94-775 Lecture 7: Interpreting Clusters, Gaussian Mixture Models, Automatically Choosing $k$

George Chen
A Sketch of How to Interpret Clusters

Demo
Gaussian Mixture Model (GMM)

GMM: assume these points generated in a particular way
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!

Red = more likely
Blue = less likely

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

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/OLWce.png
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!

Key idea: Each Gaussian corresponds to a different cluster

2D Gaussian distribution

2D Gaussian distribution

Example of a 2D probability distribution

Image source: https://www.intechopen.com/source/html/17742/media/image25.png
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 the probability distribution 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 = 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>

What do you think the probability distribution 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

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

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

Cluster 1

Probability of generating a point from cluster 1 = $\pi_1$

Gaussian mean = $\mu_1$

Gaussian std dev = $\sigma_1$

Cluster 2

Probability of generating a point from cluster 2 = $\pi_2$

Gaussian mean = $\mu_2$

Gaussian std dev = $\sigma_2$

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

<table>
<thead>
<tr>
<th>Cluster 1</th>
<th>Cluster $k$</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 $k = \pi_k$</td>
</tr>
<tr>
<td>Gaussian mean = $\mu_1$</td>
<td>Gaussian mean = $\mu_k$</td>
</tr>
<tr>
<td>Gaussian std dev = $\sigma_1$</td>
<td>Gaussian std dev = $\sigma_k$</td>
</tr>
</tbody>
</table>

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

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

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
GMM with $k$ Clusters

Cluster 1

Probability of generating a point from cluster 1 = $\pi_1$

Gaussian mean = $\mu_1$ in $\mathbb{R}^d$

Gaussian covariance = $\Sigma_1$

$d \times d$ matrix

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, $\ldots$, $k$)
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?
**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 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**

Step 0: Pick \( k \)

Step 1: Pick guesses for where cluster centers are

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)
(Rough Intuition) Learning a GMM

Step 0: Pick $k$

Step 1: Pick guesses for cluster means and covariances

Repeat until convergence:

Step 2: Assign each point a probability to belonging to each of the $k$ clusters

Step 3: Update cluster means and covariances carefully accounting for probabilities of each point belonging to each of the clusters

This algorithm is called the Expectation-Maximization (EM) algorithm specifically for GMM's (and approximately does maximum likelihood)

(Note: EM by itself is a general algorithm not just for GMM's)
Relating $k$-means to GMM's

$k$-means approximates the EM algorithm for GMM’s:

- $k$-means does "hard" assignment of each point to a cluster, whereas EM does a "soft" (probabilistic) assignment

- $k$-means does not keep track of shape/correlation information between variables (so shape is circular)

**Interpretation:** We know when $k$-means should work! It should work when the data appear as if they're from a GMM with true clusters that "look like circles"
$k$-means should do well on this
But not on this
Automatically Choosing $k$

For $k = 2, 3, \ldots$ up to some user-specified max value:

- Fit model using $k$
- Compute a score for the model
  
  But what score function should we use?

- Use whichever $k$ has the best score

There are fancier ways for choosing $k$ (e.g., DP-GMMs)

No single way of choosing $k$ is the “best” way
Here’s an example of a score function you don’t want to use

But hey it’s worth a shot
Residual Sum of Squares

Look at one cluster at a time

Cluster 1

Cluster 2
Residual Sum of Squares

Look at one cluster at a time

Cluster 1

Cluster 2
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Cluster 1

Cluster 2
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Cluster 2

Cluster 1
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Cluster 1

Cluster 2
Cluster 2

Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Cluster 1

Cluster 2

Residual sum of squares for cluster 1: sum of squared purple lengths
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Cluster 1

Cluster 2

Residual sum of squares for cluster 1:

\[ \text{RSS}_1 = \sum_{x \in \text{cluster 1}} \| x - \mu_1 \|^2 \]
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center

Repeat similar calculation for other cluster

Residual sum of squares for cluster 2:

$$\text{RSS}_2 = \sum_{x \in \text{cluster 2}} \| x - \mu_2 \|^2$$
Residual Sum of Squares

\[ \text{RSS} = \text{RSS}_1 + \text{RSS}_2 = \sum_{x \in \text{cluster 1}} \| x - \mu_1 \|^2 + \sum_{x \in \text{cluster 2}} \| x - \mu_2 \|^2 \]

In general if there are \( k \) clusters:

\[ \text{RSS} = \sum_{g=1}^{k} \text{RSS}_g = \sum_{g=1}^{k} \sum_{x \in \text{cluster } g} \| x - \mu_g \|^2 \]

Remark: \( k \)-means \( \text{tries} \) to minimize RSS (it does so \textit{approximately}, with no guarantee of optimality)

RSS only really makes sense for clusters that look like circles
Why is minimizing RSS a bad way to choose \( k \)?

What happens when \( k \) is equal to the number of data points?
A Good Way to Choose $k$

RSS measures *within-cluster variation*

$$W = \text{RSS} = \sum_{g=1}^{k} \text{RSS}_g = \sum_{g=1}^{k} \sum_{x \in \text{cluster } g} \| x - \mu_g \|^2$$

Want to also measure *between-cluster variation*

$$B = \sum_{g=1}^{k} (\text{# points in cluster } g) \| \mu_g - \mu \|^2$$

Called the **CH index** [Calinski and Harabasz 1974]

A good score function to use for choosing $k$:

$$\text{CH}(k) = \frac{B \cdot (n - k)}{W \cdot (k - 1)}$$

$n = \text{total # points}$

Pick $k$ with highest $\text{CH}(k)$

(Choose $k$ among 2, 3, ... up to pre-specified max)
Automatically Choosing $k$

Demo