AP4 - Algebra Linear Numérica

Gustavo Ramalho*

22 de Maio de 2022

1 Questão 1.a

Faça como Cobb e Douglas: use o Método dos Mínimos Quadrados para estimar os valores dos parâmetros β e α. Mostre a sua modelagem para o problema ser resolvido pelo Método dos Mínimos Quadrados.

Para começar essa questão, preciso primeiro colocar a função $P = \beta L^\alpha K^{1-\alpha}$ no formato $Ax = B$, de modo que x seja um vetor que nos dê o valor de β e α. Assim, conseguiremos utilizar nossa função de eliminação gaussiana, que vai nos dar o valor dessas variáveis.

$$
\begin{aligned}
P &= \beta L^\alpha K^{1-\alpha} && (1)\\
\ln P &= \ln \beta + \ln L^\alpha + \ln K^{1-\alpha} && (2)\\
\ln P &= \ln \beta + \alpha \ln L + (1-\alpha) \ln K && (3)\\
\ln P - \ln K &= \ln \beta + \alpha \ln L - \alpha \ln K && (4)\\
\ln \frac{P}{K} &= \ln \beta + \alpha \ln \frac{L}{K} && (5)\\
& && (6)
\end{aligned}
$$

Agora, posso dizer que tenho $b = Ax$:

$$
\underbrace{\begin{bmatrix} \ln \frac{P}{K} \\ \ln \frac{P}{K} \\ \vdots \end{bmatrix}}_{b} = \underbrace{\begin{bmatrix} 1 & \ln \frac{L}{K} \\ 1 & \ln \frac{L}{K} \\ \vdots & \vdots \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} \ln \beta \\ \alpha \end{bmatrix}}_{x} \tag{7}
$$

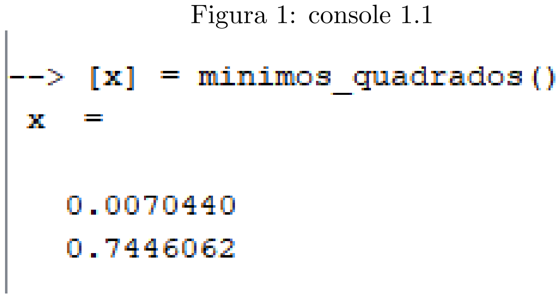
Agora, preciso que meu código encontre o vetor $x = \begin{bmatrix} \ln \beta \\ \alpha \end{bmatrix}$:

*gustavoramalho384@gmail.com

```
function [x] = minimos_quadrados()

    CD = csvRead("cobbdouglas.csv")

    P = CD(:,2)    //Aqui, pego as colunas do meu arquivo que representam P, L e K
    L = CD(:,3)
    K = CD(:,4)

    vec = log(L./K)

    A = [ones(vec), vec]  //Aqui eu gero A e b, da forma que expliquei anteriormente
    b = [log(P./K)]

    A1 = A' * A
    B1 = A' * b

    [x]=Gaussian_Elimination_4(A1,B1)

endfunction

```

Não coloquei nada como entrada da função, pois todos os valores necessários são definidos a partir do arquivo com os dados. Depois, consigo gerar A e b da forma que defini nas equações anteriores, e pela Gaussian_Elimination_4 eu encontro o meu vetor $\begin{bmatrix} \ln\beta \\ \alpha \end{bmatrix}$.

Figura 1: console 1.1



Encontramos o nosso resultado! Como 0.0070440 representa $\ln\beta$, sei que $\beta = e^{0.0070440} = 1.00706886732$.

Portanto, $\beta = 1.00706886732$ e $\alpha = 0.7446062$.

2 Questão 1.b

Agora, use a função de Cobb-Douglas encontrada no item a) e teste a sua adequação calculando os valores da produção nos anos de 1910 e 1920. Comente!

Agora, criei uma função simples que calcula a produção:

```
function [P] = testes_cobbdouglas(L, K, alpha, beta)
    P = beta * L.^(alpha) * K.^(1 - alpha)
endfunction

```

Vou testar para os anos de 1910 e 1920:

Figura 2: console 1.2

```
--> [P] = testes_cobbdouglas(147, 208, 0.7446062, 1.00706886732)
 P  =

   161.76185

--> [P] = testes_cobbdouglas(194, 407, 0.7446062, 1.00706886732)
 P  =

   236.07215
```

Os valores exatos do ano de 1910 é 159 e do ano de 1920 é 231. Isso nos diz que os valores de alfa e beta que encontramos são muito satisfatórios, tendo em vista que a diferença em 1910 foi ≈ 2.7 e em 1920 foi ≈ 5. Podemos dizer que é satisfatório porque a nossa intenção é minimizar o erro e entre os valores que encontramos e os resultados exatos, ou seja, procuramos a solução que "melhor se ajusta"aos pontos dados.

3 Questão 2

Use o seu classificador (hiperplano) e calcule a porcentagem de acertos sobre o arquivo de treinamento (de certa forma é uma medida do ajuste do seu modelo aos dados de treinamento) e sobre o arquivo de teste (de certa forma é uma medida da capacidade de generalização do seu modelo). Construa uma Matriz de Confusão (Confusion Matrix) (pesquise a respeito) com o conjunto de teste e calcule as diversas medidas daí decorrentes, tais como: acurácia, precisão, recall, probabilidade de falso alarme, probabilidade de falsa omissão de alarme. Interprete essas medidas e comente os resultados obtidos.

Criei a função a seguir, que calcula o valor de x de acordo com o arquivo de treino. Depois disso, confiro a acurácia do modelo em relação aos arquivos de entrada, que no caso são TestM e TestB. TestM irá virar Atest (com uma coluna de 1's adicional), que é essencial para que eu consiga conferir essa acurácia.

```
function [x,accuracy] = cancer(TestM, Testb)

    cancer_train = csvRead("cancer_train.csv")  //importo o arquivo de treino
    TrainM = cancer_train(:,1:10)
    Trainb = cancer_train(:, 11)

    [i j] = size(TrainM) //capturando tamanho de M treino
    A = [ones(i, 1), TrainM]  //Criando a matriz A Treino

    [x]=Gaussian_Elimination_4(A' * A, A' * Trainb)   //Encontrando Xi

    [n m] = size(TestM) //Capturando tamanho de M teste
    Atest = [ones(n, 1), TestM]  //Criando A teste
    y = Atest * x  //Conferindo os valores com Xi

    comp = y.*Testb  //Conferindo se o sinal (+ ou -) bate
    acertos = comp>=0
    accuracy = sum(acertos)/n  //calculando acuracia

endfunction
```

Criada a função, consigo calcular os valores de x_i e também a acurácia do meu modelo com o arquivo de treino e o de teste. Primeiro, vou mostrar o vetor x e a acurácia do arquivo de treino:

Figura 3: console 2.1

```
--> cancer_train = csvRead("cancer_train.csv");

--> [x, accuracy] = cancer(cancer_train(:,1:10), cancer_train(:,11))
 x  =

  -6.7579731
   29.311052
   2.0765803
  -18.730222
  -7.3665161
   1.2222756
   0.2283419
   0.0503253
   2.2385058
   0.0249405
   0.7704282
 accuracy  =

   0.93
```

Vimos que o modelo se ajustou bem ao conjunto de treinamento, obtendo 93% de acurácia. Como são 300 dados, chegamos a conclusão que tivemos 279 acertos. Agora, vou analisar o arquivo de teste:

Figura 4: console 2.2

```
--> [x, accuracy] = cancer(cancer_test(:,1:10), cancer_test(:,11))
 x  =

  -6.7579731
   29.311052
   2.0765803
  -18.730222
  -7.3665161
   1.2222756
   0.2283419
   0.0503253
   2.2385058
   0.0249405
   0.7704282
 accuracy  =

   0.7115385
```

Agora, o nosso modelo obteve uma acurácia consideravelmente menor: 71.15%. Isso quer dizer que, de 260 dados de teste, acertamos 185. Possivelmente o modelo está em overfitting, o que significa que ele tem resultados interessantes para o conjunto de treinamento, mas não tão satisfatórios para o conjunto de teste.

Agora, vou calcular a matriz de confusão. Para isso, faço uma pequena alteração nos dados de saída da minha função, que agora irá retornar y também. y vai ser necessário para comparar os valores com TestB:

Figura 5: console 2.3

```
--> FN = sum(cancer_test(:,11)>0 & y<0)
 FN  =

   0.

--> FP = sum(cancer_test(:,11)<0 & y>0)
 FP  =

   75.

--> TP = sum(cancer_test(:,11) > 0 & y>0)
 TP  =

   60.

--> TN = sum(cancer_test(:,11) < 0 & y<0)
 TN  =

   125.

-->
```

É interessante ressaltar que:

- FN = False Negative
- FP = False Positive
- TP = True Positive
- TN = True Negative

Realidade \ Previsão	Sim	Não
Sim	60	0
Não	75	125

Agora que encontramos nossa matriz, vamos calcular algumas medidas interessantes:

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{185}{260} = 0,7115385$$

$$\text{Precisão} = \frac{TP}{TP + FP} = \frac{60}{135} = 0,4444...$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{60}{60} = 1$$

$$\text{Probabilidade de falso alarme} = \frac{FP}{FP + TN} = \frac{75}{200} = 0,375$$

$$\text{Probabilidade de falsa omissão de alarme} = \frac{FN}{FN + TN} = \frac{0}{0 + 125} = 0$$

A acurácia do modelo vai representar a fração entre os acertos que ele obteve e a quantidade total de dados, a precisão é a fração entre o número de casos positivos que foram classificados corretamente e a quantidade total de casos positivos previstos, o recall é a razão entre o número de casos positivos classificados corretamente e a quantidade real de casos positivos, a probabilidade de falso alarmente é a razão entre o número de casos negativos que foi previsto como positivos e a quantidade total de casos negativos reais, a probabilidade de falsa omissão de alarme é a fração entre o número de casos positivos previstos como negativos e a quantidade total de casos negativos reais.

Podemos notar que tivemos uma acurácia razoável, apesar de não ser extremamente satisfatória. A precisão nos deu números consideravelmente ruins, uma vez que, dado o conjunto de positivos que o modelo previu, mais da metade são falsos positivos. Entretanto, quando olhamos os casos positivos reais, podemos observar que 100% foram identificados corretamente, o que diz que o modelo não previu nenhum falso negativo. A probabilidade de falso alarme mostra que $37,5\%$ dos pacientes que não tem câncer foram diagnosticados com câncer, enquanto a probabilidade de falsa omissão de alarme nos diz que nenhum paciente com câncer foi diagnosticado erroneamente.

É muito interessante que a probabilidade de falsa omissão de alarme é 0, porque isso significa que ninguém que possui câncer foi diagnosticado erroneamente, ou seja, a pessoa poderia procurar tratamento e não viver com a doença sem saber. Entretanto, $37,5\%$ de falso alarme é uma quantidade bastante alta, porque todas essas pessoas estariam diagnosticadas com câncer mesmo não estando, o que pode gerar impactos emocionais e financeiros para a pessoa.