Hierarchical Agglomerative Clustering
What is Clustering?

**Clustering** is a process of partitioning a set of data (or objects) in a set of meaningful sub-classes, called **clusters**

Helps users understand the natural grouping or structure in a data set

- **Cluster:**
  - a collection of data objects that are “similar” to one another
  - and "different" from other clusters
Clustering Methods

• **Agglomerative** *(bottom-up)*
  - iteratively combine clusters to form larger and larger clusters.

• **Divisive** *(partitional, top-down)* separate all examples into clusters. Separate clusters into subclusters.
  - Decision Trees
    - But they require labelled examples

• **K-Means** start with K "centers", assign data to nearest. Compute new centers. Repeat.
  - This week's lab!!!
Hierarchical Clustering

- Builds a tree-based hierarchical taxonomy (dendrogram) from a set of examples.
Hierarchical Agglomerative Clustering (HAC)

- Assumes a similarity function for determining the similarity of two instances.
- Repeatedly join the two clusters that are most similar until there is only one cluster.
- The history of merging forms a binary tree or hierarchy (dendrogram).
HAC Algorithm

Start with all instances in their own cluster.
Until there is only one cluster:
   Among the current clusters, determine the two clusters, $c_i$ and $c_j$, that are most similar.
   Replace $c_i$ and $c_j$ with a single cluster $c_i \cup c_j$
Cluster Similarity

• Assume a similarity function that determines the similarity of two instances: \( sim(x, y) \).

• How to compute similarity of two clusters each possibly containing multiple instances?
  – **Single Link**: Similarity of two most similar members.
  – **Complete Link**: Similarity of two least similar members.
  – **Centroid**: Average similarity between members.

– Monotonically increasing!
  – Desirable but not true of every technique
  – Yes: Single Link, Complete Link
  – No (possibly): Centroid
Single Link Agglomerative Clustering

• Use maximum similarity of pairs:
  \[ sim(c_i, c_j) = \max_{x \in c_i, y \in c_j} sim(x, y) \]

• Can result in “straggly” (long and thin) clusters due to *chaining effect*.
Complete Link Agglomerative Clustering

- Use minimum similarity of pairs:
  \[ \text{sim}(c_i, c_j) = \min_{x \in c_i, y \in c_j} \text{sim}(x, y) \]
- Makes more “tight,” spherical clusters that are typically preferable.
What is a "centroid"

- The average of the points in a cluster.
  - \( c_z = \left(1/N_z\right) \sum_{i=0..Nz} (d_i) \)
  - where \( c_z \) is the centroid of cluster \( Z \)
  - \( N_z \) is number of docs in cluster
  - \( d_i \) is a document in cluster \( Z \)
  - "sum" does each dimension independently

<table>
<thead>
<tr>
<th></th>
<th>Dim1</th>
<th>Dim2</th>
<th>Dim3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc1</td>
<td>1</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Doc2</td>
<td>17</td>
<td>20</td>
<td>-5</td>
</tr>
<tr>
<td>Doc3</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Doc4</td>
<td>2</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Doc5</td>
<td>5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>( C_Z )</td>
<td>7.0</td>
<td>8.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>
**Centroid Agglomerative Clustering**

- distance to other clusters is the distance between centroids
  - Math: the distance from a document to a cluster centroid is average of distance from that document to each document in the cluster.
- Every time you change a cluster, you have to recompute distance to every other cluster
Computational Complexity

• In the first iteration, all HAC methods need to compute similarity of all pairs of $n$ individual instances which is $O(n^2)$.

• In each of the subsequent $n-2$ merging iterations, must compute the distance between the most recently created cluster and all other existing clusters.
  • Why $n-2$?
  • Overall $O(n^3)$
    • can get $O(n^2)$ for single link
Stopping HAC

• HAC will continue until there is a single cluster
• Often this does not make sense
  • Cutoff clustering at a specified distance
  • Cutoff clustering when you get a big jump in distance
  • Cutoff when you have a set number of clusters
Distance Dendrograms -- Visualizing HAC

- Create a graph in which y axis is distance between clusters.
- Given monotonically increasing distance get a nice looking graphic
  - Each time you merge a cluster
    - draw two vertical lines from last cluster distance to new cluster distance
    - draw a horizontal line connecting vertical lines at distance between merged clusters
  - clusters all start at 0.
Dendrograms -- Example

- Suppose 4 items and using single link distance. Distances are in the table.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>--</td>
<td>2</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>B</td>
<td>--</td>
<td>1</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>--</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Step 1: Group together B, C with cost 1.
- Step 2: Group A, (B,C) with cost 2.
- Step 3: Group D, (A,B,C) with cost 3.
## Exercise

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>b</td>
<td>87</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>c</td>
<td>92</td>
<td>91</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>d</td>
<td>74</td>
<td>22</td>
<td>60</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>e</td>
<td>24</td>
<td>21</td>
<td>28</td>
<td>17</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>f</td>
<td>6</td>
<td>31</td>
<td>90</td>
<td>17</td>
<td>8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>g</td>
<td>5</td>
<td>3</td>
<td>78</td>
<td>8</td>
<td>56</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>h</td>
<td>54</td>
<td>47</td>
<td>15</td>
<td>65</td>
<td>61</td>
<td>80</td>
<td>47</td>
</tr>
</tbody>
</table>

Suppose you have 7 items (a-g) with distances between items in the table above.

Show the dendrogram for single link and complete-link clustering for the above data.
Clustering Issues

- Suppose you have a feature for which distance is not defined (or at least not well-defined)
- Features on different ranges
- Features with same range but different variance
Normalization

- Ensure features have same mean and variance
- Why?

Algorithm

Given $D$ -- an $N \times M$ array where $N$ is the number of independent data items and $M$ is the dimensionality of the items
Return -- an $N \times M$ array in which the original date has been transformed to have 0 mean and unit variance

$\text{Normalize}(D)$:

- let $R$ = an $N \times M$ array, initially 0
- for $m$ in $0..(M-1)$:
  - let $av$ = average of the $N$ items on feature $m$
  - let $sd$ = standard deviation of $N$ items on feature $m$
  - for $n$ in $0..(N-1)$:
    - $R[n][m] = (D[n][m] - av)/sd$
- return $R$
A similarity measure is a function that computes the degree of similarity between two vectors.

Using a similarity measure between two vectors:
- It is possible to give a total ordering of the distances between a target vector and a set of vectors.
- It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.
Similarity Measure
Inner Product

• Similarity between vectors for the document \(d_i\) and query \(q\) can be computed as the vector inner product (a.k.a. dot product):

\[
\text{sim}(d_j, q) = d_j \cdot q = \sum_{i=1}^{t} w_{ij} w_{iq}
\]

where \(w_{ij}\) is the weight of term \(i\) in document \(j\) and \(w_{iq}\) is the weight of term \(i\) in the query.

• For binary vectors, the inner product is the number of matched query terms in the document (size of intersection).

• For weighted term vectors, it is the sum of the products of the weights of the matched terms.
Properties of Inner Product

• The inner product is unbounded.

• Works best when features are either binary or binary-like
  • 0 indicates absence / false

• Requires that features are strictly non-negative
Cosine Similarity Measure

- Cosine similarity measures the cosine of the angle between two vectors.
- Inner product normalized by the vector lengths.

\[
\text{CosSim}(d_j, q) = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}
\]

Suppose that VS has only 3 terms (dimensions) A, B, and C, corresponding to \(t_1, t_2\) and \(t_3\) in graph.

\[D_1 = (2, 3, 5)\] \[\text{CosSim}(D_1, Q) = \frac{10}{\sqrt{(4+9+25)(0+0+4)}} = 0.81\]

\[D_2 = (3, 7, 1)\] \[\text{CosSim}(D_2, Q) = \frac{2}{\sqrt{(9+49+1)(0+0+4)}} = 0.13\]

\[Q = (0, 0, 2)\]

\(D_1\) is 6 times better than \(D_2\) using cosine similarity but only 5 times better using inner product.
"Norms"

• Properties of norms:
  • Non-negativity: It should always be non-negative.
  • Definiteness: It is zero if and only if the vector is zero, i.e., zero vector.
  • Triangle inequality: The norm of a sum of two vectors is no more than the sum of their norms.
  • Homogeneity: Multiplying a vector by a scalar multiplies the norm of the vector by the absolute value of the scalar.

• Given 2 vectors X and Y
  • Euclidian Norm (2-Norm)
    • $\sqrt{\text{sum of squares of } X_i-Y_i}$
  • 1-Norm (Manhattan Distance)
  • Infinity-Norm
    • conceptually similar to complete link
  • P-Norm