaussians with probability α_i
- Sample a point from that Gaussian



From: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/mixture.html

Gaussian Mixture Models

- Let X be a set of multivariate data points (vectors):
 $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$.
- General expression for Gaussian mixture model (with K Gaussians):

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$\text{where } \sum_{k=1}^K \pi_k = 1$$

and $\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is a Gaussian with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$.

Text classification from labeled and unlabeled documents using EM

K. Nigam et al., *Machine Learning*, 2000

- Big problem with text classification: need labeled data.
- What we have: lots of unlabeled data.
- Question of this paper: Can unlabeled data be used to increase classification accuracy?
- I.e.: Any information implicit in unlabeled data? Any way to take advantage of this implicit information?

General idea: A version of EM algorithm

- Train a classifier with small set of available labeled documents.
- Use this classifier to assign probabilistically-weighted class labels to unlabeled documents.
- Then train a new classifier using all the documents, both originally labeled and formerly unlabeled.
- Iterate.

Probabilistic framework

- Assumes data are generated with Gaussian mixture model
- Assumes one-to-one correspondence between mixture components and classes.
- “These assumptions rarely hold in real-world text data”

Probabilistic framework

Let $C = \{c_1, \dots, c_K\}$ be the classes / mixture components

Let $\theta = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K\} \cup \{\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K\} \cup \{\pi_1, \dots, \pi_K\}$ be the mixture parameters.

Assumptions: A document d_i is created by first selecting a mixture component according to the mixture weights π_j , then having this selected mixture component generate a document according to its own parameters, with distribution

$$p(d_i | c_j; \theta).$$

- Likelihood of document d_i :

$$p(d_i | \theta) = \sum_{j=1}^K \pi_k p(d_i | c_j; \theta)$$

- Now, we will apply EM to a naive Bayes Classifier

Recall naive Bayes classifier: Assume each attribute is conditionally independent, given c_j

Let $\mathbf{x} = (a_1, a_2, \dots, a_D)$

We have:

$$p(a_1, a_2, \dots, a_D | c_j) = p(a_1 | c_j) p(a_2 | c_j) \cdots p(a_D | c_j)$$

$$p(c_j | \mathbf{x}) = p(c_j) \prod_i p(a_i | c_j), i = 1, \dots, D; j = 1, \dots, K$$

To “train” naive Bayes from labeled data, estimate

$$p(c_j) \text{ and } p(a_i | c_j), j = 1, \dots, K; i = 1, \dots, D$$

These values are estimates of the parameters in θ . Call these values $\hat{\theta}$.

Note that Naive Bayes can be thought of as a generative mixture model.

Document d_i is represented as a vector of word frequencies $(w_1, \dots, w_{|V|})$, where V is the vocabulary (all known words).

There is an assumed probability distribution over words associated with each class, parameterized by θ .

We need to find estimate $\hat{\theta}$ to determine what probability distribution document $d_i = (w_1, \dots, w_{|V|})$ is most likely to come from.

Applying EM to Naive Bayes

- We have a small number of labeled documents S_{labeled} and a large number of unlabeled documents, $S_{\text{unlabeled}}$.
- The initial parameters $\hat{\theta}$ are estimated from the labeled documents S_{labeled} .
- **Expectation step:** The resulting classifier is used to assign probabilistically-weighted class labels $p(c_j | \mathbf{x})$ to each unlabeled document $\mathbf{x} \in S_{\text{unlabeled}}$.
- **Maximization step:** Re-estimate $\hat{\theta}$ using $p(c_j | \mathbf{x})$ values for $\mathbf{x} \in S_{\text{unlabeled}} \cup S_{\text{labeled}}$.
- Repeat until $p(c_j | \mathbf{x})$ or $\hat{\theta}$ has converged.

Data

- 20 UseNet newsgroups
- Web pages (WebKB)
- Newswire articles (Reuters)

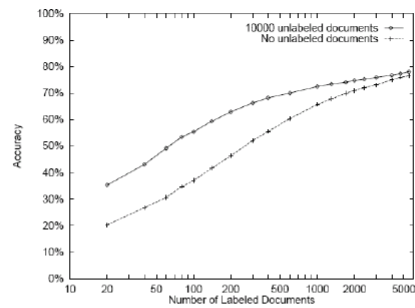


Figure 2. Classification accuracy on the 20 Newsgroups data set, both with and without 10,000 unlabeled documents. With small amounts of training data, using EM yields more accurate classifiers. With large amounts of labeled training data, accurate parameter estimates can be obtained without the use of unlabeled data, and the two methods begin to converge.

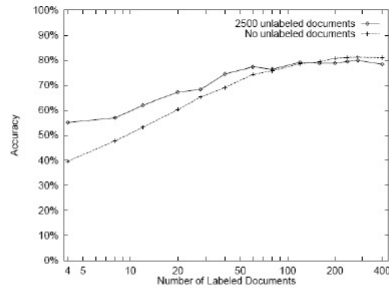
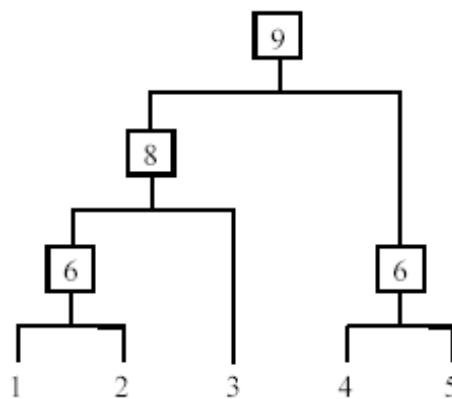


Figure 4. Classification accuracy on the WebKB data set, both with and without 2500 unlabeled documents. When there are small numbers of labeled documents, EM improves accuracy. When there are many labeled documents, however, EM degrades performance slightly—indicating a misfit between the data and the assumed generative model.

Hierarchical Clustering

- Produce a nested sequence of clusters, a **tree**, also called **Dendrogram**.



Adapted from Bing Liu, UIC

<http://www.cs.uic.edu/~liub/teach/cs583-fall-05/CS583-unsupervised-learning.ppt>

Applications of hierarchical clustering?

Benefits:

- Deterministic.
- Don't need to know the number of clusters ahead of time.
- Can assign document to more than one cluster

Disadvantages:

- Expensive

Types of hierarchical clustering

- **Agglomerative (bottom up) clustering:** Builds the dendrogram (tree) from the bottom level, and
 - merges the most similar (or nearest) pair of clusters
 - stops when all the data points are merged into a single cluster (i.e., the root cluster).
- **Divisive (top down) clustering:** Starts with all data points in one cluster, the root.
 - Splits the root into a set of child clusters. Each child cluster is recursively divided further
 - stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Adapted from CS583, Bing Liu, UIC

38

Agglomerative clustering

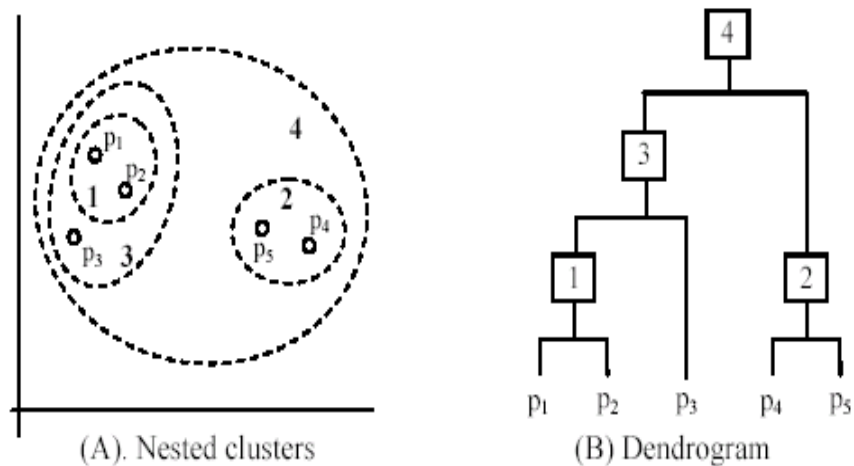
More popular than divisive methods.

- At the beginning, each data point forms a cluster (also called a node).
- Merge nodes/clusters that have the least distance.
- Go on merging
- Eventually all nodes belong to one cluster

Adapted from CS583, Bing Liu, UIC

39

Example



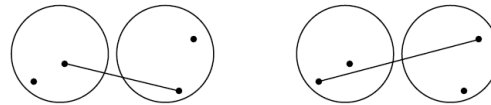
CS583, Bing Liu, UIC

40

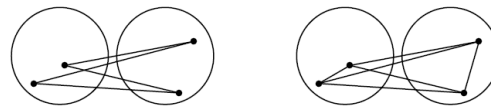
Measuring the distance of two clusters

- A few ways to measure distances of two clusters; results in different variations of the algorithm.

- Single link
- Complete link
- Average link
- Centroids



(a) single-link: maximum similarity (b) complete-link: minimum similarity



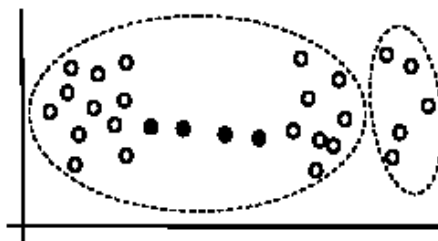
(c) centroid: average inter-similarity (d) group-average: average of all similarities

► Figure 17.3 The different notions of cluster similarity used by the four HAC algorithms. An *inter-similarity* is a similarity between two documents from different clusters.

Adapted from CS583, Bing Liu, UIC

Single link method

- The distance between two clusters is the distance between two **closest data points** in the two clusters, one data point from each cluster.
- Greedy
- It can find arbitrarily shaped clusters, but
 - It may cause the undesirable “**chain effect**” by noisy points



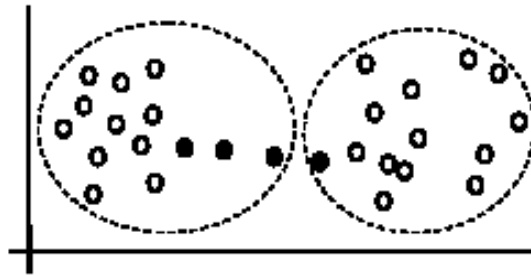
Two natural clusters are split into two

CS583, Bing Liu, UIC

42

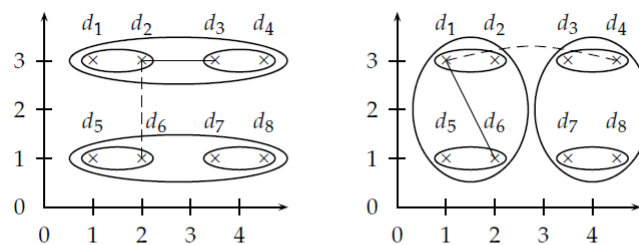
Complete link method

- The distance between two clusters is the distance of two **furthest** data points in the two clusters.
- Results in preference for compact clusters.
- However, is sensitive to outliers because they are far away



CS583, Bing Liu, UIC

43



► **Figure 17.4** A single-link (left) and complete-link (right) clustering of eight documents. The ellipses correspond to successive clustering stages. Left: The single-link similarity of the two upper two-point clusters is the similarity of d_2 and d_3 (solid line), which is greater than the single-link similarity of the two left two-point clusters (dashed line). Right: The complete-link similarity of the two upper two-point clusters is the similarity of d_1 and d_4 (dashed line), which is smaller than the complete-link similarity of the two left two-point clusters (solid line).

Group-average agglomerative clustering

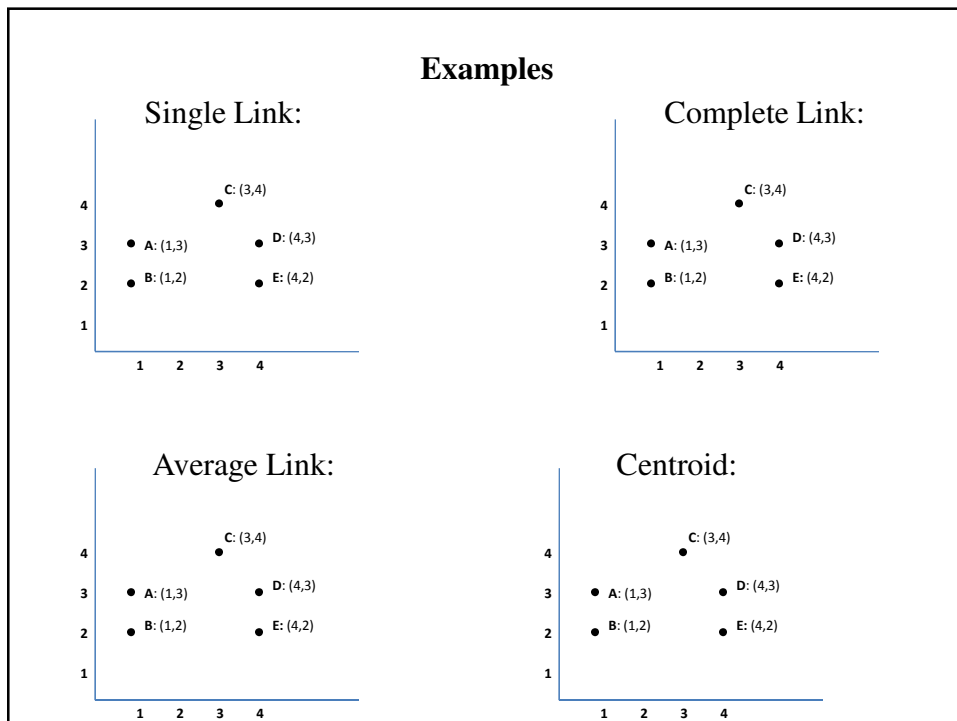
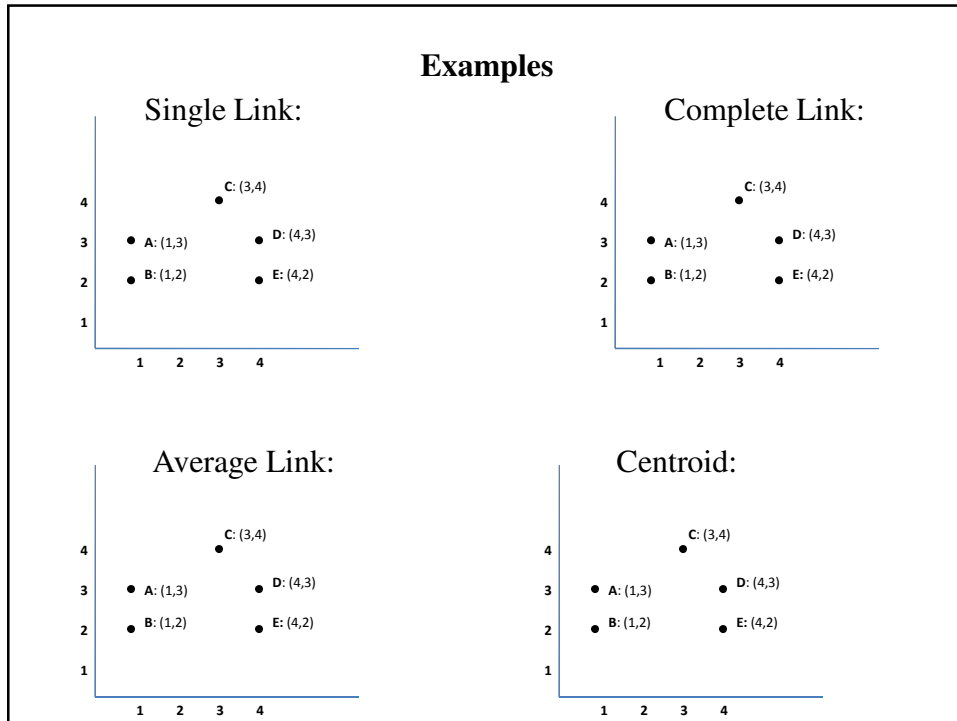
- **Average link:** A compromise between
 - the sensitivity of complete-link clustering to outliers and
 - the tendency of single-link clustering to form long chains that do not correspond to the intuitive notion of clusters as compact, spherical objects.
 - In this method, the distance between two clusters is the average distance of all pair-wise distances between the data points in two clusters.

CS583, Bing Liu, UIC

45

Centroid clustering

- **Centroid method:** In this method, the distance between two clusters is the distance between their centroids



- Hierarchical clustering demo
- http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletH.html

Divisive Clustering

- See Section 17.6