L10 -- k-means clustering
[Jeff Phillips - Utah - Data Mining]

----
k-means clustering:
  Find k points C = {c_1, ..., c_k}, s.t.
   - each p in P assigned mu(p) = arg min_{c in C} ||p - c||
   - minimize E(P,C,mu) = sum_{p in P} ||p, mu(p)||^2

(like k-center minimize max_{p in P} ||p - mu(p)||)
(k-median minimize sum_{p in P} ||p - mu(p)||)

Lloyd's algorithm (1957 -> published 1982)

Choose k points (arbitrarily?) C subset P
   1. for all p in P, find mu(p) (closest center c in C to p)
   2. for all i in [k]   let c_i = average{p in P | mu(p) = c_i}
 if (C changed, repeat)
----

say R rounds ==> O(R kn)
   (improved w/ faster NN search)

What is R?
  finite. # of distinct clusters
    each step minimizes E(P,C,mu)
  with fixed k, d -> R = O(n^{dk})   (Voronoi diagram)
  --> exponential in k,d (NP-Hard)
  R ~ 10, usually ok.

  smooth complexity: (perturb data randomly, -> O(n^{35} k^{34} d^8) :) big but poly)
  on a lattice: O(d n^4 M^2)

How to choose initial centers C?

 - random set of k points
     we know that collisions are likely (if k true clusters)
 - randomly partition data P -> {S_1, ... S_k}, take mean of each
 - MinMax
     (sensitive to outliers)

Choose first c_1 arbitrarily

C_1 = {c_1}   (generally C_i = {C_1, C_2, ..., C_i} \\ goal C_k)

Let c_{i+1} = arg max_{p in P \ C_i} d(p,mu(p))
"always pick point furtherest from set of centers C_i"

- k-means++ (guarantees polynomial time, with some probability)

Choose first c_1 arbitrarily
C_1 = {c_1}   (generally C_i = {C_1, C_2, ..., C_i} \\ goal C_k)

Choose c_{i+1} with_prob_{p in P \ C_i} ||p - mu(p)||^2
"pick point proportional to distance from set of centers C_i"

- random re-starts (try multiple times, take the best)

How accurate is Lloyd's Algo?
- can be arbitrarily bad
- (1+eps)-approx in 2^{(k/eps)^{O(1)}} nd [Kumar,Sabharwal,Sen '04]

k-means++ is O(log k) competitive (8 if well-separated)

Problems with k-means:
- Lloyd's Algo requires d(a,b) = ||a-b||
  -> can use C subset P (slower to run step 2)

- effected by outliers. squared distance makes far points more important
  (k-medians: step 1 same, step 2 harder "Fermat-Weber problem", gradient descent)

- enforces equi-sized clusters. Vornonoi partition.
  (draw mickey-mouse picture)

- EM formulation: Expectation-Maximization
  model each cluster as a Gaussian G_i (centered at c_i)
  1. for each point, find cluster with largest probability of containing that point
  2. for a cluster, find best fit Gaussian (c_i = mean, covariance = estimate each variance)

(allows for slanted (with PCA) and non-uniform clusters)

- has also been work in clustering to low-dimensional subspaces.
  Enforces that some covariances are 0, others "infinite" (at least uniform).

Speeding up k-means:

- run k-means on random sample of points.
  Once centers obtained, run on full set.

- run streaming with (k log k) clusters
  merge clusters at end
  (better: maintain hierarchy of clusters)

- BFR algorithm: Process points in batches
  - summarize batches (compact clusters as Gaussians + leftovers)
  - merge summaries