EM-алгоритм (пример)

Описание

EM-алгоритм – общий метод нахождения оценок функции правдоподобия в моделях со скрытыми переменными. В данной статье рассматривается интерпретация смеси гауссовых распределений в терминах дискретных скрытых переменных.

Помимо того, что смеси распределений позволяют строить (приближать) сложные вероятностные распепределения, с их помощью можно так же решать задачу кластеризации данных. Далее мы будем решать задачу кластеризации с помощью EM-алгоритма, предварительно приблизив решение алгоритмом K-Means.

Постановка задачи разделения смеси гауссовых распределений

Задана выборка X^l случайных и независимых наблюдений из смеси $p(x)$, в которой описание $i-ого$ элемента есть вектор $x(i) \in R^n$.
Принята модель, в которой каждая компонента смеси есть гауссина с параметрами μm и Σ, и известно число компонент смеси - K.

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Требуется оценить вектор параметров $\theta = (\pi 1, \ldots, \pi k, \mu 1, \ldots, \mu k, \Sigma 1, \ldots, \Sigma k)$, доставляющий максим функции правдоподобия

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}.$$

Алгоритм отыскания оптимальных параметров

Оптимальные параметры отыскиваются последовательно с помощью итерационного EM-алгоритма. Основная идея – вводится вспомогательный вектор скрытых переменных. Это позволяет свести сложную оптимизационную задачу к последовательности итераций по пересчету коэффициентов (скрытых переменных по текущему приближению вектора параметров - E-шаг) и максимизации правдоподобия (с целью найти следующее приближение вектора - M-шаг.

1) В начале работы алгоритма задаются параметры начального приближения, которые в данной статье получаются в результате работы алгоритма K-Means.
(Данная эвристика применяется, так как K-Means требуется намного меньше итераций до достижения стабилизации, в то время как каждый шаг EM требует больших вычислительных затрат).

Никита Спирин © 2009

2) Далее итеративно выполняется следующая пара процедур:

E-шаг: используя текущее значение вектора параметров
$\theta = (\pi 1, \ldots, \pi k, \mu 1, \ldots, \mu k, \Sigma 1, \ldots, \Sigma k)$, вычисляем значение вектора скрытых переменных γ.

$$\gamma(z_{nk}) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}.$$

M-шаг: переоценка вектора параметров, используя текущее значение вектора скрытых переменных.

$$\begin{aligned}
\boldsymbol{\mu}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \\
\boldsymbol{\Sigma}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \left(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}\right) \left(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}\right)^{\mathrm{T}} \\
\pi_k^{\text{new}} &= \frac{N_k}{N}
\end{aligned}$$

где $N_k = \sum_{n=1}^{N} \gamma(z_{nk}).$

Процедура останавливается после того, как норма разности векторов скрытых переменных на каждой итерации не будет превышать заданную константу

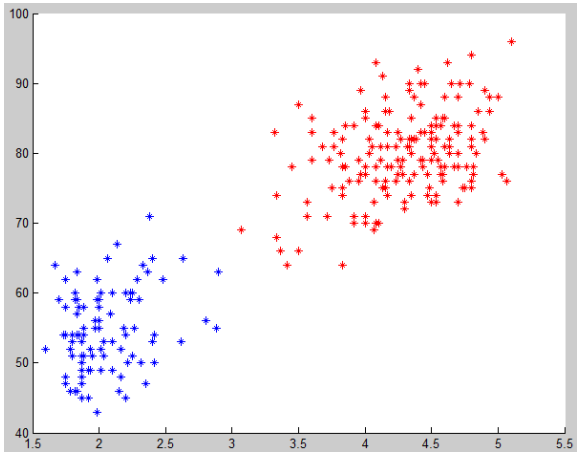
$$\delta_{\max} := \max\left\{\delta_{\max}, |g_{ij} - g_{ij}^0|\right\}; \quad \delta_{\max} > \Delta;$$

Никита Спирин © 2009

Пример на модельных данных

Продемонстрирована работа алгоритма на серии реальных и синтетических данных.

1) $Old\ Faithful$ – гидродинамический гейзер в *Yellowstone National Park*. Данные содержат информацию о 272-х извержениях, каждое из которых есть пара характеризующая длительность извержения и время до следующего извержения (в минутах).Ссылка на данные.



На графике справа показаны имходные данные. По оси x отложено значение одного признака(время извержения в мин.), по оси y – соответственно значение второго признака. Цвет точек означает принадлежность к тому или иному классу.

```
% Load the data from local directory
load 'Old Faithful.txt'
%------------------Invoke the EM-Algorithm
k=2; % set the number of classes
w = ExpMin(Old_Faithful,k,2);
%----------------Plot classified data
h = figure; hold on
idx1 = find(w.y == 1); % object indices for the 1st class
idx2 = find(w.y == 2);
plot(w.x(idx1,1), w.x(idx1,2), 'r*');
plot(w.x(idx2,1), w.x(idx2,2), 'b*');
```

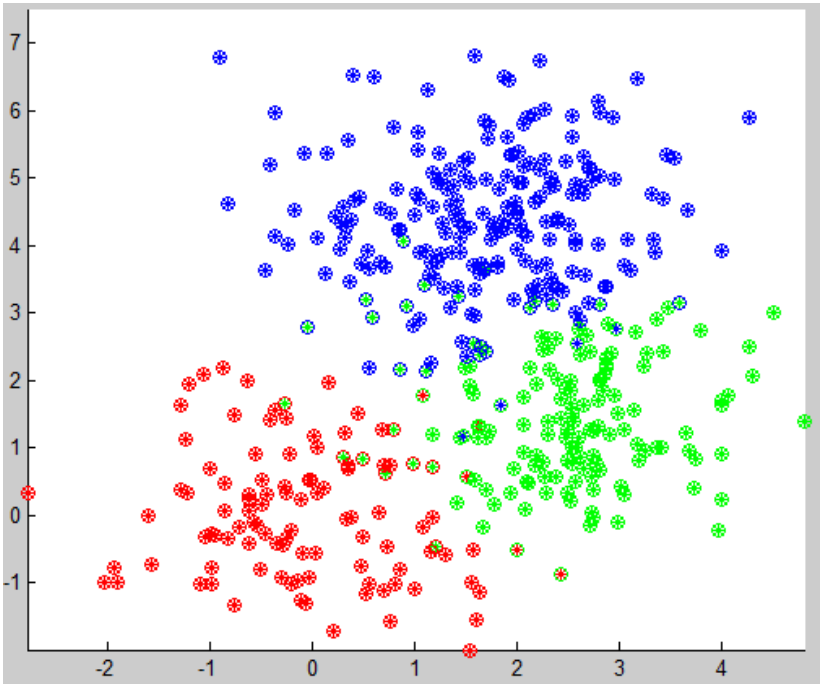
Классы определены алгоритмом абсолютно (ошибок нет), сложность задачи - легкая, так как видно что классы линейно-разделимы и расстояние между центрами классов велико.

2) Синтетические данные (легкий случай, 3 класса)

```
N = 100; % 1st class contains N objects
alpha = 1.5; % 2st class contains alpha*N ones
beta = 2; % 3rd class contains beta*N ones
sig23 = 1; % assume 2nd and 3rd classes have the same variance as the 1st
dist2 = 2.5;
dist3 = 2.0;

N2 = floor(alpha * N); % calculate the size of the 2nd class
N3 = floor(beta * N); % calculate the size of the 2nd class
cls1X = randn(N, 2); % generate random objects of the 1st class

% generate a random distance from the center of the 1st class to the center
% of the 2nd and 3rd
ShiftClass2 = repmat(dist2 * [sin(pi*rand)  cos(pi*rand)],[N2,1]);
ShiftClass3 = repmat(dist3 * [sin(pi*rand)  2*cos(pi*rand)],[N3,1]);
% generate random objects of the 2nd class
cls2X = sig23 * randn(N2, 2) + ShiftClass2;
```

Никита Спирин © 2009

```
cls3X = sig23 * randn(N3, 2) + ShiftClass3;
% combine the objects
X = [cls1X; cls2X; cls3X];
% assign class labels: 1s and 2s and 3s
y = [zeros(size(cls1X,1),1)+1; zeros(size(cls2X,1),1)+2;
zeros(size(cls3X,1),1)+3];
%end

X; y; % note that y contains only 1s and 2s and 3s
idx1 = find(y == 1); % object indices for the 1st class
idx2 = find(y == 2);
idx3 = find(y == 3);

%-------------------plot sample data

h = figure; hold on
plot(X(idx1,1), X(idx1,2), 'r*');
plot(X(idx2,1), X(idx2,2), 'g*');
plot(X(idx3,1), X(idx3,2), 'b*');
axis tight
xlabel('x_1');
ylabel('x_2');

%------------------Invoke the EM-Algorithm

k=3; % set the number of classes
w = ExpMin(X,k,2);

%----------------Plot classified data

idx1 = find(w.y == 1); % object indices for the 1st class
idx2 = find(w.y == 2);
idx3 = find(w.y == 3);
plot(w.x(idx1,1), w.x(idx1,2), 'ro');
plot(w.x(idx2,1), w.x(idx2,2), 'go');
plot(w.x(idx3,1), w.x(idx3,2), 'bo');
```

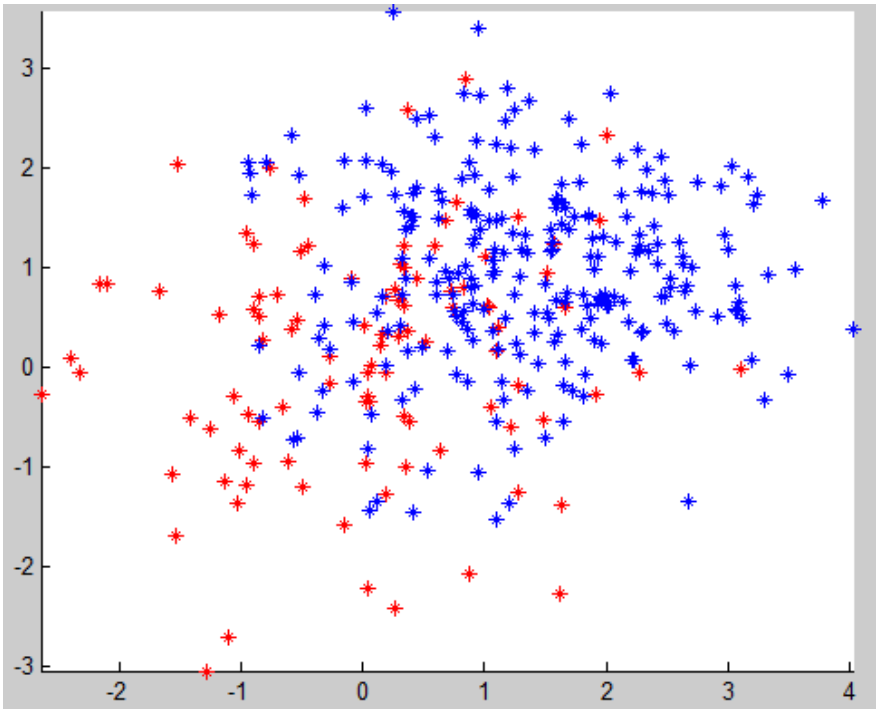


В данном примере заметно небольшое взаимное проникновение классов друг в друга, что влечет к частично ошибочной классификации (метки в результате работы алгоритма EM сравниваются с известными). Аналогично предыдущему примеру по оси абсцисс значение первого признака, по оси ординат – второго. Цвет – принадлежность классу.

Никита Спирин © 2009

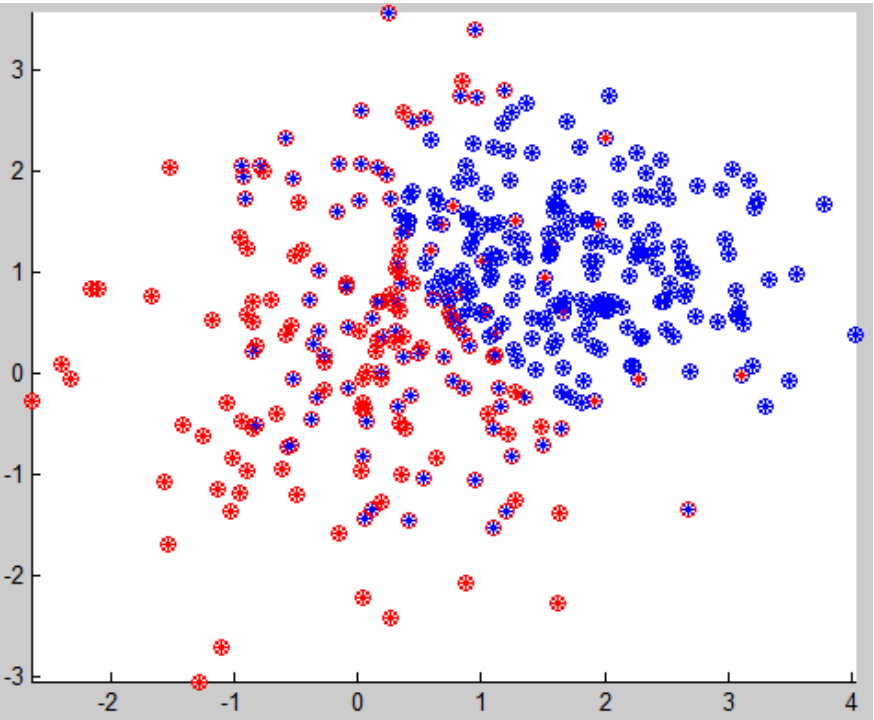
3) Синтетические данные (тяжелый случай, 2 класса)

```
N = 100; % 1st class contains N objects
alpha = 2.5; % 2st class contains alpha*N ones
sig2 = 1; % assume 2nd class has the same variance as the 1st
dist2 = 1.5;

N2 = floor(alpha * N); % calculate the size of the 2nd class
cls1X = randn(N, 2); % generate random objects of the 1st class

% generate a random distance from the center of the 1st class to the center
% of the 2nd
ShiftClass2 = repmat(dist2 * [sin(pi*rand)  cos(pi*rand)],[N2,1]);
% generate random objects of the 2nd class
cls2X = sig2 * randn(N2, 2) + ShiftClass2;
% combine the objects
X = [cls1X; cls2X];
% assign class labels: 0s and 1s
y = [zeros(size(cls1X,1),1); ones(size(cls2X,1),1)];
%------------------------------------------------------------end
X; y; % note that y contains only 1s and 0s
idx1 = find(y == 0); % object indices for the 1st class
idx2 = find(y == 1);
%------------------------------plot sample data
h = figure; hold on
plot(X(idx1,1), X(idx1,2), 'r*');
plot(X(idx2,1), X(idx2,2), 'b*');
axis tight
xlabel('x_1');
ylabel('x_2');
% close(h);
 %--------------------------------------Invoke the EM-Algorithm
k=2; % set the number of classes
w = ExpMin(X,k,2);
 %-----------------------------------------Plot classified data
idx1 = find(w.y == 1); % object indices for the 1st class
idx2 = find(w.y == 2);
plot(w.x(idx1,1), w.x(idx1,2), 'ro');
plot(w.x(idx2,1), w.x(idx2,2), 'bo');
```



Отображение исходных данных с истинными метками классов

Никита Спирин © 2009



На рисунке приведены результаты работы алгоритма на данных, изображенных на рисунке выше. Стоит заметить, что даже при сильном взаимном проникновении классов, алгоритм показывает достойный результат. Интересным является также правильная классификация нескольких точек, находящихся «глубоко» в чужом классе.

Исходный код

- Ссылка на код (скачать ExpMin.m, kMeansParam.m, DistMatrix.m)

Смотри также

- Эта статья в формате PDF
- Другие алгоритмы по практикуму
- EM алгоритм (реализация Кирилла Павлова)
- ...

Литература

- К. В. Воронцов, Лекции по статистическим (байесовским) алгоритмам классификации
- C.M. Bishop, Pattern Recognition and Machine Learning

Никита Спирин © 2009