Clustering

- Goal: group objects into subsets or clusters, such that objects within each cluster are more similar to one another than objects assigned to different clusters.
- Choice of distance measures $d(\mathbf{x}, \mathbf{z})$ is crucial. As a metric in Mathematics, a distance measure $d(\mathbf{x}, \mathbf{z})$ satisfies[a]:
  - $d(\mathbf{x}, \mathbf{z}) \geq 0$
  - $d(\mathbf{x}, \mathbf{z}) = 0$ if and only if $\mathbf{x} = \mathbf{z}$
  - $d(\mathbf{x}, \mathbf{z}) = d(\mathbf{z}, \mathbf{x})$
  - $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$ (triangle inequality)

[a] It is okay to use a distance measure that does not satisfy all the properties.

Dissimilarity Measures

- Euclidean Distances $d(\mathbf{x}, \mathbf{z})$

$$L_2 \quad : \quad \Big(\sum_{j=1}^{p} (x_j - z_j)^2 \Big)^{1/2}$$

$$L_1 \quad : \quad \sum_{j=1}^{p} |x_j - z_j| \qquad \text{(Manhattan distance)}$$

$$L_\infty \quad : \quad \max_{j=1,\ldots,p} |x_j - z_j| = \lim_{d \to \infty} \Big(\sum_{j=1}^{p} |x_j - z_j|^d \Big)^{1/d}$$

- Non-Euclidean Distances
  - Jaccard distance between sets: $1 - \frac{|A \cap B|}{|A \cup B|}$

    $\{A, C, D, E\}$ and $\{A, D, E\}$ is $1 - 3/4$.

    E.g., can measure the distance between sentences,

    - "Cluster analysis arranges similar objects in the same group."
    - "Cluster analysis divides data into groups."

    or distance between movies/restaurants based on their ratings.

– Hamming distance between two strings (of the same length): number of positions at which the corresponding symbols are different.

- "Karolin" and "Kathrin" is 3.
- 1011101 and 1001001 is 2.

For binary strings, the hamming distance is the same as L_1 distance.

E.g., can measure distance between texts, DNA/Protein sequences

– Edit distance: number of inserts and deletes to change one string into another

E.g., $\mathbf{x} = abcde$ and $\mathbf{y} = bcduve$ is 3.

– Cosine distance: angle between two vectors $\mathbf{x}$ and $\mathbf{y}$.

$$\arccos\left(\frac{\mathbf{x}^t\mathbf{y}}{\|\mathbf{x}\|\|\mathbf{y}\|}\right)$$

Two Types of Inputs for Clustering

1. data matrix, $\mathbf{X}_{n \times p}$, where rows stand for objects/samples and columns stand for variables/features.

2. dissimilarity matrix, $\mathbf{D}_{n \times n}$, where $d(i, j) = d(j, i)$ measures the difference between i and j.

We can transform the input from one form to the other:

- $\mathbf{X} \Longrightarrow \mathbf{D}$: easy with a given distance measure;
- $\mathbf{D} \Longrightarrow \mathbf{X}$: various choices, e.g., multi-dimensional scaling (MDS).

Classical Multi-dimensional Scaling (cMDS)

Given a pairwise squared ℓ_2 distance matrix $\mathbf{D}_{n\times n} = [d_{ij}]$, where $d_{ij} = \sum_{l=1}^{p}(x_{il} - x_{jl})^2$, can we retrieve the n data points $\mathbf{x}_i$'s (up to a location shift and orthonormal rotation)?

1. Double Centering $\mathbf{D}$ to obtain $\tilde{\mathbf{D}}$:

$$\tilde{d}_{ij} = -\big(d_{ij} - d_{i\cdot} - d_{\cdot j} + d_{\cdot\cdot}\big)/2,$$

   where $d_{i\cdot} = i$-th row mean, $d_{\cdot j} = j$-th column mean, $d_{\cdot\cdot} =$ overall mean of $\mathbf{D}$.

2. SVD/PCA of $\tilde{\mathbf{D}}_{n\times n} = U_{n\times p}D_{p\times p}U^t$, then $\mathbf{X}_{n\times p} = UD^{1/2}$.

Key idea: Double centering $\mathbf{D} \Longrightarrow \tilde{\mathbf{D}} = \mathbf{X}_{n\times p}\mathbf{X}^t_{p\times n}$.

$$d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{x}_i^t\mathbf{x}_i + \mathbf{x}_j^t\mathbf{x}_j - 2\mathbf{x}_i^t\mathbf{x}_j$$

Assume $\bar{\mathbf{x}} = \frac{1}{n}\sum_i \mathbf{x}_i = \mathbf{0}$ and write $C = \frac{1}{n}\sum_{i=1}^n \|\mathbf{x}_i\|^2$.

$$\begin{aligned}
d_{i\cdot} &= \frac{1}{n}\sum_{j=1}^n d_{ij} = \mathbf{x}_i^t\mathbf{x}_i + C \\
d_{\cdot j} &= \frac{1}{n}\sum_{i=1}^n d_{ij} = C + \mathbf{x}_j^t\mathbf{x}_j, \\
d_{\cdot\cdot} &= \frac{1}{n}\sum_{i=1}^n d_{i\cdot} = 2C \\
\tilde{d}_{ij} &= -(d_{ij} - d_{i\cdot} - d_{\cdot j} + d_{\cdot\cdot})/2 = -(-2\mathbf{x}_i^t\mathbf{x}_j)/2 = \mathbf{x}_i^t\mathbf{x}_j
\end{aligned}$$

Instead of using all p dimensions of $\mathbf{X}$ from the SVD of $\tilde{\mathbf{D}}$, we can just extract the first k dimensions, so $z_1, \ldots, z_n \in \mathbb{R}^k$ form a k-dimensional representation of the data that approximates the pairwise distances given by $\mathbf{D}$. Check `R` command `cmdscale`.

There are some variations of cMDS, check `isoMDS` for Kruskal's non-metricand MDS and `sammon` for Sammon's non-linear mapping in R package `MASS`.

K-means Clustering

- Input: $\mathbf{X}_{n\times p}$ (data matrix) and K (number of clusters)
- K-means clustering algorithm
  0. Start with some initial guess for the K cluster centers.
  1. For each data point, find the closest cluster center (partition step).
  2. Replace each center by the average of data points in its partition (update centers).
  3. Repeat $1+2$ until convergence.

- Goal: Partition data into K groups so the within-cluster dissimilarity is small.
- Dissimilarity measure is the squared Euclidean distance.
- Optimize the following objective function (within cluster sum of squares)

$$\Omega(z_{1:n}, \mathbf{m}_{1:K}) = \sum_{i=1}^{n} \|x_i - \mathbf{m}_{z_i}\|^2 = \sum_{k=1}^{K} \sum_{i:z_i=k} \|x_i - \mathbf{m}_k\|^2,$$

where $z_i \in \{1, 2, \ldots, K)$ and $\mathbf{m}_k \in \mathbb{R}^p$.

K-means converges to a local minimum of Ω by iterating the following two steps

Step 1 (Update Partition): Given the cluster centers $\mathbf{m}_1, \ldots, \mathbf{m}_K$,

$$z_i = \arg \min_{k=1:K} \|\mathbf{x}_i - \mathbf{m}_k\|^2.$$

Step 2 (Update Centers): Given the partition $z_1, \ldots, z_n$,

$$\mathbf{m}_k = \arg \min_{\mathbf{m}} \sum_{i:z_i=k} \|\mathbf{x}_i - \mathbf{m}\|^2 = \text{mean of } \{\mathbf{x}_i : z_i = k\}.$$

Some Practical Issues

1. Try many random starting centers and choose the solution with the smallest within-cluster SS. Check the option `nstart` in `kmeans` in R.

2. Dimension reduction via PCA or random project.

3. If using other dissimilarity measures, how should we modify the K-means algorithm? You just need to change Step 2.

4. How to choose K?

Dimension Reduction for K-means

The computation cost for K-means is $O(I \cdot n \cdot p \cdot K)$

$I = \#\text{iterations},\ n = \#\text{points},\ p = \#\text{features},\ K = \#\text{centers}$

Apply dimension reduction techniques to reduce p.

- PCA: use the top d directions to form the best approximation of the pairwise distance on average.
- Random Projection (or Johnson-Lindenstrauss type embedding): generate a random Gaussian matrix $U \in \mathbb{R}^{d \times p}$ where $d = O(\log(n)/\epsilon^2)$, project $\mathbf{x}_i$ to $\mathbf{z}_i = U\mathbf{x}_i$ then w.h.p., for any i and j:

$$(1-\epsilon)\|\mathbf{z}_i - \mathbf{z}_j\|_2 \leq \|\mathbf{x}_i - \mathbf{x}_j\|_2 \leq (1+\epsilon)\|\mathbf{z}_i - \mathbf{z}_j\|_2.$$

K-means with Other Distance Measures

To find a local minimum of

$$\Omega(z_{1:n}, \mathbf{m}_{1:K}) = \sum_{k=1}^{K} \sum_{i:z_i=k} d(\mathbf{x}_i, \mathbf{m}_k),$$

iterates the following two steps:

Step 1 (Update Partition): $z_i = \arg\min_{k=1:K} d(\mathbf{x}_i, \mathbf{m}_k)$

Step 2 (Update Centers): $\mathbf{m}_k = \arg\min_{\mathbf{m}} \sum_{i:z_i=k} d(\mathbf{x}_i, \mathbf{m})$.

E.g., $\mathbf{x} = (x_1, x_2)$ where x_2 is a categorical variable taking values $\{A, B, C\}$, and

$$d(\mathbf{x}, \mathbf{y}) = (0.4) \cdot |x_1 - y_1| + (0.6) \cdot \mathbf{I}(x_2 \neq y_2).$$

How to update centers at Step 2? Suppose cluster 1 contains the first 10 objects. Then

- the 1st dim of $\mathbf{m}_1$ is the median of the 1st dim of the 10 objects: $\min_a \sum_{i=1}^{10} |x_{i1} - a|$;
- the 2nd dim of $\mathbf{m}_1$ is the most frequent value among $\{A, B, C\}$: $\min_{a \in \{A,B,C\}} \sum_{i=1}^{10} \mathbf{I}(x_{i2} \neq a)$.

Difficulty with Step 2: what about edit distance? Sometimes, you'll find that the computation is easier if we restrict the centers $\mathbf{m}_k$ to sample points, i.e., the centers are representative points from the sample. Further, all we need is $\mathbf{D}_{n \times n}$ and no need for data matrix $X_{n \times p}$.

K-medoids Clustering

- Input: $\mathbf{D}_{n\times n}$ (pair-wise dissimilarity matrix) and K
- Cluster centers (medoids) are restricted to be data points.
- The center for cluster k,

$$\mathbf{m}_k = \arg\min_{\mathbf{x}_i : z_i = k} \sum_{j : z_j = k} d_{ij}, \tag{1}$$

  is set to be one of the data points in cluster k such that the sum of its pairwise distance to all the other points in that cluster is the smallest.

Partition Around Medoids (PAM)

PAM (Kaufman and Rousseeew, 1990) minimizes the following objective function

$$\min_{z_{1:n}, \mathbf{m}_{1:K}} \sum_{k=1}^{K} \sum_{i:z_i=k} d(\mathbf{x}_i, \mathbf{m}_k),$$

where each $\mathbf{m}_k$ is one of the data points, by iterating the following two steps:

- Partition $z_i = \arg\min_{1 \le k \le K} d(\mathbf{x}_i, \mathbf{m}_k)$;
- Update Medoids based on (1).

In addition, PAM adds a swapping step at the end to avoid local minimal: it keeps swapping $\mathbf{x}_i \leftrightarrow \mathbf{x}_j$ that decreases the objective function the most, where

$$\mathbf{x}_i \in \{\mathbf{m}_1, \ldots, \mathbf{m}_k\}, \quad \mathbf{x}_j \neq \{\mathbf{m}_1, \ldots, \mathbf{m}_k\},$$

until convergence.

How Many Clusters?

In supervised learning, the target is well-defined: predict Y well. But in unsupervised learning, there is no Y, so it is not clear how to evaluate the accuracy/effectiveness of an unsupervised procedure such as clustering.

Methods for selecting K:

- Gap statistics
- Silhouettes statistics
- Prediction strength

Gap Statistics

- Many measures of goodness for clusterings are based on the tightness of clusters, e.g., the within cluster SS:

$$SS(K) = \sum_{k=1}^{K} \sum_{z_i=k} \|x_i - m_k\|^2.$$

- Gap statistic (Tibshirani, Walther and Hastie, 2001)

$$\begin{aligned} G(K) &= \mathbb{E}_0\Big[\log SS^*(K)\Big] - \log SS_{\text{obs}}(K) \\ &\approx \frac{1}{B}\sum_{b=1}^{B} \log SS_b^*(K) - \log SS_{\text{obs}}(K) \end{aligned}$$

  where $SS_{\text{obs}}(K)$ is the within cluster SS computed based on the observed data, and $SS_b^*(K)$ is computed based on (fake) data generated from the reference distribution (which has no clustering structure).

How to generate data from the reference distribution? Two choices suggested by Tibshirani et al. (2001).

(a) Generate each feature uniformly over the range of the observed values for that feature: original data $\mathbf{X}_{n\times p}$, generate reference data $\mathbf{z} = (z_1, \ldots, z_p)^t$ by uniformly sampling z_j from the range of the j-th column of $\mathbf{X}$.

(b) Generate features from a uniform distribution over the ranges of the principal components of the data: if $\mathbf{X}_{n\times p} = UDV^t$, define $\tilde{\mathbf{X}}_{n\times p} = \mathbf{X}V = UD$, and then sample features $\tilde{\mathbf{z}}$ uniformly over the ranges of the columns of $\tilde{\mathbf{X}}$ as in (a), and then transform back to get the reference data $\mathbf{z} = \tilde{\mathbf{z}}V^t$.

One-standard-error (1se) Rule:

$$K_{\text{opt}} = \arg\min_{K}\{K : G(K) \geq G(K+1) - s_{K+1}\}$$

where $s_K = \text{sd}_0(\log SS(K))\sqrt{1+1/B}$.

Starting from $K = 1$, we would like to pick the first K whose performance is better than the one of $(K+1)$, i.e., $G(K) > G(K+1)$. Since the gap statistics $G(\cdot)$ are computed based on samples, to take account of the stochastic uncertainty, we think any value from this interval $G(K+1) \pm s_{K+1}$ can represent the performance of $K+1$, which leads to the 1se rule.

Silhouettes statistics

- The Silhouette statistic (Rousseeuw, 1987) of the ith obs measures how well it fits in its own cluster versus how well it fits in its next closest cluster.
- Adapted to K-means, define

$$a(i) = \|\mathbf{x}_i - \mathbf{m}_k\|^2, \quad b(i) = \|\mathbf{x}_i - \mathbf{m}_l\|^2,$$

where $i \in C_k$ and C_l is the next-closest cluster to $\mathbf{x}_i$. Then its silhouette is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

- $1 \geq s(i) \geq -1$; For K-means, $s(i) > 0$, but can be below 0 for other clustering methods.
- $s(i) \approx 1$: object i is well classified;
- $s(i) \approx 0$: object i lies intermediate between two clusters;
- $s(i) \approx -1$: object i is badly classified.
- The larger the (average) silhouette, the better the cluster.

- Silhouette Coefficient (SC): Average silhouette widths.

  A rule of thumb from Anja et al. "*Clustering in an Object-Oriented Environment*":

  - SC $>$ 70%: A strong structure has been found.
  - SC $>$ 50%: A reasonable structure has been found.
  - SC $>$ 26% : The structure is weak and could be artificial, try additional methods.
  - SC $<$ 26%: No substantial structure has been found.

- We can plot SC(K) over K and pick K that achieves the largest SC.

Prediction Strength

- Step 0: Divide the data into A and B two sets (training and test).
- Step 1: Cluster the m obs in B into K clusters, $C_1, \ldots, C_K$ of size $m_1, \ldots, m_K$ (the truth).
- Step 2: Cluster A into K clusters and use the corresponding clustering rule to assign the m obs from B into K clusters (the prediction).

- Can we evaluate the classification accuracy on the m obs in B with the label being their cluster membership?
  - Not appropriate, because of the labelling issue: the meaning of "cluster 1" changes when data change; cluster 1 at Step 1 is not the same as cluster 1 at Step 2.
- A better choice is to measure the error on the Association (or co-membership) matrix A: $A_{ij} = 1$ if i and j are in the same cluster, and 0, otherwise.

- Step 0: Divide the data into A and B two sets (training and test).
- Step 1: Cluster the m obs in B into K clusters, $C_1, \ldots, C_K$ of size $m_1, \ldots, m_K$ (the truth).
- Step 2: Cluster A into K clusters and use the corresponding clustering rule to assign the m obs from B into K clusters (the prediction).
- Prediction strength (Tibshirani and Walther 2005) measures the worst performance of predicting pairwise co-membership: $\min_{j\in\{1,\ldots,K\}} \frac{1}{n_j(n_j-1)} \sum_{i,i'\in C_j} I_{ii'}$, where $I_{ii'} = 1$ if i and i' assigned to the same group at Step 2.

Hierarchical Clustering Methods

- Input:
  - Pairwise dissimilarity matrix of observations
  - Rule to calculate dissimilarity between (disjoint) groups of observations
- Output: a hierarchical clustering result
  - n single-point clusters at the lowest level;
  - one cluster at the highest level;
  - clusters at one level are created by splitting/merging clusters at the next higher/lower level.

- Bottom-up clustering starts with all observations separate, and successively joins together the closest groups of observations until all observations are in a single group.
- Dissimilarty (or distance) between groups of obs
  - Single-linkage (nearest-neighbor)
  - Complete-linkage (furthest-neighbor)
  - Group average

- Top-down clustering starts with all observations together, and successively divide into two groups until all observations separate.
- Hybrid (Chipman and Tibshirani, 2006). Check the `hybridHclust` package.