Clustering

Algorithmic Methods of Data Mining, Fall 2007, Clustering 1

Clustering

- Task: group observations into groups so that the observations belonging to the same group are similar, whereas observations in different groups are different
- Lots and lots of research in various areas
- Just scratching the surface here

Topics

- Basic remarks
- k-means clustering
- kmeans++
- Hierarchical clustering
- Kleinberg's impossibility theorem

Basic questions?

- What does "similar" mean?
- What is a good partition of the objects? I.e., how is the quality of a solution measured?
- How to find a good partition of observations?

What does "similar" mean?

- Some function of the attribute values of the observations
- Usual approach: L_p distance

$$L((x_1, \ldots, x_n), (y_1, \ldots, y_n)) = (\sum_i (x_i - y_i)^p)^{1/p}$$

- Easy in 1-dimensional real case
- Already 2 dimensions cause problems: how to weigh the different dimensions?
- Lots of problems

Clustering, so what?

- Often a clustering gives lots of information
- Sometime not
- A cluster as such is not necessarily very useful
- Additional analyses are needed
- Characterizing the clusters
- Using them for something

One problem formulation: minimum k-center

- Input: Complete graph $G = (V, E)$ and distances $d(v_i, v_j) \in N$ satisfying the triangle inequality.
- Solution: A k-center set, i.e., a subset $C \subseteq V$ with $|C| = k$.
- Score function: The maximum distance from a vertex to its nearest center
  $\max_{v \in V} \min_{c \in C} d(v, c)$.
- This problem is NP-hard
- Can be approximated within a factor of 2

Another formulation: k-means problem

- Given a set X of n points in R^d and an integer k
- Task: choose a set $C \subseteq R^d$ of k points minimizing

$$\phi = \sum_{x \in X} \min_{c \in C} ||x - c||^2.$$

Properties of the problems

- There is always a solution
- Even if there is no structure in the data

What is the correct value of k?

- Model selection problem
- Difficult task
- Not covered in this course

Algorithmic properties of the k-means problem

- NP-hard if the dimension is at least 2
- Thus finding an exact solution is probably not feasible
- Dimension 1 is solvable in polynomial time by dynamic programming
- A simple iterative algorithm works quite well

k-means algorithm

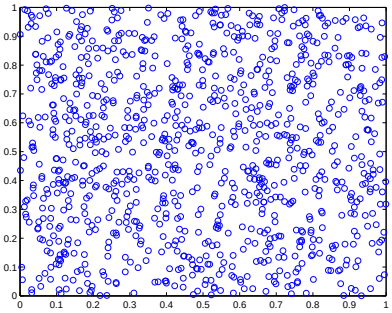
- One way of solving the k-means problem
- Randomly pick K cluster centers $C = \{c_1, \ldots, c_k\}$
- For each i, set the cluster i to be the set of points in X that are closer to c_i than they are to c_j for all $j \neq i$
- For each i let c_i be the center of cluster i
- Repeat until convergence

Examples

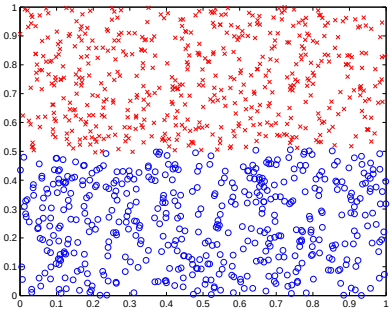
Treatment follows "k-means++: The Advantages of Careful Seeding David Arthur and Sergei Vassilvitskii", SODA 2006

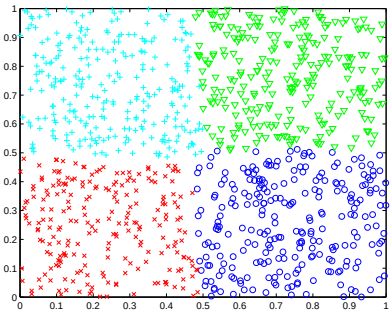
See
www.stanford.edu~sergeiv/slides/BATS-Means.pdf

Example

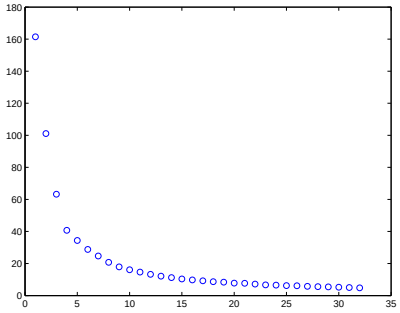


Example





Example

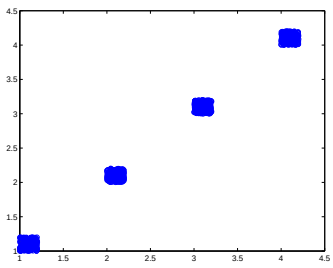


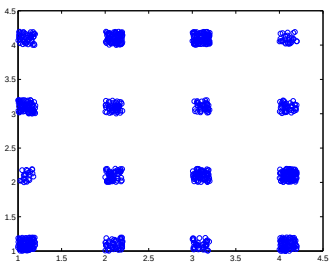
Example

1000 points from $\{0,1\}^{100}$

clustered twice into 4 clusters, cluster ids plotted

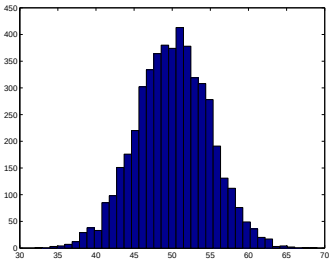
Should look like this looks like this





Why?

Random points in 100-dimensional 0-1 space are about equivdistant from each other



Where do the numbers come from?

Properties of the K-means algorithm

- Finds a local optimum
- Converges often quite quickly
- Sometimes slow convergence
- For high dimensions the choice of initial points can have a large influence

Why does this converge?

- Why does changing the centers decrease ϕ?
- Let S be a set of points with center of mass $c(S)$, and let z be an arbitrary point. Then

$$\sum_{x \in S} ||x - z||^2 - \sum_{x \in S} ||x - c(S)||^2 = |S||c(S) - z||^2.$$

- S: a current cluster, z its original center

k-means++

- Just change the initialization a little bit
- $D(x)$: distance from point $x \in X$ to an already selected center
- For $i = 2, \ldots, k$, select c_1 uniformly at random from X
- Select as center c_i a point x' from X, with probabilities

$$\frac{D(x')^2}{\sum_{x \in X} D(x)^2}$$

- Use the normal k-means algorithm

Does this make a difference?

- It does
- Suprisingly, this simple change makes the algorithm theoretically and practically better
- ϕ_{Opt}: the cost of the optimal clustering C_{Opt} (this cannot be found efficiently)
- Let C be the result of k-means++, and let ϕ be the corresponding cost function
- Then

$$E(\phi) \leq 8(\ln k + 2)\phi_{\text{Opt}}$$

- Not too far from the optimal value

Proving the bound

- First cluster: easy
- Other clusters: harder, but not very difficult

What does the result mean?

Empirical results

Hierarchical clustering

- Merge sets of points or divide sets of points
- Agglomerative or divisive
- Dendrograms (figures)

Agglomerative clustering

for $i = 1, \ldots, n$ let $C_i = \{\mathbf{x}(i)\}$;
while there is more than one cluster left **do**
  let C_i and C_j be the clusters
    minimizing the distance $\mathcal{D}(C_k, C_h)$ between
    any two clusters;
  $C_i = C_i \cup C_j$;
  remove cluster C_j ;
od;

Complexity

- Quadratic, at least, in the number of points
- Not usable for large sets of data

What is the distance between clusters?

- How to define the distance between two clusters for hierarchical clustering?
- Two sets of points
- Lots of alternatives
- Actually quite difficult to define a metric

Single-link distance

$d(\mathbf{x}, \mathbf{y})$ the distance between objects $\mathbf{x}$ and $\mathbf{y}$

$$\mathcal{D}_{sl}(C_i, C_j) = \min_{\mathbf{x},\mathbf{y}}\{d(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C_i, \mathbf{y} \in C_j\}, \quad (1)$$

chaining: long, elongated clusters

Complete link

Furthest distance

$$\mathcal{D}_{fl}(C_i, C_j) = \max_{\mathbf{x},\mathbf{y}}\{d(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C_i, \mathbf{y} \in C_j\}, \tag{2}$$

leads to equal colume (or at least diameter)

Other measures

- For vectors
- the centroid measure (the distance between two clusters is the distance between their centroids)
- the group average measure (the distance between two clusters is the average of all the distances between pairs of points, one from each cluster)
- Ward's measure for vector data (the distance between two clusters is the difference between the total within cluster sum of squares for the two clusters separately, and the within cluster sum of squares resulting from merging the two clusters discussed above)

Divisive methods

- start with a single cluster composed of all of the data points
- split this into components
- continue recursively
- *Monothetic* divisive methods split clusters using one variable at a time
- *Polythetic* divisive methods make splits on the basis of all of the variables together
- any intercluster distance measure can be used
- computationally intensive, less widely used than agglomerative methods

Kleinberg's impossibility theorem

- John Kleinberg, An impossibility theorem for clustering, NIPS 2002
- Clustering methods based on pairwise distances
- Three properties for clustering methods
- No algoritm can have all three

Computational task

- A clustering function operates on a set S of n points
- No underlying space; $S = \{1, 2, \ldots, n\}$
- Distance function: $d : S \times S \to \mathbf{R}$ with $d(i, j) \geq 0$, $d(i, j) = d(j, i)$, and $d(i, j) = 0$ only if $i = j$
- (Metric: additionally have $d(i, j) + d(j, k) \geq d(i, k)$)
- Clustering function $f : f(S, d) = \Gamma$, where Γ is a partition of S
- (A partition)

Scale invariance

$\alpha > 0$; distance function αd has values $(\alpha d)(i, j) = \alpha d(i, j)$
For any d and for any $\alpha > 0$ we have $f(d) = f(\alpha d)$

Richness

The range of f is equal to the set of partitions of S

I.e., for any S and any partition Γ of S there is a distance function d on S such that $f(S, d) = \Gamma$

Consistency

Shrinking distances between points inside a cluster and expanding distances between points in different clusters does not change the result.
Γ a partition of S
$d,\ d'$ two distance functions on S
d' is a Γ-transformation of d, if

- for all $i, j \in S$ in the same cluster of Γ we have $d'(i, j) \leq d(i, j)$
- for all $i, j \in S$ in the different cluster of Γ we have $d'(i, j) \geq d(i, j)$

Consistency: if $f(S, d) = \Gamma$ and d' is a Γ-transformation of d,
then $f(S, d') = \Gamma$

Examples

- Agglomerative clustering with single-link
- Repeatedly merge cluster whose distance is minimum
- Continue until a stopping criterion is met
  - k-cluster stopping criterion: continue until there are k connected components
  - distance-r stopping criterion: continue until all distances between clusters are larger than r
  - scale-α stopping criterion: let ρ^* be the maximum pairwise distance; continue until all distances are larger than $\alpha\rho^*$

Examples, cont.

- Single link with k-cluster stopping criterion satisfies scale-invariance and consistency
- Single link with distance-r stopping criterion satisfies richness and consistency
- Sigle link with scale-α stopping criterion satisfies richness and scale-invariance

Theorem

For each $n \geq 2$ there is no clustering function that satisfies scale-invariance, richness, and consistency

Proof of theorem

A partition Γ' is a refinement of partition Γ, if each set $C' \in \Gamma'$ is included in some set $C \in \Gamma$
A partial order between partitions: $\Gamma' \leq \Gamma$
Antichain of partitions: collection of partitions such than no one is a refinement of others
Theorem: If a clustering function f satisfies scale-invariance and concistency, then the range of f is an antichain

Γ-forcing

- partition Γ
- d (a, b)-conforms to Γ, if for all points i, j in the same cluster of Γ $d(i, j) \leq a$, and for all points i, j in different clusters of Γ $d(i, j) \geq b$
- given a clustering function f
- (a, b) is Γ-forcing, if for all d that (a, b)-conform to Γ we have $f(S, d) = \Gamma$

Forcing, cont.

- Assume f satisfies consistency; let $\Gamma \in Range(f)$
- Claim: there are $a < b$ such that (a, b) is Γ-forcing
- Γ in range of f: there is d such that $f(S, d) = \Gamma$
- a'=minimum distance of points in the same cluster in Γ
- b'=maximum distance of points in different clusters in Γ
- Choose $a < b$ such that $a < a'$ and $b < b'$
- If d' (a, b)-conforms to Γ, then d is a Γ-transformation of d
- By consistency $f(d') = \Gamma$
- Thus (a, b) is Γ-forcing

Antichains

- Assume f satisfies scale-invariance
- Let Γ_0 and Γ_1 be possible results of f, and let Γ_0 be a refinement of Γ_1
- Show that this leads to contradiction
- (a_0, b_0) Γ_0-forcing, (a_1, b_1) Γ_1-forcing
- Let $a_2 < a_1$, choose ϵ so that $0 < \epsilon < a_0 a_2 b_0^{-1}$
- Construct a d such that
  - For i, j in same cluster of Γ_0 we have $d(i, j) \leq \epsilon$
  - For i, j in same cluster of Γ_1 but not in Γ_0 we have $a_2 \leq d(i, j) \leq a_1$
  - For i, j in different clusters of Γ_1 $d(i, j) \geq b_1$

- d (a, b)-conforms to Γ_1, and thus $f(S, d) = \Gamma_1$
- $\alpha = b_0 a_2^{-1}$, and let $d' = \alpha d$
- scale-invariance: $f(d') = f(d) = \Gamma_1$
- i, j in same cluster of Γ_0 we have

$$d'(i, j) \leq \epsilon b_0 a_2^{-1} < a_0$$

- i, j in different clusters of Γ_0 we have

$$d'(i, j) \geq a_2 b_0 a_2^{-1} = b_0$$

- d' (a_0, b_0) conforms to Γ_0, and thus $f(S, d') = \Gamma_0 \neq \Gamma_1$, contradiction