Inferência Estatística
2º Trabalho

Igor Patrício Michels

16/09/2020

Introdução

Trabalho elaborado pelo aluno Igor Patrício Michels referente a disciplina de Inferência Estatística, do quarto período da Graduação em Matemática Aplicada da FGV-EMAp. Nele discutiremos o Algoritmo EM, além de visitarmos um exemplo de sua aplicação.

O enunciado e eventuais funções utilizadas para resolução deste ou de outros trabalhos podem ser encontrados **nesse repositório do GitHub**. Já o presente relatório em LaTeX pode ser encontrado **nesse link**.

O Algoritmo EM

O Algoritmo EM

O Algoritmo EM é um algoritmo iterativo que visa estimar a máxima verossimilhança quando os dados obtidos são incompletos. Historicamente, esse algoritmo foi explicado por Dempster, Laird e Rubin (veja [1]), entretanto com uma análise de convergência falha. Em 1983, Wu faz um novo estudo sobre o algoritmo e nele apresenta uma análise correta para a convergência do algoritmo (veja [8]).

Feita uma breve apresentação histórica, podemos definir o Algoritmo EM de forma similar a Wu (em [8]) e Casella (em [2]).

Tomamos dois conjuntos $\mathscr{X}$ e $\mathscr{Y}$ onde $\mathscr{X}$ representa o conjunto de dados faltantes e $\mathscr{Y}$ representa o conjunto de dados observados, assim podemos estabelecer uma relação de muitos para um entre $\mathscr{X}$ e $\mathscr{Y}$[1]. Agora, seja θ o vetor de parâmetros a serem estimados, $\theta^{(p)}$ a estimativa na p-ésima iteração do algoritmo, seja também $x \in \mathscr{X}$ um conjunto de dados faltantes, um conjunto de dados observados $y \in \mathscr{Y}$ e de forma que (y, x) representa o dado completo. Assim, $f(y, x|\theta)$ é a função de densidade de probabilidade dos dados completos com parâmetros $\theta \in \Omega$ e a função de densidade de probabilidade de y dada por $g(y|\theta)$ de forma que

$$g(y|\theta) = \int f(y, x|\theta)\ dx.$$

Agora, note que o parâmetro θ é estimado pelo método da máxima verossimilhança dos dados observados, ou seja, maximizando $L(\theta|y)$, que é a função de verossimilhança dos dados observados. Nesse ponto, podemos fazer algumas observações:

1. a função de verossimilhança dos dados completos é a função $L(\theta|y, x) = f(y, x|\theta)$;

2. maximizar uma função é equivalente a maximizar o logaritmo da mesma função, assim, por simplicidade, pode ser interessante trabalhar com a função de log-verossimilhança;

3. em geral, o mais simples é maximizar a função com os dados completos, ou seja, maximizar $L(\theta|y, x)$ é, em geral, mais simples que maximizar $L(\theta|y)$.

[1] Note que essa relação existe em virtude de que os dados faltantes podem ser qualquer um dentre os possíveis.

Feitas tais observações, iremos pensar em maximizar a função de log-verossimilhança. Para tanto, definimos uma nova função

$$k(x|\theta, y) = \frac{f(y, x|\theta)}{g(y|\theta)}$$

a qual é a densidade condicional de x dados y e θ. Assim, podemos reescrever

$$g(y|\theta) = \frac{f(y, x|\theta)}{k(x|\theta, y)},$$

o que nos dá

$$\log L(\theta|y) = \log L(\theta|y, x) - \log k(x|\theta, y).$$

Entretanto, note que x não foi observado, então podemos substituir o lado direito da equação acima elo valor esperado de $k(x|\theta^{(0)}, y)$, nos dando

$$\log L(\theta|y) = E\left[\log L(\theta|y, X)|\theta^{(0)}, y\right] - E\left[\log k(X|\theta, y)|\theta^{(0)}, y\right].$$

Agora, definido $\theta^{(0)}$, podemos iterar o Algoritmo EM da seguinte forma:

- **E-step** determine o valor esperado da função de log-verossimilhança dos dados: $E\left[\log L(\theta|y, x)|\theta^{(p)}, y\right]$;
- **M-step** escolha $\theta^{(p+1)}$ de forma a maximizar a função definida no passo anterior e, em seguida, tome $p = p + 1$.

Um outro detalhe muito importante do Algoritmo EM é que durante a execução do algoritmo obtemos uma sequência $\{\theta^p\}$ monotônica e não decrescente com respeito à verossimilhança, ou seja,

$$L\left(\theta^{(p+1)}|y\right) \geq L\left(\theta^{(p)}|y\right).$$

Podemos enunciar e demonstrar esse teorema.

Teorema 1. *A sequência $\{\theta^{(p)}\}$ obtida durante a iteração do Algoritmo EM é tal que*

$$L\left(\theta^{(p+1)}|y\right) \geq L\left(\theta^{(p)}|y\right).$$

Demonstração. Primeiramente, perceba que o Algoritmo EM realiza um procedimento que visa maximizar a a função de log-verossimilhança e, consequentemente, a função de verossimilhança. O algoritmo funciona maximizando uma função $Q(\theta, \theta^{(p)})$, fácil de maximizar e que serve de limite inferior para $\log L(\theta)$, além disso, temos que $\log L(\theta^{(p)}) = Q(\theta^{(p)}, \theta^{(p)})$. Entretanto, a cada iteração buscamos maximizar Q com $\theta^{(p+1)}$, logo, devemos ter que, ao maximizar Q, $Q \geq \log L(\theta^{(p)})$ e, como Q atua como limite inferior de $\log L(\theta)$, teremos $\log L(\theta^{(p+1)}) \geq \log L(\theta^{(p)})$.

De forma mais matemática:

$$\begin{aligned}
\log L(\theta) &= \sum \log p(y_i|\theta) \\
&= \sum \log \sum_x p(y_i, x|\theta) \\
&= \sum \log \sum_x p\left(x|y_i, \theta^{(p)}\right) \cdot \frac{p(y_i, x|\theta)}{p\left(x|y_i, \theta^{(p)}\right)} \\
&\geq \sum \sum_x p\left(x|y_i, \theta^{(p)}\right) \cdot \frac{p(y_i, x|\theta)}{p\left(x|y_i, \theta^{(p)}\right)} \\
&\equiv Q\left(\theta, \theta^{(p)}\right).
\end{aligned}$$

Note que, ao incluir $p\left(x|y_i, \theta^{(p)}\right)$ estamos calculando de forma separadamente cada amostra y_i. Além disso, a desigualdade acima surge com a aplicação da Desigualdade de Jensen.

Note também que

$$Q\left(\theta^{(p)},\theta^{(p)}\right)=\sum \log p\left(y_i|\theta^{(p)}\right)=\log L\left(\theta^{(p)}\right).$$

Agora, lembramos que durante a execução do passo M encontramos $\theta^{(p+1)}$ de forma a maximizar Q, assim

$$Q\left(\theta^{(p+1)},\theta^{(p)}\right)\geq Q\left(\theta^{(p)},\theta^{(p)}\right)=\log L\left(\theta^{(p)}\right)$$

e, usando que Q é uma cota inferior para $\log L$, temos

$$\log L\left(\theta^{(p+1)}\right)\geq Q\left(\theta^{(p+1)},\theta^{(p)}\right)\geq Q\left(\theta^{(p)},\theta^{(p)}\right)=\log L\left(\theta^{(p)}\right),$$

o que implica em

$$L\left(\theta^{(p+1)}\right)\geq L\left(\theta^{(p)}\right).$$

□

Uma aplicação do Algoritmo

Uma vez que já fomos apresentados ao Algoritmo EM, podemos vê-lo em prática em um exemplo.

Exemplo 1. *Suponha que temos duas moedas, Moeda 1 e Moeda 2 de modo que $Pr(Cara \mid Moeda = 1) = p_1$ e $Pr(Cara \mid Moeda = 2) = p_2$; Suponha que agora façamos o seguinte experimento:*

(i) selecionamos uma moeda aleatoriamente com probabilidade $\frac{1}{2}$;

(ii) lançamos a moeda selecionada m vezes;

(iii) repetimos (i) e (ii) n vezes.

Podemos representar os dados advindos desse experimento como

$$\begin{array}{cccc} X_{11} & \dots & X_{1m} & M_1 \\ X_{21} & \dots & X_{2m} & M_2 \\ \vdots & \ddots & \vdots & \vdots \\ X_{n1} & \dots & X_{nm} & M_n \end{array}$$

onde X_{ij} é a variável de Bernoulli que representa o resultado do j-ésimo lançamento da moeda na i-ésima rodada e $M_i \in \{1,2\}$ é a variável aleatória que guarda a informação sobre qual moeda foi lançada na i-ésima rodada do experimento.

Desenvolva um esquema EM para obter o EMV de $\theta = (p_1, p_2)$ quando desconhecemos os valores de M_i, isto é, quando não sabemos que moeda foi escolhida em cada uma das n rodadas.

Em nosso exemplo, temos apenas os X's, ou seja, dispomos apenas de

$$\begin{array}{ccc} X_{11} & \dots & X_{1m} \\ X_{21} & \dots & X_{2m} \\ \vdots & \ddots & \vdots \\ X_{n1} & \dots & X_{nm} \end{array}$$

e queremos fazer uma estimativa de $\theta = (p_1, p_2)$. Note que se tivéssemos os M's seria muito simples de fazer a estimativa de θ, bastava separar os dados que representam $M = 1$ e os dados em que temos $M = 2$ e calcular p_1 e p_2 separadamente. Entretanto, como não temos os M's, vamos utilizar o algoritmo que acabamos de ver.

Para utilizar o Algoritmo EM, vamos definir que $X_{ij} \in \{0,1\}$, sendo que $X_{ij} = 1$ representa que o j-ésimo lançamento da moeda na i-ésima rodada resultou em cara e $X_{ij} = 0$ irá representar que tal lançamento resultou em coroa.

Agora, seguindo a ideia descrita pelo algoritmo EM, vamos considerar $\theta^{(0)} = \left(\frac{1}{2}, \frac{1}{2}\right)$.[2] Feito isso, podemos ir para o passo E, ou seja, vamos escrever a função de log-verossimilhança

$$
\begin{aligned}
Q\left(\theta|\theta^{(0)}\right) &= \ln\left(\prod_{i=1}^{n}\left(\frac{1}{2}\prod_{j=1}^{m} Pr\left(X_{ij} = x_{ij}|M_i = 1\right) + \frac{1}{2}\prod_{j=1}^{m} Pr\left(X_{ij} = x_{ij}|M_i = 2\right)\right)\right) \\
&= \sum_{i=1}^{n} \ln\left(\frac{1}{2}\prod_{j=1}^{m} Pr(X_{ij} = x_{ij}|M_i = 1) + \frac{1}{2}\prod_{j=1}^{m} Pr(X_{ij} = x_{ij}|M_i = 2)\right)
\end{aligned}
$$

onde $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 1\right) = \frac{1}{2}$ se $x_{ij} = 1$ e $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 1\right) = \frac{1}{2}$ se $x_{ij} = 0$. Analogamente, $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 2\right) = \frac{1}{2}$ se $x_{ij} = 1$ e $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 2\right) = \frac{1}{2}$ se $x_{ij} = 0$.[3]

Definida a função de log-verossimilhança podemos ir para o passo M: encontrar $\theta^{(1)}$ de forma que $\theta^{(1)}$ maximize $Q\left(\theta|\theta^{(0)}\right)$.

Feito isso, finalizamos a primeira iteração e podemos ficar repetindo esse processo.

Para ilustrar esse exemplo, coloquei no repositório do GitHub, um script "Simulação Moedas.py"[4], o qual faz a simulação do Exemplo 1 com os resultados da estimativa de θ para atribuições distintas de $\theta^{(0)}$.

No script, o usuário pode definir os parâmetros θ, m e n, a partir dos quais cria-se uma amostra aleatória de "lançamentos de moedas". Em seguida, podemos fazer a estimativa de θ da amostra gerada, basta dar um valor inicial para θ (definir um $\theta^{(0)}$) que as funções lá definidas irão retornar o valor estimado para θ, bem como o valor da função de log-verossimilhança negativa.[5]

```
Theta inicial: [0.3, 0.9]
Theta final: [0.1     0.70725]
Função log-verossimilhança negativa: 8770.221081994141

Theta inicial: [0.5, 0.5]
Theta final: [0.70725 0.1    ]
Função log-verossimilhança negativa: 8770.22108199414

Theta inicial: [0.1, 0.7]
Theta final: [0.1     0.70725]
Função log-verossimilhança negativa: 8770.221081994141

Theta usado para gerar os dados: [0.1, 0.7]
Função log-verossimilhança negativa: 8771.226955684731
```

Figura 1: Retorno do Scrip

O script citado é uma ferramenta interessante para ilustrar o fato de que o algoritmo nem sempre irá nos levar a um máximo global da função de log-verossimilhança, mas a um máximo local. Além do script, o repositório do GitHub contém um Notebook[6] com a mesma simulação e com algumas informações sobre o desenvolvimento da simulação.

Uma outra aplicação do Algoritmo EM é dada na estimação de parâmetros para exames que utilizam a TRI - Teoria de Resposta ao Item. Para mais detalhes sobre essa aplicação veja [4] ou então [6].

Mais um pouco sobre o Algoritmo EM

Conforme visto nas seções anteriores, o Algoritmo EM é um algoritmo iterativo que busca os parâmetros que irão maximizar a função de verossimilhança quando algumas variáveis não foram observadas e/ou co-

[2] Aqui também poderíamos tomar $\theta^{(0)} = \left(p_1^{(0)}, p_2^{(0)}\right)$ qualquer, uma vez que iremos atualizar o valor de θ durante as iterações.

[3] No caso geral, vale que $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 1\right) = p_1$ se $x_{ij} = 1$ e $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 1\right) = 1 - p_1$ se $x_{ij} = 0$ e, de forma análoga, $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 2\right) = p_2$ se $x_{ij} = 1$ e $Pr\left(X_{ij} = x_{ij}|\theta^{(0)}, M_i = 2\right) = 1 - p_2$ se $x_{ij} = 0$.

[4] Executei o script no VS Code utilizando Python 3.8.0.

[5] Em python, utilizando a biblioteca scipy.optimize, podemos utilizar a função minimize, a qual irá buscar o valor mínimo de uma função já definida, dessa forma, se queremos maximizar a função de log-verossimilhança, podemos minimizar a função de log-verossimilhança negativa.

[6] Para o Notebook, utilizei o Jupyter Notebook (Anaconda3), com o Python 3.

letadas. Além disso, o Teorema 1 nos afirma que a verossimilhança nunca diminui de uma iteração para outra, ou seja, conforme vamos aumentando o número de iterações, o valor da verossimilhança aumenta ou se mantém, assim, para uma sequência limitada $L(\theta^{(p)})$, isso significa que $L(\theta^{(p)})$ converge para um $L*$ (para mais detalhes, veja [8]).

Quanto a sua aplicabilidade, note que nem sempre os dados não coletados possuem relevância no problema, por exemplo, ao estimar a altura média de uma população não precisamos do nome dos indivíduos que observamos, entretanto é interessante observar o sexo dos mesmos para fazer uma estimativa separando os indivíduos pelo sexo. Dessa forma, se coletarmos apenas a altura do indivíduo e a proporção de homem e mulheres podemos estimar qual a altura média da população masculina e feminina.

Agora, embora o algoritmo se mostre um ferramental interessante, nem sempre é compensatório utilizá-lo, pois, mesmo em computadores com alto poder de processamento, suas iterações podem ficar lentas quando o volume de dados é muito grande e/ou quando a quantidade de dados faltantes é alta.[7] Além disso, pelo fato do Algoritmo EM nem sempre convergir para o valor de máxima verossimilhança, mas para um máximo local, pode ser interessante reaplicar o algoritmo diversas vezes com valores iniciais distintos para chegarmos, com maior probabilidade, no máximo global ([3]). Além disso, em muitos casos pode ser até mesmo mais simples utilizar um estimador de máxima verossimilhança, o qual apresentou resultados similares nas comparações realizadas por Stephen Walker em [7] ou então utilizar uma das variações do Algoritmo EM, as quais podem ser vistas brevemente em [5].

Conclusão

Nesse trabalho vimos a definição do Algoritmo EM, bem como um exemplo de sua aplicação e uma breve análise do mesmo.

Conforme visto, o Algoritmo EM nos dá uma forma iterativa de calcular parâmetros para uma série de dados, mesmo que incompleta. Além disso, embora ocorra do algoritmo ficar lento, o mesmo não deixa de ser relevante, uma vez que o mesmo suporta séries de dados relativamente grandes dependendo do computador a ser utilizado e ainda ele é de fácil implementação computacional.

Referências

[1] A. P. Dempster, N. M. Laird e D. B. Rubin. «Maximum Likelihood from Incomplete Data via the EM Algorithm». Em: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 1–38. ISSN: 00359246. URL: `http://www.jstor.org/stable/2984875`.

[2] Roger L. Berger George Casella. *Statical Inference*. 2ª ed. Cengage Learning, 2001. ISBN: 0534243126.

[3] Stephanie Glen. *EM Algorithm (Expectation-maximization): Simple Definition*. URL: `https://www.statisticshowto.com/em-algorithm-expectation-maximization/`. (accessedo: 11.09.2020).

[4] Yaowen Hsu, Terry A. Ackerman e Meichu Fan. «The Relationship between the Bock-Aitkin Procedure and the EM Algorithm for IRT Model Estimation». Em: (1999).

[5] Alexis Roche. *EM algorithm and variants: an informal tutorial*. 2011. arXiv: `1105.1476 [stat.CO]`.

[6] Donald B. Rubin e Neal Thomas. «Using Parameter Expansion to Improve the Performance of the EM Algorithm for Multidimensional IRT Population-Survey Models». Em: *Essays on Item Response Theory*. Ed. por Anne Boomsma, Marijtje A. J. van Duijn e Tom A. B. Snijders. New York, NY: Springer New York, 2001, pp. 193–204. ISBN: 978-1-4613-0169-1. DOI: `10.1007/978-1-4613-0169-1_11`. URL: `https://doi.org/10.1007/978-1-4613-0169-1_11`.

[7] Stephen Walker. «An EM Algorithm for Nonlinear Random Effects Models». Em: *Biometrics* 52.3 (1996), pp. 934–944. ISSN: 0006341X, 15410420. URL: `http://www.jstor.org/stable/2533054`.

[8] C. F. Jeff Wu. «On the Convergence Properties of the EM Algorithm». Em: *Ann. Statist.* 11.1 (mar. de 1983), pp. 95–103. DOI: `10.1214/aos/1176346060`. URL: `https://doi.org/10.1214/aos/1176346060`.

[7] Note que, a cada iteração, buscamos o máximo de uma função que nem sempre é simples.