LECTURE OUTLINE

1. Unsupervised learning
2. Problems and types of clustering
3. Clustering techniques
   - $k$-means clustering
   - hierarchical clustering
   - probability-based clustering
What is Clustering?

Also called *unsupervised learning*, sometimes called *classification* by statisticians and *sorting* by psychologists and *segmentation* by people in marketing.

- Organizing data into classes such that there is
  - high intra-class similarity
  - low inter-class similarity
- Finding the class labels and the number of classes directly from the data (in contrast to classification).
- More informally, finding natural groupings among objects.
What is a natural grouping among these objects?
What is a natural grouping among these objects?

Clustering is subjective

Simpson's Family
School Employees
Females
Males
What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

Webster's Dictionary

Similarity is hard to define, but…
“We know it when we see it”

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.
Defining Distance Measures

**Definition:** Let $O_1$ and $O_2$ be two objects from the universe of possible objects. The distance (dissimilarity) between $O_1$ and $O_2$ is a real number denoted by $D(O_1, O_2)$. 

0.23  

3  

342.7  

Peter  Piotr
What properties should a distance measure have?

- \( D(A,B) = D(B,A) \) \hspace{1cm} \text{Symmetry}
- \( D(A,A) = 0 \) \hspace{1cm} \text{Constancy of Self-Similarity}
- \( D(A,B) = 0 \) if \( A = B \) \hspace{1cm} \text{Positivity (Separation)}
- \( D(A,B) \leq D(A,C) + D(B,C) \) \hspace{1cm} \text{Triangular Inequality}
Intuitions behind desirable distance measure properties

\[ D(A, B) = D(B, A) \quad \text{Symmetry} \]
Otherwise you could claim “Alex looks like Bob, but Bob looks nothing like Alex.”

\[ D(A, A) = 0 \quad \text{Constancy of Self-Similarity} \]
Otherwise you could claim “Alex looks more like Bob, than Bob does.”

\[ D(A, B) = 0 \text{ Iff } A = B \quad \text{Positivity (Separation)} \]
Otherwise there are objects in your world that are different, but you cannot tell apart.

\[ D(A, B) \leq D(A, C) + D(B, C) \quad \text{Triangular Inequality} \]
Otherwise you could claim “Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl.”
TWO TYPES OF CLUSTERING

- **Partitional algorithms**: Construct various partitions and then evaluate them by some criterion (we will see an example called BIRCH)
- **Hierarchical algorithms**: Create a hierarchical decomposition of the set of objects using some criterion
DESIRABLE PROPERTIES OF A CLUSTERING ALGORITHM

• Scalability (in terms of both time and space)
• Ability to deal with different data types
• Minimal requirements for domain knowledge to determine input parameters
• Able to deal with noise and outliers
•Insensitive to order of input records
• Incorporation of user-specified constraints
• Interpretability and usability
HIERARCHICAL CLUSTERING
A USEFUL TOOL FOR SUMMARIZING SIMILARITY MEASUREMENTS

In order to better appreciate and evaluate the examples given in the early part of this talk, we will now introduce the dendrogram.

The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.
There is only one dataset that can be perfectly clustered using a hierarchy...
Note that hierarchies are commonly used to organize information, for example in a web portal.

Yahoo’s hierarchy is manually created, we will focus on automatic creation of hierarchies in data mining.
**A DEMONSTRATION OF HIERARCHICAL CLUSTERING USING STRING EDIT DISTANCE**

Pedro (Portuguese), Petros (Greek), Peter (English), Piotr (Polish), Peadar (Irish), Pierre (French), Peder (Danish), Peka (Hawaiian), Pietro (Italian), Piero (Italian Alternative), Petr (Czech), Pyotr (Russian)

Cristovao (Portuguese), Christoph (German), Christophe (French), Cristobal (Spanish), Cristoforo (Italian), Kristoffer (Scandinavian), Krystof (Czech), Christopher (English)

Miguel (Portuguese), Michalis (Greek), Michael (English), Mick (Irish)
Pedro (Portuguese/Spanish)
Petros (Greek), Peter (English), Piotr (Polish),
Peadar (Irish), Pierre (French), Peder (Danish),
Pekā (Hawaiian), Pietro (Italian), Piero (Italian Alternative), Petr (Czech), Pyotr (Russian)
Hierarchal clustering can sometimes show patterns that are meaningless or spurious

- For example, in this clustering, the tight grouping of Australia, Anguilla, St. Helena etc is meaningful, since all these countries are former UK colonies.

- However the tight grouping of Niger and India is completely spurious, there is no connection between the two.
• The flag of Niger is orange over white over green, with an orange disc on the central white stripe, symbolizing the sun. The orange stands the Sahara desert, which borders Niger to the north. Green stands for the grassy plains of the south and west and for the River Niger which sustains them. It also stands for fraternity and hope. White generally symbolizes purity and hope.

• The Indian flag is a horizontal tricolor in equal proportion of deep saffron on the top, white in the middle and dark green at the bottom. In the center of the white band, there is a wheel in navy blue to indicate the Dharma Chakra, the wheel of law in the Sarnath Lion Capital. This center symbol or the 'CHAKRA' is a symbol dating back to 2nd century BC. The saffron stands for courage and sacrifice; the white, for purity and truth; the green for growth and auspiciousness.
We can look at the dendrogram to determine the “correct” number of clusters. In this case, the two highly separated subtrees are highly suggestive of two clusters. 

(Things are rarely this clear cut, unfortunately)
One potential use of a dendrogram is to detect outliers.

The single isolated branch is suggestive of a data point that is very different to all others.
**HOW-TO HIERARCHICAL CLUSTERING**

The number of dendrograms with $n$ leaves is given by:

$$\text{leaves} = \frac{(2^n - 3)!}{[(2^{n-2})(n-2)!]}$$

<table>
<thead>
<tr>
<th>Number of Leafs</th>
<th>Number of Possible Dendrograms</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
</tr>
<tr>
<td>...</td>
<td>…</td>
</tr>
<tr>
<td>10</td>
<td>34,459,425</td>
</tr>
</tbody>
</table>

Since we cannot test all possible trees, we will have to heuristic search of all possible trees. We could do this.

**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

**Top-Down (divisive):** Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.
We begin with a distance matrix which contains the distances between every pair of objects in our database.

$$D(\text{, }) = 8$$

$$D(\text{, }) = 1$$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>8</th>
<th>8</th>
<th>7</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>2</th>
<th>4</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges...  
Choose the best
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges... Choose the best

Consider all possible merges... Choose the best
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges...

Choose the best

Consider all possible merges...

Choose the best

Consider all possible merges...

Choose the best
**Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges...

Choose the best

Consider all possible merges...

Choose the best

Consider all possible merges...

Choose the best
HOW TO DEFINE THE DISTANCE BETWEEN TWO CLUSTERS?

• **Single linkage (nearest neighbor):** In this method the distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters.

• **Complete linkage (furthest neighbor):** In this method, the distances between clusters are determined by the greatest distance between any two objects in the different clusters (i.e., by the "furthest neighbors").

• **Group average linkage:** In this method, the distance between two clusters is calculated as the average distance between all pairs of objects in the two different clusters.

• **Wards Linkage:** In this method, we try to minimize the variance of the merged clusters.
Summary of Hierarchical Clustering Methods

- No need to specify the number of clusters in advance.
- Hierarchical nature maps nicely onto human intuition for some domains.
- They do not scale well: time complexity of at least $O(n^2)$, where $n$ is the number of total objects.
- Like any heuristic search algorithms, local optima are a problem.
- Interpretation of results is (very) subjective.
UP TO THIS POINT WE HAVE SIMPLY ASSUMED THAT WE CAN MEASURE SIMILARITY, BUT HOW DO WE MEASURE SIMILARITY?
A generic technique for measuring similarity

To measure the similarity between two objects, transform one of the objects into the other, and measure how much effort it took. The measure of effort becomes the distance measure.

The distance between Patty and Selma.
- Change dress color, 1 point
- Change earring shape, 1 point
- Change hair part, 1 point

\[ D(\text{Patty}, \text{Selma}) = 3 \]

The distance between Marge and Selma.
- Change dress color, 1 point
- Add earrings, 1 point
- Decrease height, 1 point
- Take up smoking, 1 point
- Lose weight, 1 point

\[ D(\text{Marge}, \text{Selma}) = 5 \]

This is called the “edit distance” or the “transformation distance”
Edit Distance Example

It is possible to transform any string $Q$ into string $C$, using only Substitution, Insertion and Deletion. Assume that each of these operators has a cost associated with it.

The similarity between two strings can be defined as the cost of the cheapest transformation from $Q$ to $C$.

Note that for now we have ignored the issue of how we can find this cheapest transformation.

How similar are the names “Peter” and “Piotr”? Assume the following cost function

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Substitution</td>
<td>1 Unit</td>
</tr>
<tr>
<td>Insertion</td>
<td>1 Unit</td>
</tr>
<tr>
<td>Deletion</td>
<td>1 Unit</td>
</tr>
</tbody>
</table>

$D(\text{Peter}, \text{Piotr})$ is 3
Partitional Clustering

- Nonhierarchichal, each instance is placed in exactly one of \( K \) nonoverlapping clusters.
- Since only one set of clusters is output, the user normally has to input the desired number of clusters \( K \).
EXAMPLE: VECTOR QUANTIZATION

Small true color picture of size 100x100 requires
10000*24 = 29.3 kB;
(N=10000)

If we can represent the same picture but use only k=32 colors
➢ only 5 bits are needed to encode the color of each pixel
➢ Memory reduction to 6.1 kB
➢ Codebook size: 32*24 bits < 0.1 kB
1. Decide on a value for $k$.
2. Initialize the $k$ cluster centers (randomly, if necessary).
3. Decide the class memberships of the $N$ objects by assigning them to the nearest cluster center.
4. Re-estimate the $k$ cluster centers, by assuming the memberships found above are correct.
5. If none of the $N$ objects changed membership in the last iteration, exit. Otherwise goto 3.

$$Cost(c_1, \ldots, c_k \mid D) = \sum_{i=1}^{N} \min_{j=1, \ldots, k} \|x_i - c_j\|^2$$
Squared Error

$$se_{K_i} = \sum_{j=1}^{m} \| t_{i,j} - C_k \|^2$$

$$se_K = \sum_{j=1}^{k} se_{K_j}$$
K-MEANS CLUSTERING: STEP 1

Algorithm: k-means, Distance Metric: Euclidean Distance

![Graph showing k-means clustering with three clusters: k₁, k₂, and k₃. The data points are scattered across a 2D space with clusters centered around the respective k values.](image-url)
K-MEANS CLUSTERING: STEP 2

Algorithm: k-means, Distance Metric: Euclidean Distance
K-MEANS CLUSTERING: STEP 3

Algorithm: k-means, Distance Metric: Euclidean Distance
K-MEANS CLUSTERING: STEP 4

Algorithm: k-means, Distance Metric: Euclidean Distance
K-MEANS CLUSTERING: STEP 5

Algorithm: k-means, Distance Metric: Euclidean Distance
Example with $k=2$
COMMENTS ON THE K-MEANS METHOD

Strength

- Relatively efficient: $O(tkn)$, where $n$ is # objects, $k$ is # clusters, and $t$ is # iterations. Normally, $k, t << n$.
- Often terminates at a local optimum. The global optimum may be found using techniques such as: deterministic annealing and genetic algorithms

Weakness

- Applicable only when mean is defined, then what about categorical data?
- Need to specify $k$, the number of clusters, in advance
- Unable to handle noisy data and outliers
- Not suitable to discover clusters with non-convex shapes
**Theorem:** For any set of points $X \subset \mathbb{R}^n$ and for any point $z \in \mathbb{R}^n$ we have

$$Cost(z \mid X) = Cost(mean(X) \mid X) + |X| \cdot \|mean(X) - z\|^2$$
THE *K-MEDOIDS* CLUSTERING METHOD

Find *representative* objects, called *medoids*, in clusters

*PAM* (Partitioning Around Medoids, 1987)

- starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
- *PAM* works effectively for small data sets, but does not scale well for large data sets
How can we tell the *right* number of clusters?

In general, this is an unsolved problem. However, there are many approximate methods. In the next few slides, we will see an example.

For our example, we will use the familiar *katydid/grasshopper* dataset.

However, in this case, we are imagining that we do NOT know the class labels. We are only clustering on the X and Y axis values.
When $k = 1$, the objective function is 873.0.
When $k = 2$, the objective function is 173.1
When $k = 3$, the objective function is 133.6
We can plot the objective function values for $k$ equals 1 to 6…

The abrupt change at $k = 2$, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is known as “knee finding” or “elbow finding”.

Note that the results are not always as clear cut as in this toy example.
FUZZY CLUSTERING
EM Algorithm

- Initialize K cluster centers
- Iterate between two steps
  - **Expectation step**: assign points to clusters
    \[
    P(d_i \in c_k) = w_k \Pr(d_i \mid c_k) \bigg/ \sum_j w_j \Pr(d_i \mid c_j)
    \]
    \[
    w_k = \frac{i}{\sum \Pr(d_i \in c_k)}
    \]
  - **Maximation step**: estimate model parameters
    \[
    \mu_k = \frac{1}{m} \sum_{i=1}^{m} \frac{d_i P(d_i \in c_k)}{\sum_k P(d_i \in c_j)}
    \]
Iteration 1

The cluster means are randomly assigned

Mean Likelihood = -13.1162400084091007

0.3225806451612903

0.322580645161290
Mean Likelihood = -12.501313295068318

0.23392524956102798

0.4801378834773863

0.231996098004228

Iteration 2
Mean Likelihood = -11.879896828880106

Iteration 5
Mean Likelihood = -11.13452288716779

Iteration 25
Fuzzy C-Means

- Soft clustering
- Minimize functional

\[
E(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left\| \mathbf{x}_j - \mathbf{v}_i \right\|^2
\]

- \( U = \begin{bmatrix} u_{ij} \end{bmatrix}_{k \times n} \): fuzzy partition matrix \( u_{ij} \in [0,1] \)

- \( \sum_{i=1}^{k} u_{ij} = 1 \quad \forall j = 1, \ldots, n \)

- \( m \in [1, \infty] \): fuzzification parameter, usually set to 2

Data set: \( X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \)
Clusters: \( C_1, C_2, \ldots C_k \)
Codebook: \( V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\} \)
Partition matrix: \( \Gamma = \{\gamma_{ij}\} \gamma_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_j \in C_i \\ 0 & \text{otherwise} \end{cases} \)

K-means: \( E(\Gamma, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} \gamma_{ij} \left\| \mathbf{x}_j - \mathbf{v}_i \right\|^2 \)
FUZZY C-MEANS

Minimize

\[ E(U,V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left\| x_j - \bar{v}_i \right\|^2 \]

subject to

\[ \sum_{i=1}^{k} u_{ij} = 1 \quad \forall j = 1, \ldots, n \]
FUZZY C-MEANS

- Minimize

\[ E(U,V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \| \bar{x}_j - \bar{v}_i \|^2 \]

subject to

\[ \sum_{i=1}^{k} u_{ij} = 1 \quad \forall j = 1, \ldots, n \]

- How to solve this constrained optimization problem?
Fuzzy C-Means

- Minimize

\[ E(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \| \bar{x}_j - \bar{v}_i \|^2 \]

subject to

\[ \sum_{i=1}^{k} u_{ij} = 1 \quad \forall j = 1, \ldots, n \]

- How to solve this constrained optimization problem?
  - Introduce Lagrangian multipliers

\[ L_j (U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \| \bar{x}_j - \bar{v}_i \|^2 + \alpha_j \left( \sum_{i=1}^{k} u_{ij} - 1 \right) \]
Fuzzy C-Means

- Introduce Lagrangian multipliers
  \[ L_j(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left\| \bar{x}_j - \bar{v}_i \right\|^2 + \alpha_j \left( \sum_{i=1}^{k} u_{ij} - 1 \right) \]

- Iterative optimization
  - Fix \( V \), optimize w.r.t. \( U \)
    \[ u_{ij} = \frac{1}{\sum_{l=1}^{c} \left( \frac{\left\| \bar{x}_j - \bar{v}_i \right\|^2}{\left\| \bar{x}_j - \bar{v}_l \right\|^2} \right)^{2/(m-1)}} \]
  - Fix \( U \), optimize w.r.t. \( V \)
    \[ \bar{v}_i = \frac{\sum_{j=1}^{n} (u_{ij})^m \bar{x}_j}{\sum_{j=1}^{n} (u_{ij})^m} \]
APPLICATION TO IMAGE SEGMENTATION

Original images

Segmentations

Homogenous intensity corrupted by 5% Gaussian noise

Accuracy = 96.02%

Sinusoidal inhomogenous intensity corrupted by 5% Gaussian noise

Image: Dao-Qiang Zhang, Song-Can Chen

Accuracy = 94.41%
KERNEL SUBSTITUTION TRICK

\[ \left\| \Phi(\bar{x}_j) - \Phi(\bar{v}_i) \right\|^2 \]

\[ = \Phi(\bar{x}_j)^T \Phi(\bar{x}_j) - \Phi(\bar{x}_j)^T \Phi(\bar{v}_i) - \Phi(\bar{v}_i)^T \Phi(\bar{x}_j) + \Phi(\bar{v}_i)^T \Phi(\bar{v}_i) \]

\[ = K(\bar{x}_j, \bar{x}_j) - 2K(\bar{x}_j, \bar{v}_i) + K(\bar{v}_i, \bar{v}_i) \]

Kernel K-means

\[ E(\Gamma, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} \gamma_{ij} \left\| \Phi(\bar{x}_j) - \Phi(\bar{v}_i) \right\|^2 \]

Kernel fuzzy c-means

\[ E(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left\| \Phi(\bar{x}_j) - \Phi(\bar{v}_i) \right\|^2 \]
**KERNEL SUBSTITUTION TRICK**

Kernel fuzzy c-means

\[
E(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left\| \Phi(\bar{x}_j) - \Phi(\bar{v}_i) \right\|^2
\]

Confine ourselves to Gaussian RBF kernel

\[
E(U, V) = 2 \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left( 1 - K(\bar{x}_j, \bar{v}_i) \right)
\]

Introduce a penalty term containing neighborhood information

\[
E(U, V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left( 1 - K(\bar{x}_j, \bar{v}_i) \right) + \frac{\alpha}{|N_j|} \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \sum_{\bar{x}_r \in N_j} (1-u_{ir})^m
\]

Equation: Dao-Qiang Zhang, Song-Can Chen
SPATIALLY CONSTRAINED KFCM

\[ E(U,V) = \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \left(1 - K(\bar{x}_j, \bar{v}_i)\right) + \frac{\alpha}{|N_j|} \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij})^m \sum_{\bar{x}_r \in N_j} (1 - u_{ir})^m \]

- \( N_j \): the set of neighbors that exist in a window around \( x_j \)
- \( |N_j| \): the cardinality of \( N_j \)
- \( \alpha \): the cardinality of \( \sum_{i=1}^{k} \sum_{j=1}^{n} \)

The penalty term is minimized when

- Membership value for \( x_i \) is large and also large at neighboring pixels
- Vice versa

Equation: Dao-Qiang Zhang, Song-Can Chen
FCM APPLIED TO SEGMENTATION

Original images

Homogenous intensity corrupted by 5% Gaussian noise

Image: Dao-Qiang Zhang, Song-Can Chen
FCM APPLIED TO SEGMENTATION

Original images

Sinusoidal inhomogenous intensity corrupted by 5% Gaussian noise

Image: Dao-Qiang Zhang, Song-Can Chen
FCM APPLIED TO SEGMENTATION

Original MR image corrupted by 5% Gaussian noise

<table>
<thead>
<tr>
<th></th>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCM</td>
<td>0.88</td>
<td>0.32</td>
<td>0.93</td>
</tr>
<tr>
<td>KFCM</td>
<td>0.79</td>
<td>0.19</td>
<td>0.91</td>
</tr>
<tr>
<td>SFCM</td>
<td>0.99</td>
<td>0.75</td>
<td>0.94</td>
</tr>
<tr>
<td>SKFCM</td>
<td>1.00</td>
<td>0.78</td>
<td>0.95</td>
</tr>
</tbody>
</table>

Image: Dao-Qiang Zhang, Song-Can Chen
INCREMENTAL CLUSTERING
Nearest Neighbor Clustering

Not to be confused with Nearest Neighbor Classification

• Items are iteratively merged into the existing clusters that are closest.

• Incremental

• Threshold, t, used to determine if items are added to existing clusters or a new cluster is created.

What happens if the data is streaming...
Threshold $t$
New data point arrives...

It is within the threshold for cluster 1, so add it to the cluster, and update cluster center.
New data point arrives…

It is not within the threshold for cluster 1, so create a new cluster, and so on..

Algorithm is highly order dependent…

It is difficult to determine $t$ in advance…
PARTITIONAL CLUSTERING ALGORITHMS

Clustering algorithms have been designed to handle very large datasets

E.g. the Birch algorithm

• Main idea: use an in-memory R-tree to store points that are being clustered

• Insert points one at a time into the R-tree, merging a new point with an existing cluster if is less than some $\delta$ distance away

• If there are more leaf nodes than fit in memory, merge existing clusters that are close to each other

• At the end of first pass we get a large number of clusters at the leaves of the R-tree
  ▪ Merge clusters to reduce the number of clusters
PARTITIONAL CLUSTERING ALGORITHMS

We need to specify the number of clusters in advance, I have chosen 2

The Birch algorithm

Data nodes containing points
PARTITIONAL CLUSTERING ALGORITHMS

The Birch algorithm

Data nodes containing points
PARTITIONAL CLUSTERING ALGORITHMS

The Birch algorithm
CLUSTER VALIDATION
CLUSTER VALIDITY

For cluster analysis, the question is how to evaluate the “goodness” of the resulting clusters?

But “clusters are in the eye of the beholder”!

Then why do we want to evaluate them?
- To avoid finding patterns in noise
- To compare clustering algorithms
- To compare two sets of clusters
- To compare two clusters
CLUSTERS FOUND IN RANDOM DATA

Random Points

K-means

DBSCAN

Complete Link
DIFFERENT ASPECTS OF CLUSTER VALIDATION

1. Determining the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.

2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.

3. Evaluating how well the results of a cluster analysis fit the data without reference to external information.
   - Use only the data

4. Comparing the results of two different sets of cluster analyses to determine which is better.

5. Determining the ‘correct’ number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.
FRAMEWORK FOR CLUSTER VALIDITY

Need a framework to interpret any measure.

- For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?

Statistics provide a framework for cluster validity

- The more “atypical” a clustering result is, the more likely it represents valid structure in the data
- Can compare the values of an index that result from random data or clusterings to those of a clustering result.
  - If the value of the index is unlikely, then the cluster results are valid
- These approaches are more complicated and harder to understand.

For comparing the results of two different sets of cluster analyses, a framework is less necessary.

- However, there is the question of whether the difference between two index values is significant
MEASURES OF CLUSTER VALIDITY

Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.

- **External Index**: Used to measure the extent to which cluster labels match externally supplied class labels.
  - Entropy

- **Internal Index**: Used to measure the goodness of a clustering structure without respect to external information.
  - Sum of Squared Error (SSE)

- **Relative Index**: Used to compare two different clusterings or clusters.
  - Often an external or internal index is used for this function, e.g., SSE or entropy

Sometimes these are referred to as criteria instead of indices

- However, sometimes criterion is the general strategy and index is the numerical measure that implements the criterion.
Algorithm 21.4: Algorithm for matching partitions and clusters

\[\text{MatchPartitionCluster}(P,C,\text{match}):\]

1. \textbf{foreach} \( p \in P \) \textbf{do}
2. \hspace{1em} \text{match}(p) \leftarrow \emptyset
3. \hspace{1em} \textbf{foreach} \( c \in C \) \textbf{do}
4. \hspace{2em} \text{overlap}(p, c) \leftarrow \frac{|p \cap c|}{|p|}
5. \hspace{1em} \textbf{while} \text{overlap} \neq \emptyset \textbf{do}
6. \hspace{2em} (p_{\text{max}}, c_{\text{max}}) \leftarrow \text{GetMaxOverlap(overlap)}
7. \hspace{2em} \text{match}(p_{\text{max}}) \leftarrow c_{\text{max}}
8. \hspace{2em} \text{overlap} \leftarrow \text{overlap} - \{\text{overlap}(p_{\text{max}}, \cdot), \text{overlap}(\cdot, c_{\text{max}})\}
PURITY-BASED MEASURE

Purity

\[ \text{Purity} = \max_j \rho_{ij} \quad \text{purity}_C = \sum_r \frac{|c_i|}{|c|} \text{purity}_i, \]

Precision/Recall/F-Measure

\[ \text{prec}(i,j), \text{recall}(i,j), \]

Entropy

\[ e_i = -\sum_q \rho_{ij} \log_2 \rho_{ij}. \]

\[ e_C = \sum_r \frac{|c_i|}{|c|} e_i. \]
MATCHING MEASURE

1. $x_C = y_C \land x_P = y_P$
2. $x_C = y_C \land x_P \neq y_P$
3. $x_C \neq y_C \land x_P = y_P$
4. $x_C \neq y_C \land x_P \neq y_P$

Rand Statistic:

$$\text{Rand}_{P,C} = \frac{CP + \overline{CP}}{m}$$

Jaccard Coefficient:

$$\text{Jaccard}_{P,C} = \frac{CP}{CP + C\overline{P} + \overline{C}P}$$
**CORRELATION MEASURE**

- Hubert’s Tau
  \[
  \Gamma = \frac{1}{m} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_P(i, j)X_C(i, j)
  \]

- Normalized Tau Statistics:
  \[
  \hat{\Gamma} = \frac{1}{m} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{(X_P(i, j) - \mu_P)(X_C(i, j) - \mu_C)}{\sigma_P \sigma_C}
  \]

where \(\mu_P\) and \(\mu_C\) are the means and \(\sigma_P\) and \(\sigma_C\) are the variances of the matrices \(X_C\) and \(X_P\).
MEASURING CLUSTER VALIDITY VIA CORRELATION

Two matrices
- Proximity Matrix
- “Incidence” Matrix
  - One row and one column for each data point
  - An entry is 1 if the associated pair of points belong to the same cluster
  - An entry is 0 if the associated pair of points belongs to different clusters

Compute the correlation between the two matrices
- Since the matrices are symmetric, only the correlation between \( n(n-1) / 2 \) entries needs to be calculated.

High correlation indicates that points that belong to the same cluster are close to each other.

Not a good measure for some density or contiguity based clusters.
MEASURING CLUSTER VALIDITY VIA CORRELATION

Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets.

\[
\text{Corr} = -0.9235 \quad \text{Corr} = -0.5810
\]
Using similarity matrix for cluster validation

Order the similarity matrix with respect to cluster labels and inspect visually.
**USING SIMILARITY MATRIX FOR CLUSTER VALIDATION**

Clusters in random data are not so crisp.

![Heatmap and scatter plot](Image)

**DBSCAN**
USING SIMILARITY MATRIX FOR CLUSTER VALIDATION

Clusters in random data are not so crisp

K-means
USING SIMILARITY MATRIX FOR CLUSTER VALIDATION

Complete Link
USING SIMILARITY MATRIX FOR CLUSTER VALIDATION

DBSCAN

0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
500 1000 1500 2000 2500 3000
500
1000
1500
2000
2500
3000

1 2 3 4 5 6 7
INTERNAL MEASURES: SSE

Clusters in more complicated figures aren’t well separated

Internal Index: Used to measure the goodness of a clustering structure without respect to external information
  - SSE

SSE is good for comparing two clusterings or two clusters (average SSE).

Can also be used to estimate the number of clusters.
INTERNAL MEASURES: SSE

SSE curve for a more complicated data set

SSE of clusters found using K-means
INTERNAL MEASURES: COHESION AND SEPARATION

Cluster Cohesion: Measures how closely related are objects in a cluster
  • Example: SSE

Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters

Example: Squared Error
  • Cohesion is measured by the within cluster sum of squares (SSE)
    \[ WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2 \]
  • Separation is measured by the between cluster sum of squares
    \[ BSS = \sum_{i} |C_i| (m - m_i)^2 \]
  • Where \(|C_i|\) is the size of cluster \(i\)
Example: SSE

- BSS + WSS = constant

K=1 cluster:

\[ WSS = (1 - 3)^2 + (2 - 3)^2 + (4 - 3)^2 + (5 - 3)^2 = 10 \]
\[ BSS = 4 \times (3 - 3)^2 = 0 \]
\[ Total = 10 + 0 = 10 \]

K=2 clusters:

\[ WSS = (1 - 1.5)^2 + (2 - 1.5)^2 + (4 - 4.5)^2 + (5 - 4.5)^2 = 1 \]
\[ BSS = 2 \times (3 - 1.5)^2 + 2 \times (4.5 - 3)^2 = 9 \]
\[ Total = 1 + 9 = 10 \]
A proximity graph based approach can also be used for cohesion and separation.

- Cluster cohesion is the sum of the weight of all links within a cluster.
- Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.
\[ \text{BetaCV} = \frac{d_{\text{intra}}}{d_{\text{inter}}} \]

\[ d_{\text{intra}} = \text{avg}d(i,j)|C_i = C_j \]

\[ d_{\text{inter}} = \text{avg}d(i,j)|C_i \neq C_j \]
INTERNAL MEASURES: SILHOUETTE COEFFICIENT

Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings.

For an individual point, $i$

- Calculate $a =$ average distance of $i$ to the points in its cluster
- Calculate $b =$ min (average distance of $i$ to points in another cluster)
- The silhouette coefficient for a point is then given by

$$s = 1 - \frac{a}{b} \quad \text{if } a < b,$$

(or $s = \frac{b}{a} - 1 \quad \text{if } a \geq b$, not the usual case)

- Typically between 0 and 1.
- The closer to 1 the better.

Can calculate the Average Silhouette width for a cluster or a clustering.
**EXTERNAL MEASURES OF CLUSTER VALIDITY: ENTROPY AND PURITY**

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Entertainment</th>
<th>Financial</th>
<th>Foreign</th>
<th>Metro</th>
<th>National</th>
<th>Sports</th>
<th>Entropy</th>
<th>Purity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>5</td>
<td>40</td>
<td>506</td>
<td>96</td>
<td>27</td>
<td>1.2270</td>
<td>0.7474</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>7</td>
<td>280</td>
<td>29</td>
<td>39</td>
<td>2</td>
<td>1.1472</td>
<td>0.7756</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>4</td>
<td>671</td>
<td>0.1813</td>
<td>0.9796</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>162</td>
<td>3</td>
<td>119</td>
<td>73</td>
<td>2</td>
<td>1.7487</td>
<td>0.4390</td>
</tr>
<tr>
<td>5</td>
<td>331</td>
<td>22</td>
<td>5</td>
<td>70</td>
<td>13</td>
<td>23</td>
<td>1.3976</td>
<td>0.7134</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>358</td>
<td>12</td>
<td>212</td>
<td>48</td>
<td>13</td>
<td>1.5523</td>
<td>0.5525</td>
</tr>
<tr>
<td>Total</td>
<td>354</td>
<td>555</td>
<td>341</td>
<td>943</td>
<td>273</td>
<td>738</td>
<td>1.1450</td>
<td>0.7203</td>
</tr>
</tbody>
</table>

**Entropy** For each cluster, the class distribution of the data is calculated first, i.e., for cluster $j$ we compute $p_{ij}$, the ‘probability’ that a member of cluster $j$ belongs to class $i$ as follows: $p_{ij} = m_{ij}/m_j$, where $m_j$ is the number of values in cluster $j$ and $m_{ij}$ is the number of values of class $i$ in cluster $j$. Then using this class distribution, the entropy of each cluster $j$ is calculated using the standard formula $e_j = \sum_{i=1}^{L} p_{ij} \log_2 p_{ij}$, where the $L$ is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster, i.e., $e = \sum_{i=1}^{K} \frac{m_j}{m} e_j$, where $m_j$ is the size of cluster $j$, $K$ is the number of clusters, and $m$ is the total number of data points.

**Purity** Using the terminology derived for entropy, the purity of cluster $j$, is given by $\text{purity}_j = \max p_{ij}$ and the overall purity of a clustering by $\text{purity} = \sum_{i=1}^{K} \frac{m_j}{m} \text{purity}_j$. 
FINAL COMMENT ON CLUSTER VALIDITY

“The validation of clustering structures is the most difficult and frustrating part of cluster analysis.

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.”

*Algorithms for Clustering Data*, Jain and Dubes
Example

- Compare SSE of 0.005 against three clusters in random data
- Histogram shows SSE of three clusters in 500 sets of random data points of size 100 distributed over the range 0.2 – 0.8 for x and y values
STATISTICAL FRAMEWORK FOR CORRELATION

Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets.

Corr = -0.9235

Corr = -0.5810
REFERENCES


- Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar. *Introduction to Data Mining (Second Edition)*
  https://www-users.cs.umn.edu/~kumar001/dmbook/index.php

- Dr Eamonn Keogh from University of California, Riverside
  www.cs.ucr.edu/~eamonn/teaching/cs235/clustering.ppt

- Dr Ruoming Jin from Kent State University
  www.cs.kent.edu/~jin/DM11/ClusterValidation.ppt