Projekt 2 - Metody klasyfikacji i redukcji wymiaru

Klaudia Weigel

1 Ogólna motywacja

Niech $\{\mathbf{x}_i\},\ i = 1, \ldots, n$ będzie zbiorem niezależnych obserwacji o takim samym rozkładzie, oraz niech θ oznacza nieznany parametr/parametry tego rozkładu. Możemy wyestymować ten parametr za pomocą estymatora największej wiarogodności:

$$\hat{\theta}_{\text{MLE}} = \arg\ \max_{\theta} \prod_{i=1}^{n} p_{\theta}(\mathbf{x}_i)$$

Funkcja wiarogodności, może okazać się trudna lub niemożliwa do wyznaczenia. W takim przypadku możemy skorzystać z algorytmu EM(Expectation Maximization).

2 Specyfikacja problemu

Mamy dane ciągi $\boldsymbol{x}_1, ..., \boldsymbol{x}_k$ określone: $\boldsymbol{x_1} = (x_{11}, ..., x_{1w})$, gdzie każdy element $x_{1i} \in \{A, C, G, T\} \equiv \{1, 2, 3, 4\}$. Każdy z ciągów jest realizacją zmiennej losowej $X = (X_1, ...X_w)$.
Rozważmy następujące rozkłady:

1. Dla $\boldsymbol{\theta}_j = (\theta_{1,j}, ..., \theta_{4,j})^T$, określamy:

   $$P(X_j = a) = \theta_{a,j},\ a = 1, ..., 4$$

   Zatem $\theta_{a,j}$ jest prawdopodobieństwem, że na j-tej pozycji znajduje się litera a.
   Czyli prawdopodobieństo otrzymania ciągu $\boldsymbol{x}_1 = (x_{11}, ..., x_{1w})$ jest równe:

   $$P(\boldsymbol{x}_1; \boldsymbol{\theta}) = \prod_{i=1}^{w} \theta_{x_{1j},j}$$

2. Dla $\boldsymbol{\theta}^b = (\theta_1^b, \theta_2^b, \theta_3^b, \theta_4^b)$:

   $$P(X_j = a) = \theta_a^b,\ a = 1, ..., 4$$

   Kolejne litery losowane są niezależnie, bez względu na pozycję.
   Prawdopodobieństo otrzymania ciągu $\boldsymbol{x}_1 = (x_{11}, ..., x_{1w})$ jest równe:

   $$P(\boldsymbol{x}_1; \boldsymbol{\theta}^b) = \prod_{i=1}^{w} \theta_{x_{1j}}^b$$

Załóżmy teraz, że losowo wybieramy liczbę 0 albo 1, gdzie prawdopodobieństwo wylosowania 1 to α, a prawdopodobieństwo wylosowania 0 to $1-\alpha$. Czyli, oznaczając przez Z_i zmienną losową, reprezentującą wynik i-tego rzutu:

$$P(Z_i = 1) = \alpha, \quad P(Z_i = 0) = 1-\alpha, \quad i = 1, ..., k$$

Na podstawie tego, jaka wypadła liczba wybieramy z jakiego rozkładu losujemy $\boldsymbol{x}_i$. Niech $Z_i = z_i$, wtedy jeśli $z_i = 1$, to $\boldsymbol{x}_i$ losujemy z rozkładu opisanego w punkcie 1 powyżej, a jeśli $z_i = 1$, to $\boldsymbol{x}_i$ losujemy z rozkładu opisanego w 2.
Chcemy wyestymować parametry $\boldsymbol{\theta}$ oraz $\boldsymbol{\theta}^b$. Gdyby $\boldsymbol{z} = (z_1, ..., z_k)$ było znane, moglibyśmy bezpośrednio wyznaczyć estymator największej wiarogodności ze wzoru:

$$L(\boldsymbol{\theta}, \boldsymbol{\theta}^b) = \prod_{i=1}^{k} z_i(P(\boldsymbol{x}_i; \boldsymbol{\theta}) + (1-z_i)P(\boldsymbol{x}_i; \boldsymbol{\theta}^b))$$

Zakładamy jednak, że nie znamy $\boldsymbol{z}$, do estymacji $\boldsymbol{\theta}$ oraz $\boldsymbol{\theta}^b$ posłuży nam algorytm EM.

3 Algorytm EM

3.1 Ogólny opis algorytmu EM

Algorytm EM iteracyjnie szuka maksimum funcji wiarogodności, w celu wyznaczenia estymatorów nieznanych parametrów rozkładu. Jego stosowanie jest przydatne w przypadkach kiedy bezpośrednia estymacja jest trudna bądź niemożliwa, na przykład kiedy w modelu probabilistycznym występują ukryte zmienne (jak w naszym przypadku, gdzie ukrytymi zmiennymi są $Z_1, ..., Z_k$). Algorytm EM składa się z dwóch kroków:

1. Krok E(xpectation)
   Znaleźć dolne ograniczenie dla logarytmu funkcji wiarogodności dla obecnych parametrów.

2. Krom M(aximization)
   Zoptymalizować dolną granicę i odpoweidnio uaktualnić parametry.

Czyli przy założeniu, że nasze obserwacje to $\boldsymbol{X} = \boldsymbol{x}_1, ..., \boldsymbol{x}_n$, $\boldsymbol{x}_i \in \mathbb{R}^d$, realizacje zmiennych ukrytych $Z_i \in \{1, ..., M\}$ to $\boldsymbol{z} = (z_1, ..., z_n)$, a nieznane parametry to $\boldsymbol{\theta}$, chcemy znaleźć:

$$\mathcal{L}(\boldsymbol{\theta}) = logP(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{\theta}) = \sum_{i=1}^{n} log \sum_{z_i} P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})$$

Załóżmy, że dla każdego $i \in \{1, ..., n\}$, Q_i jest pewnym rozkładem zmiennej losowej Z_i ($Q_i(z) = P(Z_i = z)$, $\sum_z Q_i(z) = 1$, $Q_i(z) \geq 0$). Wtedy:

$$\sum_{i=1}^{n} log \sum_{z_i} P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta}) = \sum_{i=1}^{n} log \sum_{z_i} Q_i(z_i) \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)} \geq \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)}$$

Ostatnia nierówność wynika z nierówności Jensena, która mówi, że jeśli funkcja jest wklęsła to $f(\mathrm{E}[X]) \geq \mathrm{E}[\mathrm{f(X)}]$ (równośc otrzymujemy kiedy X jest stałą). Ponieważ logarytm jest funkcją wklęsłą, a wartość:

$$\sum_{z_i} Q_i(z_i) \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)}$$

jest wartością oczekiwaną funkcji $h(z_i) = \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)}$ zmiennej losowej Z_i z rozkładu Q_i. Teraz chcemy dobrać tak Q_i aby ograniczenie było jak najlepsze, czyli zamiast nierówności otrzymać równość. Czyli musi być spełnione:

$$\frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)} = c$$

Gdzie c jest stałą. Wystarczy zatem, że wybierzemy Q_i proporcjonalne do $P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})$. Stąd i z faktu, że $\sum_z Q_i(z) = 1$, możemy zatem określić Q_i jako:

$$\begin{aligned} Q_i(z_i) &= \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{\sum_z P(\boldsymbol{x}_i, z; \boldsymbol{\theta})} && (1) \\ &= \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{P(\boldsymbol{x}_i); \boldsymbol{\theta}} && (2) \\ &= P(z_i | \boldsymbol{x}_i; \boldsymbol{\theta}) && (3) \\ &= P(Z_i = z_i | \boldsymbol{x}_i; \boldsymbol{\theta}) && (4) \end{aligned}$$

Czyli tak dobrana funkcja Q_i daje nam dolną granicę logarytmu funkcji wiarogodności. W kroku E, obliczymy Q_i dla obecnych parametrów. W kroku M zoptymalizujemy dolną granicę ze względu na θ:

$$\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log \frac{P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})}{Q_i(z_i)}$$

Ponieważ szukamy maksimum tylko po $\boldsymbol{\theta}$, to powyższy wzór można zapisać:

$$\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})$$

Czyli algorytm EM wygląda następująco:

Algorithm 1 EM Algorithm

Require: Observations $\boldsymbol{X} = \boldsymbol{x}_1, ..., \boldsymbol{x}_n, \quad \boldsymbol{x}_i \in \mathbb{R}^d$

Set: $t = 0$. Initialize $\boldsymbol{\theta}^{(0)}$

E(xpectation step). Compute for each i $Q_i(z_i) = P(z_i|\boldsymbol{x}_i; \boldsymbol{\theta})$

M(aximization step).

  Compute $\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{n} \sum_{z_i} Q_i(z_i) log P(\boldsymbol{x}_i, z_i; \boldsymbol{\theta})$

  Find $\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(t)})$

  Repeat until converged

3.2 Algorytm EM dla danych

Mając ogólny opis algorytmu EM, musismy teraz wyprowadzić wykorzystywane w nim wzory dla problemu opisanego w sekcji 2. Mamy więc:

$$
\begin{aligned}
Q_i^{(t)}(0) &= P(Z_i = 0|\boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)}) && (5)\\
&= \frac{P(Z_i = 0)P(\boldsymbol{x}_i|Z_i = 0; \boldsymbol{\Theta}^{(t)})}{P(\boldsymbol{x}_i; \boldsymbol{\Theta}^{(t)})} && (6)\\
&= \frac{P(Z_i = 0)P(\boldsymbol{x}_i|Z_i = 0; \boldsymbol{\Theta}^{(t)})}{\alpha P(\boldsymbol{x}_i; \boldsymbol{\theta}^{(t)}) + (1-\alpha)P(\boldsymbol{x}_i; \boldsymbol{\theta}^{b,(t)})} && (7)\\
& \frac{(1-\alpha)\prod_{i=1}^{w} \theta_{x_{ij}}^{b,(t)}}{\alpha \prod_{j=1}^{w} \theta_{x_{1j},j}^{(t)} + (1-\alpha)\prod_{j=1}^{w} \theta_{x_{ij}}^{b,(t)}} && (8)
\end{aligned}
$$

$$
Q_i^{(t)}(1) = 1 - Q_i^{(t)}(0) \tag{9}
$$

W kroku M musimy wyznaczyć:

$$
\begin{aligned}
\boldsymbol{\Theta}^{(t+1)} &= \arg\max_{\boldsymbol{\Theta}} \sum_{i=1}^{k} \sum_{z_i} Q_i^{(t)}(z_i) log P(\boldsymbol{x}_i, z_i; \boldsymbol{\Theta}) && (10)\\
&= \arg\max_{\boldsymbol{\Theta}} \sum_{i=1}^{k} \left[Q_i^{(t)}(0)(1-\alpha)\sum_{j=1}^{w} log\ \theta_{x_{ij}}^{b} + Q_i^{(t)}(1)\alpha \sum_{j=1}^{w} log\ \theta_{x_{ij},j} \right] && (11)\\
&= \arg\max_{\boldsymbol{\Theta}} Q_1(\boldsymbol{\theta}^b) + Q_2(\boldsymbol{\theta}) && (12)
\end{aligned}
$$

Ponieważ:

$$
\boldsymbol{\Theta}^{(t+1)} = (\boldsymbol{\theta}^{(t+1)}, \boldsymbol{\theta}^{b,(t+1)})
$$

$$
\boldsymbol{\theta}^{b,(t+1)} = \arg\max_{\boldsymbol{\theta}^b} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{(t+1)})
$$

$$
\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{(t+1)})
$$

Czyli szukamy maksimum $Q_1(\boldsymbol{\theta}^b)$ ze względu na $\boldsymbol{\theta}^b$, oraz $Q_2(\boldsymbol{\theta})$ ze względu na $\boldsymbol{\theta}$. Dla $Q_1(\boldsymbol{\theta}^b)$:

$$\begin{aligned} &\text{maximize } Q_1(\boldsymbol{\theta}^b) \\ &\text{subject to } \theta_1^b + \theta_2^b + \theta_3^b + \theta_4^b = 1 \end{aligned}$$

Po rozwiązaniu metodą mnożników Lagrange'a otrzymujemy:

$$\theta_1^b = \frac{\sum_{i=1}^k Q_i^{(t)}(0)|\{j:\ x_{ij}=1\}|}{w\sum_{i=1}^k Q_i^{(t)}(0)} \tag{13}$$

$$\theta_2^b = \frac{\sum_{i=1}^k Q_i^{(t)}(0)|\{j:\ x_{ij}=2\}|}{w\sum_{i=1}^k Q_i^{(t)}(0)} \tag{14}$$

$$\theta_3^b = \frac{\sum_{i=1}^k Q_i^{(t)}(0)|\{j:\ x_{ij}=3\}|}{w\sum_{i=1}^k Q_i^{(t)}(0)} \tag{15}$$

$$\theta_4^b = \frac{\sum_{i=1}^k Q_i^{(t)}(0)|\{j:\ x_{ij}=4\}|}{w\sum_{i=1}^k Q_i^{(t)}(0)} \tag{16}$$

Postępując analogicznie dla $Q_2(\boldsymbol{\theta})$, otrzymujemy:

$$\theta_{t_1 t_2} = \frac{\sum_{i=1, x_{it_2}=t_1}^k Q_i^{(t)}(1)}{\sum_{i=1}^k Q_i^{(t)}(1)} \tag{17}$$

4 Implementacja w Pythonie

Program został napisany w Pythonie wersji 3.7.1.

4.1 Generowanie danych

Na podstawie danych $\boldsymbol{\theta}$, $\boldsymbol{\theta}^b$, w, k oraz $alpha$ możemy wysymulować macierz $\boldsymbol{X}$, tak jak zostało to opisane w specyfikacji problemu. Program generujący dane należy uruchomić poleceniem:

```
python3 NAZWA_PROGRAMU --params param_file.json --output generated_data.json
```

Gdzie `params_file` to plik zawierający dane, a `generated_data.json` to plik, do którego zostanie zapisana wygenerowana macierz.

4.2 Estymacja parametrów

Program do estymacji parametrów $\boldsymbol{\theta}$ i $\boldsymbol{\theta}^b$ uruchamiany jest poleceniem:

```
python3 NAZWA_PROGRAMU --input generated_data.json -- output estimated_params.json
```

4.3 Działanie algorytmu

Algorytm EM wymaga początkowej inicjalizacji szukanych parametrów. W tym przypadku macierze zostały zainicjalizowane losowo. Jakość wyestymowanych parametrów została obliczona za pomocą MSE (Mean Squared Error). Przykładowe wyniki otrzymane dla różnych w i k oraz przybliżenia dla $\boldsymbol{\theta}^b$:

```
w = 50, k = 100 ,alpha = 0.3, MSE Theta = 0.003657, MSE ThetaB = 0.000477
Original ThetaB = [0.25 0.25 0.25 0.25]
Estimated ThetaB = [0.26344315 0.2782388 0.22371683 0.23460122]
w = 50, k = 120 ,alpha = 0.4, MSE Theta = 0.008031, MSE ThetaB = 0.005872
Original ThetaB = [0.39759036 0.03212851 0.37349398 0.19678715]
Estimated ThetaB = [0.32628343 0.13392961 0.29670589 0.24308106]
w = 50, k = 500 ,alpha = 0.3, MSE Theta = 0.003166, MSE ThetaB = 0.009603
Original ThetaB = [0.15533981 0.22815534 0.48058252 0.13592233]
Estimated ThetaB = [0.20108117 0.24352806 0.31923179 0.23615899]
```

Iteracje algorytmu wykonujemy dopóki różnice w logarytmie funkcji wiarogodności stana się nieznaczne.