Clustering

Groups together “similar” instances in the data sample

**Basic clustering problem:**
- distribute data into \( k \) different groups such that data points **similar** to each other are in the same group
- **Similarity** between data points is typically defined in terms of some distance metric (can be chosen)
Clustering

Groups together “similar” instances in the data sample

**Basic clustering problem:**

- distribute data into \( k \) different groups such that data points similar to each other are in the same group
- Similarity between data points is typically defined in terms of some distance metric (can be chosen)

Clustering example

Clustering could be applied to different types of data instances

**Example:** partition patients into groups based on similarities

<table>
<thead>
<tr>
<th>Patient #</th>
<th>Age</th>
<th>Sex</th>
<th>Heart Rate</th>
<th>Blood pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>55</td>
<td>M</td>
<td>85</td>
<td>125/80</td>
</tr>
<tr>
<td>Patient 2</td>
<td>62</td>
<td>M</td>
<td>87</td>
<td>130/85</td>
</tr>
<tr>
<td>Patient 3</td>
<td>67</td>
<td>F</td>
<td>80</td>
<td>126/86</td>
</tr>
<tr>
<td>Patient 4</td>
<td>65</td>
<td>F</td>
<td>90</td>
<td>130/90</td>
</tr>
<tr>
<td>Patient 5</td>
<td>70</td>
<td>M</td>
<td>84</td>
<td>135/85</td>
</tr>
</tbody>
</table>
Clustering example

Clustering could be applied to different types of data instances

**Example:** partition patients into groups based on similarities

<table>
<thead>
<tr>
<th>Patient #</th>
<th>Age</th>
<th>Sex</th>
<th>Heart Rate</th>
<th>Blood pressure …</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>55</td>
<td>M</td>
<td>85</td>
<td>125/80</td>
</tr>
<tr>
<td>Patient 2</td>
<td>62</td>
<td>M</td>
<td>87</td>
<td>130/85</td>
</tr>
<tr>
<td>Patient 3</td>
<td>67</td>
<td>F</td>
<td>80</td>
<td>126/86</td>
</tr>
<tr>
<td>Patient 4</td>
<td>65</td>
<td>F</td>
<td>90</td>
<td>130/90</td>
</tr>
<tr>
<td>Patient 5</td>
<td>70</td>
<td>M</td>
<td>84</td>
<td>135/85</td>
</tr>
</tbody>
</table>

**Key question: How to define similarity between instances?**

---

Similarity and dissimilarity measures

- **Dissimilarity measure**
  - Numerical measure of how different two data objects are
  - Often expressed in terms of a distance metric
- **Example:** Euclidean:
  
  \[ d(a, b) = \sqrt{\sum_{i=1}^{k} (a_i - b_i)^2} \]

- **Similarity measure**
  - Numerical measure of how alike two data objects are
- **Examples:**
  - **Cosine similarity:**
    
    \[ K(a, b) = a^T b \]
  - **Gaussian kernel:**
    
    \[ K(a, b) = \frac{1}{(2\pi h^2)^{d/2}} \exp \left[ -\frac{\| a - b \|^2}{2h^2} \right] \]
**Distance metrics**

Dissimilarity is often measured with the help of a distance metrics.

**Properties of distance metrics:**

Assume 2 data entries \( a, b \)

- **Positiveness:** \( d(a, b) \geq 0 \)
- **Symmetry:** \( d(a, b) = d(b, a) \)
- **Identity:** \( d(a, a) = 0 \)
- **Triangle inequality:** \( d(a, c) \leq d(a, b) + d(b, c) \)

**Distance metrics**

Assume 2 real-valued data-points:

- \( a=(6, 4) \)
- \( b=(4, 7) \)

What distance metric to use?
Distance metrics

Assume 2 real-valued data-points:

\( a = (6, 4) \)
\( b = (4, 7) \)

What distance metric to use?

**Euclidian:**

\[
d(a, b) = \sqrt{\sum_{i=1}^{k} (a_i - b_i)^2}
\]
Distance metrics

Assume 2 real-valued data-points:

\[ a = (6, 4) \]
\[ b = (4, 7) \]

What distance metric to use?

**Squared Euclidean:** works for an arbitrary k-dimensional space

\[ d^2(a, b) = \sum_{i=1}^{k} (a_i - b_i)^2 \]

**Manhattan distance:** works for an arbitrary k-dimensional space

\[ d(a, b) = \sum_{i=1}^{k} |a_i - b_i| \]
Distance measures

**Generalized distance metric:**

\[ d^2 (\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^T \Gamma^{-1} (\mathbf{a} - \mathbf{b}) \]

- \( \Gamma \) is a semi-definite positive matrix.
- \( \Gamma^{-1} \) is a matrix that weights attributes proportionally to their importance. Different weights lead to a different distance metric.

If \( \Gamma = I \) we get **squared Euclidean**

- \( \Gamma = \Sigma \) (covariance matrix) – we get the **Mahalanobis distance** that takes into account correlations among attributes.

---

**Distance measures**

**Generalized distance metric:**

\[ d^2 (\mathbf{a}, \mathbf{b}) = (\mathbf{a} - \mathbf{b})^T \Gamma^{-1} (\mathbf{a} - \mathbf{b}) \]

**Special case:** \( \Gamma = I \) we get **squared Euclidean**

**Example:**

- \( \mathbf{a} = \begin{bmatrix} 6 \\ 4 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 4 \\ 7 \end{bmatrix}, \Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \Gamma^{-1} \)

\[
\begin{align*}
    d^2 (\mathbf{a}, \mathbf{b}) &= (\mathbf{a} - \mathbf{b})^T \Gamma^{-1} (\mathbf{a} - \mathbf{b}) \\
    &= \begin{bmatrix} 2 & -3 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ -3 \end{bmatrix} \\
    &= 2^2 + (-3)^2 = 13
\end{align*}
\]
Distance measures

Generalized distance metric:

\[ d^2(a, b) = (a - b)^T \Gamma^{-1} (a - b) \]

Special case: \( \Gamma = \Sigma \) defines Mahalanobis distance

Example: Assume dimensions are independent in data

Covariance matrix \( \Sigma \) and Inverse covariance \( \Sigma^{-1} \):

\[
\Sigma = \begin{pmatrix}
\sigma_1^2 & 0 \\
0 & \sigma_2^2
\end{pmatrix}
\]

\[
\Sigma^{-1} = \begin{pmatrix}
\frac{1}{\sigma_1^2} & 0 \\
0 & \frac{1}{\sigma_2^2}
\end{pmatrix}
\]

\[ d^2(a, b) = [2 -3] \begin{pmatrix}
\frac{1}{\sigma_1^2} & 0 \\
0 & \frac{1}{\sigma_2^2}
\end{pmatrix}^{-2} [2 -3] = \frac{2^2}{\sigma_1^2} + \frac{(-3)^2}{\sigma_2^2} \]

Contribution of each dimension to the squared Euclidean is normalized (rescaled) by the variance of that dimension.

Distance measures

Assume categorical data where integers represent the different categories:

0 1 1 0 0
1 0 3 0 1
2 1 1 0 2
1 1 1 1 2
...

What distance metric to use?
Distance measures

Assume categorical data where integers represent the different categories:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

... 

What distance metric to use?

**Hamming distance**: The number of values that need to be changed to make them the same

Distance measures.

Assume pure binary values data:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

...

One metric is the **Hamming distance**: The number of bits that need to be changed to make the entries the same

How about squared Euclidean?

\[ d^2(a, b) = \sum_{i=1}^{k} (a_i - b_i)^2 \]
Distance measures.

Assume pure binary values data:

\[
\begin{align*}
0 & \ 1 \ 1 \ 0 \ 1 \\
1 & \ 0 \ 1 \ 0 \ 1 \\
0 & \ 1 \ 1 \ 0 \ 1 \\
1 & \ 1 \ 1 \ 1 \ 1 \\
\ldots
\end{align*}
\]

One metric is the **Hamming distance**: The number of bits that need to be changed to make the entries the same.

How about the squared Euclidean?

\[
d^2(a, b) = \sum_{i=1}^{k} (a_i - b_i)^2
\]

**The same as Hamming distance.**

---

Distance measures

Combination of real-valued and categorical attributes

<table>
<thead>
<tr>
<th>Patient #</th>
<th>Age</th>
<th>Sex</th>
<th>Heart Rate</th>
<th>Blood pressure</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>55</td>
<td>M</td>
<td>85</td>
<td>125/80</td>
<td></td>
</tr>
<tr>
<td>Patient 2</td>
<td>62</td>
<td>M</td>
<td>87</td>
<td>130/85</td>
<td></td>
</tr>
<tr>
<td>Patient 3</td>
<td>67</td>
<td>F</td>
<td>80</td>
<td>126/86</td>
<td></td>
</tr>
<tr>
<td>Patient 4</td>
<td>65</td>
<td>F</td>
<td>90</td>
<td>130/90</td>
<td></td>
</tr>
<tr>
<td>Patient 5</td>
<td>70</td>
<td>M</td>
<td>84</td>
<td>135/85</td>
<td></td>
</tr>
</tbody>
</table>

What distance metric to use?
Distance measures

Combination of real-valued and categorical attributes

<table>
<thead>
<tr>
<th>Patient #</th>
<th>Age</th>
<th>Sex</th>
<th>Heart Rate</th>
<th>Blood pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Patient 1</td>
<td>55</td>
<td>M</td>
<td>85</td>
<td>125/80</td>
</tr>
<tr>
<td>Patient 2</td>
<td>62</td>
<td>M</td>
<td>87</td>
<td>130/85</td>
</tr>
<tr>
<td>Patient 3</td>
<td>67</td>
<td>F</td>
<td>80</td>
<td>126/86</td>
</tr>
<tr>
<td>Patient 4</td>
<td>65</td>
<td>F</td>
<td>90</td>
<td>130/90</td>
</tr>
<tr>
<td>Patient 5</td>
<td>70</td>
<td>M</td>
<td>84</td>
<td>135/85</td>
</tr>
</tbody>
</table>

What distance metric to use? Solutions:

- **A weighted sum approach**: e.g. a mix of Euclidian and Hamming distances for subsets of attributes
- **Generalized distance metric** (weighted combination, use one-hot representation of categories)

More complex solutions: tensors and decompositions

Distance metrics and similarity

- **Dissimilarity/distance measure**
- **Similarity measure**
  - Numerical measure of how alike two data objects are
  - Do not have to satisfy the properties like the ones for the distance metric
  - **Examples**:
    - **Cosine similarity**: \( K(a, b) = a^T b \)
    - **Gaussian kernel**:
      \[
      K(a, b) = \frac{1}{(2\pi h^2)^{d/2}} \exp\left(-\frac{||a - b||_2^2}{2h^2}\right)
      \]
Clustering

Clustering is useful for:

- **Similarity/Dissimilarity analysis**
  Analyze what data points in the sample are close to each other
- **Dimensionality reduction**
  High dimensional data replaced with a group (cluster) label
- **Data reduction**: Replaces many data-points with a point representing the group mean

**Challenges:**

- How to measure similarity (problem/data specific)?
- How to choose the number of groups?
  - Many clustering algorithms require us to provide the number of groups ahead of time

**Clustering algorithms**

**Algorithms covered:**

- **K-means algorithm**
- **Hierarchical methods**
  - Agglomerative
  - Divisive
K-means clustering algorithm

- An iterative clustering algorithm
- works in the d-dimensional $R$ space representing $x$

**K-Means clustering algorithm:**

**Initialize** randomly $k$ values of means (centers)

**Repeat**
- Partition the data according to the current set of means (using the similarity measure)
- Move the means to the center of the data in the current partition

**Until** no change in the means

---

K-means: example

- **Initialize the cluster centers**

---
K-means: example

- Calculate the distances of each point to all centers

K-means: example

- For each example pick the best (closest) center
K-means: example

- Recalculate the new mean from all data examples assigned to the same cluster center

K-means: example

- Shift the cluster center to the new mean
K-means: example

• Shift the cluster centers to the new calculated means

K-means: example

• And repeat the iteration …
• Till no change in the centers
K-means clustering algorithm

K-Means algorithm:
- **Initialize** randomly \( k \) values of means (centers)
- **Repeat**
  - Partition the data according to the current set of means (using the similarity measure)
  - Move the means to the center of the data in the current partition
- **Until** no change in the means

Properties:
- Minimizes the sum of squared center-point distances for all clusters
  \[
  \min_s \sum_{i=1}^{k} \sum_{x_j \in S_i} \| x_j - u_i \|^2 \\
  u_i = \text{center of cluster } S_i
  \]

K-means clustering algorithm

- **Properties**: 
  - **converges** to centers minimizing the sum of squared center-point distances (still local optima)
  - The result is sensitive to the initial means’ values
- **Advantages**: 
  - Simplicity
  - Generality – can work for more than one distance measure
- **Drawbacks**: 
  - Can perform poorly with overlapping regions
  - Lack of robustness to outliers
  - Good for attributes (features) with continuous values
    - Allows us to compute cluster means
    - k-medoid algorithm used for discrete data
Hierarchical clustering

- Builds a hierarchy of clusters (groups) with singleton groups at the bottom and ‘all points’ group on the top

Uses many different dissimilarity measures
- Pure real-valued data-points:
  - Euclidean, Manhattan, Minkowski
- Pure categorical data:
  - Hamming distance,
  - Combination of real-valued and categorical attributes
  - Weighted, or Euclidean

Hierarchical clustering

Two versions of the hierarchical clustering
- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- **Divisive approach:**
  - Splits clusters in top-down fashion, starting from one complete cluster
Hierarchical (agglomerative) clustering

Approach:
- Compute dissimilarity matrix for all pairs of points
  - uses standard or other distance measures
- Construct clusters greedily:
  - Agglomerative approach
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- Stop the greedy construction when some criterion is satisfied
  - E.g. fixed number of clusters
Hierarchical (agglomerative) clustering

Approach:
- Compute dissimilarity matrix for all pairs of points
  - uses standard or other distance measures

N datapoints, O(N^2) pairs, O(N^2) distances

Hierarchical (agglomerative) clustering

Approach:
- Compute dissimilarity matrix for all pairs of points
  - uses standard or other distance measures
- Construct clusters greedily:
  - Agglomerative approach
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
Hierarchical (agglomerative) clustering

Approach:
- Compute dissimilarity matrix for all pairs of points
  - uses standard or other distance measures
- Construct clusters greedily:
  - Agglomerative approach
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
Hierarchical (agglomerative) clustering

Approach:
- Compute dissimilarity matrix for all pairs of points
  - uses standard or other distance measures
- Construct clusters greedily:
  - Agglomerative approach
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters

Cluster merging

- Agglomerative approach
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on cluster (or linkage) distances. Defined in terms of point distances. Examples:

  Min distance $d_{\text{min}}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$
Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

  \[
  \text{Max distance } d_{\text{max}}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)
  \]

Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

  \[
  \text{Mean distance } d_{\text{mean}}(C_i, C_j) = \left| \frac{1}{|C_i|} \sum_i P_i - \frac{1}{|C_j|} \sum_j q_j \right|
  \]
Hierarchical (agglomerative) clustering

Approach:
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - **Stop the greedy construction** when some criterion is satisfied
    - E.g. fixed number of clusters

Hierarchical (divisive) clustering

Approach:
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard distance or other dissimilarity measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - **Divisive approach:**
    - Splits clusters in top-down fashion, starting from one complete cluster
  - **Stop the greedy construction** when some criterion is satisfied
    - E.g. fixed number of clusters
Hierarchical clustering example

- Dendogram
Hierarchical clustering

- **Advantage:**
  - Smaller computational cost; avoids scanning all possible clusters

- **Disadvantage:**
  - Greedy choice fixes the order in which clusters are merged; cannot be repaired

- **Partial solution:**
  - Combine hierarchical clustering with iterative algorithms like k-means algorithm