Chapter 2: Finite Mixture Model

Nguyen Manh Khiem

2023-10-17

2 Finite Mixture Model

Tài liệu chương 2: https://math.univ-cotedazur.fr/~cbouveyr/MBCbook/

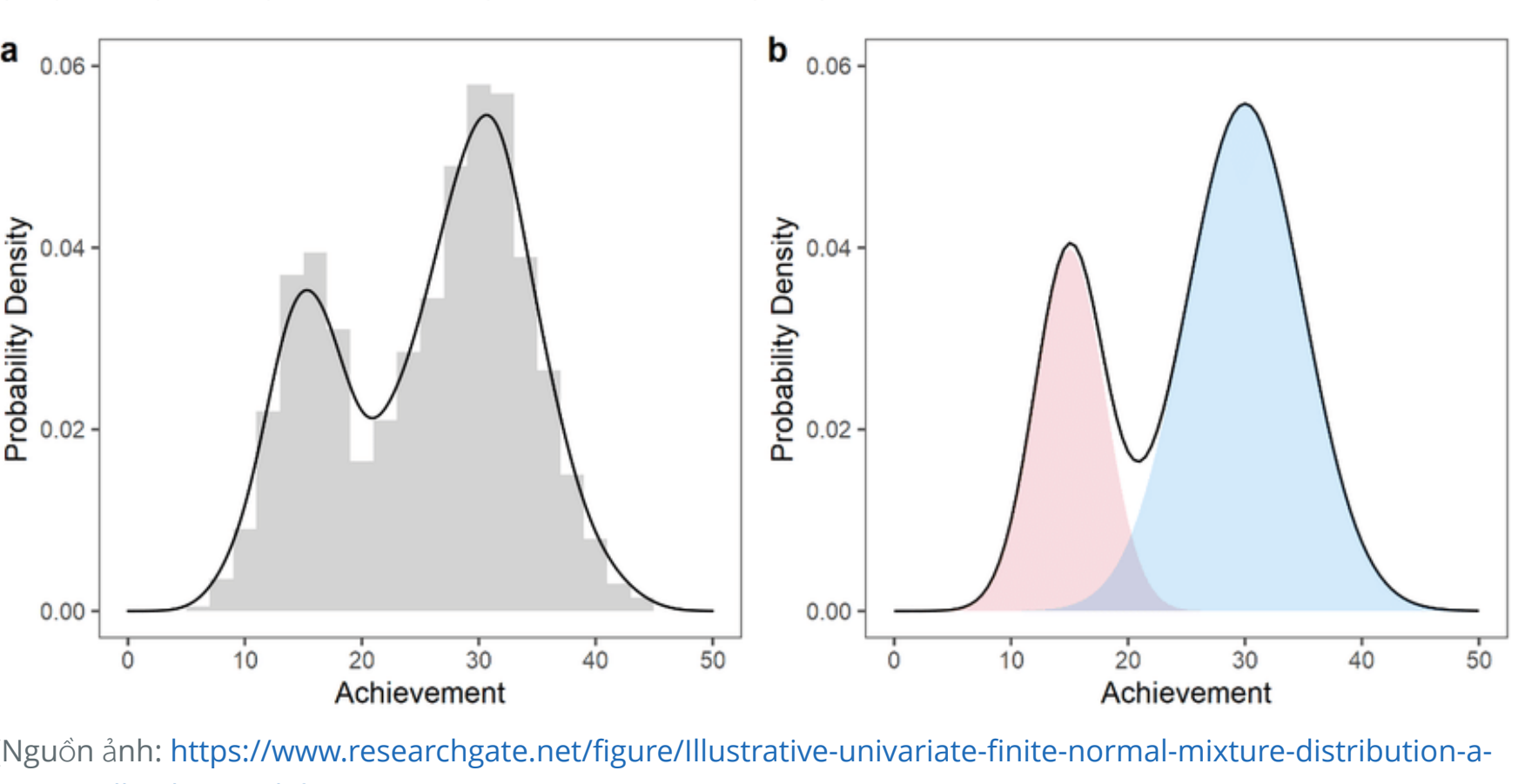
Đôi lời về cuốn sách:

- Các công thức khá hàn lâm, yêu cầu bạn phải học qua 1 khóa về Non-parameter estimate để hiểu lý thuyết. Một số thuật toán không được dạy ở cấp bậc Đại học như: EM Algorithm, K-mean Clustering (thuộc về Machine Learning)
- Vài chỗ giải thích khá khó hiểu, cần phải chấp nhận, đọc tiếp về sau mới hiểu ngược lại bên trên.
- Cần quen thuộc với code R, nắm được ý tưởng bức tranh toàn cảnh, đôi khi không cần hiểu hết các tham số. Đòi hỏi người đọc phải tự mày mò từng bước để hiểu hết ý tưởng.
- Tuy nhiên nếu đã có kiến thức nền tảng thì cuốn sách này liên kết các công thức với ý nghĩa thực tế, nhiều hình ảnh trực quan (nhưng đòi hỏi suy ngẫm).

Kết luận: Không dành cho người mới bắt đầu, chuyên ngành kinh tế không có background về toán, thống kê, code.

Page 15 - Introduction

Khi vẽ hàm mật độ hay histogram của một bộ dữ liệu, nếu nó có đa đỉnh thì theo lý thuyết, có vẻ cách thu thập dữ liệu của bạn bị sai khi có quá nhiều đối tượng bị trộn lẫn với nhau. Lúc đó cách tốt nhất là nên thu thập lại, tách riêng từng object và thực hiện các phương pháp thống kê cơ bản như bình thường. Nhưng nếu điều đó quá phức tạp hoặc quá tốn kém chi phí, thì bắt buộc ta phải phân tích dựa trên bộ dữ liệu đó. (Hình a)



(Nguồn ảnh: https://www.researchgate.net/figure/Illustrative-univariate-finite-normal-mixture-distribution-a-empirically-observed_fig1_353424179)

Finite Mixture Model là ta giả sử đã biết trong mixture dataset đó có bao nhiêu object (hay từ giờ sẽ gọi là component). ví dụ: bài toán về thu nhập người dân thì sẽ là người có thu nhập cao/trung bình/thấp. Nếu ta chưa biết trước được số lượng cluster (nói cách khác là chưa có domain knowledge về vấn đề đang làm), cách dễ nhất là nhìn vào hình dạng Histogram, có bao nhiêu đỉnh thì bấy nhiêu cluster. Mở rộng hơn sẽ có một số phương pháp giúp lựa chọn số cluster bằng tay hoặc tự động.

Mỗi component trong mixture dataset như vậy lại tuân theo một phân phối xác suất. Trong khuôn khổ của cuốn sách này và hầu hết trong thực tế khi bạn có một bộ dữ liệu đủ lớn, tất cả component đều tuân theo phân phối chuẩn *(định lý giới hạn trung tâm - central limit theorem)*. (Hình b)

2.1 Finite Mixture Model

Giả sử chúng ta có

- n quan trắc (n hàng), $y_1, \ldots, y_n$
- Mỗi hàng sẽ có d chiều (d cột), : $y_i = (y_{i,1}, \ldots, y_{i,d})$.

A **finite mixture model** được định nghĩa là:

- Hàm phân phối xác suất / Hàm mật độ ước lượng p của multivariate observation (quan trắc đa biến) y_i,
- được tạo từ trung bình trọng số (weighted average) τ của G hàm ước lượng xác suất thành phần (density mixture component) f_g.

$$p(y_i) = \sum_{g=1}^{G} \tau_g f_g (y_i \mid \theta_g) . \tag{2.1}$$

Page 16 - Giải thích công thức Finite Mixture Model

Trong đó:

- τ_g là xác suất observation thuộc về component thứ g
- $\tau_g \geq 0$
- $g = 1, \ldots, G,$
- $\sum_{g=1}^{G} \tau_g = 1,$
- $f_g(\cdot \mid \theta_g)$ là hàm density của component thứ g với tham số là θ_g.

Vector trọng số τ ở đây nhằm mục đích: giá trị nằm ở phần overlap giữa 2 phân phối, trọng số của phân phối nào cao hơn sẽ có khả năng thuộc về phân phối đó lớn hơn.

Tham số θ ở đây là tham số của phân phối. Ví dụ $f(\ . \mid \theta \ )$ có phân phối chuẩn thì sẽ có 2 tham số trung bình và phương sai $\mathcal{N}(\mu, \ \sigma^2)$

Dataset faithful and mclust pakage:

Mạch nước Old Faithful ở Công viên Quốc gia Yellowstone, Wyoming phun trào cứ sau 35–120 phút trong khoảng một đến năm phút.

Sẽ rất hữu ích cho các kiểm lâm viên khi có thể dự đoán thời gian xảy ra vụ phun trào tiếp theo.

Thời gian xảy ra vụ phun trào tiếp theo và thời gian kéo dài mỗi lần phun của nó có liên quan với nhau, theo đó, vụ phun trào kéo dài càng lâu thì thời gian cho đến lần phun trào tiếp theo càng lâu.

Bộ dữ liệu được xem xét trong cuốn sách này bao gồm các quan sát về 272 vụ phun trào Azzhini và Bowman (1990).

Dữ liệu về hai biến số được đo lường:

- thời gian từ lần phun trào này đến lần phun trào tiếp theo
- thời gian phun trào.

Bộ dataset này được cài đặt sẵn trong R.

```
waiting = faithful$waiting
n = length(waiting)
```

Thư viện **mclust** là thư viện được dùng để cluster cho những object tuân theo phân phối chuẩn. Tuy nhiên hiện tại ta tạm thời bỏ qua việc nó có thể làm gì, ta chỉ mượn nó để tạo ra một bộ dataset fake tổ hợp của 2 object có phân phối chuẩn.

Page 17 - Fake dataset for examples:

```
options(warn=-1)
library(mclust)
```

```
## Package 'mclust' version 6.0.1
## Type 'citation("mclust")' for citing this R package in publications.
```

```
waiting.Mclust <- Mclust(waiting, model = "V", G = 2)
```

Để vẽ đồ thị, ta cần tạo tọa độ x theo giá trị của waiting:

```
x = seq(from = min(waiting), to = max(waiting), length = 1000)
```

2 cluster, mỗi cluster có trung bình và phương sai tương ứng:

```
mean1 = waiting.Mclust$parameters$mean[1]
mean2 = waiting.Mclust$parameters$mean[2]

std1 = sqrt(waiting.Mclust$parameters$variance$sigmasq[1])
std2 = sqrt(waiting.Mclust$parameters$variance$sigmasq[2])

print(mean1)
```

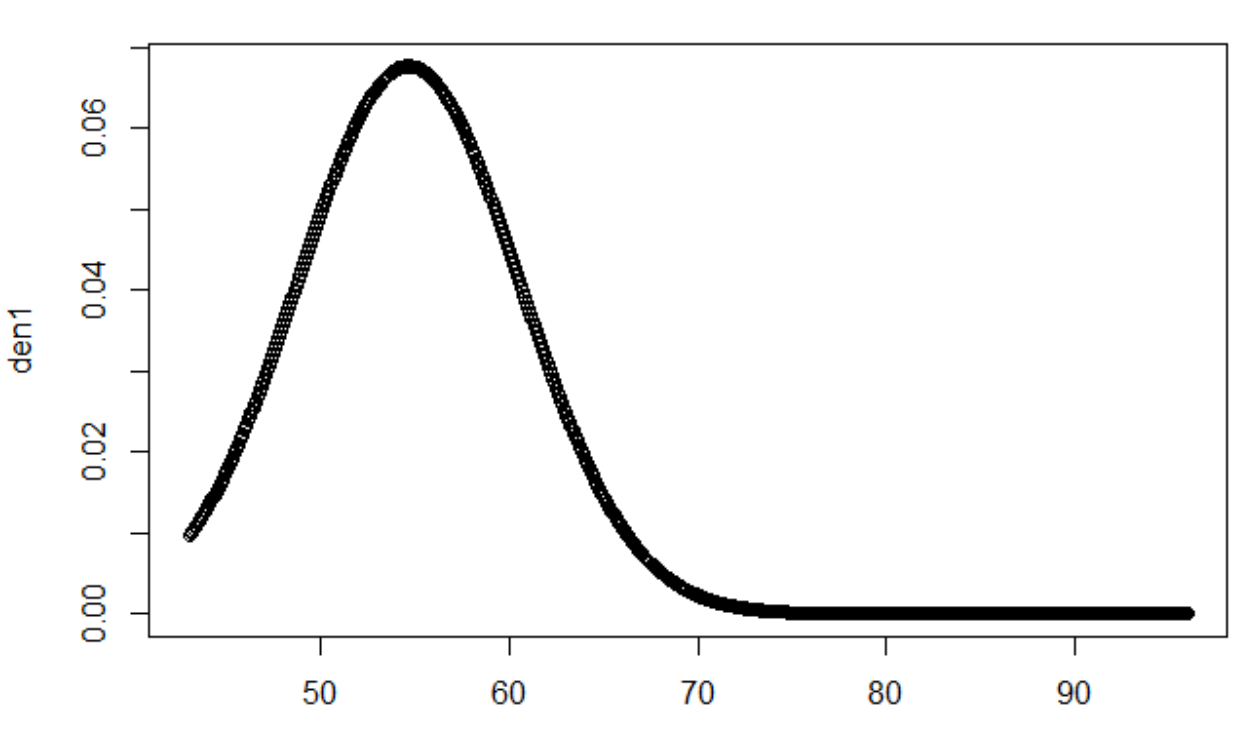
```
##        1
## 54.64666
```

```
print(mean2)
```

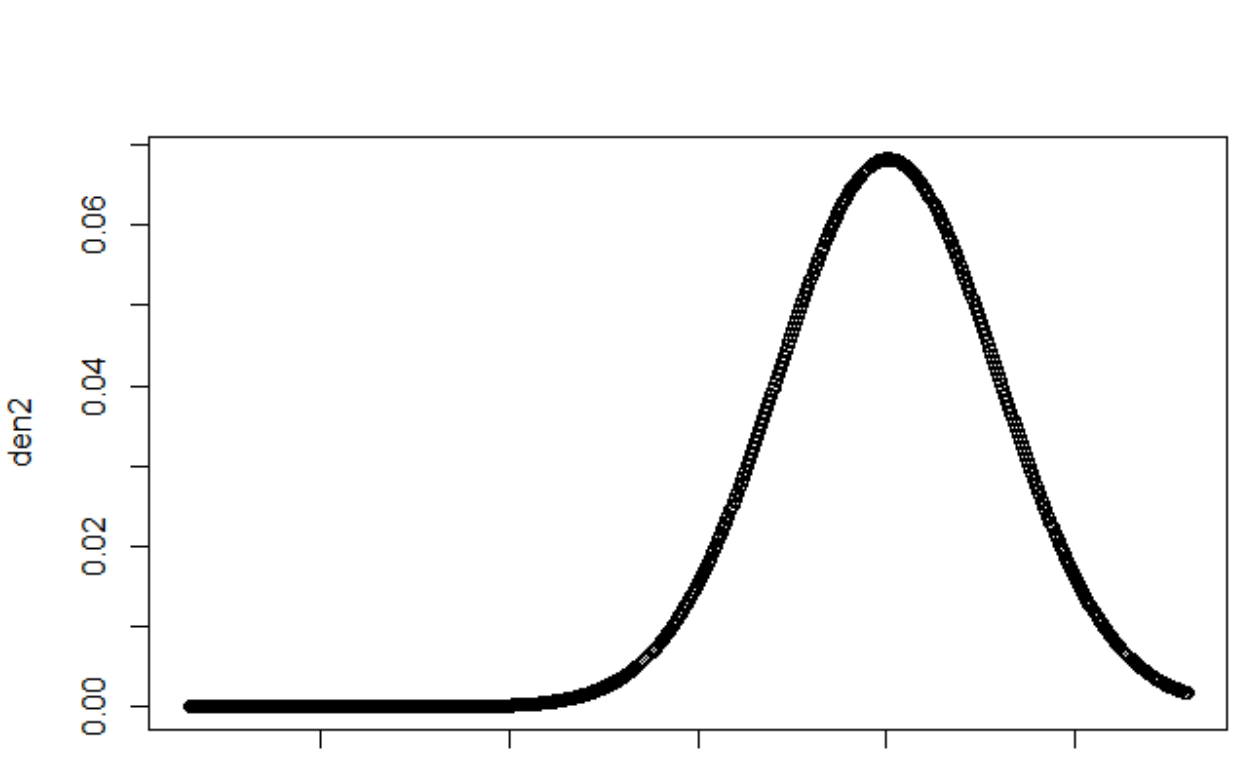
```
##        2
## 80.11096
```

```
# Plot densities:
den1 = dnorm(x, mean = mean1, sd = std1)
den2 = dnorm(x, mean = mean2, sd = std2)

plot(x, den1)
```



```
plot(x, den2)
```



Ta mượn vector trọng số τ từ việc xấp xỉ của mclust:

```
tau1 = waiting.Mclust$parameters$pro[1]
tau2 = waiting.Mclust$parameters$pro[2]
print(tau1)
```

```
## [1] 0.3618359
```

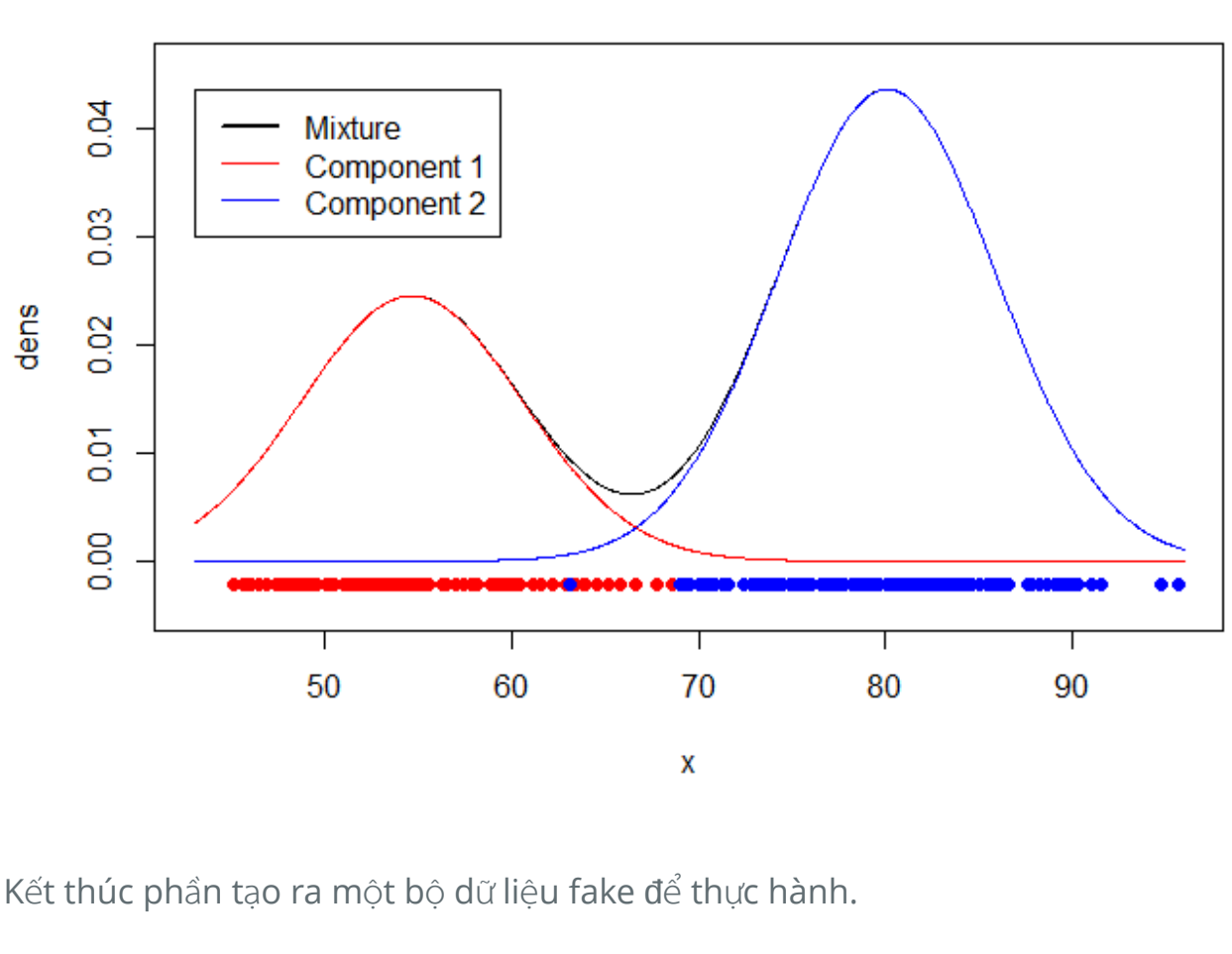
```
print(tau2)
```

```
## [1] 0.6381641
```

```
dens = tau1*den1 + tau2*den2
```

```
plot(x, dens, type = "l", ylim=c(-max(dens)/10,1.05*max(dens)))
lines (x, tau1*den1, col = "red")
lines (x, tau2*den2, col = "blue")
legend(x = min(x), y = max(dens), legend = c("Mixture", "Component 1", "Component 2"), col =

# Simulate and plot a sample from the distribution:
sim.results = simV(waiting.Mclust$parameters, n, seed = 0)
ysim = sim.results[, 2]
groupsim = sim.results[, "group"]
ysim1 = ysim[groupsim == 1]
ysim2 = ysim[groupsim == 2]
points(ysim1, rep(-max(dens)/20, length(ysim1)), col = "red", pch=19)
points(ysim2, rep(-max(dens)/20, length(ysim2)), col = "blue", pch=19)
```



Kết thúc phần tạo ra một bộ dữ liệu fake để thực hành.

Page 18 - Gaussian Mixture Function and Covariance matrix

Hàm mật độ xác suất trên có hai thành phần mix với nhau. Từng mật độ thành phần riêng lẻ nhân với trọng số của chúng lần lượt được hiển thị bằng màu đỏ và xanh lam, và mật độ hỗn hợp tổng thể thu được (tổng của các đường cong màu đỏ và xanh lam) là đường cong màu đen. Các dấu chấm hiển thị một mẫu có kích thước 272, với màu sắc biểu thị thành phần hỗn hợp mà chúng được khởi tạo.

- Một số điểm màu xanh lẫn vào những điểm màu đỏ.

→ không chắc nó thuộc về cluster nào.

- Có thể được giải quyết bằng model-based clustering.
- Khi bộ dữ liệu f_g đa biến thì nó thường sẽ có phân phối chuẩn hoặc Gaussian ϕ_g, tham số bởi vector trung bình μ_g và ma trận hiệp phương sai Σ_g.

Nghĩa là tăng lên phân phối chuẩn nhiều chiều, với trung bình của mỗi component giữ nguyên, còn phương sai đổi thành ma trận hiệp phương sai.

$$\phi_g(y_i \mid \mu_g, \Sigma_g) = \frac{1}{|2\pi\Sigma_g|} e^{-\frac{1}{2}(y_i-\mu_g)^T \Sigma_g^{-1}(y_i-\mu_g)} \tag{2.2}$$

Trước khi đi tiếp thì bạn cần xem qua khái niệm về **density contours**:

https://www.youtube.com/watch?v=a2t2EQd5SC4

- Khu vực có màu đậm hơn sẽ có mật độ các điểm dữ liệu dày hơn
- Các điểm nằm trên đường viền sẽ có cùng chiều cao (xác suất)

Figure 2.2 là biểu đồ Density contours cho 2 mixture components có phân phối chuẩn. Trong đó, các tham số của cluster như trung bình và phương sai được chỉnh sửa sao cho phù hợp với mục đích tác giả muốn đưa ra:

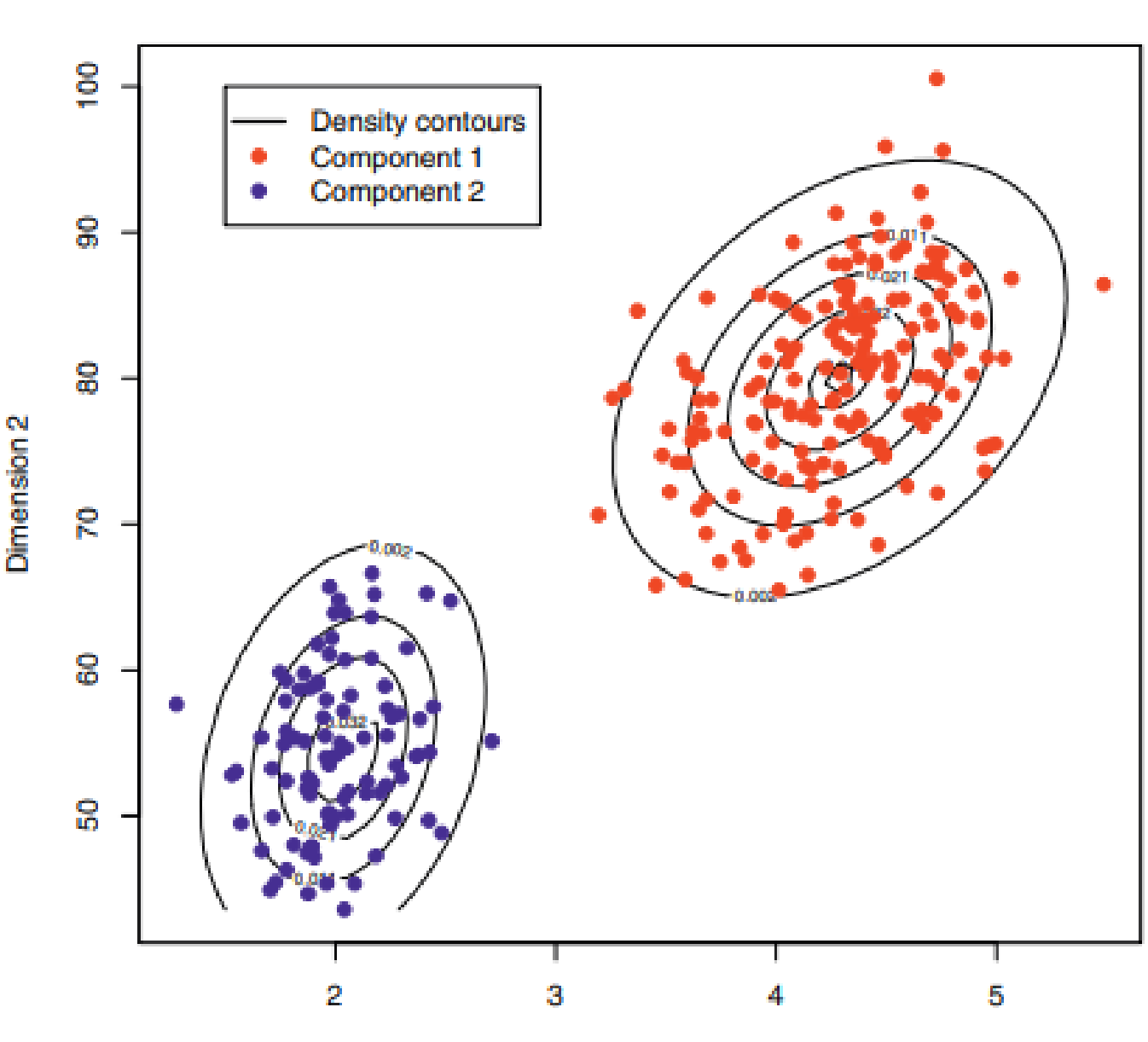


Figure 2.2 Contours of the density function for a two-dimensional finite bivariate normal mixture model with two mixture components (see text for parameter values). The dots show a sample of size 272 simulated from the density, with the colors indicating the mixture component from which they were generated.

- Trong trường hợp này, cả 2 cluster không bị overlap với nhau. 6 đường contours (từ trong ra ngoài) thể hiện cho 5%, 25%, 50%, 75%, 95% and 99% giá trị xác suất (% số lượng điểm dữ liệu nằm bên trong).

- Vậy một cái tips nhỏ thường thấy trong Machine Learning là đôi khi người ta sẽ chấp nhận tăng độ phức tạp để tăng số chiều lên, nhưng nó đem lại kết quả tốt hơn.

$$\mu_1 = (4.29, 79.97), \mu_2 = (2.04, 54.48), \tau_1 = 0.644, \tau_2 = 0.356,$$
$$\Sigma_1 = \begin{bmatrix} 0.170 & 0.938 \\ 0.938 & 36.017 \end{bmatrix}, \text{ and } \Sigma_2 = \begin{bmatrix} 0.069 & 0.437 \\ 0.437 & 33.708 \end{bmatrix}.$$

Nhìn vào các thông số này, ta có thể nhận xét như sau:

- Giá trị của cả 2 ma trận hiệp phương sai đều dương, cho thấy cả hai nhóm đều có mối quan hệ thuận giữa hai biến, nhưng mối quan hệ này mạnh hơn ở nhóm 1 (0.938 so với 0.437)
- Nhóm 1 có trọng số lớn hơn (0.644 so với 0.356), cho thấy dữ liệu có xu hướng thuộc về nhóm này hơn
- Trung bình của nhóm 1 cao hơn so với nhóm 2 trên cả hai chiều.
- Ma trận covariance cho thấy dữ liệu của cả hai nhóm đều phân tán nhiều hơn theo hướng thứ hai. Tuy nhiên, sự phân tán của nhóm 1 lớn hơn một chút so với nhóm 2 (36.017 so với 33.708)

2.2 Geometrically Constrained Multivariate Normal Mixture Models

Mỗi tham số của mô hình cần được ước lượng ta xem nó là một biến cần lưu. Công thức để tính số tham số cần ước lượng như sau:

$$(G-1) + Gd + G\{d(d+1)/2\}$$

Trong đó:

- $G-1$: Số lượng tham số của G-1 component, không cần tính của G cái vì ta có thể tự suy ra (giống với bậc tự do).
- Gd: mỗi component có d chiều
- $G\{d(d+1)/2\}$: ma trận hiệp phương sai $(d \times d)$ tham số, nhưng vì nó là ma trận đối xứng, nên chỉ cần (d(d + 1)/2) phần tử duy nhất.

Ví dụ:

- Data faithful: $d = \ 2, \ G = 2 \rightarrow$ 11 params.
- $d = 27, \ G = 3 \rightarrow$ 1,217 params.

Số lượng lớn params gây khó khăn trong việc tính toán lẫn biểu diễn kết quả. Chiếm phần lớn nhất đến từ covariance matrix.

Do vậy để khắc phục điều này, người ta dùng **eigenvalue decomposition** (hay Singular Value Decomposition) thay cho covariance matrix Σ_g

Page 20 - eigenvalue decomposition

$$\Sigma_g = \lambda_g D_g A_g D_g^T . \tag{2.3}$$

Trong đó: - λ_g: vector tỉ lệ thể tích (Volume).

- D_g: ma trận eigenvalue thể hiện cho hướng (Direction) phân tán của cluster.
- A_g: ma trận đường chéo eigenvalue. thể hiện cho độ lớn phân tán.

Nếu xét giữa các component khác nhau:

- Nếu λ_g là hằng số, nghĩa là thể tích của các component (cluster) bằng nhau.
- Nếu mọi component đều có hướng phân tán giống nhau ở mọi Direction $D_g = D$. Ta dùng từ Orientation để chỉ điều này.
- Nếu giá trị của mọi component đều bằng nhau $A_g = A$ thì nghĩa là mức độ phân tán của nó bằng nhau ở mọi hướng.

Nếu xét trong cùng 1 component, ta chỉ quan tâm đến A vì λ, D không thay đổi, nghĩa là thể tích và hướng giống nhau nếu xét trong cùng 1 component.

- Nếu $A_{1,g} > A_{2,g}$: nghĩa là mức độ phân tán theo direction 1 lớn hơn direction 2. Hay nói cách khác, các điểm dữ liệu trong cluster thứ g tập trung gần trục đường thẳng hướng thứ 2 hơn.
- Nếu $A_{1,g} \approx A_{2,g} > A_{3,g}$ thì component thứ g tập trung xung quanh hyperplane tạo bởi direction 1 và 2.

Tổng kết và Câu hỏi:

- EM clustering giống và khác với K-means ở điểm nào?
- Tại sao người ta lại dùng mixture component thay vì cluster?