Кластеризация и алгоритм EM

Сергей Николенко

Академический Университет, весенний семестр 2011

Outline

Суть лекции

- *Кластеризация* — типичная задача статистического анализа: задача классификации объектов одной природы в несколько групп так, чтобы объекты в одной группе обладали одним и тем же свойством.
- Под свойством обычно понимается близость друг к другу относительно выбранной метрики.

Чуть более формально

- Есть набор тестовых примеров $X = \{x_1, \ldots, x_n\}$ и функция расстояния между примерами ρ.
- Требуется разбить X на непересекающиеся подмножества (кластеры) так, чтобы каждое подмножество состояло из похожих объектов, а объекты разных подмножеств существенно различались.

Идея

- Есть точки $x_1, x_2, \ldots, x_n$ в пространстве. Нужно кластеризовать.
- Считаем каждую точку кластером. Затем ближайшие точки объединяем, далее считаем единым кластером. Затем повторяем.
- Получается дерево.

Алгоритм

`HierarchyCluster`$(X = \{x_1, \ldots, x_n\})$

- Инициализируем $C = X$, $G = X$.
- Пока в C больше одного элемента:
  - Выбираем два элемента C c_1 и c_2, расстояние между которыми минимально.
  - Добавляем в G вершину $c_1 c_2$, соединяем её с вершинами c_1 и c_2.
  - $C := C \cup \{c_1 c_2\} \setminus \{c_1, c_2\}$.
- Выдаём G.

Результат

- В итоге получается дерево кластеров, из которого потом можно выбрать кластеризацию с требуемой степенью детализации (обрезать на том или ином максимальном расстоянии).
- Всё ли понятно?

Результат

- В итоге получается дерево кластеров, из которого потом можно выбрать кластеризацию с требуемой степенью детализации (обрезать на том или ином максимальном расстоянии).
- Всё ли понятно?
- Остаётся вопрос: как подсчитывать расстояние между кластерами?

Single-link vs. complete-link

- *Single-link* алгоритмы считают *минимум* из возможных расстояний между парами объектов, находящихся в кластере.
- *Complete-link* алгоритмы считают *максимум* из этих расстояний
- Какие особенности будут у single-link и complete-link алгоритмов? Чем они будут отличаться?

Упражнение

Упражнение. Реализовать single-link и complete-link алгоритмы агломеративной кластеризации для точек из евклидова пространства размерности n.

Очевидный алгоритм

- Нарисуем полный граф с весами, равными расстоянию между объектами.
- Выберем лимит расстояния r и выбросим все рёбра длиннее r.
- Компоненты связности полученного графа — это наши кластеры.

Минимальное остовное дерево

- Минимальное остовное дерево — дерево, содержащее все вершины (связного) графа и имеющее минимальный суммарный вес своих рёбер.
- Алгоритм Краскала (Kruskal): выбираем на каждом шаге ребро с минимальным весом, если оно соединяет два дерева, добавляем, если нет, пропускаем.
- Алгоритм Борувки (Boruvka).

Кластеризация

- Как использовать минимальное остовное дерево для кластеризации?

Кластеризация

- Как использовать минимальное остовное дерево для кластеризации?
- Построить минимальное остовное дерево, а потом выкидывать из него рёбра максимального веса.
- Сколько рёбер выбросим, столько кластеров получим.

Outline

Постановка задачи

- Часто возникает ситуация, когда в имеющихся данных некоторые переменные присутствуют, а некоторые — отсутствуют.
- Даны результаты сэмплирования распределения вероятностей с несколькими параметрами, из которых известны не все.

Постановка задачи

- Эти неизвестные параметры тоже расцениваются как случайные величины.
- Задача — найти наиболее вероятную гипотезу, то есть ту гипотезу h, которая максимизирует

$$E[\ln p(D|h)].$$

Частный случай

Построим один из простейших примеров применения алгоритма EM. Пусть случайная переменная x сэмплируется из суммы двух нормальных распределений. Дисперсии даны (одинаковые), нужно найти только средние μ_1, μ_2.

Два распределения

- Теперь нельзя понять, какие x_i были порождены каким распределением — классический пример *скрытых переменных*.
- Один тестовый пример полностью описывается как тройка $\langle x_i, z_{i1}, z_{i2} \rangle$, где $z_{ij} = 1$ iff x_i был сгенерирован j-м распределением.

Суть алгоритма EM

- Сгенерировать какую-нибудь гипотезу $h = (\mu_1, \mu_2)$.
- Пока не дойдем до локального максимума:
  - Вычислить ожидание $E(z_{ij})$ в предположении текущей гипотезы (E–шаг).
  - Вычислить новую гипотезу $h' = (\mu_1', \mu_2')$, предполагая, что z_{ij} принимают значения $E(z_{ij})$ (M–шаг).

В примере с гауссианами

В примере с гауссианами:

$$E(z_{ij}) = \frac{p(x = x_i|\mu = \mu_j)}{p(x = x_i|\mu = \mu_1) + p(x = x_i|\mu = \mu_2)} = \\ = \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{e^{-\frac{1}{2\sigma^2}(x_i - \mu_1)^2} + e^{-\frac{1}{2\sigma^2}(x_i - \mu_2)^2}}.$$

Мы подсчитываем эти ожидания, а потом подправляем гипотезу:

$$\mu_j \leftarrow \frac{1}{m}\sum_{i=1}^{m} E(z_{ij})x_i.$$

Обоснование алгоритма EM

- Дадим формальное обоснование алгоритма EM.
- Мы решаем задачу максимизации правдоподобия по данным $\mathcal{X} = \{x_1, \ldots, x_N\}$.

$$L(\theta \mid \mathcal{X}) = p(\mathcal{X} \mid \theta) = \prod p(x_i \mid \theta)$$

  или, что то же самое, максимизации $\ell(\theta \mid \mathcal{X}) = \log L(\theta \mid \mathcal{X})$.
- EM может помочь, если этот максимум трудно найти аналитически.

Обоснование алгоритма EM

- Давайте предположим, что в данных есть *скрытые компоненты*, такие, что если бы мы их знали, задача была бы проще.
- Замечание: совершенно не обязательно эти компоненты должны иметь какой-то физический смысл. :) Может быть, так просто удобнее.
- В любом случае, получается набор данных $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ с совместной плотностью

$$p(z \mid \theta) = p(x, y \mid \theta) = p(y \mid x, \theta) p(x \mid \theta).$$

- Получается полное правдоподобие $L(\theta \mid \mathcal{Z}) = p(\mathcal{X}, \mathcal{Y} \mid \theta)$. Это случайная величина (т.к. $\mathcal{Y}$ неизвестно).

Обоснование алгоритма EM

- Заметим, что настоящее правдоподобие $L(\theta) = E_Y\left[p(\mathcal{X}, \mathcal{Y} \mid \theta) \mid \mathcal{X}, \theta\right]$.
- E-шаг алгоритма EM вычисляет условное ожидание (логарифма) полного правдоподобия при условии $\mathcal{X}$ и текущих оценок параметров θ_n:

$$Q(\theta, \theta_n) = E\left[\log p(\mathcal{X}, \mathcal{Y} \mid \theta) \mid \mathcal{X}, \theta_n\right].$$

- Здесь θ_n – текущие оценки, а θ – неизвестные значения (которые мы хотим получить в конечном счёте); т.е. $Q(\theta, \theta_n)$ – это функция от θ.

Обоснование алгоритма EM

- E-шаг алгоритма EM вычисляет условное ожидание (логарифма) полного правдоподобия при условии $\mathcal{X}$ и текущих оценок параметров θ:

$$Q(\theta, \theta_n) = E\left[\log p(\mathcal{X}, \mathcal{Y} \mid \theta) \mid \mathcal{X}, \theta_n\right].$$

- Условное ожидание – это

$$E\left[\log p(\mathcal{X}, \mathcal{Y} \mid \theta) \mid \mathcal{X}, \theta_n\right] = \int_y \log p(\mathcal{X}, y \mid \theta) p(y \mid \mathcal{X}, \theta_n) \mathrm{d}y,$$

где $p(y \mid \mathcal{X}, \theta_n)$ – маргинальное распределение скрытых компонентов данных.

- EM лучше всего применять, когда это выражение легко подсчитать, может быть, даже аналитически.
- Вместо $p(y \mid \mathcal{X}, \theta_n)$ можно подставить $p(y, \mathcal{X} \mid \theta_n) = p(y \mid \mathcal{X}, \theta_n) p(\mathcal{X} \mid \theta_n)$, от этого ничего не изменится.

Обоснование алгоритма EM

- В итоге после E-шага алгоритма EM мы получаем функцию $Q(\theta, \theta_n)$.
- На M-шаге мы максимизируем

$$\theta_{n+1} = \arg\max_{\theta} Q(\theta, \theta_n).$$

- Затем повторяем процедуру до сходимости.
- В принципе, достаточно просто находить θ_{n+1}, для которого $Q(\theta_{n+1}, \theta_n) > Q(\theta_n, \theta_n)$ – Generalized EM.
- Осталось понять, что значит $Q(\theta, \theta_n)$ и почему всё это работает.

Обоснование алгоритма EM

- Мы хотели перейти от θ_n к θ, для которого $\ell(\theta) > \ell(\theta_n)$.

$$\ell(\theta) - \ell(\theta_n) =$$

$$= \log\left(\int_y p(\mathcal{X} \mid y, \theta) p(y \mid \theta) \mathrm{d}y\right) - \log p(\mathcal{X} \mid \theta_n) =$$

$$= \log\left(\int_y p(y \mid \mathcal{X}, \theta_n) \frac{p(\mathcal{X} \mid y, \theta) p(y \mid \theta)}{p(y \mid \mathcal{X}, \theta_n)} \mathrm{d}y\right) - \log p(\mathcal{X} \mid \theta_n) \geq$$

$$\geq \int_y p(y \mid \mathcal{X}, \theta_n) \log\left(\frac{p(\mathcal{X} \mid y, \theta) p(y \mid \theta)}{p(y \mid \mathcal{X}, \theta_n)}\right) \mathrm{d}y - \log p(\mathcal{X} \mid \theta_n) =$$

$$= \int_y p(y \mid \mathcal{X}, \theta_n) \log\left(\frac{p(\mathcal{X} \mid y, \theta) p(y \mid \theta)}{p(\mathcal{X} \mid \theta_n) p(y \mid \mathcal{X}, \theta_n)}\right) \mathrm{d}y.$$

Обоснование алгоритма EM

- Получили

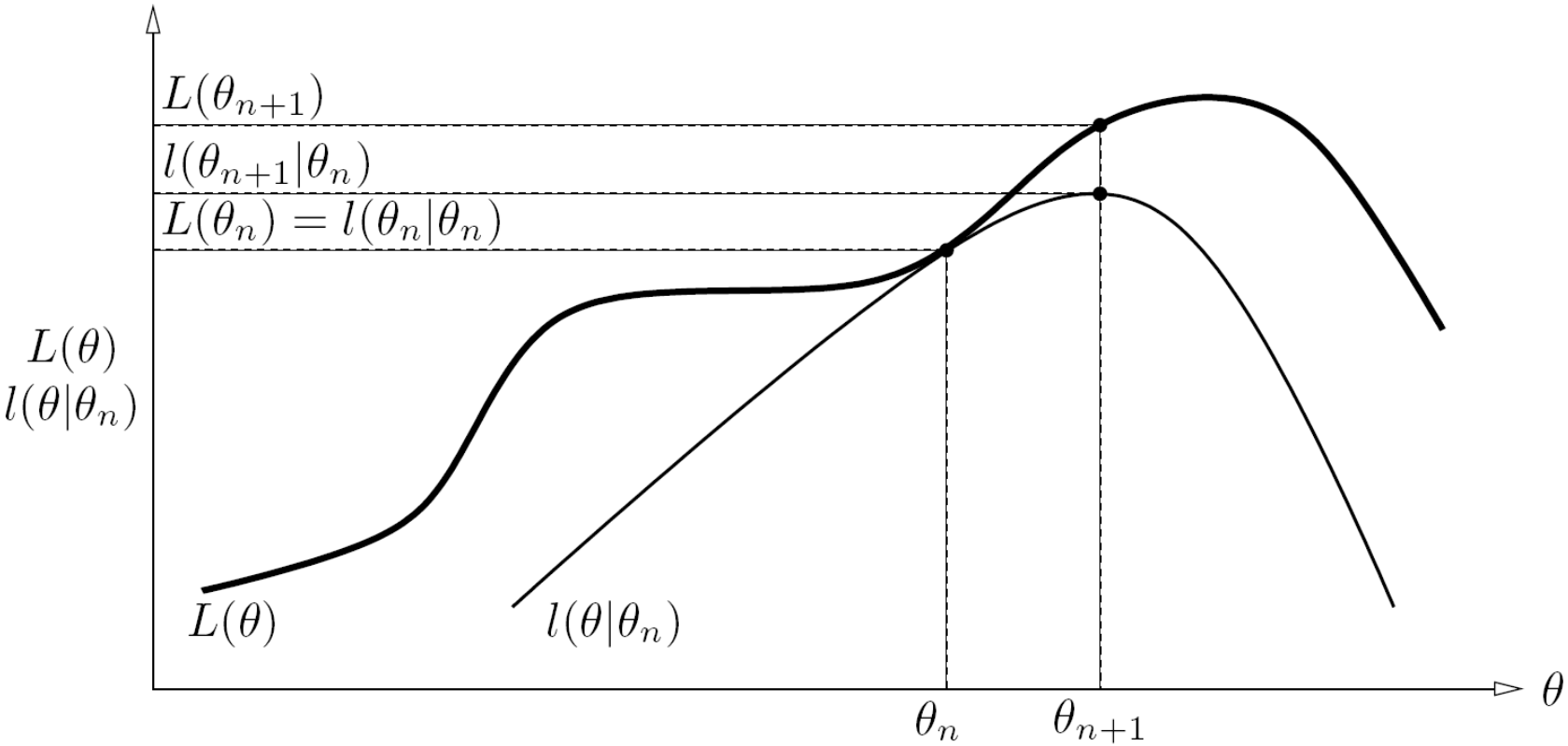
$$\ell(\theta) \geq I(\theta, \theta_n) =$$
$$= \ell(\theta_n) + \int_y p(y \mid \mathcal{X}, \theta_n) \log \left(\frac{p(\mathcal{X} \mid y, \theta) p(y \mid \theta)}{p(\mathcal{X} \mid \theta_n) p(y \mid \mathcal{X}, \theta_n)} \right) \mathrm{d}y.$$

Упражнение. Докажите, что $I(\theta_n, \theta_n) = \ell(\theta_n)$.

Обоснование алгоритма EM

- Иначе говоря, мы нашли нижнюю оценку на $\ell(\theta)$ везде, касание происходит в точке θ_n.
- Т.е. мы нашли нижнюю оценку для правдоподобия и смещаемся в точку, где она максимальна (или хотя бы больше текущей).
- Такая общая схема называется *MM-алгоритм* (minorization-maximization). Мы к ним, возможно, ещё вернёмся.

Обоснование алгоритма EM



Обоснование алгоритма EM

- Осталось только понять, что максимизировать можно Q.

$$\theta_{n+1} = \arg\max_\theta I(\theta, \theta_n) = \arg\max_\theta \{\ell(\theta_n) + \\ + \int_y f(y \mid \mathcal{X}, \theta_n) \log \left(\frac{p(\mathcal{X} \mid y, \theta) f(y \mid \theta)}{p(\mathcal{X} \mid \theta_n) f(y \mid \mathcal{X}, \theta_n)} \right) \mathrm{d}y \Big\} =$$

$$= \arg\max_\theta \left\{ \int_y p(y \mid \mathcal{X}, \theta_n) \log \left(p(\mathcal{X} \mid y, \theta) p(y \mid \theta) \right) \mathrm{d}y \right\} =$$

$$= \arg\max_\theta \left\{ \int_y p(y \mid \mathcal{X}, \theta_n) \log p(\mathcal{X}, y \mid \theta) \mathrm{d}y \right\} =$$

$$= \arg\max_\theta \{Q(\theta, \theta_n)\},$$

а остальное от θ не зависит. Вот и получился EM.

Мысли?

- Какие есть мысли о применении алгоритма EM к задачам кластеризации?

Гипотезы

- Чтобы воспользоваться статистическим алгоритмом, нужно сформулировать гипотезы о распределении данных.
- *Гипотеза о природе данных*: тестовые примеры появляются случайно и независимо, согласно вероятностному распределению, равному смеси распределений кластеров

$$p(x) = \sum_{c \in C} w_c p_c(x), \quad \sum_{c \in C} w_c = 1,$$

где w_c — вероятность появления объектов из кластера c, p_c — плотность распределения кластера c.

Гипотезы cont'd

- Остается вопрос: какими предположить распределения p_c?

Гипотезы cont'd

- Остается вопрос: какими предположить распределения p_c?
- Часто берут сферические гауссианы, но это не слишком гибкий вариант: кластер может быть вытянут в ту или иную сторону.

Гипотезы cont'd

- Остается вопрос: какими предположить распределения p_c?
- Часто берут сферические гауссианы, но это не слишком гибкий вариант: кластер может быть вытянут в ту или иную сторону.
- Мы будем брать эллиптические гауссианы.
- *Гипотеза* 2: Каждый кластер c описывается d–мерной гауссовской плотностью с центром $\mu_c = \{\mu_{c1}, \ldots, \mu_{cd}\}$ и диагональной матрицей ковариаций $\Sigma_c = \text{diag}(\sigma_{c1}^2, \ldots, \sigma_{c2}^2)$ (т.е. по каждой координате своя дисперсия).

Постановка задачи и общий вид алгоритма

- В этих предположениях получается в точности задача разделения смеси вероятностных распределений. Для этого и нужен EM–алгоритм.
- Каждый тестовый пример описывается своими координатами $(f_1(x), \ldots, f_n(x))$.
- Скрытые переменные в данном случае — вероятности g_{ic} того, что объект x_i принадлежит кластеру $c \in C$.

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

$$g_{ic} = \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}.$$

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

$$g_{ic} = \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}.$$

- M–шаг: с использованием g_{ic} уточняются параметры кластеров w, μ, σ:

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

$$g_{ic} = \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}.$$

- M–шаг: с использованием g_{ic} уточняются параметры кластеров w, μ, σ:

$$w_c = \frac{1}{n} \sum_{i=1}^{n} g_{ic},$$

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

$$g_{ic} = \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}.$$

- M–шаг: с использованием g_{ic} уточняются параметры кластеров w, μ, σ:

$$w_c = \frac{1}{n} \sum_{i=1}^{n} g_{ic}, \quad \mu_{cj} = \frac{1}{n w_c} \sum_{i=1}^{n} g_{ic} f_j(x_i),$$

Идея алгоритма

- E–шаг: по формуле Байеса вычисляются скрытые переменные g_{ic}:

$$g_{ic} = \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}.$$

- M–шаг: с использованием g_{ic} уточняются параметры кластеров w, μ, σ:

$$w_c = \frac{1}{n} \sum_{i=1}^{n} g_{ic}, \quad \mu_{cj} = \frac{1}{n w_c} \sum_{i=1}^{n} g_{ic} f_j(x_i),$$

$$\sigma_{cj}^2 = \frac{1}{n w_c} \sum_{i=1}^{n} g_{ic} \left(f_j(x_i) - \mu_{cj}\right)^2.$$

Алгоритм

EMCluster($X, |C|$):

- Инициализировать $|C|$ кластеров; начальное приближение:
  $w_c := 1/|C|$, $\mu_c :=$ случайный x_i,
  $\sigma^2_{cj} := \frac{1}{n|C|} \sum_{i=1}^{n} (f_j(x_i) - \mu_{cj})^2$.
- Пока принадлежность кластерам не перестанет изменяться:
  - E–шаг: $g_{ic} := \frac{w_c p_c(x_i)}{\sum_{c' \in C} w_{c'} p_{c'}(x_i)}$.
  - M–шаг: $w_c = \frac{1}{n} \sum_{i=1}^{n} g_{ic}$, $\mu_{cj} = \frac{1}{n w_c} \sum_{i=1}^{n} g_{ic} f_j(x_i)$,

$$\sigma^2_{cj} = \frac{1}{n w_c} \sum_{i=1}^{n} g_{ic} (f_j(x_i) - \mu_{cj})^2 .$$

  - Определить принадлежность x_i к кластерам:

$$\text{clust}_i := \arg\max_{c \in C} g_{ic}.$$

Упражнение

Упражнение. Докажите, что E-шаг и M-шаг действительно в данном случае так выглядят.

Проблема

- Остается проблема: нужно задавать количество кластеров.

Суть алгоритма k–средних

- Один из самых известных алгоритмов кластеризации – алгоритм k-средних – это фактически упрощение алгоритма EM.
- Разница в том, что мы не считаем вероятности принадлежности кластерам, а жестко приписываем каждый объект одному кластеру.
- Кроме того, в алгоритме k–средних форма кластеров не настраивается (но это не так важно).

Цель

- Цель алгоритма k–средних — минимизировать меру ошибки

$$E(X, C) = \sum_{i=1}^{n} \|x_i - \mu_i\|^2,$$

где μ_i — ближайший к x_i центр кластера.

- Т.е. мы не относим точки к кластерам, а двигаем центры, а принадлежность точек определяется автоматически.

Алгоритм неформально

- Идея та же, что в EM:
  - Проинициализировать.
  - Классифицировать точки по ближайшему к ним центру кластера.
  - Перевычислить каждый из центров.
  - Если ничего не изменилось, остановиться, если изменилось — повторить.

Алгоритм

$\texttt{kMeans}(X, |C|)$:

- Инициализировать центры $|C|$ кластеров $\mu_1, \ldots, \mu_{|C|}$.
- Пока принадлежность кластерам не перестанет изменяться:
  - Определить принадлежность x_i к кластерам:

$$\text{clust}_i := \arg\min_{c \in C} \rho(x_i, \mu_c).$$

  - Определить новое положение центров кластеров:

$$\mu_c := \frac{\sum_{\text{clust}_i = c} f_j(x_i)}{\sum_{\text{clust}_i = c} 1}.$$

Semi–supervised clustering

- И EM, и k–means хорошо обобщаются на случай частично обученных кластеров.
- То есть про часть точек уже известно, какому кластеру они принадлежат.
- Как это учесть?

Semi–supervised clustering

- Чтобы учесть информацию о точке x_i, достаточно для EM положить скрытую переменную g_{ic} равной тому кластеру, которому нужно, с вероятностью 1, а остальным — с вероятностью 0, и не пересчитывать.
- Для k–means то же самое, но для clust_i.

Thank you!

Спасибо за внимание!