**Machine Learning**

**Session 7**

1. Unsupervised Learning Motivation
   1. Unsupervised Learning:
      1. Too much data: We need to save memory/computation.
         1. Reduce the data to a more manageable amount
      2. Don’t understand the data
         1. Exploratory data analysis
         2. What underlying knowledge is there?
         3. Discover patterns & trends
      3. Clustering
         1. Focus on instances (rows)
      4. Dimension reduction
         1. Focus on dimensions (columns)
2. Overview
   1. Clustering Algorithms
      1. K-means
      2. Hierarchical Clustering
   2. Further topics
      1. Choosing the number of clusters
      2. Advanced algorithms
   3. Some applications
      1. ..besides marketing!
3. Clustering: Unsupervised Learning Context
   1. Clustering versus Dimensionality Reduction
      1. Both unsupervised
      2. Group rows (clustering) versus columns (dimensionality reduction)
4. Clustering
   1. Problem definition:
      1. Given a set of examples (x1 ,x2 ,..,xN )
      2. Divide them into subsets of similar examples.
   2. Alternative view:
      1. How to do classification if you have no labeled training examples
   3. Conceptual & algorithmic challenges:
      1. How to quantify similarity?
      2. How to measure the goodness of a partition?  
           
         Figure:  
         Given a graph with points, separate according to closeness
5. Clustering: Why do it?
   1. Market segmentation
      1. Teenagers, Mothers, Empty-nesters
      2. Targeted products/marketing for each “cluster” of customers
   2. Data-center organization
   3. Social network analysis (find recommended friends)
   4. Group articles on your website or blog
   5. Group websites on your aggregator
   6. T-shirts: Given height & weight in the population:
      1. How big should size of S, M, L be?
      2. Should you offer S, M, L or XS, S, M, L, XL?
   7. Stock choice in retail: There are more possible products to stock than we have space for. Which ones to include?
      1. Try to cover the range of possible options so a consumer of every preference can find something they like.
      2. How? Clustering then pick one example from each cluster.
      3. Question: Have you covered the space well?
6. K-Means Algorithm
   1. Input:
      1. N point dataset, D={x1 ,x2 ,…xn },
      2. Number of clusters K.
   2. Initialize randomly K centers Ctr\_1 ,..,Ctr\_k
   3. Repeat
      1. For i=1:N
         1. Labels\_i=Cluster centroid closest to xi
      2. For k=1:K
         1. Ctr\_k = average of points assigned to k
   4. Cost Function:
      1. Find cluster centers u\_1:k and cluster assignments c\_1:N so as to minimize the sum squared distances of points from assigned clusters:   
           
         E\_{KM} (D, C\_1:N, mu\_1:K) = sum of (x\_i – mu\_c\_i)^2
   5. Algorithm:
      1. “E-step”: Find cluster closest to each point (fix u, minimize for c)
      2. “M-step”: Find new center of each cluster (fix c, minimize for u)
   6. Aside:
      1. We have seen algorithms with exact & gradient solutions to problems.
      2. This is our first alternating minimization solution
      3. An exact solution to each part of the problem given the other: Iterate
   7. Pseudocode:
      1. Find cluster centers u\_1:k and cluster assignments c\_1:N
      2. Data structures:
         1. N is #dimensions,
         2. D is #samples,
         3. K is #clusters
         4. X (data matrix) NxD reals
         5. c (cluster assignments) N integers in {1..K}
         6. u (cluster centres) KxD reals
      3. Algorithm:
         1. “E-step”: Find cluster closest to each point (fix u, minimize for c)
         2. “M-step”: Find new center of each cluster (fix c, minimize for u)
      4. “E-step”: Find cluster closest to each point (fix u, minimize for c)
         1. For 1< i < N
            1. for 1<k

distance[I,k] = L2\_norm(X[i,:], u[k,:]];

* + - * 1. end
        2. c[i] = argmin\_k(distance[I,k]); % c[i] is the index of the closest cluster centre
      1. end
    1. M-step: “Find new center of each cluster (fix c, minimize for u)
       1. tmp = zeros(K,D);
       2. tmpcount = zeros(K); # How many points in cluster K
       3. For 1< i < N
          1. tmp[c[i],:] = tmp[c[i],:] + X(i,:]; tmpcount[c[i],:] += 1;
       4. End
       5. For 1 < k< K
          1. u[k,:] = tmp[k,:] / tmpcount[k];
       6. end

1. Formalizing K-means
   1. Cost Function: E\_{KM} (D, C\_1:N, mu\_1:K) = sum of (x\_i – mu\_c\_i)^2
      1. Find cluster centers u\_1:k and cluster assignments c\_1:N so as to minimize the distance of each point from it’s assigned cluster
2. K-means Properties
   1. Recall the algorithm:
      1. Repeat S times:
      2. “E-step”: Find cluster closest to each point (fix u, minimize for c)
      3. “M-step”: Find new center of each cluster (fix c, minimize for u)
   2. Computation time?
      1. O(NK), Or O(NKDS) if dimension and iterations included
      2. Fast relative to O(N^2), slow relative to O(N). (i.e., if large K)
   3. Distance Metric
      1. Typically use Euclidean
         1. May or may not be appropriate depending on data.
         2. May not be robust to outliers
      2. What happens if you have categorical data?
         1. use 1-of-N encoding
      3. Convergence:
         1. It converges to a local minima only
         2. In practice repeat with many random initializations and pick the best
         3. Different distances lead to changes in both steps!!
      4. When it (doesn’t) work
         1. Sensitive to data scaling
         2. Renormalize in [0,1] or by standard deviation  
              
            Figure:  
            Given a graph with two clusters, without scaling the points are closer together, and with scaling they are further apart.
      5. Works if:
         1. Clusters are spherical
         2. Clusters are well separated
         3. Clusters are of similar volumes
         4. Clusters have same number of points
3. Summary
   1. K-means groups data into K groups
   2. K-means as an iterative algorithm that that optimizes a cost
   3. K-groups of data – K given.
   4. Influence of scaling
   5. When k-means work
4. Hierarchical Clustering:
   1. Sometimes you want a tree of similarity rather than a flat clustering
      1. (And K-means clusters discovered can be sensitive to chosen K)
   2. Algorithm: Agglomerative (or Divisive)
      1. Start with one cluster per example
      2. Merge two nearest clusters
         1. E.g., min, max, mean distance.
      3. Repeat until one cluster
5. Summary
   1. Clustering identifies typical groups.
   2. Need to understand the groups.
      1. What do they have in common? Two options:
         1. Manually examine elements of a cluster.
         2. Use a supervised classifier!
      2. Use the cluster labels as a supervision for a classifier.
      3. Run the classifier and examine the weights on each feature. The weights will say what is unique about each cluster.
6. Choosing Number of Clusters For K-means
   1. Elbow Method: Plot EKM/GMM as a function of k, and choose the elbow point (the point where the graph changes derivative substantially.
   2. Present results to end users see what they prefer. Broader or more specialized groups
   3. Cross-validation
      1. For K = 1…Large
         1. Learn GMM/KM clusters on a train set.
         2. Evaluate Quality(K) = quality on validation set.
      2. Pick K with the highest validation set quality
   4. BIC/AIC Criterion
      1. (ML people: An approximation to the integration required in the Bayesian model selection)
      2. Adds a penalty to the cost that penalises more complex models.
         1. P: Number of parameters in model. N: Number of data points.
      3. Evaluate modified cost EK BIC for many values of K.
      4. Pick the K with best cost:  
           
         E^{K}\_{BIC} = E^K – (p/2) log N
7. Applications:
   1. Bioinformatics
      1. Clustering of Gene Activity from Microarrays
         1. Input: Gene activations
         2. Output: Gene clusters
         3. Discover which genes activate at the same time (appeared in the same cluster)
         4. Help discover relation between functions of dissimilar genes
   2. Finance
      1. Clustering companies
         1. Input: Corporate data (e.g., financials)
         2. Output: Clusters of similar companies
      2. Application:
         1. Develop hedging/investment strategies
         2. Predict if one stock price will rise or fall
   3. News Summarisation
      1. Clustering News articles:
         1. Input: One news article per row (E.g., as bag of words)
         2. Output: Clusters of similar news articles.
      2. Application:
         1. Get an overview of today’s news by one article in each cluster.
            1. No redundancy in stories
   4. Politics Clustering voting records
      1. Input: Votes of each MP
      2. Output: Clusters of voting patterns
         1. Who tend to vote together / vote against each other.