Глава 13 Репрезентативная кластеризация

Для набора данных с n точками в d-мерном пространстве, $\mathbf{D} = \{\mathbf{x}_i\}_{i=1}^n$, и количеств желаемых кластеров k, цель репрезентативной кластеризации состоит в том, чтобы разделить набор данных на k групп или кластеров, обозначающихся как $C = \{C_1, C_2, \dots, C_k\}$. Далее, для каждого кластера C_i существует репрезентативная точка, которая суммирует кластер. Обычно выбирается среднее значение (также называемое центроидом) $\boldsymbol{\mu}_i$ всех точек в кластере, т. е.,

$$\boldsymbol{\mu}_i = \frac{1}{\boldsymbol{n}_i} \sum_{x_j \in C_i} \mathbf{x}_j ,$$

где $\boldsymbol{n}_i = |C_i|$ – количество точек в кластере C_i.

Полный перебор или исчерпывающий алгоритм поиска хорошей кластеризации состоит в том, чтобы просто сгенерировать все возможные разбиения n точек на k кластеров, произвести оценку оптимизации для каждого из них и сохранить кластеризацию, которая дает лучший результат. Точное количество способов разбить n точек на k непустых и непересекающихся частей можно получить с помощью чисел Стирлинга второго рода, заданных как:

$$S(n,k) = \frac{1}{k!}\sum_{t=0}^{k}(-1)^t \binom{k}{t}(k-t)^n$$

Неформально каждую точку можно назначить любому из k кластеров, поэтому существует не более k^n возможных кластеров. Однако любая перестановка k кластеров в рамках данной кластеризации дает эквивалентную кластеризацию; следовательно, имеется $O(k^n/k!)$ кластеризаций n точек в k групп. Понятно, что исчерпывающее перечисление и оценка всех возможных кластеров практически неосуществимы. В этой главе мы описываем два подхода к репрезентативной кластеризации, а именно алгоритм K-средних и EM-алгоритм.

13.1 Алгоритм K-средних

Для кластеризации $C = \{C_1, C_2, \dots, C_k\}$, нам нужна некоторая функция оценки, которая оценивает её качество. Эта функция оценки суммы квадратов ошибок определяется как:

$$SSE(C) = \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in C_i} \left\|\mathbf{x}_j - \boldsymbol{\mu}_i\right\|^2 \tag{13.1}$$

Цель состоит в том, чтобы найти кластеризацию, которая минимизирует оценку SSE:

$$C^* = \arg\min_{C}\{SSE(C)\}$$

K-средних использует жадный итерационный подход для поиска кластеризации, которая минимизирует цель SSE [уравнение. (13.1)]. Таким образом, он может сходиться к локальным оптимумам вместо глобально оптимальной кластеризации.

K-средних инициализирует средние кластеров путем случайной генерации k точек в пространстве данных. Обычно это делается путем генерации равномерной случайной величины в пределах диапазона для каждого измерения. Каждая итерация K-средних состоит из двух шагов: (1) назначение кластера и (2) обновление центроида. Для k средних кластера на этапе назначения кластера каждой точке $\mathbf{x}_j \in \mathbf{D}$ присваивается ближайшее среднее значение, что вызывает кластеризацию, причем каждый кластер C_i содержит точки, которые ближе к $\boldsymbol{\mu}_i$, чем любое другое среднее значение кластера. То есть каждая точка $\mathbf{x}_j$ относится к кластеру C_{j^*}, где

$$j^* = \arg \min_{i=0}^{k} \left\{ \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2 \right\} \tag{13.2}$$

Для набора кластеров $C = \{C_1, C_2, \ldots, C_k\}$, на этапе обновления центроида, новые средние значения вычисляются для каждого кластера из точек в C_i. Этапы назначения кластера и обновления центроида выполняются итеративно, пока мы не достигнем фиксированной точки или локальных минимумов. С практической точки зрения можно предположить, что K-средние сошлись, если центроиды не меняются от одной итерации к другой. Например, мы можем остановиться, если $\sum_{i=1}^{k} \left\| \boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1} \right\|^2 \leq \epsilon$, где $\epsilon > 0$ – порог сходимости, t обозначает текущую итерацию, и $\boldsymbol{\mu}_i^t$ – среднее для кластера C_i в итерации t.

Поскольку метод K-средних начинается со случайного предположения для начальных центроидов, K-средних обычно запускается несколько раз, и для отчета окончательной кластеризации выбирается запуск с наименьшим значением SSE. Также стоит отметить, что K-средних генерирует кластеры выпуклой формы, поскольку область в пространстве данных, соответствующем каждому кластеру, может быть получена как пересечение полупространств, возникающих из гиперплоскостей, которые делят пополам и перпендикулярны линейным сегментам, которые соединяют пары центроидов кластера.

С точки зрения вычислительной сложности K-средних, мы можем видеть, что этап назначения кластера занимает $O(nkd)$ времени, потому что для каждой из n точек мы должны вычислить расстояние до каждого из k кластеров, что требует d операций в d размерностях. Шаг повторного вычисления центроида занимает $O(nd)$ времени, потому что мы должны добавить в общей сложности n d-мерных точек. Предполагая, что существует t итераций, общее время для K-средних задается как $O(nkd)$. С точки зрения затрат ввода-вывода для этого требуется $O(t)$ полных сканирований базы данных, потому что мы должны читать всю базу данных на каждой итерации.

Пример 13.1. Рассмотрим одномерные данные, показанные на рисунке 13.1a. Предположим, что мы хотим кластеризовать данные в $k = 2$ кластера. Пусть начальные центроиды равны $\mu_1 = 2$ и $\mu_2 = 4$. На первой итерации мы сначала вычисляем кластеры, присваивая каждой точке ближайшее среднее значение, чтобы получить

$$C_1 = \{2,3\}, C_2 = \{4,10,11,12,20,25,30\}$$

затем мы обновляем средние следующим образом:

$$\mu_1 = \frac{2+3}{2} = \frac{5}{2} = 2.5$$

$$\mu_2 = \frac{4+10+11+12+20+25+30}{7} = \frac{112}{7} = 16$$

Новые центроиды и кластеры после первой итерации показаны на рисунке 13.1b. Для второго шага мы повторяем шаги назначения кластера и обновления центроида, как показано на рисунке 13.1c, чтобы получить новые кластеры:

$$C_1 = \{2,3,4\}, C_2 = \{10,11,12,20,25,30\}$$

и новые средние

$$\mu_1 = \frac{2+3+4}{3} = \frac{9}{3} = 3$$

$$\mu_2 = \frac{10+11+12+20+25+30}{6} = \frac{108}{6} = 18$$

Полный процесс до схождения показан на рисунке 13.1. Конечные кластеры представлены как

$$C_1 = \{2,3,4,11,12\}, C_2 = \{20,25,30\}$$

с представителями $\mu_1 = 7$ и $\mu_2 = 25$.

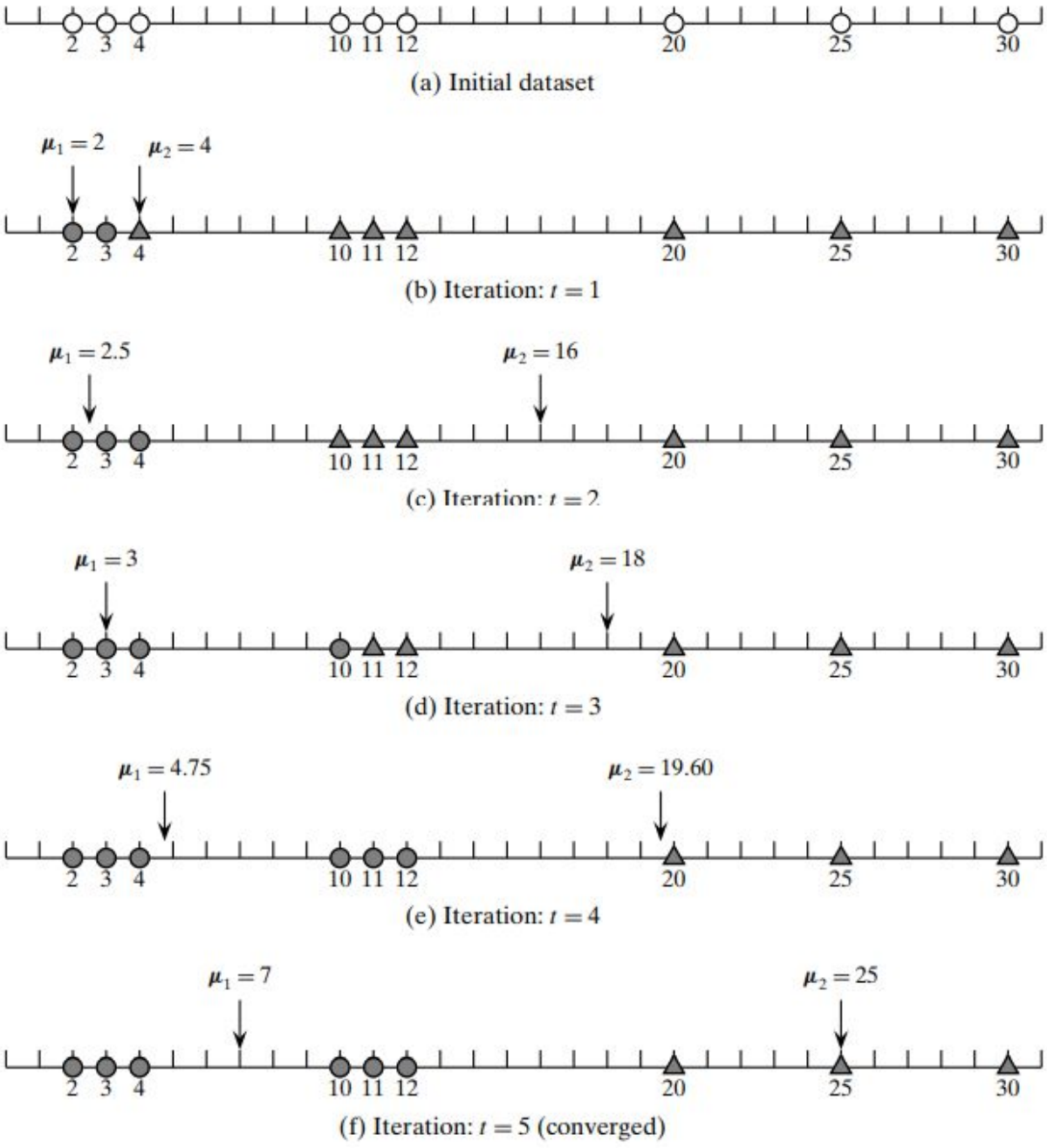

2 3 4 10 11 12 20 25 30
(a) Initial dataset
$\mu_1 = 2$
$\mu_2 = 4$
2 3 4 10 11 12 20 25 30
(b) Iteration: $t = 1$
$\mu_1 = 2.5$
$\mu_2 = 16$
2 3 4 10 11 12 20 25 30
(c) Iteration: $t = 2$
$\mu_1 = 3$
$\mu_2 = 18$
2 3 4 10 11 12 20 25 30
(d) Iteration: $t = 3$
$\mu_1 = 4.75$
$\mu_2 = 19.60$
2 3 4 10 11 12 20 25 30
(e) Iteration: $t = 4$
$\mu_1 = 7$
$\mu_2 = 25$
2 3 4 10 11 12 20 25 30
(f) Iteration: $t = 5$ (converged)

Рисунок 13.1

ALGORITHM 13.1. K-means Algorithm

K-MEANS $(\mathbf{D}, k, \epsilon)$:

$t = 0$

Randomly initialize k centroids: $\boldsymbol{\mu}_1^t, \boldsymbol{\mu}_2^t, \ldots, \boldsymbol{\mu}_k^t \in \mathbb{R}^d$

repeat

  $t \leftarrow t + 1$

  $C_j \leftarrow \emptyset$ for all $j = 1, \cdots, k$

  `// Cluster Assignment Step`

  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**

    $j^* \leftarrow \arg\min_i \left\{ \left\| \mathbf{x}_j - \boldsymbol{\mu}_i^{t-1} \right\|^2 \right\}$ `// Assign` $\mathbf{x}_j$ `to closest centroid`

    $C_{j^*} \leftarrow C_{j^*} \cup \{\mathbf{x}_j\}$

  `// Centroid Update Step`

  foreach $i = 1$ *to* k **do**

    $\boldsymbol{\mu}_i^t \leftarrow \frac{1}{|C_i|} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j$

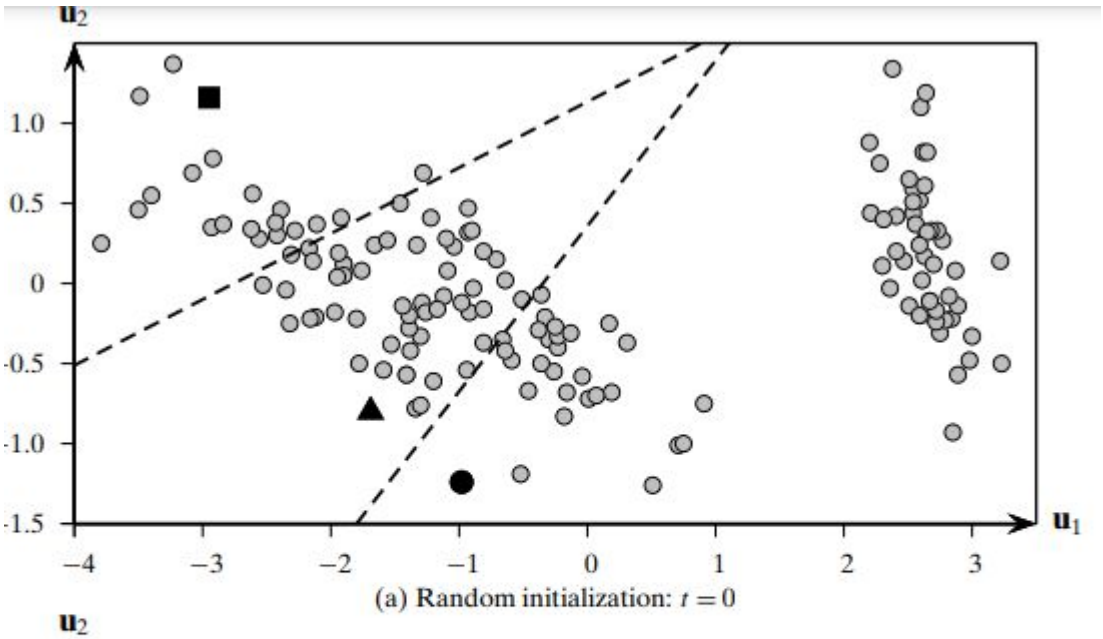
until $\sum_{i=1}^{k} \left\| \boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1} \right\|^2 \leq \epsilon$

Пример 13.2(K-средних в двух измерениях). На рисунке 13.2 мы проиллюстрировали алгоритм K-средних для набора данных Iris, используя первые два основных компонента в качестве двух измерений. Ирис имеет $n = 150$ точек, и мы хотим найти $k = 3$ кластера, соответствующих трем типам ирисов. Случайная инициализация кластера дает

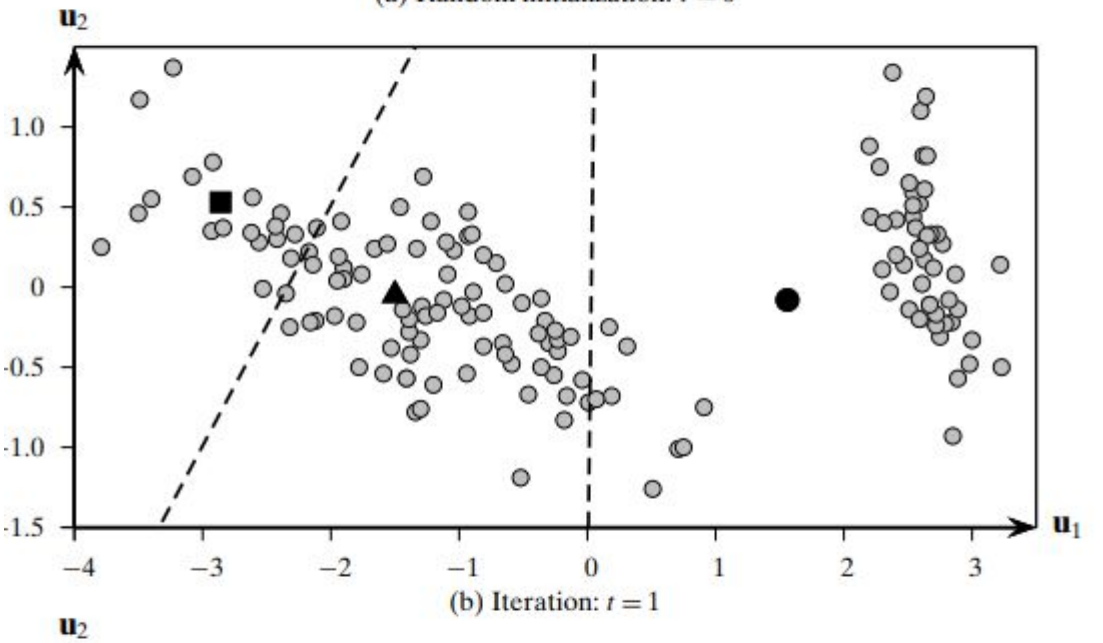
$$\boldsymbol{\mu}_1 = (-0.98, -1.24)^T, \boldsymbol{\mu}_2 = (-2.96, 1.16)^T, \boldsymbol{\mu}_3 = (-2.96, 1.16)^T$$

как показано на рисунке 13.2a. С этими начальными кластерами K-средних требуется восемь итераций для схождения. На рисунке 13.2b показаны кластеры и их средние значения после одной итерации:

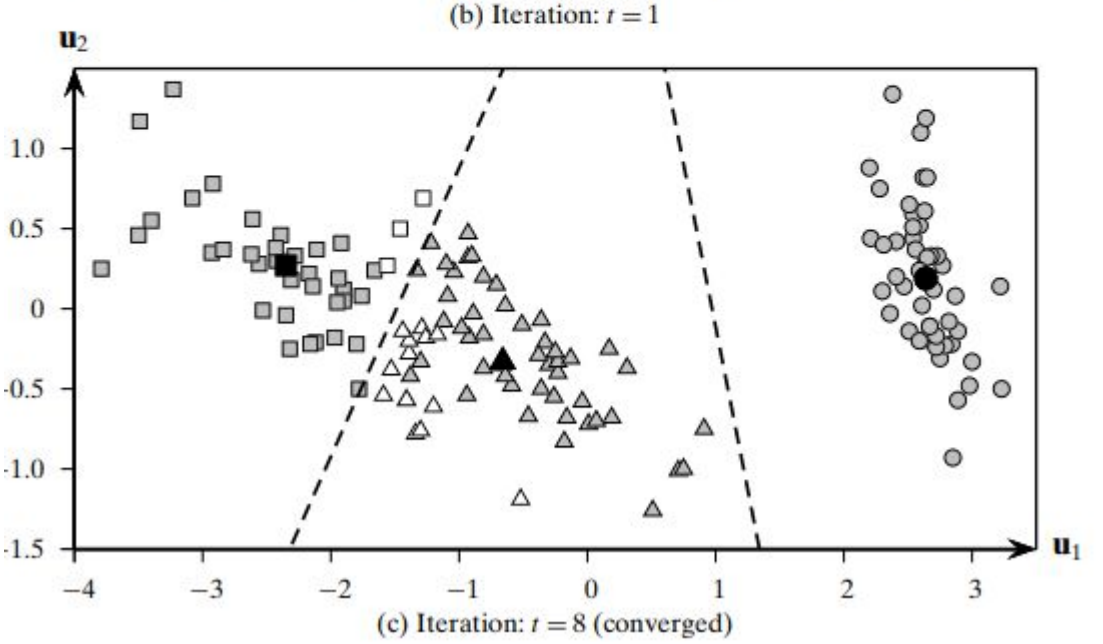
$$\boldsymbol{\mu}_1 = (1.56, -0.08)^T, \boldsymbol{\mu}_2 = (-2.86, 0.53)^T, \boldsymbol{\mu}_3 = (-1.50, -0.05)^T$$



(a) Random initialization: $t = 0$



(b) Iteration: $t = 1$



(c) Iteration: $t = 8$ (converged)

Рисунок 13.2

На рис. 13.2c показаны кластеры при сходимости. Окончательные средние следующие:

$$\boldsymbol{\mu}_1 = (2.64,0.19)^T, \boldsymbol{\mu}_2 = (-2.35,0.27)^T, \boldsymbol{\mu}_3 = (-0.66,-0.33)^T$$

На рисунке 13.2 средние значения кластера показаны черными точками, также показаны выпуклые области пространства данных, соответствующие каждому из трех кластеров. Пунктирные линии (гиперплоскости) - это серединные перпендикулярные отрезки прямых, соединяющих два центра кластеров. Получившееся выпуклое разбиение точек составляет кластеризацию.

На рисунке 13.2c показаны последние три кластера: C_1 в виде окружностей, C_2 в виде квадратов и C_3 в виде треугольников. Белые точки указывают на неправильную группировку по сравнению с известными типами ирисов. Таким образом, мы видим, что C_1 идеально соответствует iris-setosa, а большинство точек C_2 соответствует iris-virginica, а C_3 - iris-versicolor. Например, три точки (белые квадраты) типа iris-versicolor ошибочно сгруппированы вC_2, а 14 точек от iris-virginica ошибочно сгруппированы в C_3 (белые треугольники). Конечно, поскольку метка класса Iris не используется в кластеризации, разумно ожидать, что мы не получим идеальной кластеризации.

13.2 Ядерный алгоритм K-средних

В K-средних разделяющая граница между кластерами линейна. Ядерный K-средних позволяет извлекать нелинейные границы между кластерами с помощью хитрости с ядром, описанной в главе 5. Таким образом, метод может быть использован для обнаружения невыпуклых кластеров.

В ядерном K-средних основная идея состоит в концептуальном отображении точки данных $\mathbf{x}_i$ во входном пространстве в точку $\phi(\mathbf{x}_i)$ в некотором многомерном пространстве признаков посредством соответствующего нелинейного отображения ϕ. Однако трюк с ядром позволяет нам выполнять кластеризацию в пространстве признаков исключительно в терминах функции ядра $K(\mathbf{x}_i, \mathbf{x}_j)$, которая может быть вычислена во входном пространстве, но соответствует скалярному (или внутреннему) произведению $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ в пространстве признаков.

Предположим пока, что все точки $\mathbf{x}_i \in \mathbf{D}$ сопоставлены с соответствующими отображениями $\phi(\mathbf{x}_i)$ в пространстве признаков. Пусть $\mathbf{K} = \{K(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1,\ldots,n}$ обозначает симметричную ядерную матрицу $n \times n$, где $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$. Пусть $\{C_1, \ldots, C_k\}$ задает разбиение n точек на k кластеров, и пусть соответствующие средние значения кластеров в пространстве признаков задаются как $\{\boldsymbol{\mu}_i^\phi, \ldots, \boldsymbol{\mu}_k^\phi\}$, где

$$\boldsymbol{\mu}_i^\phi = \frac{1}{n_i}\sum_{\mathbf{x}_j \in C_i} \phi(\mathbf{x}_j)$$

среднее кластера C_i в пространстве признаков и $n_i = |C_i|$.

В пространстве признаков ядерная K-средних сумма квадратов отклонений может быть записана как

$$\min_C SSE(C) = \sum_{i=1}^{k}\sum_{\mathbf{x}_j \in C_i} \left\|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\phi\right\|^2$$

Расширяя ядерную SSE в терминах функции ядра, мы получаем

$$SSE(C) = \sum_{i=1}^{k}\sum_{\mathbf{x}_j \in C_i} \left\|\phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^\phi\right\|^2$$

$$= \sum_{i=1}^{k}\sum_{\mathbf{x}_j \in C_i} \left\|\phi(\mathbf{x}_j)\right\|^2 - 2\phi(\mathbf{x}_j)^T\boldsymbol{\mu}_i^\phi + \left\|\boldsymbol{\mu}_i^\phi\right\|^2$$

$$= \sum_{i=1}^{k}\left(\left(\sum_{\mathbf{x}_j \in C_i} \left\|\phi(\mathbf{x}_j)\right\|^2\right) - 2n_i\left(\frac{1}{n_i}\sum_{\mathbf{x}_j \in C_i} \phi(\mathbf{x}_j)\right)^T\boldsymbol{\mu}_i^\phi + n_i\left\|\boldsymbol{\mu}_i^\phi\right\|^2\right)$$

$$= \left(\sum_{i=1}^{k} \sum_{\mathbf{x}_j \in C_i} \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_j) \right) - \left(\sum_{i=1}^{k} n_i \left\| \boldsymbol{\mu}_i^{\phi} \right\|^2 \right)$$

$$= \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in C_i} K(\mathbf{x}_j, \mathbf{x}_j) - \sum_{i=1}^{k} \frac{1}{n_i} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b)$$

$$= \sum_{j=1}^{n} K(\mathbf{x}_j, \mathbf{x}_j) - \sum_{i=1}^{k} \frac{1}{n_i} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b) \tag{13.3}$$

Таким образом, ядерная K-средних SSE целевая функция может быть выражена исключительно в терминах функции ядра. Как и K-средних, для минимизации целевой функции SSE мы применяем жадный итеративный подход. Основная идея состоит в том, чтобы присвоить каждой точке ближайшее среднее значение в пространстве признаков, что приведет к новой кластеризации, которая, в свою очередь, может быть использована для получения новых оценок для средних значений кластера. Однако основная трудность заключается в том, что мы не можем явно вычислить среднее значение каждого кластера в пространстве признаков. К счастью, явное получение средних кластера не требуется; все операции могут быть выполнены в терминах функции ядра $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$.

Рассмотрим расстояние от точки $\phi(\mathbf{x}_j)$ до среднего значения $\boldsymbol{\mu}_i^{\phi}$ в пространстве признаков, которое можно вычислить как

$$\left\| \phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^{\phi} \right\|^2 = \left\| \phi(\mathbf{x}_j) \right\|^2 - 2\phi(\mathbf{x}_j)^T \boldsymbol{\mu}_i^{\phi} + \left\| \boldsymbol{\mu}_i^{\phi} \right\|^2$$

$$= \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) - \frac{2}{n_i} \sum_{\mathbf{x}_a \in C_i} \phi(\mathbf{x}_j)^T \phi(\mathbf{x}_a) + \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} \phi(\mathbf{x}_a)^T \phi(\mathbf{x}_b)$$

$$= K(\mathbf{x}_j, \mathbf{x}_j) - \frac{2}{n_i} \sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j) + \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b) \tag{13.4}$$

Таким образом, расстояние от точки до среднего значения кластера в пространстве признаков можно вычислить, используя только ядерные операции. На этапе назначения кластера ядерного K-средних мы присваиваем точку ближайшему среднему кластеру следующим образом:

$$C^*(\mathbf{x}_j) = \arg\min_i \left\{ \left\| \phi(\mathbf{x}_j) - \boldsymbol{\mu}_i^{\phi} \right\|^2 \right\}$$

$$= \arg\min_i \left\{ K(\mathbf{x}_j, \mathbf{x}_j) - \frac{2}{n_i} \sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j) + \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b) \right\}$$

$$= \arg\min_i \left\{ \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b) - \frac{2}{n_i} \sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j) \right\} \quad (13.5)$$

где мы опускаем член $K(\mathbf{x}_j, \mathbf{x}_j)$, потому что он остается одинаковым для всех k кластеров и не влияет на решение о назначении кластера. Также обратите внимание, что первый член – это просто среднее попарное ядерное значение для кластера C_i, которое не зависит от точки $\mathbf{x}_j$. Фактически, это квадрат нормы кластерного среднего в пространстве признаков. Второй член в два раза больше среднего значения ядра для точек в C_i относительно $\mathbf{x}_j$.

Алгоритм 13.2 показывает псевдокод для метода ядерного K-средних. Он начинается с первоначального случайного разбиения точек на k кластеров. Затем он итеративно обновляет назначения кластера, переназначая каждую точку ближайшему среднему значению в пространстве признаков с помощью уравнения (13.5). Чтобы облегчить вычисление расстояния, он сначала вычисляет среднее значение ядра, то есть квадрат нормы среднего значения кластера, для каждого кластера (для цикла в строке 5). Затем он вычисляет среднее значение ядра для каждой точки $\mathbf{x}_j$ с точками в кластере C_i (для цикла в строке 7). На этапе назначения основного кластера эти значения используются для вычисления расстояния xj от каждого из кластеров C_i и $\mathbf{x}_j$ присваиваются ближайшему среднему значению. Эта информация о переназначении используется для повторного разделения точек на новый набор кластеров. То есть все точки $\mathbf{x}_j$, которые ближе к среднему значению C_i, составляют новый кластер для следующей итерации. Этот итерационный процесс повторяется до сходимости.

ALGORITHM 13.2. Kernel K-means Algorithm

KERNEL-KMEANS($\mathbf{K}, k, \epsilon$):

$t \leftarrow 0$
$\mathcal{C}^t \leftarrow \{C_1^t, \ldots, C_k^t\}$// Randomly partition points into k clusters
repeat
  $t \leftarrow t+1$
  foreach $C_i \in \mathcal{C}^{t-1}$ **do** // Compute squared norm of cluster means
    $\text{sqnorm}_i \leftarrow \frac{1}{n_i^2} \sum_{\mathbf{x}_a \in C_i} \sum_{\mathbf{x}_b \in C_i} K(\mathbf{x}_a, \mathbf{x}_b)$
  foreach $\mathbf{x}_j \in \mathbf{D}$ **do** // Average kernel value for $\mathbf{x}_j$ and C_i
    foreach $C_i \in \mathcal{C}^{t-1}$ **do**
      $\text{avg}_{ji} \leftarrow \frac{1}{n_i} \sum_{\mathbf{x}_a \in C_i} K(\mathbf{x}_a, \mathbf{x}_j)$
  // Find closest cluster for each point
  foreach $\mathbf{x}_j \in \mathbf{D}$ **do**
    foreach $C_i \in \mathcal{C}^{t-1}$ **do**
      $d(\mathbf{x}_j, C_i) \leftarrow \text{sqnorm}_i - 2 \cdot \text{avg}_{ji}$
    $j^* \leftarrow \arg\min_i \{d(\mathbf{x}_j, C_i)\}$
    $C_{j^*}^t \leftarrow C_{j^*}^t \cup \{\mathbf{x}_j\}$ // Cluster reassignment
  $\mathcal{C}^t \leftarrow \{C_1^t, \ldots, C_k^t\}$
until $1 - \frac{1}{n} \sum_{i=1}^k \left| C_i^t \cap C_i^{t-1} \right| \leq \epsilon$

Для проверки сходимости мы проверяем, есть ли какие-либо изменения в кластерных назначениях точек. Количество точек, не меняющих кластеры, задается как сумма $\sum_{i=1}^k \left| C_i^t \cap C_i^{t-1} \right|$, где t задает текущую итерацию. Доля точек, переназначенных другому кластеру в текущей итерации, задается как

$$\frac{n - \sum_{i=1}^k \left| C_i^t \cap C_i^{t-1} \right|}{n} = 1 - \frac{1}{n} \sum_{i=1}^k \left| C_i^t \cap C_i^{t-1} \right|$$

Ядерный K-средних останавливается, когда доля точек с новыми назначениями кластера падает ниже некоторого порога $\epsilon \geq 0$. Например, можно выполнять итерацию, пока никакие точки не изменяют кластеры.

Вычислительная сложность

Вычисление среднего значения ядра для каждого кластера C_i занимает время $O(n^2)$ по всем кластерам. Вычисление среднего значения ядра для каждой точки по отношению к каждому из k кластеров также занимает $O(n^2)$ времени. Наконец, вычисление ближайшего среднего для каждой точки и переназначения кластера занимает $O(kn)$ времени. Общая вычислительная сложность

ядерного K-средних составляет $O(tn^2)$, где t – это количество итераций до сходимости. Сложность I/O составляет $O(t)$ проходов по ядерной матрице **K**.

13.3 КЛАСТЕРИЗАЦИЯ С МАКСИМИЗАЦИЕЙ ОЖИДАНИЙ

Метод k-средних является примером «жесткой» кластеризации, где каждая точка может принадлежать только одному кластеру. Теперь мы обобщаем подход для рассмотрения «мягкого» распределения в кластеры, так что у каждой точки есть вероятность принадлежать каждому кластеру.

Пусть **D** состоит из n точек $\boldsymbol{x}_j$ в d-мерном пространстве $\mathbb{R}^d$. Пусть X_a – случайная величина, соответствующая a-ому атрибуту. Мы также используем X_a для обозначения a-того вектора столбца, соответствующий n выборкам данных из X_a. Пусть $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ – векторная случайная величина, определенная через d-атрибуты с помощью x_j, являющимися образцами данных из **X**.

Модель Гауссовой смеси

Мы предполагаем, что каждый кластер характеризуется многомерным нормальным распределением, то есть:

$$f_i(\mathbf{x}) = f\left(\mathbf{x}_j \middle| \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) = \frac{1}{(2\pi)^{-\frac{d}{2}} \left|\boldsymbol{\Sigma}_i^{-1}\right|^{\frac{1}{2}}} exp\left\{-\frac{(\mathbf{x}_j - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_j - \boldsymbol{\mu}_i)}{2}\right\} \quad (13.6)$$

где $\boldsymbol{\mu}_i \in \mathbb{R}^d$ – среднее значение кластера и $\boldsymbol{\Sigma}_i \in \mathbb{R}^{d \times d}$ – ковариационная матрица неизвестные параметры. $f_i(\mathbf{x})$ – плотность вероятности **x**, принадлежности к кластеру C_i. Мы предполагаем, что функция плотности вероятности **X** задается как *модель гауссовой смеси* для всех k нормалей кластеров, определяемых как

$$f(\mathbf{x}) = \sum_{j=1}^{k} f_i(\mathbf{x}) P(C_i) = \sum_{j=1}^{k} f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) P(C_i) \quad (13.7)$$

где априорные вероятности $P(C_i)$ называются *параметрами смеси*, которые должны удовлетворять условию

$$\sum_{i=1}^{k} P(C_i) = 1$$

Таким образом, модель гауссовой смеси характеризуется средним значением $\boldsymbol{\mu}_i$, ковариационной матрицей $\boldsymbol{\Sigma}_i$, и вероятностью смеси $P(C_i)$ для каждого из k нормальных распределений. Мы компактно запишем множество всех параметров модели в виде:

$$\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, P(C_1), \ldots, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, P(C_k)\}$$

Оценка максимального правдоподобия

С учетом набора данных **D**, определим правдоподобие $\boldsymbol{\theta}$ как условную вероятность данных **D** с учетом параметров модели $\boldsymbol{\theta}$, обозначенную как $P(\mathbf{D}|\boldsymbol{\theta})$. Потому что каждая из n точек x_j считается случайной выборкой из X (т. е. независимой, с распределением как **X**), правдоподобие $\boldsymbol{\theta}$ задается как

$$P(\mathbf{D}|\boldsymbol{\theta}) = \prod_{j=1}^{k} f(\mathbf{x}_j)$$

Целью оценки максимального правдоподобия (MLE) является выбор параметров $\boldsymbol{\theta}$, которые максимизируют вероятность, то есть:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\{P(\mathbf{D}|\boldsymbol{\theta})\}$$

Максимизация логарифма функции правдоподобия – это типичное решение, поскольку оно превращает произведение по точкам в суммирование, а максимальное значение правдоподобия и логарифмической вероятности совпадают. Т.е. MLE максимизирует

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\{\ln P(\mathbf{D}|\boldsymbol{\theta})\}$$

где *функция логарифмического правдоподобия* задается как:

$$\ln P(\mathbf{D}|\boldsymbol{\theta}) = \sum_{j=1}^{n} \ln f(\mathbf{x}_j) = \sum_{j=1}^{n} \ln\left(\sum_{i=1}^{k} f(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) P(C_i)\right) \qquad (13.8)$$

Непосредственная максимизация логарифмической вероятности над $\boldsymbol{\theta}$ – весьма трудная задача. Вместо этого мы можем использовать подход максимизации ожидания (EM) для нахождения оценок максимального правдоподобия для параметров $\boldsymbol{\theta}$. EM - это двухэтапный итерационный подход, который начинается с начального предположения для параметров $\boldsymbol{\theta}$. Учитывая текущие оценки для $\boldsymbol{\theta}$, на *шаге ожидания* EM вычисляет апостериорные вероятности кластера $P(C_i|\mathbf{x}_j)$ теорему Байеса:

$$P(C_i|\mathbf{x}_j) = \frac{P(C_i \text{ and } \mathbf{x}_j)}{P(\mathbf{x}_j)} = \frac{P(\mathbf{x}_j|C_i)P(C_i)}{\sum_{a=1}^{k} P(\mathbf{x}_j|C_a)P(C_a)}$$

Поскольку каждый кластер моделируется как многомерное нормальное распределение [ур-е (13.6)], вероятность $\mathbf{x}_j$ данного кластера можно получить, рассматривая небольшой интервал $\varepsilon > 0$ с центром в точке $\mathbf{x}_j$, следующим образом:

$$P(\mathbf{x}_j|C_i) \simeq 2\varepsilon \cdot f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = 2\varepsilon \cdot f_i(\mathbf{x}_j)$$

Апостериорная вероятность C_i с $\mathbf{x}_j$:

$$P(C_i|\mathbf{x}_j) = \frac{f_i(\mathbf{x}_j)P(C_i)}{\sum_{a=1}^{k} f_a(\mathbf{x}_j)P(C_a)} \qquad (13.9)$$

И $P(C_i|\mathbf{x}_j)$ можно рассматривать как вес или вклад точки $\mathbf{x}_j$ в кластеризацию C_i. Далее, на этапе максимизации, используя веса $P(C_i|\mathbf{x}_j)$ EM заново пересчитывает $\boldsymbol{\theta}$, то есть он заново оценивает параметры $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$, и $P(C_i)$ для каждого кластера C_i. Пересчитанное среднее значение задается как средневзвешенное значение всех точек, пересчитанная ковариационная матрица задается как взвешенная ковариация по всем парам измерений, а пересчитанная априорная вероятность для каждого кластера задается как доля весов, которые вносят вклад в этот кластер. В разделе 13.3.3 мы формально выводим выражения для оценок MLE для параметров кластера, а в разделе 13.3.4 мы более подробно описываем общий подход EM. Мы начнем с применения алгоритма кластеризации EM для одномерных и общих d-мерных случаев.

13.3.1 EM в одномерном пространстве

Рассмотрим набор данных $\mathbf{D}$, состоящий из одного атрибута $\mathbf{X}$, где каждая точка $\boldsymbol{x}_j \in \mathbb{R}$ $(j = 1, \ldots, n)$ – случайная выборка из $\mathbf{X}$. Для модели смеси [ур-е 13.7], мы используем одномерные нормали для каждого кластера:

$$f_i(\mathbf{x}) = f(\mathrm{x}_j|\mu_i, \sigma_i^2) = \frac{1}{\sqrt{2\pi}\sigma_j} exp\left\{-\frac{(\mathbf{x}_j - \boldsymbol{\mu}_i)^2}{2\sigma_i^2}\right\}$$

с параметрами кластера μ_i, σ_i^2 и $P(C_i)$. Подход EM состоит из трех этапов: инициализации, шага ожидания и шага максимизации.

Инициализация

Для каждого кластера C_i для $(i = 1, 2, \ldots, k)$, мы можем случайным образом инициализировать параметры кластер μ_i, σ_i^2 и $P(C_i)$. Среднее значение μ_i выбирается равномерно случайным образом из диапазона возможных значений для **X**. Обычно предполагается, что начальная дисперсия задается в виде $\sigma_i^2 = 1$. Наконец, предыдущие вероятности кластера инициализируются в $P(C_i) = \frac{1}{k}$, так что каждый кластер имеет равную вероятность.

Шаг ожидания

Предположим, что для каждого из k кластеров мы имеем оценку параметров, а именно: среднее значение μ_i, дисперсия σ_i^2 и априорная вероятность $P(C_i)$. Учитывая эти значения апостериорные вероятности кластеров вычисляются с помощью ур-я (13.9):

$$P(C_i|\mathbf{x}_j) = \frac{f(\mathrm{x}_j|\mu_i, \sigma_i^2)P(C_i)}{\sum_{a=1}^{k} f(\mathrm{x}_j|\mu_a, \sigma_a^2)P(C_a)}$$

Для удобства мы используем обозначение $w_{ij} = P(C_i|x_j)$, рассматривая апостериорные вероятности как вес или вклад точки x_j в кластер C_i. Далее, пусть:

$$\boldsymbol{w}_i = (w_1, w_2, \ldots, w_n)^T$$

обозначим весовой вектор для кластера C_i через все *n* точек.

Шаг максимизации

При условии, что все апостериорные значения вероятности или веса $w_{ij} = P(C_i|x_j)$ известны, шаг максимизации, как следует из названия, вычисляет оценки максимального правдоподобия параметров кластера путем пересчета μ_i, σ_i^2 и $P(C_i)$. Пересчитанное значение для среднего значения кластера, μ_i, вычисляется как средневзвешенное значение из всех пунктов:

$$\mu_i = \frac{\sum_{j=1}^{n} w_{ij}\mathrm{x}_j}{\sum_{j=1}^{n} w_{ij}}$$

В терминах вектора веса $\boldsymbol{w}_i$ и вектора атрибутов $X = (x_1, x_2, \ldots, x_n)^T$ - мы можем переписать как:

$$\mu_i = \frac{\boldsymbol{w}_i{}^T X}{\boldsymbol{w}_i{}^T \mathbf{1}}$$

Пересчитанное значение кластерной дисперсии вычисляется как взвешенная дисперсия по всем точкам:

$$\sigma_i^2 = \frac{\sum_{j=1}^{n} w_{ij}(x_j - \mu_i)^2}{\sum_{j=1}^{n} w_{ij}}$$

Пусть $Z_i = X - \mu_i \mathbf{1} = (x_1 - \mu_i, x_2 - \mu_i, \ldots, x_n - \mu_i)^T = (z_{1i}, z_{2i}, \ldots, z_{in})^T$

будет отцентрированным вектором атрибутов для кластера C_i, и пусть Z_i^S –квадратный вектор, заданный как $Z_i = ({z_{i1}}^2, {z_{i2}}^2, \ldots, {z_{in}}^2)^T$. Дисперсия может быть компактно выражена в терминах произведения между вектором веса и квадратом центрированного вектора:

$$\sigma_i^2 = \frac{\boldsymbol{w}_i{}^T Z_i^{s\,T}}{\boldsymbol{w}_i{}^T \mathbf{1}}$$

Наконец, априорная вероятность кластера C_i пересчитывается как доля веса, принадлежащего C_i, от общего веса:

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{\sum_{a=1}^{k}\sum_{j=1}^{n} w_{aj}} = \frac{\sum_{j=1}^{n} w_{ij}}{\sum_{j=1}^{k} 1} = \frac{\sum_{j=1}^{n} w_{ij}}{n} \tag{13.10}$$

где мы воспользовались тем фактом, что

$$\sum_{i=1}^{n} w_{ij} = \sum_{i=1}^{n} P(C_i|x_j) \; = 1$$

В векторной нотации априорная вероятность может быть записана как:

$$P(C_i) = \frac{\boldsymbol{w}_i{}^T \mathbf{1}}{n}$$

Итерация

Начиная с исходного набора значений параметров кластера μ_i, σ_i^2 и $P(C_i)$ для всех i $= 1, \ldots,$ k, алгоритм EM применяет шаг ожидания для вычисления весов $w_{ij} = P(C_i|x_j)$. Эти значения затем используются на этапе максимизации для вычисления обновленных параметров кластера μ_i, σ_i^2 и $P(C_i)$. Как шаг ожидания, так и максимизация применяются итеративно до тех пор, пока, например, не произойдет конвергенция, например, пока средние не изменятся очень мало от одной итерации к другой.

13.3.2 EM в *d*-мерном пространстве

Теперь рассмотрим метод ЭМ в *d* измерениях, где каждый кластер

характеризуется многомерным нормальным распределением [ур-е. (13.6)], с параметрами $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$ и $P(C_i)$. Таким образом, для каждого кластера C_i, нам нужно оценить d-мерный вектор среднего:

$$\boldsymbol{\mu}_i = (\mu_1, \mu_2, \ldots, \mu_n)^T$$

и ковариационной матрицей $d \times d$:

$$\boldsymbol{\Sigma} = \begin{pmatrix} (\sigma_1^i)^2 & \sigma_{12}^i & \cdots & \sigma_{1d}^i \\ \sigma_{21}^i & (\sigma_2^i)^2 & & \sigma_{1d}^i \\ \vdots & & \ddots & \vdots \\ \sigma_{d1}^i & \sigma_{d2}^i & \cdots & (\sigma_d^i)^2 \end{pmatrix}$$

Поскольку ковариационная матрица симметрична, мы должны оценить: $\binom{d}{2} = \frac{d(d-1)}{2}$ попарных ковариаций и d дисперсии, в общей сложности $\frac{d(d+1)}{2}$ параметры для $\mathbf{\Sigma}_i$. Такое количество может быть большим для практических целей, потому что у нас может быть недостаточно данных, чтобы надежно оценить все из них. Например, если d = 100, то мы должны оценить $100 * \frac{101}{2} =$ 5050100 * 101/2 = 5050 параметров! Одним из упрощений является предположение, что все измерения являются независимыми, что приводит к диагональной ковариационной матрице:

$$\mathbf{\Sigma}_i = \begin{pmatrix} (\sigma_1^i)^2 & 0 & \cdots & 0 \\ 0 & (\sigma_2^i)^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & (\sigma_d^i)^2 \end{pmatrix}$$

При предполагаемой независимости у нас есть только d параметров для оценки диагональной ковариационной матрицы.

Инициализация

Для каждого кластера C_i, где $i = 1, \ldots, k$, мы случайным образом инициализируем среднее значение $\boldsymbol{\mu}_i$ путем выбора наугад значения μ_{ia} для каждого измерения X_a равномерно распределённого на диапазоне X_a. Ковариационная матрица инициализируется как матрица тождества $d \times d$ $\mathbf{\Sigma}_i = \mathbf{I}$. Наконец, апостериорные вероятности кластера инициализируются $P(C_i) = 1$, так что каждый кластер имеет равную вероятность.

Шаг ожидания

На шаге ожидания мы вычисляем апостериорную вероятность кластера C_i для точки $\boldsymbol{x}_j$, использую ур-е. (13.9), при $i = 1, \ldots, k$и $j = 1, \ldots, n$. Как и прежде, мы используем сокращенные обозначение $w_{ij} = P(C_i|x_j)$, чтобы обозначить тот факт, что $P(C_i|x_j)$ можно рассматривать как вес вклада точки $\boldsymbol{x}_j$ в кластер C_i, и мы используем обозначение $\boldsymbol{w}_i = (w_1, w_2, \ldots, w_n)^T$ чтобы определить весовой вектор для кластера C, через все n точек.

Шаг максимизации

При заданных весах w_{ij}, на этапе максимизации мы повторно вычисляем, $\boldsymbol{\mu}_i$, $\mathbf{\Sigma}_i$ и $P(C_i)$. Среднее значение $\boldsymbol{\mu}_i$ для кластера C_i можно оценить как:

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{n} w_{ij} \mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}} \tag{13.11}$$

которое может быть компактно выражено в матричном виде как:

$$\boldsymbol{\mu}_i = \frac{\mathbf{D}^T \boldsymbol{w}_i}{\boldsymbol{w}_i{}^T \mathbf{1}}$$

Пусть $Z_i = \mathbf{D} - \mathbf{1}\boldsymbol{\mu}_i{}^T$ – центрированная матрица данных для кластера C_i. Пусть $\boldsymbol{z}_{ji} = \boldsymbol{x}_j - \boldsymbol{\mu}_i \in \mathbf{R}^d$ – j-я центрированную точку в Z_i. Мы можем выразить $\mathbf{\Sigma}_i$:

$$\mathbf{\Sigma}_i = \frac{\sum_{j=1}^{n} w_{ij} \mathbf{z}_{ji} \mathbf{z}_{ji}{}^T}{\boldsymbol{w}_i{}^T \mathbf{1}} \tag{13.12}$$

Учитывая попарное представление атрибутов, ковариация между измерениями X_a и X_b оценивается как:

$$\sigma_{ab}^{i} = \frac{\sum_{j=1}^{n} w_{ij}\left(\mathrm{x}_{ja} - \mu_{ia}\right)\left(\mathrm{x}_{jb} - \mu_{ib}\right)}{\sum_{j=1}^{n} w_{ij}}$$

где x_{ja} и μ_{ia} – значения a-го измерения для $\boldsymbol{x}_j$ и $\boldsymbol{\mu}_i$, соответственно.

Наконец, априорная вероятность $P(C_i)$ для каждого кластера – это то же самое, что и одномерный случай [ур-е. (13.10)], приведенное в виде:

$$P(C_i) = \frac{\sum_{j=1}^{k} w_{ij}}{n} = \frac{\boldsymbol{w}_i{}^T\mathbf{1}}{n} \tag{13.13}$$

Формальный вывод этих переоценок для $\boldsymbol{\mu}_i$ [ур-е. (13.11)], $\boldsymbol{\Sigma}_i$ [ур-е (13.12)] и $P(C_i)$ [ур-е (13.13)] приводится в разделе 13.3.3.

Алгоритм кластеризации ЭМ

Псевдокод для многомерного алгоритма кластеризации ЭМ приведен в

алгоритме 13.3. После инициализации $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$ и $P(C_i)$ для всех $i = 1, \ldots, k$, шаги ожидания и максимизации повторяются до тех пор, пока не произойдет конвергенция. Для проверки конвергенции мы проверяем, что $\sum_i \left\|\boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1}\right\|^2 \leq \varepsilon$, где $\varepsilon > 0$ – порог сходимости, и t обозначает итерацию. Другими словами, итерационный процесс продолжается до тех пор, пока изменение в среднем кластера не станет очень маленьким.

А Л Г О Р И Т Н М 13.3. Алгоритм максимизации ожидания (EM)

Максимизация ожидания (**D**, k, ε):

1. $t \leftarrow 0$ // Инициализация
2. Случайным образом инициализировать $\boldsymbol{\mu}_i^t$, $i = 1, \ldots, k$
3. $\boldsymbol{\Sigma}_i^t \leftarrow \mathbf{1}$, $i = 1, \ldots, k$
4. $P(C_i) \leftarrow \frac{1}{k}$, $i = 1, \ldots, k$
5. Повторять
   a. $t \leftarrow t - 1$

   // Шаг ожидания

   b. Для $i = 1, \ldots, k$ и $j = 1, \ldots, n$ делать:
      i. $w_{ij} \leftarrow \frac{f\left(\mathbf{x}_j \middle| \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) P(C_i)}{\sum_{a=1}^{k} f\left(\mathbf{x}_j \middle| \boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a\right) P(C_a)}$ // апостериорная вероятность $P(C_i|\mathbf{x}_j)$,

   // Шаг максимизации

   c. Для $i = 1, \ldots, k$ делать
      i. $\boldsymbol{\mu}_i \leftarrow \frac{\sum_{j=1}^{n} w_{ij}\mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}}$
      ii. $\boldsymbol{\Sigma}_i \leftarrow \frac{\sum_{j=1}^{n} w_{ij}\mathbf{z}_{ji}\mathbf{z}_{ji}{}^T}{\boldsymbol{w}_i{}^T\mathbf{1}}$
      iii. $P(C_i) \leftarrow \frac{\boldsymbol{w}_i{}^T\mathbf{1}}{n}$
6. До тех пор пока не $\sum_i \left\|\boldsymbol{\mu}_i^t - \boldsymbol{\mu}_i^{t-1}\right\|^2 \leq \varepsilon$

Вычислительная сложность

Для шага ожидания, чтобы вычислить апостериорные вероятности кластера, нам нужно инвертировать $\mathbf{\Sigma}_i$ и вычислить его определитель $|\mathbf{\Sigma}_i||$, который занимает $O(d^3)$. Для k кластеров время равно $O(kd^3)$. На шаге ожидания, оценка плотности $f(\mathbf{x}_j|\boldsymbol{\mu}_i, \mathbf{\Sigma}_i)$ требуется $O(d^2)$, для n точек и k кластеров это потребует $O(knd^2)$. На шаге максимизации, время обновления для $\mathbf{\Sigma}_i$ для k кластеров по требует $O(knd^2)$. Таким образом, вычислительная сложность метода EM, составляет $O(t(kd^3 + knd^2))$где t-число итераций. Если мы используем диагональную ковариационную матрицу, то обратная матрица и определитель могут быть вычислены за $O(d)$. Вычисление плотности в точке занимает $O(d)$ времени, так что время для шага ожидания равно $O(knd)$. Шаг максимизации также занимает $O(knd)$ для повторной оценки $\mathbf{\Sigma}_i$. Таким образом, общее время для диагональной ковариационной матрицы равно $O(tknd)$.. Сложность ввода-вывода для алгоритма EM – это $O(t)$, полное сканирование базы данных, потому что мы читаем весь набор точек в каждой итерации.

Метод K-средних как частный случай EM

Хотя мы предположили нормальную модель смеси для кластеров, подход EM может быть использован с другими моделями распределения плотности кластеров $P(\mathbf{x}_j|C_i)$. Например, k-среднее можно рассматривать как частный случай алгоритма EM, полученный следующим образом:

$$P(\mathbf{x}_j|C_i) = \begin{cases} 1 \text{ если } C_i = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left\{ \|\mathbf{x}_j - \boldsymbol{\mu}_a\|^2 \right\} \\ \text{иначе } 0 \end{cases}$$

Используя ур-е (13.9), апостериорная вероятность $P(C_i|\mathbf{x}_j)$ задается какЖ

$$P(C_i|\mathbf{x}_j) = \frac{P(\mathbf{x}_j|C_i)P(C_i)}{\sum_{a=1}^{k} P(\mathbf{x}_j|C_a)P(C_a)}$$

Можно видеть, что если $P(\mathbf{x}_j|C_i) = 0$, тогда $P(C_i|\mathbf{x}_j) = 0$. В противном случае, если $P(\mathbf{x}_j|C_i) = 1$, то $P(C_i|\mathbf{x}_a) = 0$ для всех $a \neq i$, и таким образом $P(C_i|\mathbf{x}_j) = \frac{1 \cdot P(C_i)}{1 \cdot P(C_i)} = 1$. Если сложить все это вместе, то апостериорная вероятность задается в виде:

$$P(C_i|\mathbf{x}_j) = \begin{cases} 1 \text{ если } \mathbf{x}_j \in C_i, \text{т. е. } C_i = \underset{C_a}{\operatorname{argmax}} \left\{ \|\mathbf{x}_j - \boldsymbol{\mu}_a\|^2 \right\} \\ \text{иначе } 0 \end{cases}$$

Понятно, что для метода k-средних параметры кластера равны $\boldsymbol{\mu}_i$и $P(C_i)$, мы можем игнорировать ковариационную матрицу.

13.3.3 Оценка максимального правдоподобия

В этом разделе мы получаем оценки максимального правдоподобия для параметров кластера $\boldsymbol{\mu}_i$, $\mathbf{\Sigma}_i$ и $P(C_i)$. Мы делаем это, беря производную от логарифмической функции правдоподобия по отношению к каждому из этих параметров и устанавливая производную в ноль. Частичная производная логарифмической функции правдоподобия [ур-е. (13.8)] в отношении какого-то параметра $\boldsymbol{\theta}_i$ для кластера C_i задается как:

$$\frac{\partial}{\partial \boldsymbol{\theta}_i}\ln\big(P(\mathbf{D}|\boldsymbol{\theta})\big) = \frac{\partial}{\partial \boldsymbol{\theta}_i}\sum_{j=1}^{n}\ln\Big(f\big(\mathbf{x}_j\big)\Big)$$

$$= \sum_{j=1}^{n}\left(\frac{1}{f\big(\mathbf{x}_j\big)}\frac{\partial f\big(\mathbf{x}_j\big)}{\partial \boldsymbol{\theta}_i}\right)$$

$$= \sum_{j=1}^{n}\left(\frac{1}{f\big(\mathbf{x}_j\big)}\sum_{j=1}^{n}\frac{\partial}{\partial \boldsymbol{\theta}_i}\Big(f\big(\mathbf{x}_j\big|\boldsymbol{\mu}_a,\boldsymbol{\Sigma}_a\big)P(C_a)\Big)\right)$$

$$= \sum_{j=1}^{n}\left(\frac{1}{f\big(\mathbf{x}_j\big)}\frac{\partial}{\partial \boldsymbol{\theta}_i}\big(f\big(\mathbf{x}_j\big|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\big)P(C_i)\big)\right)$$

Последний шаг вытекает из того, что поскольку $\boldsymbol{\theta}_i$ является параметром для i-го кластера компоненты смеси для других кластеров являются константами относительно $\boldsymbol{\theta}_i$. Используя тот факт, что $|\boldsymbol{\Sigma}_i| = \frac{\mathbf{1}}{|\boldsymbol{\Sigma}_i{}^{-\mathbf{1}}|}$, многомерная нормальная плотность в ур-и (13.6) может быть записана следующим образом:

$$f\big(\mathbf{x}_j\big|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\big) = (2\pi)^{-\frac{d}{2}}\big|\boldsymbol{\Sigma}_i{}^{-1}\big|^{\frac{1}{2}}exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\} \tag{13.14}$$

где

$$g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i) = -\frac{1}{2}\big(\mathbf{x}_j-\boldsymbol{\mu}_i\big)^T\boldsymbol{\Sigma}_i{}^{-1}\big(\mathbf{x}_j-\boldsymbol{\mu}_i\big) \tag{13.15}$$

Таким образом, производная логарифмической функции правдоподобия может быть записана в виде

$$\frac{\partial}{\partial \boldsymbol{\theta}_i}\ln\big(P(\mathbf{D}|\boldsymbol{\theta})\big)$$

$$= \tfrac{1}{2}\textstyle\sum_{j=1}^{n}\frac{1}{f(\mathbf{x}_j)}\frac{\partial}{\partial \boldsymbol{\theta}_i}\Big((2\pi)^{-\frac{d}{2}}\big|\boldsymbol{\Sigma}_i{}^{-1}\big|^{\frac{1}{2}}exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\}P(C_i)\Big) \tag{13.16}$$

Далее мы используем тот факт, что

$$\frac{\partial}{\partial \boldsymbol{\theta}_i}exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\} = exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\}\frac{\partial}{\partial \boldsymbol{\theta}_i}g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i) \tag{13.17}$$

Оценка среднего значения

Для получения оценки максимального правдоподобия для среднего значения $\boldsymbol{\mu}_i$, мы должны взять производную логарифмического правдоподобия относительно $\boldsymbol{\theta}_i = \boldsymbol{\mu}_i$. Согласно ур-ю (13.16), единственное слагаемое, включающее $\boldsymbol{\mu}_i$, это $exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\}$. Используя тот факт, что

$$\frac{\partial}{\partial \boldsymbol{\mu}_i}exp\{g(\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\} = \boldsymbol{\Sigma}_i{}^{-1}\big(\mathbf{x}_j-\boldsymbol{\mu}_i\big) \tag{13.18}$$

и используя ур-е. (13.17), частичная производная логарифмического правдоподобия [ур-е. (13.16)] в отношении $\boldsymbol{\mu}_i$:

$$\frac{\partial}{\partial \boldsymbol{\mu}_i}\ln\big(P(\mathbf{D}|\boldsymbol{\theta})\big)$$

$$= \sum_{j=1}^{n} \frac{1}{f(\mathbf{x}_j)}(2\pi)^{-\frac{d}{2}}\left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}P(C_i)\boldsymbol{\Sigma}_i{}^{-1}(\mathbf{x}_j - \boldsymbol{\mu}_i)$$

$$= \sum_{j=1}^{n} \frac{f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)P(C_i)}{f(\mathbf{x}_i)}\boldsymbol{\Sigma}_i{}^{-1}(\mathbf{x}_j - \boldsymbol{\mu}_i)$$

$$= \sum_{j=1}^{n} w_{ij}\boldsymbol{\Sigma}_i{}^{-1}(\mathbf{x}_j - \boldsymbol{\mu}_i)$$

где мы использовали ур-я (13.14) и (13.9), а также тот факт, что

$$w_{ij} = P(C_i|\mathbf{x}_j) = \frac{f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)P(C_i)}{f(\mathbf{x}_i)}$$

Устанавливая частную производной логарифмического правдоподобия на нулевой вектор и умножая обе стороны на $\boldsymbol{\Sigma}_i$, мы получаем:

$\sum_{j=1}^{n} w_{ij}(\mathbf{x}_j - \boldsymbol{\mu}_i) = 0$, это означает, что

$\sum_{j=1}^{n} w_{ij}\mathbf{x}_j = \boldsymbol{\mu}_i \sum_{j=1}^{n} w_{ij}$, и поэтому

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{n} w_{ij}\mathbf{x}_j}{\sum_{j=1}^{n} w_{ij}} \qquad (13.19)$$

это именно та формула переоценки, которую мы использовали в ур-и. (13.11).

Оценка ковариационной матрицы

Для переоценки ковариационной матрицы $\boldsymbol{\Sigma}_i$, возьмем частную производную от ур-я (13.16) по $\boldsymbol{\Sigma}_i{}^{-1}$, используя правила для дифференциации $\left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}$.

Используя тот факт, что для любой квадратной матрицы $\mathbf{A}$ мы имеем $\frac{\partial|\mathbf{A}|}{\partial \mathbf{A}} = |\mathbf{A}| \cdot (\mathbf{A}^{-1})^T$, то производная от $\left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{\frac{1}{2}}$ в отношении $\boldsymbol{\Sigma}_i{}^{-1}$:

$$\frac{\partial\left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{\frac{1}{2}}}{\partial \boldsymbol{\Sigma}_i{}^{-1}} = \frac{1}{2} \cdot \left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{-\frac{1}{2}} \cdot \left|\boldsymbol{\Sigma}_i{}^{-1}\right| \cdot \boldsymbol{\Sigma}_i = \frac{1}{2}\left|\boldsymbol{\Sigma}_i{}^{-1}\right|^{\frac{1}{2}} \cdot \boldsymbol{\Sigma}_i \qquad (13.20)$$

Далее, используя тот факт, что для квадратной матрицы $\mathbf{A} \in \mathbf{R}^{d\times d}$ векторы $\mathbf{a}, \mathbf{b} \in \mathbf{R}^d$, у нас есть $\frac{\partial}{\partial \mathbf{A}}\mathbf{a}^T\mathbf{A}\mathbf{b} = \mathbf{a}\mathbf{b}^T$, производная от $exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}$ по $\boldsymbol{\Sigma}_i{}^{-1}$ получается из ур-я (13.17) следующим образом:

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_i{}^{-1}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\} = -\frac{1}{2} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}(\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T \quad (13.21)$$

Используя правила вывода на ур-ях (13.20) и (13.21), получаем

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_i^{-1}} \left|\boldsymbol{\Sigma}_i^{-1}\right|^{\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}$$

$$= \frac{1}{2}\left|\boldsymbol{\Sigma}_i^{-1}\right|^{\frac{1}{2}}\boldsymbol{\Sigma}_i exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\} - \frac{1}{2}\left|\boldsymbol{\Sigma}_i^{-1}\right|^{-\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T$$

$$= \tfrac{1}{2}\left|\boldsymbol{\Sigma}_i^{-1}\right|^{\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}(\boldsymbol{\Sigma}_i - \left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T) \qquad (13.22)$$

Подставляя ур-е (13.22) в ур-е (13.16) производная логарифмической функции правдоподобия относительно $\boldsymbol{\Sigma}_i^{-1}$ задается как:

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_i^{-1}} \ln\left(P(\mathbf{D}|\boldsymbol{\theta})\right)$$

$$= \frac{1}{2}\sum_{j=1}^{n} \frac{(2\pi)^{-\frac{d}{2}}\left|\boldsymbol{\Sigma}_i^{-1}\right|^{\frac{1}{2}} exp\{g(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)\}P(C_i)}{f(\mathbf{x}_i)}\left(\boldsymbol{\Sigma}_i - \left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T\right)$$

$$= \frac{1}{2}\sum_{j=1}^{n} \frac{f\left(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)P(C_i)}{f(\mathbf{x}_i)}\left(\boldsymbol{\Sigma}_i - \left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T\right)$$

$$= \frac{1}{2}\sum_{j=1}^{n} w_{ij}\left(\boldsymbol{\Sigma}_i - \left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T\right)$$

Устанавливая производную на нулевую матрицу $\mathbf{0}_{d\times d}$, мы можем решить для $\boldsymbol{\Sigma}_i$:

$\sum_{j=1}^{n} w_{ij}\left(\boldsymbol{\Sigma}_i - \left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T\right) = \mathbf{0}_{d\times d}$, из чего следует, что

$$\boldsymbol{\Sigma}_i = \frac{\sum_{j=1}^{n} w_{ij}\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)^T}{\sum_{j=1}^{n} w_{ij}} \qquad (13.23)$$

Таким образом, мы видим, что оценка максимального правдоподобия для ковариационной матрицы задается как взвешенная форма внешнего произведения в ур-и (13.12).

Оценка априорной вероятности: параметры смеси

Для получения оценки максимального правдоподобия для параметров смеси или априорных вероятностей $P(C_i)$, мы должны взять частную производную логарифмического правдоподобия [ур-е (13.16)] в отношении $P(C_i)$. Однако мы должны ввести множитель Лагранжа α для ограничения $\sum_{a=1}^{k} P(C_a) = 1$. Таким образом, мы берем производное:

$$\frac{\partial}{\partial P(C_i)}\left(\ln\left(P(\mathbf{D}|\boldsymbol{\theta})\right) + \alpha\left(\sum_{a=1}^{k} P(C_a) - 1\right)\right) \qquad (13.24)$$

Частная производная логарифмического правдоподобия в ур-и (13.16) по $P(C_i)$:

$$\frac{\partial}{\partial P(C_i)} \ln\left(P(\mathbf{D}|\boldsymbol{\theta})\right) = \sum_{j=1}^{n} \frac{f\left(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)}{f(\mathbf{x}_i)}$$

Таким образом, производная в ур-и (13.24):

$$\left(\sum_{j=1}^{n} \frac{f\left(\mathbf{x}_j \middle| \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)}{f(\mathbf{x}_i)}\right) + \alpha$$

Устанавливая производную в ноль и умножая с обеих сторон на $P(C_i)$, мы получаем

$$\sum_{j=1}^{n} \frac{f\left(\mathbf{x}_j \middle| \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) P(C_i)}{f(\mathbf{x}_i)} = -\alpha P(C_i)$$

$$\sum_{j=1}^{n} w_{ij} = -\alpha P(C_i) \tag{13.25}$$

Делая суммирование ур-я (13.25) по всем кластерам, получим:

$$\sum_{i=1}^{i} \sum_{j=1}^{n} w_{ij} = -\alpha \sum_{i=1}^{n} P(C_i)$$

$$\text{or } n = -\alpha \tag{13.26}$$

Последний шаг вытекает из того, что $\sum_{j=1}^{k} w_{ij} = 1$. Подставляя ур-е (13.26) в ур-е (13.25), получаем оценку максимального правдоподобия для $P(C_i)$ следующим образом:

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{n} \tag{13.27}$$

что соответствует формуле в ур-и (13.13).

Мы видим, что все три параметра $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$ и $P(C_i)$ для кластера C_i зависят от весов w_{ij}, которые соответствуют апостериорным вероятностям кластера $P(C_i|\mathbf{x}_j)$. Таким образом, уравнения (13.19), (13.23) и (13.27) не представляют собой замкнутое решение для максимизации логарифмической функции правдоподобия. Вместо этого мы используем итеративный подход EM для вычисления w_{ij} на шаге ожидания, а затем мы повторно оцениваем $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$ и $P(C_i)$ на этапе максимизации. Далее мы опишем структуру EM более подробно.

13.3.4 EM подход

Непосредственная максимизация логарифмической функции правдоподобия весьма трудна [ур-е. (13.8)], потому что слагаемое смеси появляется внутри логарифма. Проблема в том, что для любой точки $\mathbf{x}_j$ мы не знаем, из какого нормального или смешанного компонента она происходит. Предположим, что мы знали эту информацию, то есть предположим, что каждая точка $\mathbf{x}_j$ имеет связанное значение, указывающее на кластер, который генерирует точку. Как мы увидим, гораздо легче максимизировать логарифмическую вероятность, учитывая эту информацию.

Категориальный атрибут, соответствующий метке кластера, может быть смоделирован как векторная случайная величина $\mathbf{C} = (C_1, C_2, \ldots, C_k)$, где C_i является случайной величиной Бернулли (см. раздел 3.1.2 для получения подробной информации о том, как моделировать категориальную переменную). Если заданная точка генерируется из кластера C_i, то $C_i = 1$, в противном случае $C_i = 0$. Параметр $P(C_i)$ – это вероятность $P(C_i = 1)$. Потому что каждая точка может быть сгенерирована

только из одного кластера, если $C_a = 1$ для данной точки, тогда $C_i = 0$ для всех $i \neq a$. отсюда следует, что $\sum_{i=1}^{k} P(C_i) = 1$.

Для каждой точки $\mathbf{x}_j$, пусть его кластерный вектор равен $\mathbf{c}_j = (\mathrm{c}_{j1}, \mathrm{c}_{j2}, \ldots, \mathrm{c}_{jk})^T$. Только один компонент из $\mathbf{c}_j$ имеет значение 1. Если $\mathbf{c}_{ji} = 11$, это означает, что $C_i = 1$, то есть кластер C_i генерирует точка $\mathbf{x}_j$. Вероятностная массовая функция для **C** задается в виде:

$$P(\mathbf{C} = \mathbf{c}_j) = \prod_{i=1}^{k} P(C_i)^{\mathbf{c}_{ji}}$$

Учитывая кластерную информацию $\mathbf{c}_j$ для каждой точки $\mathbf{x}_j$, функция плотности условной вероятности для **X** задается в виде:

$$f(\mathbf{x}_j|\mathbf{c}_j) = \prod_{i=1}^{k} f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)^{\mathbf{c}_{ji}}$$

Только один кластер может генерировать $\mathbf{x}_j$, пусть C_a, в этом случае $\mathrm{c}_{ja} = 1$, и приведенное выше выражение упростилось бы до $f(\mathbf{x}_j|\mathbf{c}_j) = f(\mathbf{x}_j|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a)$.

Пара $(\mathbf{x}_j, \mathbf{c}_j)$ является случайной выборкой, полученной из совместного распределения вектора случайных величин $\mathbf{X} = (X_1, X_2, \ldots, X_d)$ и $\mathbf{C} = (C_1, C_2, \ldots, C_k)$, соответствующие *d* атрибутам данных и *k* атрибутам кластера. Совместная функция плотности **X** и **C** задается в виде:

$$f(\mathbf{x}_j \text{ and } \mathbf{c}_j) = f(\mathbf{x}_j|\mathbf{c}_j)P(\mathbf{c}_j) = \prod_{i=1}^{k} \left(f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)P(C_i)\right)^{\mathbf{c}_{ji}}$$

Логарифмическая вероятность для данных, приведенных в кластерной информации, выглядит следующим образом:

$$\begin{aligned}
\ln\left(P(\mathbf{D}|\boldsymbol{\theta})\right) &= \ln \prod_{i=1}^{k} f(\mathbf{x}_j \text{ and } \mathbf{c}_j|\boldsymbol{\theta}) \\
&= \sum_{j=1}^{n} \ln f(\mathbf{x}_j \text{ and } \mathbf{c}_j|\boldsymbol{\theta}) \\
&= \sum_{j=1}^{n} \ln \left(\prod_{i=1}^{k} \left(f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)P(C_i)\right)^{\mathbf{c}_{ji}} \right) \\
&= \sum_{j=1}^{n} \sum_{i=1}^{k} \mathrm{c}_{ji} \left(\ln f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) + \ln P(C_i)\right) \qquad (13.28)
\end{aligned}$$

Шаг ожидания

На шаге ожидания мы вычисляем ожидаемое значение логарифмической вероятности для помеченных данных, приведенных в уравнении. (13.28). Ожидание недостающей кластерной информации $\mathbf{c}_j$ описывается с помощью $\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ и $P(C_i)$ и фиксированного $\mathbf{x}_j$. Вследствие линейности ожидания, ожидаемое значение логарифмической вероятности задается в виде:

$$E[ln\ P\ (\mathbf{D}|\boldsymbol{\theta})] = \sum_{j=1}^{n}\sum_{i=1}^{k} E\left[\mathrm{c}_{ji}\right]\left(\ln f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) + \ln P(C_i)\right)$$

Ожидаемое значение $E\left[\mathrm{c}_{ji}\right]$ может быть вычислено как

$$E\left[\mathrm{c}_{ji}\right] = 1 \cdot P\left(\mathrm{c}_{ji} = 1\middle|\mathbf{x}_j\right) + 0 \cdot P\left(\mathrm{c}_{ji} = 0\middle|\mathbf{x}_j\right) = P\left(\mathrm{c}_{ji} = 1\middle|\mathbf{x}_j\right) = P(C_i|\mathbf{x}_j)$$

$$= \frac{P(C_i|\mathbf{x}_j)P(C_i)}{P\left(\mathbf{x}_j\right)} = \frac{f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)P(C_i)}{f\left(\mathbf{x}_j\right)}$$

$$= w_{ji} \qquad (13.29)$$

Таким образом, на шаге ожидания мы используем значения $\boldsymbol{\theta} = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, P(C_i)\}_{i=1}^{k}$, чтобы оценить апостериорные вероятности или веса w_{ji} для каждой точки для каждого кластера. Используя $E\left[\mathrm{c}_{ji}\right] = w_{ji}$, ожидаемое значение логарифмической функции правдоподобия можно переписать следующим образом:

$$E[ln\ P\ (\mathbf{D}|\boldsymbol{\theta})] = \sum_{j=1}^{n}\sum_{i=1}^{k} w_{ij}\left(\ln f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) + \ln P(C_i)\right) \qquad (13.30)$$

Шаг максимизации

На шаге максимизации мы максимизируем ожидаемое значение логарифмического правдоподобия [ур-е (13.30)]. Беря производную по отношению к $\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ или $P(C_i)$ мы можем не обращать внимания на слагаемые всех остальных кластеров. Производная от ур-я (13.30) в отношении $\boldsymbol{\mu}_i$ задается как:

$$\frac{\partial}{\partial \boldsymbol{\mu}_i} E[ln\ P\ (\mathbf{D}|\boldsymbol{\theta})]$$

$$= \frac{\partial}{\partial \boldsymbol{\mu}_i}\sum_{j=1}^{n} w_{ij} \ln f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)$$

$$= \sum_{j=1}^{n} w_{ij}\frac{1}{f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)} f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right){\boldsymbol{\Sigma}_i}^{-1}\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)$$

$$= \sum_{j=1}^{n} w_{ij}{\boldsymbol{\Sigma}_i}^{-1}\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)$$

где мы использовали наблюдение, что

$$\frac{\partial}{\partial \boldsymbol{\mu}_i} f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) = f\left(\mathbf{x}_j\middle|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right){\boldsymbol{\Sigma}_i}^{-1}\left(\mathbf{x}_j - \boldsymbol{\mu}_i\right)$$

что следует из ур-й (13.14), (13.17) и (13.18). Устанавливая производную от

ожидаемого значения логарифмического правдоподобия в нулевой вектор и умножая с обеих сторон на $\boldsymbol{\Sigma}$, мы получаем

$$\boldsymbol{\mu}_i = \frac{\sum_{i=1}^{n} w_{ij}\mathbf{x}_j}{\sum_{i=1}^{n} w_{ji}}$$

что соответствует формуле в ур-и (13.11).

Используя уравнения (13.22) и (13.14), получаем производную от ур-я (13.30) от $\boldsymbol{\Sigma}_i{}^{-1}$ следующим образом:

$$\frac{\partial}{\partial \boldsymbol{\Sigma}_i{}^{-1}} \ln E[P(\mathbf{D}|\boldsymbol{\theta})]$$

$$= \sum_{i=1}^{n} w_{ij} \cdot \frac{1}{f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} \cdot \frac{1}{2} f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \left(\boldsymbol{\Sigma}_i - (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T\right)$$

$$= \frac{1}{2} \sum_{i=1}^{n} w_{ij} \cdot \left(\boldsymbol{\Sigma}_i - (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T\right)$$

Устанавливая производную на нулевую матрицу $d \times d$ и решеная для $\boldsymbol{\Sigma}_i$:

$$\boldsymbol{\Sigma}_i = \frac{\sum_{i=1}^{n} w_{ij} \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{i=1}^{n} w_{ij}}$$

это то же самое, что и в ур-и (13.12).

Используя множитель Лагранжа α для ограничения $\sum_{i=1}^{k} P(C_i) = 1$, и отмечая, что в логарифмической функции правдоподобия [ур-е. (13.30)], слагаемое $\ln f(\mathbf{x}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ – постоянная величина, дифференцируя по $P(C_i)$, мы получаем следующее:

$$\frac{\partial}{\partial P(C_i)} \left(\ln E[P(\mathbf{D}|\boldsymbol{\theta})] + \alpha \left(\sum_{a=1}^{k} P(C_a) - 1 \right) \right) = \frac{\partial}{\partial P(C_i)} \left(w_{ij} \ln P(C_i) + \alpha P(C_i) \right)$$

$$= \left(\sum_{j=1}^{n} w_{ij} \frac{1}{P(C_i)} \right) + \alpha$$

Устанавливая производную в ноль, получим:

$$\sum_{j=1}^{n} w_{ij} = -\alpha P(C_i)$$

Используя тот же вывод, что и в ур-и (13.26) получаем

$$P(C_i) = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

что идентично формуле переоценки в ур-и (13.13).

Глава 14 Иерархическая кластеризация

Для n точек в d-мерном пространстве, цель иерархической кластеризации – создать последовательность вложенных разделов, которую можно удобно визуализировать с помощью дерева или иерархии кластеров, также называемой дендрограммой кластеров. Кластеры в иерархии варьируются от слабо детализированных до сильно детализированных – нижний уровень дерева (листья) состоит из каждой точки в своем собственном кластере, тогда как самый высокий уровень (корень) состоит из всех точек в одном кластере. Обе эти группы можно рассматривать как тривиальные кластеры. На каком-то промежуточном уровне мы можем найти значимые кластеры. Если пользователь предоставляет k, желаемое количество кластеров, мы можем выбрать уровень, на котором будет находиться k кластеров.

14.1 Исходные данные

Для датасета $\mathbf{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, где $\mathbf{x}_i \in \mathbb{R}^d$, кластеризация $C = \{C_1, C_2, \ldots, C_k\}$ это разделение $\mathbf{D}$, при котором каждый кластер представляет собой набор точек $C_i \subseteq \mathbf{D}$ такой, что кластеры попарно не пересекаются $C_i \cap C_j = \emptyset$(для всех $i \neq j$), и $\bigcup_{i=1}^{k} C_i = \mathbf{D}$. Кластеризация $\mathcal{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_r\}$ считается вложенной в другую кластеризацию $\mathcal{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_s\}$, тогда и только тогда, когда $r > s$ и для каждого $\mathcal{A}_i \in \mathcal{A}$ существует кластер $\mathcal{B}_j \in \mathcal{B}$ такой, что $\mathcal{A}_i \subseteq \mathcal{B}_j$. Иерархическая кластеризация дает последовательность из n вложенных разделов, начиная от тривиальной кластеризации $C_1 = \{\{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_n\}\}$, где каждая точка находится в отдельном кластере по отношении к другой тривиальной кластеризации $C_n = \{\{\mathbf{x}_1\}, \ldots, \{\mathbf{x}_n\}\}$, где все точки располагаются в одном кластере. В общем случае, кластеризация C_{t-1} вложена в кластеризацию C_t. Дендрограмма кластеризации – это бинарное дерево с корнем, которое фиксирует эту вложенную структуру с ребрами между кластером $C_i \in C_{t-1}$ и кластером $C_j \in C_t$, если C_i вложен в C_j, то есть, если $C_i \subset C_j$. Таким образом, дендрограмма фиксирует всю последовательность вложенных кластеров.

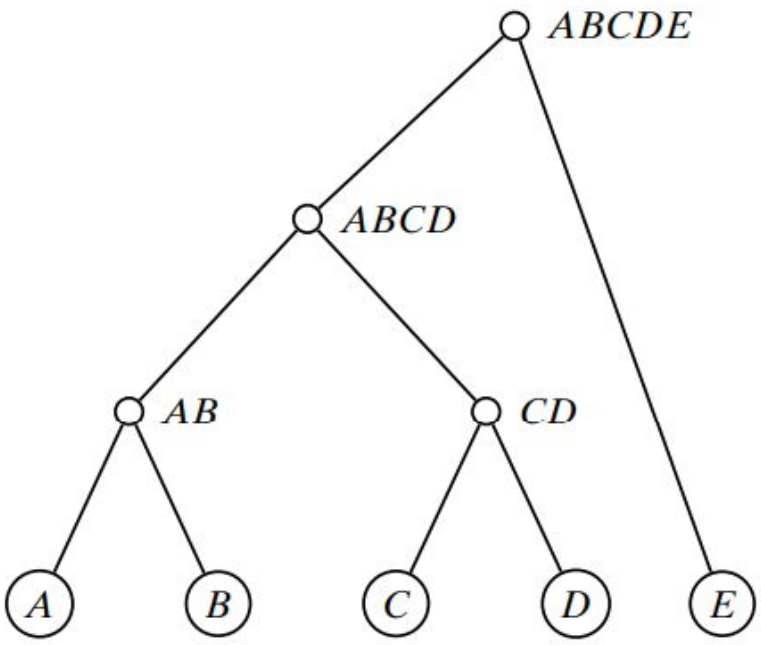


Рисунок 14.1 – Дендрограмма иерархической кластеризации

Пример 14.1. На рисунке 14.1 показан пример иерархической кластеризации пяти отмеченных точек: *A, B, C, D* и *E*. Дендрограмма представляет следующую последовательность вложенных разделов:

Кластеризация	**Кластеры**
C_1	$\{A\},\{B\},\{C\},\{\mathrm{D}\},\{E\}$
C_2	$\{AB\},\{C\},\{\mathrm{D}\},\{E\}$
C_3	$\{AB\},\{C\mathrm{D}\},\{E\}$
C_4	$\{ABC\mathrm{D}\},\{E\}$
C_5	$\{ABC\mathrm{D}E\}$

где $C_i \subset C_{t-1}$ для $t = 2, \ldots, 5$. Мы предполагаем, что *A* и *B* объединены до *C* и *D*.

Количество иерархических кластеризаций

Количество различных вложенных или иерархических кластеризаций соответствует количеству различных двоичных корневых деревьев или дендрограмм с *n* листами с различными метками. Любое дерево с *t* вершинами имеет $t - 1$ ребро. Кроме того, любое корневое двоичное дерево с *m* листьями имеет $m - 1$ внутренних узлов. Таким образом, дендрограмма с *m* листовыми узлами имеет всего $t = m + m - 1 = 2m - 1$ узлов и, следовательно, $t - 1 = 2m - 2$ ребер. Чтобы подсчитать количество различных топологий дендрограммы, давайте рассмотрим, как мы можем расширить дендрограмму с *m* листами, добавив дополнительный лист, чтобы получить дендрограмму с $m + 1$ листом. Обратите внимание, что мы можем добавить дополнительный лист, разделив (т. е. создав ответвление) любое из $2m - 2$ ребер. Кроме того, мы также можем добавить новый лист в качестве потомка нового корня, получая $2m - 2 + 1 = 2m - 1$ новых дендрограмм с $m + 1$ листом. Таким образом, общее количество различных дендрограмм с *n* листами получается с помощью следующего произведения:

$$\prod_{m=1}^{n-1}(2m-1) = 1 \times 3 \times 5 \times 7 \times \ldots \times (2n-3) = (2n-3)!! \qquad (14.1)$$

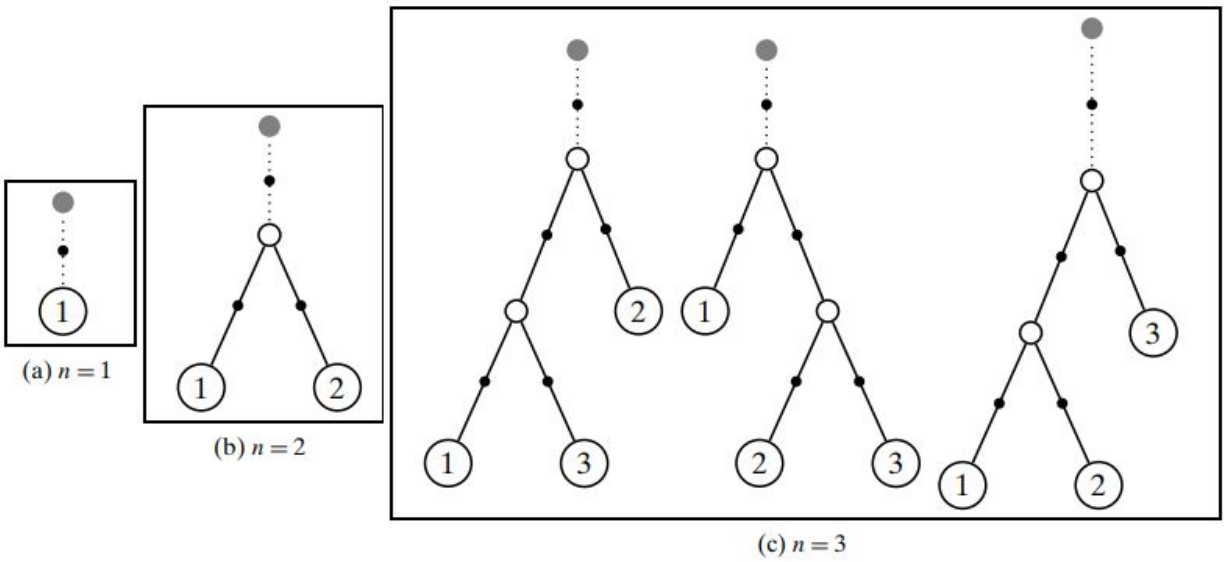


Рисунок 14.2 – Количество иерархических кластеризаций

Индекс m в формуле (14.1) увеличивается до $n - 1$, потому что последний член в произведении обозначает количество дендрограмм, которые можно получить, когда мы расширяем дендрограмму с $n - 1$ листом, добавляя еще один лист, чтобы получить дендрограммы с *n* листами.

Таким образом, количество возможных иерархических кластеризаций задается как $(2n - 3)!!$, что дает очень быстрый рост. Очевидно, что наивный подход к перечислению всех возможных иерархических кластеризаций просто невозможен.

Пример 14.2. На рисунке 14.2 показано количество деревьев с одним, двумя и тремя листьями. Серые узлы – это виртуальные корни, а черные точки указывают места, где можно добавить новый лист. Возможно только одно дерево с одним листом, как показано на рисунке 14.2a. Его можно расширить только одним способом, чтобы получить уникальное дерево с двумя листьями, показанное на рис. 14.2b. Однако у этого дерева есть три возможных места, где можно добавить третий лист. Каждый из этих случаев показан на рисунке 14.2c. Далее мы можем видеть, что каждое из деревьев с m = 3 листьями имеет пять мест, где можно добавить четвертый лист и так далее, что подтверждает уравнение для количества иерархических кластеризаций в уравнении (14.1).

14.2 Агломеративная иерархическая кластеризация

В агломеративной иерархической кластеризации мы начинаем с каждой из n точек в отдельном кластере. Мы многократно объединяем два ближайших кластера, пока все точки не станут членами одного кластера, как показано в псевдокоде, приведенном в алгоритме 14.1. Формально, имея набор кластеров $C = \{C_1, C_2, \ldots, C_m\}$, мы находим ближайшую пару кластеров C_i и C_j и объединяем их в новый кластер $C_{ij} = C_i \cup C_j$. Затем мы обновляем набор кластеров, удаляя C_i и C_j и добавляя C_{ij}, таким образом $\mathrm{C} = \big(\mathrm{C} \setminus \{C_i, C_j\}\big) \cup \{C_{ij}\}$. Мы повторяем процесс до тех пор, пока C не будет содержать только один кластер. Поскольку количество кластеров уменьшается на один на каждом этапе, этот процесс приводит к последовательности n вложенных кластеров. Если надо, мы можем остановить процесс слияния, когда останется ровно k кластеров.

ALGORITHM 14.1. Agglomerative Hierarchical Clustering Algorithm

```
AGGLOMERATIVECLUSTERING(D, k):
1 C ← {C_i = {x_i} | x_i ∈ D} // Each point in separate cluster
2 Δ ← {δ(x_i, x_j): x_i, x_j ∈ D} // Compute distance matrix
3 repeat
4     Find the closest pair of clusters C_i, C_j ∈ C
5     C_ij ← C_i ∪ C_j // Merge the clusters
6     C ← (C \ {C_i, C_j}) ∪ {C_ij} // Update the clustering
7     Update distance matrix Δ to reflect new clustering
8 until |C| = k
```

Расстояние между кластерами

Основным этапом алгоритма является определение ближайшей пары кластеров. Некоторые меры расстояния, такие как одиночная связь, полная связь, среднее по группе и другие, обсуждаемые в следующих параграфах, могут использоваться для вычисления расстояния между любыми двумя кластерами. Расстояние между кластерами в конечном итоге основано на расстоянии между двумя точками, которое обычно вычисляется с использованием евклидова расстояния или L_2-нормы

$$\delta(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \left(\sum_{i=1}^{d} (x_i - y_i)^2\right)^{1/2}$$

Тем не менее, можно использовать другие метрики расстояния или, если она доступна, матрицу расстояний, задаваемую пользователем.

Метод одиночной связи

Для двух кластеров C_i и C_j расстояние между ними, обозначаемое $\delta(C_i, C_j)$, определяется как минимальное расстояние между точкой в C_i и точкой в C_j.

$$\delta(C_i, C_j) = \min\{\delta(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in C_i, \boldsymbol{y} \in C_j\}$$

Название *одиночная связь* происходит из наблюдения, что если мы выберем минимальное расстояние между точками в двух кластерах и соединим эти точки, то (как правило) между этими кластерами будет существовать только одна связь, потому что все другие пары точек будут дальше.

Метод полной связи

Расстояние между двумя кластерами определяется как максимальное расстояние между точкой в C_i и точкой в C_j:

$$\delta(C_i, C_j) = \max\{\delta(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in C_i, \boldsymbol{y} \in C_j\}$$

Название *полная связь* передает тот факт, что если мы соединим все пары точек из двух кластеров с расстоянием не более $\delta(C_i, C_j)$, то все возможные пары будут соединены, то есть мы получим полную связь.

Метод средней связи

Расстояние между двумя кластерами определяется как среднее попарное расстояние между точкой в C_i и точкой в C_j:

$$\delta(C_i, C_j) = \frac{\sum_{\boldsymbol{x} \in C_i} \sum_{\boldsymbol{y} \in C_i} \delta(\boldsymbol{x}, \boldsymbol{y})}{n_i \cdot n_j}$$

Где $n_i = |C_i|$ обозначает количество точек в кластере C_i.

Центроидный метод

Расстояние между двумя кластерами определяется как расстояние между средними или центроидами двух кластеров:

$$\delta(C_i, C_j) = \delta(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) \tag{14.2}$$

где $\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$.

Минимальная дисперсия: метод Уорда

Расстояние между двумя кластерами определяется как увеличение суммы квадратов ошибок (SSE) при объединении двух кластеров. SSE для данного кластера C_i задается как:

$$SSE_i = \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

Которая также может быть записана как

$$SSE_i = \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

$$= \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}^T \boldsymbol{x} - 2 \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}^T \boldsymbol{\mu}_i + \sum_{\boldsymbol{x} \in C_i} \boldsymbol{\mu}_i{}^T \boldsymbol{\mu}_i$$

$$= \left(\sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}^T \boldsymbol{x} \right) - n_i \boldsymbol{\mu}_i{}^T \boldsymbol{\mu}_i \tag{14.3}$$

SSE для кластеризации $C = \{C_1, \ldots, C_m\}$ задается как

$$SSE = \sum_{i=1}^{m} SSE_i = \sum_{i=1}^{m} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

Мера Уорда определяет расстояние между двумя кластерами C_i и C_j как чистое изменение значения SSE, когда мы объединяем C_i и C_j в C_{ij}

$$\delta(C_i, C_j) = \Delta SSE_{ij} = SSE_{ij} - SSE_i - SSE_j \tag{14.4}$$

Мы можем получить более простое выражение для меры Уорда, подставив уравнение (14.3) в уравнение (14.4) и отмечая, что, поскольку $C_{ij} = C_i \cup C_j$ и $C_i \cap C_j = \emptyset$, мы имеем $|C_{ij}| = n_{ij} = n_i + n_j$, и поэтому

$$\begin{aligned}
&\delta(C_i, C_j) = \Delta SSE_{ij} \\
&= \sum_{\boldsymbol{z} \in C_{ij}} \|\boldsymbol{z} - \boldsymbol{\mu}_{ij}\|^2 - \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2 - \sum_{\mathbf{y} \in C_j} \|\mathbf{y} - \boldsymbol{\mu}_j\|^2 \\
&= \sum_{\boldsymbol{z} \in C_{ij}} \boldsymbol{z}^T \boldsymbol{z} - n_{ij} \boldsymbol{\mu}_{ij}^T \boldsymbol{\mu}_{ij} - \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}^T \boldsymbol{x} + n_i \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i - \sum_{\mathbf{y} \in C_j} \mathbf{y}^T \mathbf{y} + n_j \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j \\
&= n_i \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i + n_j \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j - (n_i + n_j) \boldsymbol{\mu}_{ij}^T \boldsymbol{\mu}_{ij} \qquad (14.5)
\end{aligned}$$

Последний шаг следует из того, что $\sum_{\boldsymbol{z} \in C_{ij}} \boldsymbol{z}^T \boldsymbol{z} = \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}^T \boldsymbol{x} + \sum_{\mathbf{y} \in C_j} \mathbf{y}^T \mathbf{y}$. Отмечая, что

$$\boldsymbol{\mu}_{ij} = \frac{n_i \boldsymbol{\mu}_i + n_j \boldsymbol{\mu}_j}{n_i + n_j}$$

мы получаем

$$\boldsymbol{\mu}_{ij}^T \boldsymbol{\mu}_{ij} = \frac{1}{(n_i + n_j)^2} \left(n_i^2 \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i + 2 n_i n_j \boldsymbol{\mu}_i^T \boldsymbol{\mu}_j + n_j^2 \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j \right)$$

Подставив вышеизложенное в уравнение (14.5) окончательно получаем

$$\begin{aligned}
&\delta(C_i, C_j) = \Delta SSE_{ij} \\
&= n_i \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i + n_j \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j - \frac{1}{(n_i + n_j)} \left(n_i^2 \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i + 2 n_i n_j \boldsymbol{\mu}_i^T \boldsymbol{\mu}_j + n_j^2 \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j \right) \\
&= \frac{n_i (n_i + n_j) \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i + n_j (n_i + n_j) \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j - n_i^2 \boldsymbol{\mu}_i^T \boldsymbol{\mu}_i - 2 n_i n_j \boldsymbol{\mu}_i^T \boldsymbol{\mu}_j - n_j^2 \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j}{n_i + n_j} \\
&= \frac{n_i n_j (\boldsymbol{\mu}_i^T \boldsymbol{\mu}_i - 2 \boldsymbol{\mu}_i^T \boldsymbol{\mu}_j + \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j)}{n_i + n_j} \\
&= \left(\frac{n_i n_j}{n_i + n_j} \right) \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2
\end{aligned}$$

Таким образом, мера Уорда является взвешенной версией меры среднего расстояния, потому что, если мы используем евклидово расстояние, среднее расстояние в уравнении (14.2) можно переписать как

$$\delta(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2 \qquad (14.6)$$

Мы можем видеть, что единственная разница в том, что мера Уорда взвешивает расстояние между средними значениями как половину гармонического среднего размера кластеров, где гармоническое среднее двух чисел n_1 и n_2 задано как $\frac{2}{\frac{1}{n_1} + \frac{1}{n_2}} = \frac{2 n_1 n_2}{n_1 + n_2}$.

Обновление матрицы расстояний

Каждый раз, когда два кластера C_i и C_j объединяются в C_{ij}, нам необходимо обновить матрицу расстояний путем пересчета расстояний от вновь созданного кластера C_{ij} до всех других кластеров C_r ($r \neq i$ и $r \neq j$). Формула Ланса-Вильямса дает общее уравнение для пересчета расстояний для всех мер кластерной близости, которые мы рассматривали ранее:

$$\delta(C_{ij}, C_r) = \alpha_i \cdot \delta(C_i, C_r) + \alpha_j \cdot \delta(C_j, C_r) + \beta \cdot \delta(C_i, C_j) + \gamma \cdot |\delta(C_i, C_r) - \delta(C_j, C_r)| \quad (14.7)$$

Коэффициенты α_i, α_j, β и γ различаются от одной меры к другой. Пусть $n_i = |C_i|$ обозначает мощность кластера C_i; тогда коэффициенты для различных мер расстояния будут такими, как показано в таблице 14.1.

Table 14.1. Lance–Williams formula for cluster proximity

Measure	α_i	α_j	β	γ
Single link	$\frac{1}{2}$	$\frac{1}{2}$	0	$-\frac{1}{2}$
Complete link	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$
Group average	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	0	0
Mean distance	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$\frac{-n_i \cdot n_j}{(n_i+n_j)^2}$	0
Ward's measure	$\frac{n_i+n_r}{n_i+n_j+n_r}$	$\frac{n_j+n_r}{n_i+n_j+n_r}$	$\frac{-n_r}{n_i+n_j+n_r}$	0

Вычислительная сложность

В агломеративной кластеризации нам необходимо вычислить расстояние от каждого кластера до всех остальных кластеров, и на каждом шаге количество кластеров уменьшается на 1. Первоначально для создания матрицы попарного расстояния требуется $O(n^2)$ времени, если только оно не указано как вход в алгоритм.

На каждом этапе слияния расстояния от объединенного кластера до других кластеров должны быть пересчитаны, тогда как расстояния между другими кластерами остаются прежними. Это означает, что на шаге t мы вычисляем $O(n-t)$ расстояний. Другая основная операция – найти ближайшую пару в матрице расстояний. Для этого мы можем сохранить n^2 расстояний в структуре данных куча, что позволяет нам найти минимальное расстояние за время $O(1)$; создание кучи занимает время $O(n^2)$. Удаление/обновление расстояний из кучи занимает время $O(\log n)$ для каждой операции, что составляет общее время на всех этапах слияния $O(n^2 \log n)$. Таким образом, вычислительная сложность иерархической кластеризации составляет $O(n^2 \log n)$.