Unstructured Data Analysis

Lecture 7: Distance and similarity functions, clustering

George Chen
Last time: 2D t-SNE plot of handwritten digit images shows clumps that correspond to real digits — this is an example of clustering structure showing up in real data.
Let’s look at a structured dataset (easier to explain clustering): drug consumption data
Drug Consumption Data
Clustering Shows Up Often in Real Data!

- Example: crime might happen more often in specific hot spots
- Example: people applying for micro loans have a few specific uses in mind (education, electricity, healthcare, etc)
- Example: users in a recommendation system can share similar taste in products

To come up with clusters, we first need to define what it means for two things to be “similar”
The Art of Defining Similarity

- Popular: define a distance first and then turn it into a similarity

  **Example**: Euclidean distance \[ \|X_i - X_j\| \]

  Turn into similarity with decaying exponential \[ \exp(-\gamma \|X_i - X_j\|^2) \]
  where \( \gamma > 0 \)

- There is no “best” distance function to use

- Can also directly define similarity function

  **Example**: cosine similarity \[ \frac{\langle X_i, X_j \rangle}{\|X_i\|\|X_j\|} \]

There exist methods for automatically learning distance or similarity functions
Example: Time Series

How would you compute a distance between these?

Only observe time steps between 0 and $T$
Example: Time Series

How would you compute a distance between these?

$X_i$

$X_j$

Only observe time steps between 0 and $T$
Example: Time Series

How would you compute a distance between these?

$X_jX_i$

Distance could be defined as the area of this purple shaded in region

One solution: Align them first

In practice: for time series, very popular to use "dynamic time warping" (aligns two time series in a nonlinear manner)
Dynamic Time Warping aims to align time series into some common coordinate system. Then in the common coordinate system, can use usual distance functions like Euclidean, Manhattan, etc. “Aligning” data points is important in other problems too, not just for time series analysis.
Example: Spell Check

Distance between “apple” and “ap;ple”?  

One way to compute: find minimum number of single-letter insertions/deletions/substitutions to convert one to the other (called the Levenshtein distance)
Brain Image “Alignment”

FreeSurfer software: convert different people’s brain scans into spherical coordinates for comparison
Is a Distance/Similarity Function Any Good?

Easy thing to try:

• Pick a data point (for example, randomly)

• Compute its similarity to all the other data points, and sort them from most similar to least similar (or smallest distance to largest)

• Manually examine the most similar (closest) data points

If the most similar/closest points are not interpretable, it's quite likely that your distance/similarity function isn't very good  =(
Clustering methods aim to group together data points that are “similar” into “clusters”, while having different clusters be “dissimilar”

Clustering methods will either directly assume a specific choice of distance/similarity function, or some allow you to specify the distance/similarity
Going from Similarities to Clusters

There's a whole zoo of clustering methods

Several main categories (although there are other categories!):

**Generative models**

1. Pretend data generated by specific model with parameters
2. Learn the parameters ("fit model to data")
3. Use fitted model to determine cluster assignments

We mainly focus on this

**Hierarchical clustering**

Top-down: Start with everything in 1 cluster and decide on how to recursively split

Bottom-up: Start with everything in its own cluster and decide on how to iteratively merge clusters

**Density-based clustering**

Based on finding parts of the data with higher density
We're going to start with perhaps the most famous of clustering methods. It won't yet be apparent what this method has to do with generative models.
**k-means**

**Step 0: Pick** \( k \)

We’ll pick \( k = 2 \)

**Step 1: Pick guesses for where cluster centers are**

Example: choose \( k \) of the points uniformly at random to be initial guesses for cluster centers
$k$-means

Step 0: Pick $k$
We’ll pick $k = 2$

Step 1: Pick guesses for where cluster centers are
Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers
(There are many ways to make the initial guesses)
$k$-means

Step 0: Pick $k$

We’ll pick $k = 2$

Step 1: Pick guesses for where cluster centers are

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers
(There are many ways to make the initial guesses)

Step 2: Assign each point to belong to the closest cluster
**k-means**

**Step 0: Pick $k$**

We’ll pick $k = 2$

**Step 1: Pick guesses for where cluster centers are**

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

**Step 2: Assign each point to belong to the closest cluster**

**Step 3: Update cluster means (to be the center of mass per cluster)**
**k-means**

Step 0: Pick $k$

We’ll pick $k = 2$

Step 1: Pick guesses for where cluster centers are

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

Step 2: Assign each point to belong to the closest cluster

Step 3: Update cluster means (to be the center of mass per cluster)
**k-means**

Step 0: Pick $k$

We’ll pick $k = 2$

Step 1: Pick guesses for where cluster centers are

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

Repeat

Step 2: Assign each point to belong to the closest cluster

Step 3: Update cluster means (to be the center of mass per cluster)
**k-means**

**Step 0: Pick k**
We’ll pick $k = 2$

**Step 1: Pick guesses for where cluster centers are**
Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers
(There are many ways to make the initial guesses)

**Step 2: Assign each point to belong to the closest cluster**

**Repeat**

**Step 3: Update cluster means (to be the center of mass per cluster)**
**k-means**

**Step 0: Pick** $k$

We’ll pick $k = 2$

**Step 1: Pick guesses for where cluster centers are**

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

**Step 2: Assign each point to belong to the closest cluster**

**Repeat**

**Step 3: Update cluster means (to be the center of mass per cluster)**
**k-means**

**Step 0:** Pick $k$

We’ll pick $k = 2$

**Step 1:** Pick guesses for where cluster centers are

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

**Repeat**

**Step 2:** Assign each point to belong to the closest cluster

**Step 3:** Update cluster means (to be the center of mass per cluster)
**k-means**

**Step 0: Pick $k$**

We’ll pick $k = 2$

**Step 1: Pick guesses for where cluster centers are**

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

**Step 2: Assign each point to belong to the closest cluster**

**Repeat**

**Step 3: Update cluster means (to be the center of mass per cluster)**
Step 0: Pick $k$

We’ll pick $k = 2$

Step 1: Pick guesses for where cluster centers are

Example: choose $k$ of the points uniformly at random to be initial guesses for cluster centers

(There are many ways to make the initial guesses)

Repeat until convergence:

Step 2: Assign each point to belong to the closest cluster

Step 3: Update cluster means (to be the center of mass per cluster)
**$k$-means**

Final output: cluster centers, cluster assignment for every point

Remark: Very sensitive to choice of $k$ and initial cluster centers

How to pick $k$?

We’ll discuss this in more detail next lecture

Suggested way to pick initial cluster centers: “$k$-means++” method
(rough intuition: incrementally add centers; favor adding center far away from centers chosen so far)
When does $k$-means work well?

$k$-means is related to a more general model, which will help us understand $k$-means.
When does $k$-means work well?

$k$-means is related to a more general model, which will help us understand $k$-means.
Gaussian Mixture Model (GMM)

What random process could have generated these points?
Generative Process

Think of flipping a coin

each outcome: heads or tails

Each flip doesn't depend on any of the previous flips
Generative Process

Think of flipping a coin

each outcome: 2D point

Each flip doesn't depend on any of the previous flips

Okay, maybe it's bizarre to think of it as a coin…

If it helps, just think of it as you pushing a button and a random 2D point appears…
Gaussian Mixture Model (GMM)

We now discuss a way to generate points in this manner.
Gaussian Mixture Model (GMM)

Assume: points sampled independently from a probability distribution

Example of a 2D probability distribution

This is the sum of two 2D Gaussian distributions!

how probable point generated at \((x, y)\) is

Red = more likely
Blue = less likely

Example of a 2D probability distribution

Image source: https://www.intechopen.com/source/html/17742/media/image25.png
Quick Reminder: 1D Gaussian

This is a 1D Gaussian distribution

Image source: https://matthew-brett.github.io/teaching//smoothing_intro-3.hires.png
2D Gaussian

This is a 2D Gaussian distribution

Image source: https://i.stack.imgur.com/OIWce.png
Gaussian Mixture Model (GMM)

Assume: points sampled independently from a probability distribution

how probable point generated at \((x, y)\) is

This is the sum of two 2D Gaussian distributions!

Red = more likely
Blue = less likely

Example of a 2D probability distribution

Key idea: Each Gaussian corresponds to a different cluster

Image source: https://www.intechopen.com/source/html/17742/media/image25.png
Gaussian Mixture Model (GMM)

For a fixed value $k$ and dimension $d$, a GMM is the sum of $k$ $d$-dimensional Gaussian distributions so that the overall probability distribution looks like $k$ mountains.

- Each mountain corresponds to a different cluster.
- Different mountains can have different peak heights.
- One missing thing we haven't discussed yet: different mountains can have different shapes.

(We've been looking at $d = 2$.)
2D Gaussian Shape

In 1D, you can have a skinny Gaussian or a wide Gaussian

![Gaussians](https://www.cs.colorado.edu/~mozer/Teaching/syllabi/ProbabilisticModels2013/homework/assign5/a52dgauss.jpg)

Less uncertainty  More uncertainty

In 2D, you can more generally have ellipse-shaped Gaussians

Ellipse enables encoding relationship between variables

Can't have arbitrary shapes

Top-down view of an example 2D Gaussian distribution

Image source: https://www.cs.colorado.edu/~mozer/Teaching/syllabi/ProbabilisticModels2013/homework/assign5/a52dgauss.jpg
Gaussian Mixture Model (GMM)

• For a fixed value $k$ and dimension $d$, a GMM is the sum of $k$ $d$-dimensional Gaussian distributions so that the overall probability distribution looks like $k$ mountains (We've been looking at $d = 2$)
  
  • Each mountain corresponds to a different cluster
  
  • Different mountains can have different peak heights
  
  • Different mountains can have different ellipse shapes (captures "covariance" information)
Example: 1D GMM with 2 Clusters

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability of generating a point from cluster 1 = 0.5</td>
<td>Probability of generating a point from cluster 2 = 0.5</td>
</tr>
<tr>
<td>Gaussian mean = −5</td>
<td>Gaussian mean = 5</td>
</tr>
<tr>
<td>Gaussian std dev = 1</td>
<td>Gaussian std dev = 1</td>
</tr>
</tbody>
</table>

What do you think this looks like?
Example: 1D GMM with 2 Clusters

Cluster 1

Probability of generating a point from cluster 1 = 0.5
Gaussian mean = −5
Gaussian std dev = 1

Cluster 2

Probability of generating a point from cluster 2 = 0.5
Gaussian mean = 5
Gaussian std dev = 1
## Example: 1D GMM with 2 Clusters

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability of generating a point from cluster 1 = <strong>0.7</strong></td>
<td>Probability of generating a point from cluster 2 = <strong>0.3</strong></td>
</tr>
<tr>
<td>Gaussian mean = −5</td>
<td>Gaussian mean = 5</td>
</tr>
<tr>
<td>Gaussian std dev = 1</td>
<td>Gaussian std dev = 1</td>
</tr>
</tbody>
</table>

What do you think this looks like?
Example: 1D GMM with 2 Clusters

Cluster 1

- Probability of generating a point from cluster 1 = 0.7
- Gaussian mean = -5
- Gaussian std dev = 1

Cluster 2

- Probability of generating a point from cluster 2 = 0.3
- Gaussian mean = 5
- Gaussian std dev = 1
Example: 1D GMM with 2 Clusters

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability of generating a point from cluster 1 = 0.7</td>
<td>Probability of generating a point from cluster 2 = 0.3</td>
</tr>
<tr>
<td>Gaussian mean = −5</td>
<td>Gaussian mean = 5</td>
</tr>
<tr>
<td>Gaussian std dev = 1</td>
<td>Gaussian std dev = 1</td>
</tr>
</tbody>
</table>

How to generate 1D points from this GMM:
1. Flip biased coin (with probability of heads 0.7)
2. If heads: sample 1 point from Gaussian mean -5, std dev 1
   If tails: sample 1 point from Gaussian mean 5, std dev 1
### Example: 1D GMM with 2 Clusters

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability of generating a point from cluster 1 = $\pi_1$</td>
<td>Probability of generating a point from cluster 2 = $\pi_2$</td>
</tr>
<tr>
<td>Gaussian mean = $\mu_1$</td>
<td>Gaussian mean = $\mu_2$</td>
</tr>
<tr>
<td>Gaussian std dev = $\sigma_1$</td>
<td>Gaussian std dev = $\sigma_2$</td>
</tr>
</tbody>
</table>

How to generate 1D points from this GMM:

1. Flip biased coin (with probability of heads $\pi_1$)
2. If heads: sample 1 point from Gaussian mean $\mu_1$, std dev $\sigma_1$
   
   If tails: sample 1 point from Gaussian mean $\mu_2$, std dev $\sigma_2$
Example: 1D GMM with $k$ Clusters

Cluster 1

Probability of generating a point from cluster 1 = $\pi_1$
Gaussian mean = $\mu_1$
Gaussian std dev = $\sigma_1$

Cluster $k$

Probability of generating a point from cluster $k = \pi_k$
Gaussian mean = $\mu_k$
Gaussian std dev = $\sigma_k$

How to generate 1D points from this GMM:

1. Flip biased $k$-sided coin (the sides have probabilities $\pi_1, \ldots, \pi_k$)
2. Let $Z$ be the side that we got (it is some value 1, \ldots, $k$)
3. Sample 1 point from Gaussian mean $\mu_Z$, std dev $\sigma_Z$
Example: 2D GMM with $k$ Clusters

Cluster 1

- Probability of generating a point from cluster 1 = $\pi_1$
- Gaussian mean = $\mu_1$ 2D point
- Gaussian covariance = $\Sigma_1$ 2x2 matrix

Cluster $k$

- Probability of generating a point from cluster $k$ = $\pi_k$
- Gaussian mean = $\mu_k$ 2D point
- Gaussian covariance = $\Sigma_k$ 2x2 matrix

How to generate 2D points from this GMM:

1. Flip biased $k$-sided coin (the sides have probabilities $\pi_1$, $\ldots$, $\pi_k$)
2. Let $Z$ be the side that we got (it is some value $1$, $\ldots$, $k$)
3. Sample 1 point from Gaussian mean $\mu_Z$, covariance $\Sigma_Z$
GMM with $k$ Clusters

Cluster 1

Probability of generating a point from cluster 1 = $\pi_1$
Gaussian mean = $\mu_1$
Gaussian covariance = $\Sigma_1$

Cluster $k$

Probability of generating a point from cluster $k = \pi_k$
Gaussian mean = $\mu_k$
Gaussian covariance = $\Sigma_k$

How to generate points from this GMM:
1. Flip biased $k$-sided coin (the sides have probabilities $\pi_1, \ldots, \pi_k$)
2. Let $Z$ be the side that we got (it is some value 1, ..., $k$)
3. Sample 1 point from Gaussian mean $\mu_Z$, covariance $\Sigma_Z$
High-Level Idea of GMM

- Generative model that gives a *hypothesized* way in which data points are generated

  In reality, data are unlikely generated the same way!

  In reality, data points might not even be independent!
“All models are wrong, but some are useful.”

–George Edward Pelham Box

Photo: “George Edward Pelham Box, Professor Emeritus of Statistics, University of Wisconsin-Madison” by DavidMCEddy is licensed under CC BY-SA 3.0
High-Level Idea of GMM

• Generative model that gives a hypothesized way in which data points are generated

  In reality, data are unlikely generated the same way!

  In reality, data points might not even be independent!

• Learning ("fitting") the parameters of a GMM
  • Input: $d$-dimensional data points, your guess for $k$
  • Output: $\pi_1, \ldots, \pi_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k$

• After learning a GMM:
  • For any $d$-dimensional data point, can figure out probability of it belonging to each of the clusters

  How do you turn this into a cluster assignment?