Search Engines

WS 2009 / 2010

Lecture 12, Thursday January 28th, 2010
(Clustering, Clustering, Clustering)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

UNI FREIBURG

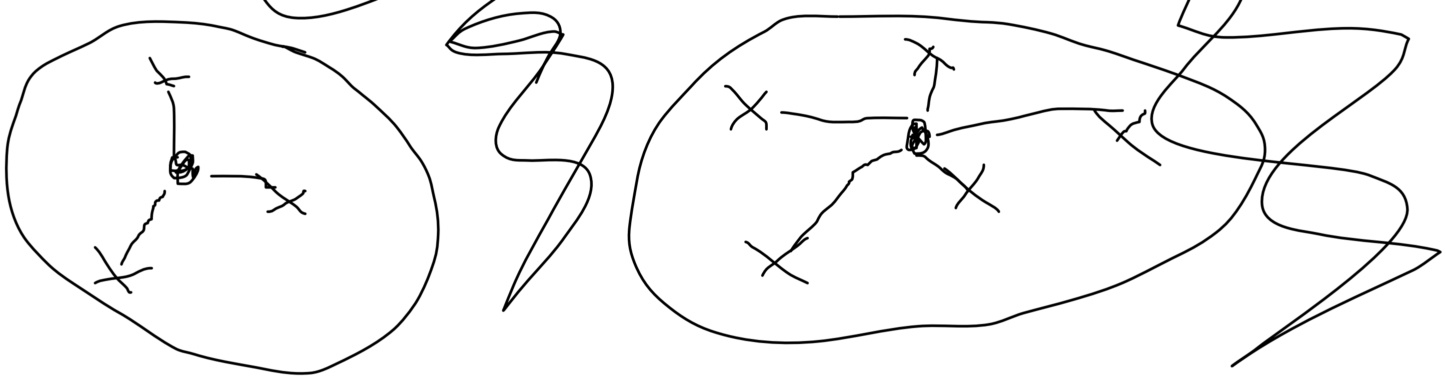
Overview of Today's Lecture

- Learn how to cluster
  - What is clustering and how is it different from classification?
  - The simplest of all clustering algorithms: k-means
  - Hierarchical clustering
  - A very nice impossibility result (Kleinberg, NIPS 2002)
    - No single clustering algorithms can achieve all desirable goals at the same time

      (a bit like Heisenberg's uncertainty principle)

What is Clustering

- Given a set of points, find a "good" partitioning



- Difference to classification:
  - Classification is supervised
    - we need training data where we know the classes
  - Clustering is unsupervised
    - we are just given the objects / points

When is a Clustering "Good" ?

- Many different ways to define quality
  - quality relative to a ground truth:
    - define precision and recall as usual
  - without a ground truth, just data-dependent:
    - intuitively, a clustering is good if it has
      - high intra-cluster similarity
      - low inter-cluster similarity
    - one formalization of this: RSS = residual sum of squares
      - Centroid of a cluster = average of points in the cluster
      - assume clusters $C_1,\ldots,C_k$ with centroids $\mu_1,\ldots,\mu_k$
      - then $RSS = \Sigma_{i=1,\ldots,k} \Sigma_{x \text{ in } Ci} |x - \mu_i|^2$

K-Means

- The simplest of all clustering algorithms
  - used a lot in practice (because of its simplicity, like Naive Bayes)
  - k-means tries to minimize RSS from last slide
  - assume we know the optimal centroids
    - (A) then best to assign each point to its nearest centroid
  - assume we know the optimal clustering
    - (B) then best to take centroid = average of points in cluster
  - but initially we know neither the centroids nor the clustering
    - so guess some initial centroids
    - from that compute clustering according to (A)
    - from that compute centroids according to (B)
    - and so on ... DEMO

Proof of optimality of (A) and (B)

RSS

$$RSS = \sum_{i=1}^{2} \sum_{x \in C_i} |x - \mu_i|^2$$

(A) decreases RSS : obvious!

(B) decreases RSS :

which μ minimizes $\sum_{i=1}^{m} |x_i - \mu|^2 =: R$

$$\frac{\partial R}{\partial \mu} = \sum_{i=1}^{m} -2(x_i - \mu) = -2\left(\sum_{i=1}^{m} x_i - m \cdot \mu\right) = 0$$

$$\Longrightarrow \mu = \sum_{i=1}^{m} x_i / m$$

K-Means — Code

- Code live in a VNC session ...
  - with points = integers — Exercise: points = text
- Possible abort criteria
  - after a fixed number of iterations
    - simple, but how to guess a good number?
  - until assignment of points to clusters remains constant
    - very reasonable, but can take very long for large data sets
  - terminate when RSS falls below given threshold
    - makes sense, but RSS may never fall below given threshold
    - combine with bound on number of iterations
  - terminate when decrease in RSS falls below given threshold
    - good, because we stop when we are close to convergence
    - must also combine with bound on number of iterations

K-Means — Convergence

- Proof of convergence to a local optimum
  - RSS decreases in assignment step (A)
    - this follows from our optimality proof for (A)
  - RSS decreases in centroid computation step (B)
    - this follows from our optimality proof for (B)
  - stop when there is no more decrease
  - only finitely many clusterings → termination
  - however, we must pay attention to proper tie breaking
    - tie = two centroids are equally close
    - for example, always prefer centroid with smaller index
    - otherwise may cycle forever between clusters of equal quality

K-Means — Time Complexity

- The time complexity is
  - each assigment step (A) takes time $O(n \cdot k)$
    - where n = #points and k = #clusters
  - each centroid computation step (B) takes time $O(n)$
  - so with I iterations we get $O(I \cdot n \cdot k)$
  - but this assumes that adding two points takes time $O(1)$
    - not true for vectors in high-dimensional space
    - however, these vectors are usually sparse (e.g. docs)
    - then cost of addition is O(#non-zero entries)
    - however, the centroids quickly become not sparse!
    - simple trick: centroid truncation
      - set components with small values to zero

K-Means — Choice of K

- Idea: try to base it on RSS
  - Idea 1: choose the K with smallest RSS
    - bad idea, because RSS is always minimized for $K = n$
  - Idea 2: choose K with smallest $RSS + \lambda \cdot K$
    - makes sense: RSS becomes larger as K becomes smaller
    - λ is a tuning parameter
    - now we have shifted the problem to finding a good λ
    - but for a given application, λ is often a constant while the best K may vary from instance to instance
    - this formula has an information-theoretic justification

Hierarchical Clustering

- General bottom-up idea:
  - start with clustering, where each point is its own cluster
  - iteratively merge the two clusters that are "closest"
  - natural visualization of hierarchy as a dendrogram

Which Clusters To Merge

- Similarity measure between clusters sim(C_i, C_j)
  - in each step merge C_i and C_j with smallest sim(C_i, C_j)
- Four common similarity measures
  - Single-Link: similarity of closest points
  - Complete-Link: similarity of farthest points
  - Centroid: average inter-similarity
  - Group-Average: average of all similarities

Single-Link and Complete-Link

- Single-Link Problem
  - only the closest pair counts → tendency to straggly clusters

- Complete-Link Problem
  - high sensitivity even to single outlier

- Graph-theoretic interpretation
  - let $s_k = sim(C_i, C_j)$ in k-th merging step
  - let G_k be the graph with an edge between all points with $d \leq s_k$
  - then single-link clusters = connected components of G_k
  - and complete-link clusters = maximal cliques of G_k

Hierarchical Clustering — Time Complexity

- Naive algorithm
  - assume we proceed until we have k clusters
  - compute all pairwise distances for all cluster pairs
    - this is on the order of n^2 (n = #points)
  - so that gives a total time complexity of $O(k \cdot n^2)$
  - n^2 is prohibitive for large data
- Improvement
  - using a priority queue we can achieve $O(k \cdot n \cdot \log n)$
  - this is ok; recall that k-means needs $O(I \cdot k \cdot n)$
  - for single-link we can even achieve $O(k \cdot n)$
  - we will not go into the details of these algorithms here
    - read in the references in case you are interested

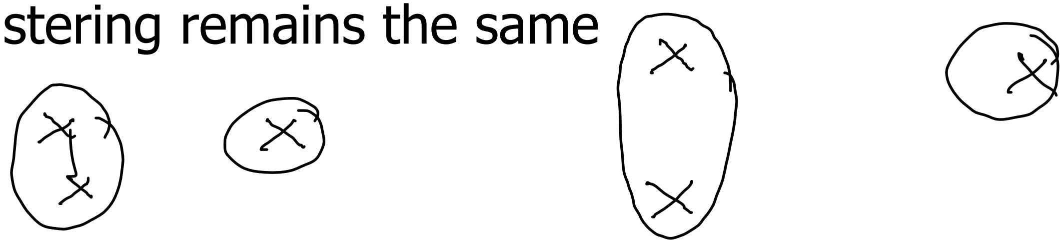
An Impossibility Theorem for Clustering

- Naive but valid question:
  - Can't there a single clustering that is always the best?
  - Each clustering algorithm we know has some drawbacks
  - but that does not answer our question
    - maybe just no one has been smart enough yet?
  - let's formulate three natural properties which every clustering algorithm should have

Three Properties ...

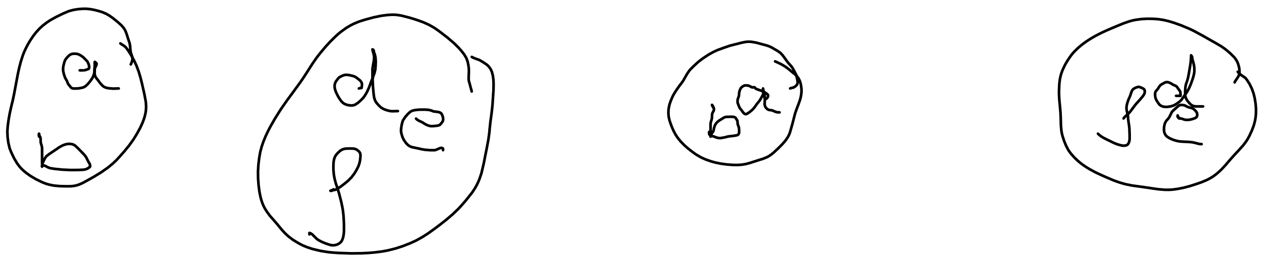
UNI FREIBURG

- ... every clustering algorithm should have
  - **Scale Invariance:** when we multiply all distances by a constant factor, the clustering remains the same



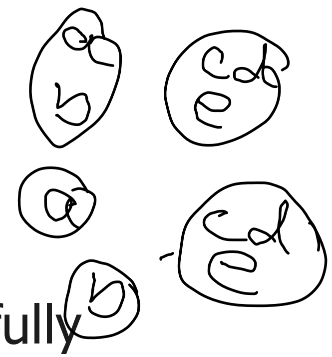
  - **Richness:** each possible partioning is achieved on some dataset

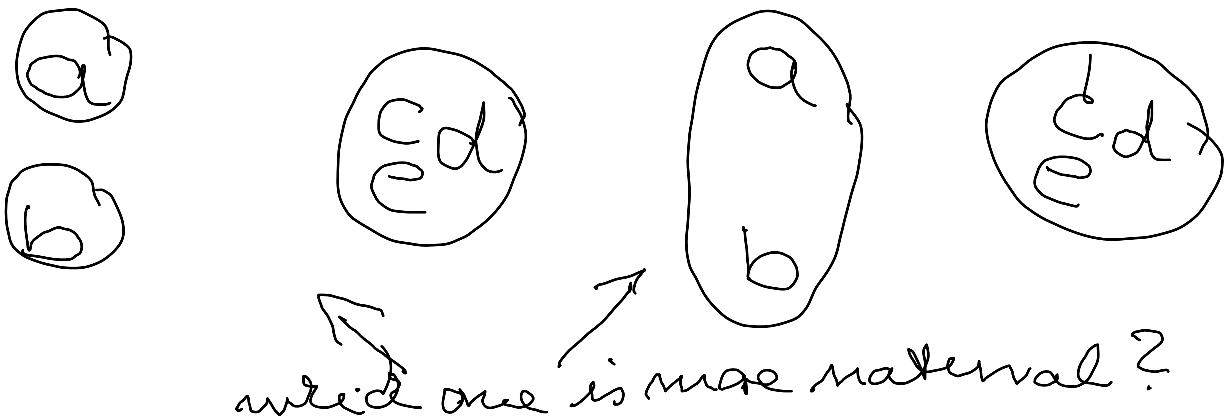
  - **Consistency:** if we shrink the intra-cluster distances and increase the inter-cluster distances, the clustering stays the same



Impossibility theorem

- No clustering algorithm can achieve all three!
  - here is the basic proof idea
  - define an antichain as a set of partionings where no partioning is a refinement of another partioning
  - then proof that Scale-Invariance + Consistency imply that the set of achievable partionings is an antichain thus contradicting Richness
  - here is some intuition which you will only be able to fully understand after you have understood the proof:





Proof of Impossibility Theorem 1

Let A be any clustering algorithm which satisfies SCALE-INVARIANCE and CONSISTENCY

Assume (by way of contradiction) that $\exists\, \Gamma_1, \Gamma_2$ where $\Gamma_1 \leq \Gamma_2$, both prod. by A

$\exists\, \alpha, \beta$ with the following property

if all intra-cluster dist $\leq \alpha$, all inter-cluster dist $\geq \beta$ $\Longrightarrow$ A always prod Γ_1

where $\alpha < \beta$. Follows from CONSISTENCY. For example $(1, 5)$ i.e. α, β

$\exists\, \beta, \gamma$ with the following property

if all intra ... $\leq \beta$, all inter $\geq \gamma$ $\Longrightarrow$ A produces Γ_2

Follows from CONSISTENCY + SCALE e.g. $(5, 10)$

Proof of Impossibility Theorem 2

$(1, 5) \Rightarrow \Gamma_1$ (1)

$(5, 10) \Rightarrow \Gamma_2$ (2)

Now take clustering Γ_2 and bring all intra-cluster distances of $\Gamma_1 \leq 1$.

(1) $\Rightarrow$ A produces Γ_1

(2) $\Rightarrow$ A produces Γ_2 ↯

Proof of Impossibility Theorem 3

References

- K-Means and Hierarchical Clustering
  - Again, the Wikipedia articles are ok
    http://en.wikipedia.org/wiki/K-means
    http://en.wikipedia.org/wiki/Hierarchical_clustering
  - Here is the textbook which I also consulted
    Introduction to Information Retrieval
- The impossibility theorem
  - The NIPS 2002 paper by Jon Kleinberg
    An impossibility theorem for Clustering