

Universidade Federal do Rio de Janeiro
Escola Politécnica
Departamento de Engenharia Eletrônica e de Computação
Engenharia Eletrônica e de Computação
Álgebra Linear 2

TRABALHO DE ÁLGEBRA LINEAR 2

Aluna: Karen dos Anjos Arcoverde

Professor: Marcello Luiz Rodrigues de Campos

Rio de Janeiro
2021

Sumário

0	Introdução	3
0.1	Conteúdo	3
0.2	Software e linguagem	3
0.3	Biblioteca	3
0.4	Base de dados	3
1	Observações	3
2	Questão 1	3
2.1	Resultados	3
2.1.1	Iris-Setosa	3
2.1.2	Iris-Versicolor	4
2.1.3	Iris-Virginica	4
3	Questão 2	4
3.1	Resultados	4
3.1.1	Iris-Setosa	4
3.1.2	Iris-Versicolor	5
3.1.3	Iris-Virginica	6
4	Questão 3	7
4.1	Resultados	7
4.1.1	Iris-Setosa	7
4.1.2	Iris-Versicolor	8
4.1.3	Iris-Virginica	9
5	Questão 4	10
5.1	Resultados	10
6	Código	10
7	Bibliografia	20

0 Introdução

0.1 Conteúdo

O relatório contém os resultados encontrados para cada questão passada pelo professor e o código final em linguagem Python.

0.2 Software e linguagem

O software usado para programação foi o Spyder e a linguagem foi o Python 3.8.5.

0.3 Biblioteca

A biblioteca utilizada para construir o código em Python foi:

- Numpy

0.4 Base de dados

O conjunto de dados "Iris" selecionado para o trabalho foi:

ESPÉCIES	DE	PARA
Iris-setosa	25	39
Iris-versicolor	75	89
Iris-virginica	125	139

Tabela 1: Base de dados "Iris" selecionada

1 Observações

A variável y da coluna PetalWidthCm foi escrita em função das outras variáveis x_1, x_2, x_3 das colunas SepalLengthCm, SepalWidthCm, PetalLengthCm, respectivamente. De forma que $y = a \cdot x_1 + b \cdot x_2 + c \cdot x_3$ sem o termo independente. Com o termo independente: $y = a \cdot x_1 + b \cdot x_2 + c \cdot x_3 + k$.

Além disso, a equação normal é: $x^T \cdot x \cdot w = x^T \cdot y$ ($R \cdot w = p$, $R = x^T \cdot x$ e $p = x^T \cdot y$), onde w é o vetor de coeficientes, o vetor y é a coluna PetalWidthCm e a matriz x é as colunas SepalLengthCm, SepalWidthCm, PetalLengthCm (se for com o termo independente, possui uma coluna a mais que só contém valores 1).

2 Questão 1

2.1 Resultados

2.1.1 Iris-Setosa

```
1 Iris-Setosa
2
3 SEM O TERMO INDEPENDENTE:
4 y = a*x1 + b*x2 + c*x3
5 [a b c] = [ 0.07455282 -0.06602361  0.03673264]
6
```

```

7 COM O TERMO INDEPENDENTE:
8 y = a*x1 + b*x2 + c*x3 + k
9 [a b c k] = [ 0.13497209 -0.07886596  0.10365958 -0.36144997]

```

2.1.2 Iris-Versicolor

```

1 Iris-Versicolor
2
3 SEM O TERMO INDEPENDENTE:
4 y = a*x1 + b*x2 + c*x3
5 [a b c] = [-0.14382683  0.17051618  0.40397714]
6
7 COM O TERMO INDEPENDENTE:
8 y = a*x1 + b*x2 + c*x3 + k
9 [a b c k] = [-0.06826027  0.24149193  0.39877688 -0.63863516]

```

2.1.3 Iris-Virginica

```

1 Iris-Virginica
2
3 SEM O TERMO INDEPENDENTE:
4 y = a*x1 + b*x2 + c*x3
5 [a b c] = [-0.11329721  0.3990124  0.25976859]
6
7 COM O TERMO INDEPENDENTE:
8 y = a*x1 + b*x2 + c*x3 + k
9 [a b c k] = [-0.11686468  0.35346044  0.22325742  0.36734017]

```

3 Questão 2

3.1 Resultados

3.1.1 Iris-Setosa

```

1 Iris-Setosa
2
3 R = VΛV^(T)
4
5 SEM O TERMO INDEPENDENTE:
6 R = [[381.33 255.73 112.57]
7      [255.73 172.5  75.51]
8      [112.57 75.51  33.84]]
9
10 V = [[-0.80618188 -0.45661854  0.37625828]
11      [-0.54157827  0.31342349 -0.78003762]
12      [-0.23825146  0.8326255  0.49997102]]
13
14 Λ = [[5.86392634e+02 0.00000000e+00 0.00000000e+00]
15      [0.00000000e+00 5.29776824e-01 0.00000000e+00]
16      [0.00000000e+00 0.00000000e+00 7.47589571e-01]]

```

```

17
18 V^T = [[-0.80618188 -0.54157827 -0.23825146]
19 [-0.45661854 0.31342349 0.8326255 ]
20 [ 0.37625828 -0.78003762 0.49997102]]
21
22
23 COM O TERMO INDEPENDENTE:
24 R = [[381.33 255.73 112.57 75.5 ]
25 [255.73 172.5 75.51 50.6 ]
26 [112.57 75.51 33.84 22.4 ]
27 [ 75.5 50.6 22.4 15. ]]
28
29 V = [[-0.79610972 0.1628056 -0.47792329 -0.33360603]
30 [-0.53478061 -0.03864172 0.33762985 0.77364243]
31 [-0.23529777 0.18572124 0.80860497 -0.50626137]
32 [-0.15765144 -0.96825037 0.06126507 -0.18407563]]
33
34 Λ = [[6.01336860e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00]
35 [0.00000000e+00 2.79352229e-02 0.00000000e+00 0.00000000e+00]
36 [0.00000000e+00 0.00000000e+00 5.31947766e-01 0.00000000e+00]
37 [0.00000000e+00 0.00000000e+00 0.00000000e+00 7.73257486e-01]]
38
39 V^T = [[-0.79610972 -0.53478061 -0.23529777 -0.15765144]
40 [ 0.1628056 -0.03864172 0.18572124 -0.96825037]
41 [-0.47792329 0.33762985 0.80860497 0.06126507]
42 [-0.33360603 0.77364243 -0.50626137 -0.18407563]]

```

3.1.2 Iris-Versicolor

```

1 Iris-Versicolor
2
3 R = VΛV^T
4
5 SEM O TERMO INDEPENDENTE:
6 R = [[555.38 256.64 397.59]
7 [256.64 119.98 184.35]
8 [397.59 184.35 286.5 ]]
9
10 V = [[ 0.76039163 0.64759888 -0.0491961 ]
11 [ 0.35223975 -0.47485674 -0.80649751]
12 [ 0.54564799 -0.59592513 0.58918716]]
13
14 Λ = [[9.59570396e+02 0.00000000e+00 0.00000000e+00]
15 [0.00000000e+00 1.33162776e+00 0.00000000e+00]
16 [0.00000000e+00 0.00000000e+00 9.57976568e-01]]
17
18 V^T = [[ 0.76039163 0.35223975 0.54564799]
19 [ 0.64759888 -0.47485674 -0.59592513]
20 [-0.0491961 -0.80649751 0.58918716]]
21
22
23 COM O TERMO INDEPENDENTE:

```

```

24 R = [[555.38 256.64 397.59 91. ]
25      [256.64 119.98 184.35 42.2 ]
26      [397.59 184.35 286.5 65.2 ]
27      [ 91.      42.2 65.2 15.  ]]
28
29 V = [[-0.7545525 0.11812875 0.64474223 0.03167953]
30      [-0.34954066 0.11526607 -0.46962259 0.80248968]
31      [-0.54144512 -0.01367345 -0.60234554 -0.58637025]
32      [-0.1237297 -0.98619084 0.030691 0.1057197 ]]
33
34 Λ = [[9.74487597e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00]
35      [0.00000000e+00 7.14132919e-02 0.00000000e+00 0.00000000e+00]
36      [0.00000000e+00 0.00000000e+00 1.33280153e+00 0.00000000e+00]
37      [0.00000000e+00 0.00000000e+00 0.00000000e+00 9.68188359e-01]]
38
39 V^T = [[-0.7545525 -0.34954066 -0.54144512 -0.1237297 ]
40         [ 0.11812875 0.11526607 -0.01367345 -0.98619084]
41         [ 0.64474223 -0.46962259 -0.60234554 0.030691 ]
42         [ 0.03167953 0.80248968 -0.58637025 0.1057197 ]]

```

3.1.3 Iris-Virginica

```

1 Iris-Virginica
2
3 R = VΛV^T
4
5 SEM 0 TERMO INDEPENDENTE:
6 R = [[676.15 304.85 562.72]
7      [304.85 138.7 253.94]
8      [562.72 253.94 469.3 ]]
9
10 V = [[-0.72592866 -0.63483621 0.2645951 ]
11       [-0.32772296 -0.01894783 -0.94458385]
12       [-0.60466953 0.77241438 0.19429563]]
13
14 Λ = [[1.28249881e+03 0.00000000e+00 0.00000000e+00]
15      [0.00000000e+00 5.79253032e-01 0.00000000e+00]
16      [0.00000000e+00 0.00000000e+00 1.07193335e+00]]
17
18 V^T = [[-0.72592866 -0.32772296 -0.60466953]
19         [-0.63483621 -0.01894783 0.77241438]
20         [ 0.2645951 -0.94458385 0.19429563]]
21
22
23 COM 0 TERMO INDEPENDENTE:
24 R = [[676.15 304.85 562.72 100.3 ]
25      [304.85 138.7 253.94 45.4 ]
26      [562.72 253.94 469.3 83.6 ]
27      [100.3 45.4 83.6 15.  ]]
28
29 V = [[-7.21743001e-01 -2.76297507e-01 6.34623278e-01 -1.50933108e-04]
30      [-3.25843186e-01 9.35769444e-01 3.68031853e-02 -1.29642937e-01]]

```

```

31 [-6.01187515e-01 -1.93722166e-01 -7.68083597e-01 -1.05322749e-01]
32 [-1.07176625e-01 1.02308151e-01 -7.71129602e-02 9.85951218e-01]]
33
34  $\Lambda =$  [[1.29740071e+03 0.00000000e+00 0.00000000e+00 0.00000000e+00]
35 [0.00000000e+00 1.08243546e+00 0.00000000e+00 0.00000000e+00]
36 [0.00000000e+00 0.00000000e+00 5.82311582e-01 0.00000000e+00]
37 [0.00000000e+00 0.00000000e+00 0.00000000e+00 8.45463191e-02]]
38
39  $V^T =$  [[-7.21743001e-01 -3.25843186e-01 -6.01187515e-01 -1.07176625e-01]
40 [-2.76297507e-01 9.35769444e-01 -1.93722166e-01 1.02308151e-01]
41 [ 6.34623278e-01 3.68031853e-02 -7.68083597e-01 -7.71129602e-02]
42 [-1.50933108e-04 -1.29642937e-01 -1.05322749e-01 9.85951218e-01]]

```

4 Questão 3

4.1 Resultados

4.1.1 Iris-Setosa

```

1 Iris-Setosa
2
3  $R = U \Sigma V^T$ 
4
5 SEM O TERMO INDEPENDENTE:
6  $R =$  [[381.33 255.73 112.57]
7 [255.73 172.5 75.51]
8 [112.57 75.51 33.84]]
9
10  $U =$  [[ 0.80618188 -0.45661854 -0.37625828]
11 [ 0.54157827 0.31342349 0.78003762]
12 [ 0.23825146 0.8326255 -0.49997102]]
13
14  $\Sigma =$  [[5.86392634e+02 0.00000000e+00 0.00000000e+00]
15 [0.00000000e+00 5.29776824e-01 0.00000000e+00]
16 [0.00000000e+00 0.00000000e+00 7.47589571e-01]]
17
18  $V^T =$  [[ 0.80618188 0.54157827 0.23825146]
19 [-0.45661854 0.31342349 0.8326255 ]
20 [-0.37625828 0.78003762 -0.49997102]]
21
22
23 COM O TERMO INDEPENDENTE:
24  $R =$  [[381.33 255.73 112.57 75.5 ]
25 [255.73 172.5 75.51 50.6 ]
26 [112.57 75.51 33.84 22.4 ]
27 [ 75.5 50.6 22.4 15. ]]
28
29  $U =$  [[ 0.79610972 -0.33360603 -0.47792329 -0.1628056 ]
30 [ 0.53478061 0.77364243 0.33762985 0.03864172]
31 [ 0.23529777 -0.50626137 0.80860497 -0.18572124]
32 [ 0.15765144 -0.18407563 0.06126507 0.96825037]]

```

```

33
34  $\Sigma = \begin{bmatrix} 6.01336860e+02 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 7.73257486e-01 & 0.00000000e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 0.00000000e+00 & 5.31947766e-01 & 0.00000000e+00 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 2.79352230e-02 \end{bmatrix}$ 
35
36
37
38
39  $V^T = \begin{bmatrix} 0.79610972 & 0.53478061 & 0.23529777 & 0.15765144 \\ -0.33360603 & 0.77364243 & -0.50626137 & -0.18407563 \\ -0.47792329 & 0.33762985 & 0.80860497 & 0.06126507 \\ -0.1628056 & 0.03864172 & -0.18572124 & 0.96825037 \end{bmatrix}$ 
40
41
42

```

4.1.2 Iris-Versicolor

```

1 Iris-Versicolor
2
3 R = U $\Sigma$ V^T
4
5 SEM O TERMO INDEPENDENTE:
6 R =  $\begin{bmatrix} 555.38 & 256.64 & 397.59 \\ 256.64 & 119.98 & 184.35 \\ 397.59 & 184.35 & 286.5 \end{bmatrix}$ 
7
8
9
10 U =  $\begin{bmatrix} 0.76039163 & 0.64759888 & -0.0491961 \\ 0.35223975 & -0.47485674 & -0.80649751 \\ 0.54564799 & -0.59592513 & 0.58918716 \end{bmatrix}$ 
11
12
13
14  $\Sigma = \begin{bmatrix} 9.59570396e+02 & 0.00000000e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 1.33162776e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 0.00000000e+00 & 9.57976568e-01 \end{bmatrix}$ 
15
16
17
18  $V^T = \begin{bmatrix} 0.76039163 & 0.35223975 & 0.54564799 \\ 0.64759888 & -0.47485674 & -0.59592513 \\ -0.0491961 & -0.80649751 & 0.58918716 \end{bmatrix}$ 
19
20
21
22
23 COM O TERMO INDEPENDENTE:
24 R =  $\begin{bmatrix} 555.38 & 256.64 & 397.59 & 91. \\ 256.64 & 119.98 & 184.35 & 42.2 \\ 397.59 & 184.35 & 286.5 & 65.2 \\ 91. & 42.2 & 65.2 & 15. \end{bmatrix}$ 
25
26
27
28
29 U =  $\begin{bmatrix} 0.7545525 & -0.64474223 & -0.11812875 & 0.03167953 \\ 0.34954066 & 0.46962259 & -0.11526607 & 0.80248968 \\ 0.54144512 & 0.60234554 & 0.01367345 & -0.58637025 \\ 0.1237297 & -0.030691 & 0.98619084 & 0.1057197 \end{bmatrix}$ 
30
31
32
33
34  $\Sigma = \begin{bmatrix} 9.74487597e+02 & 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 1.33280153e+00 & 0.00000000e+00 & 0.00000000e+00 \\ 0.00000000e+00 & 0.00000000e+00 & 7.14132919e-02 & 0.00000000e+00 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 9.68188359e-01 \end{bmatrix}$ 
35
36
37
38
39  $V^T = \begin{bmatrix} 0.7545525 & 0.34954066 & 0.54144512 & 0.1237297 \end{bmatrix}$ 

```



```

40 [-0.64474223  0.46962259  0.60234554 -0.030691  ]
41 [-0.11812875 -0.11526607  0.01367345  0.98619084]
42 [ 0.03167953  0.80248968 -0.58637025  0.1057197  ]]

```

4.1.3 Iris-Virginica

```

1  Iris-Virginica
2
3  R = UΣV^(T)
4
5  SEM 0 TERMO INDEPENDENTE:
6  R = [[676.15 304.85 562.72]
7        [304.85 138.7  253.94]
8        [562.72 253.94 469.3  ]]
9
10 U = [[ 0.72592866 -0.63483621 -0.2645951  ]
11        [ 0.32772296 -0.01894783  0.94458385]
12        [ 0.60466953  0.77241438 -0.19429563]]
13
14 Σ = [[1.28249881e+03 0.00000000e+00 0.00000000e+00]
15        [0.00000000e+00 5.79253032e-01 0.00000000e+00]
16        [0.00000000e+00 0.00000000e+00 1.07193335e+00]]
17
18 V^T = [[ 0.72592866  0.32772296  0.60466953]
19          [-0.63483621 -0.01894783  0.77241438]
20          [-0.2645951  0.94458385 -0.19429563]]
21
22
23 COM 0 TERMO INDEPENDENTE:
24 R = [[676.15 304.85 562.72 100.3  ]
25        [304.85 138.7  253.94  45.4  ]
26        [562.72 253.94 469.3   83.6  ]
27        [100.3  45.4   83.6   15.   ]]
28
29 U = [[ 7.21743001e-01 -2.76297507e-01 -6.34623278e-01 -1.50933128e-04]
30        [ 3.25843186e-01  9.35769444e-01 -3.68031855e-02 -1.29642937e-01]
31        [ 6.01187515e-01 -1.93722166e-01  7.68083597e-01 -1.05322749e-01]
32        [ 1.07176625e-01  1.02308151e-01  7.71129601e-02  9.85951218e-01]]
33
34 Σ = [[1.29740071e+03 0.00000000e+00 0.00000000e+00 0.00000000e+00]
35        [0.00000000e+00 1.08243546e+00 0.00000000e+00 0.00000000e+00]
36        [0.00000000e+00 0.00000000e+00 5.82311582e-01 0.00000000e+00]
37        [0.00000000e+00 0.00000000e+00 0.00000000e+00 8.45463191e-02]]
38
39 V^T = [[ 7.21743001e-01  3.25843186e-01  6.01187515e-01  1.07176625e-01]
40          [-2.76297507e-01  9.35769444e-01 -1.93722166e-01  1.02308151e-01]
41          [-6.34623278e-01 -3.68031855e-02  7.68083597e-01  7.71129601e-02]
42          [-1.50933128e-04 -1.29642937e-01 -1.05322749e-01  9.85951218e-01]]

```

5 Questão 4

5.1 Resultados

```
1 SEM O TERMO INDEPENDENTE:
2 A = Iris-Versicolor
3 B = Iris-Setosa
4 C = Iris-Setosa
5 D = Iris-Versicolor
6 E = Iris-Virginica
7
8 COM O TERMO INDEPENDENTE:
9 A = Iris-Versicolor
10 B = Iris-Setosa
11 C = Iris-Setosa
12 D = Iris-Versicolor
13 E = Iris-Virginica
```

6 Código

```
1 # Programa codigo.py
2 # Autora: Karen dos Anjos Arcoverde
3 # Data: 06/02/2021
4 #
5
6
7 import numpy as np
8 import sys
9
10
11
12 ##### Funcoes #####
13 def pegarDados(tipo_iris):
14
15     # tipo_iris = 1 Setosa , tipo_iris = 2 Versicolor,
16     # tipo_iris = 3 Virginica
17     dados = []
18     IDs = []
19
20     Setosa = range(25,39+1)
21     Versicolor = range(75,89+1)
22     Virginica = range(125,139+1)
23
24     arquivo= open("dados_13.csv",'r')
25     arquivo.readline() # ignora a primeira linha
26
27     if (tipo_iris == 1):
28         IDs = Setosa
29     if (tipo_iris == 2):
30         IDs = Versicolor
31     if (tipo_iris == 3):
```

```

32     IDs = Virginica
33
34     for i in range(1,46): # percorre todo o banco de dados 1-45
35         linha = (arquivo.readline()).split(',') #separa os dados por virgula
36
37
38         if (int(linha[0]) in IDs):# percorre os ids selecionados
39             linha.pop(0) # retira o ID dos dados
40             linha.pop(-1) # retira a especie dos dados
41             for j in range(4): #para cada dado
42                 linha[j] = float(linha[j]) #transforma em numero
43
44             dados.append(linha) #adiciona na lista de dados
45
46     arquivo.close()
47     return dados
48
49 # -----
50 def construir_equacao_normal (dados):
51
52     #equacao normal - minimos quadrados:
53     #  $(x^T).x.w = (x^T).y$ 
54     #  $(x\_transposta).x.w = (x\_transposta).y$ 
55     #  $R.w = p$ 
56     #  $R = (x^T).x$ 
57     #  $p = (x^T).y$ 
58
59     ##### sem termo independente #####
60     #  $y = a*x1 + b*x2 + c*x3$ 
61     x = []
62     y = []
63
64     #achar  $(x^T)$  e x
65     # x - colunas SepalLengthCm, SepalWidthCm, PetalLengthCm
66     x = np.array(dados)
67     x = np.delete(x.reshape(15,4),3,1) #deleta a ultima coluna de x
68     x_transposta = np.transpose(x) #faz a transposta de x:  $(x^T)$ 
69
70     #achar R
71     R = np.dot(x_transposta,x) #multiplica  $(x^T)$  por x
72
73     #achar y - coluna PetalWidthCm
74     y = np.array(dados)
75     y = np.delete(y.reshape(15,4),0,1) #deleta a primeira coluna de y
76     y = np.delete(y.reshape(15,3),0,1) #deleta a segunda coluna de y
77     y = np.delete(y.reshape(15,2),0,1) #deleta a terceira coluna de y
78
79     #achar p
80     p = np.dot(x_transposta,y)
81
82     ##### com termo independente #####
83     #  $y = a*x1 + b*x2 + c*x3 + k$ 

```

```

84
85     for i in range (0,15):
86         dados[i][3] = 1
87
88     #achar (x^T) e x
89     x = np.array(dados)
90     x_transposta = np.transpose(x) #faz a transposta de x: (x^T)
91     #achar R
92     R1= np.dot(x_transposta,x) #multiplica (x^T) por x
93     #achar p
94     p1 = np.dot(x_transposta,y)
95
96     return R,p,R1,p1
97
98 # -----
99 def PLU(R,p):
100     indice_coluna = 0
101     indice_linha = 1
102     indice_L_1s = 0
103     tamanho_R = len(R)
104
105     while (indice_coluna < tamanho_R):
106         #constroi a matriz L, triangular inferior
107         if (R[indice_coluna][indice_coluna] != 0):
108             L = np.zeros((tamanho_R, tamanho_R))
109
110             indice_coluna_aux = indice_coluna + 1
111             while (indice_coluna_aux < tamanho_R):
112                 L[indice_coluna_aux][indice_coluna] = - R[indice_coluna_aux][
113                     indice_coluna]/R[indice_coluna][indice_coluna]
114                 indice_linha += 1
115                 indice_coluna_aux += 1
116
117             while (indice_L_1s < tamanho_R):
118                 L[indice_L_1s][indice_L_1s] = 1
119                 indice_L_1s += 1
120
121             indice_L_1s = 0
122             indice_linha = 1
123
124             #multiplica a matriz R por L
125             R = np.dot (L,R)
126             p = np.dot (L,p)
127
128         #se o pivo for zero, necessario multiplicar por uma matriz P de
129         #permutacao
130         if (R[indice_coluna][indice_coluna] == 0):
131             P = np.zeros((tamanho_R, tamanho_R))
132             P[indice_coluna][indice_coluna + 1] = 1
133             P[indice_coluna + 1][indice_coluna] = 1

```

```

134         i = 0
135         j = 0
136         guarda_1 = False
137         while (i < tamanho_R):
138             while (j < tamanho_R):
139                 if (P[i][j] == 1):
140                     guarda_1 = True
141                     j += 1
142             if (guarda_1 == False):
143                 P[i][i] = 1
144             i += 1
145
146         R = np.dot(P,R)
147         p = np.dot (P,p)
148
149         indice_coluna += 1
150
151     #backsubstitution
152     w = np.linalg.solve(R, p)
153
154     return w
155
156 # -----
157 def decomposicao_espectral(R):
158
159     # R = VDV^(T)
160     #determinando autovalores e autovetores
161     autovalores, autovetores = np.linalg.eig(R)
162     # matriz diagonal de autovalores
163     matrizDiagonal = np.diag(autovalores)
164
165     return autovetores, matrizDiagonal
166
167 # -----
168 def contruir_svd (R):
169     s = []
170     i = 0
171     j = 0
172
173     # R = U.s.VT
174     #determinando autovalores e autovetores
175     ### U
176     ## autovetores de R.(R^T)
177     autovaloresU, autovetoresU = np.linalg.eig(np.dot(R,np.transpose(R)))
178
179     ### V
180     ## autovetores de (R^T).R
181     autovaloresV, autovetoresV = np.linalg.eig(np.dot(np.transpose(R),R))
182
183     tamanho_autovalores_U = autovaloresU.shape
184     tamanho_autovalores_V = autovaloresV.shape
185

```

```

186 while (i < tamanho_autovalores_U [0]):
187     while (j < tamanho_autovalores_V [0]):
188         if (autovaloresU [i] == autovaloresV [j]):
189             s.append(np.sqrt(autovaloresU [i]))
190         elif (autovaloresV [j] == 0):
191             s.append(0)
192         j += 1
193
194     if (autovaloresU [i] == 0):
195         s.append(0)
196
197     j = 0
198     i += 1
199
200     return autovetoresU, np.array(s),np.transpose(autovetoresV)
201
202 # -----
203 def estimar_amostras(amostra, w_setosa,w_versicolor,w_virginica,c_independente
204 ):
205
206     lista_erros = []
207     estimativa = ""
208     indice = 0
209     amostra_x = []
210     indice_erros_modulo = 0
211
212     while (indice < 3):
213         amostra_x.append (amostra[indice])
214         indice += 1
215
216     if (c_independente == True):
217         amostra_x.append(1)
218
219     amostra_x = np.array(amostra_x)
220
221     #produto interno <x,y> = (x^T).y
222
223     estimativa_setosa = np.dot(amostra_x,w_setosa)
224     estimativa_versicolor = np.dot(amostra_x,w_versicolor)
225     estimativa_virginica = np.dot(amostra_x,w_virginica)
226
227     erro_setosa = estimativa_setosa[0] - amostra[3]
228     lista_erros.append(erro_setosa)
229
230     erro_versicolor = estimativa_versicolor[0] - amostra[3]
231     lista_erros.append(erro_versicolor)
232
233     erro_virginica = estimativa_virginica[0] - amostra[3]
234     lista_erros.append(erro_virginica)
235
236     while (indice_erros_modulo < len(lista_erros)):
237         lista_erros [indice_erros_modulo] = abs(lista_erros[indice_erros_modulo]

```

```

    ])
    indice_erros_modulo += 1

237
238
239 if (min(lista_erros) == abs(erro_setosa)):
240     estimativa = "Iris-Setosa"
241
242     return estimativa
243
244 if (min(lista_erros) == abs(erro_versicolor)):
245     estimativa = "Iris-Versicolor"
246
247     return estimativa
248
249 if (min(lista_erros) == abs(erro_virginica)):
250     estimativa = "Iris-Virginica"
251
252     return estimativa
253
254 ##### Programa Principal #####
255 def menu():
256     resultado = 0
257     tipo_iris = 0
258     coeficientes_sem_aux = []
259     coeficientes_com_aux = []
260     indice = 0
261
262
263 while (resultado != 5):
264     print()
265     print('##### MENU PRINCIPAL
266         #####')
267     print("Digite somente o numero da questao que voce deseja ver o
268         resultado: ")
269     print("1 = Questao 1")
270     print("2 = Questao 2")
271     print("3 = Questao 3")
272     print("4 = Questao 4")
273     print("5 = SAIR")
274     print('
275         #####
276         ')
277     print()
278
279     resultado = int(input())
280
281     if (resultado == 5):
282         sys.exit(0)
283
284     if (resultado == 1):
285
286         while (tipo_iris != 4):
287             print('##### MENU IRIS

```

```

#####')
284 print("Digite qual especie voce deseja fazer a regressao
      linear: ")
285 print("1 = Iris-Setosa")
286 print("2 = Iris-Versicolor")
287 print("3 = Iris Virginica")
288 print("4 = VOLTAR AO MENU PRINCIPAL")
289 print('
      #####')
290 print()
291
292 tipo_iris = int(input())
293
294 if (tipo_iris == 4):
295     menu()
296
297 else:
298     dados = pegarDados (tipo_iris)
299     R,p,R1,p1 = construir_equacao_normal(dados)
300     w = PLU(R,p)
301     w1 = PLU(R1,p1)
302
303     print()
304     if (tipo_iris == 1):
305         print("Iris-Setosa\n")
306     elif (tipo_iris == 2):
307         print("Iris-Versicolor\n")
308     elif (tipo_iris == 3):
309         print("Iris-Virginica\n")
310
311     print("SEM O TERMO INDEPENDENTE: ")
312     print("y = a*x1 + b*x2 + c*x3")
313     print("[a b c] = ",end="")
314
315
316     while (indice < len(w)):
317         coeficientes_sem_aux.append(w[indice][0])
318         coeficientes_sem = np.array(coeficientes_sem_aux)
319         indice += 1
320
321     print(coeficientes_sem)
322     coeficientes_sem_aux = []
323
324
325     print()
326
327     print("COM O TERMO INDEPENDENTE: ")
328     print("y = a*x1 + b*x2 + c*x3 + k")
329     print("[a b c k] = ",end="")
330
331     indice = 0
332     while (indice < len(w1)):

```



```

333         coeficientes_com_aux.append(w1[indice][0])
334         coeficientes_com = np.array(coeficientes_com_aux)
335         indice += 1
336
337     print(coeficientes_com)
338     coeficientes_com_aux = []
339     indice = 0
340
341     print()
342
343 if (resultado == 2):
344
345     while (tipo_iris != 4):
346         print('##### MENU IRIS
347             #####')
348         print("Digite qual especie voce deseja fazer a decomposicao
349             espectral: ")
350         print("1 = Iris-Setosa")
351         print("2 = Iris-Versicolor")
352         print("3 = Iris Virginica")
353         print("4 = VOLTAR AO MENU PRINCIPAL")
354         print('
355             #####
356             ')
357         print()
358
359         tipo_iris = int(input())
360
361         if (tipo_iris == 4):
362             menu()
363
364         else:
365             dados = pegarDados (tipo_iris)
366             R,p,R1,p1 = construir_equacao_normal(dados)
367             autovetores, matrizDiagonal = decomposicao_espectral(R)
368             autovetores1, matrizDiagonal1 = decomposicao_espectral(R1)
369
370             if (tipo_iris == 1):
371                 print("Iris-Setosa\n")
372             elif (tipo_iris == 2):
373                 print("Iris-Versicolor\n")
374             elif (tipo_iris == 3):
375                 print("Iris-Virginica\n")
376
377             print("R = V\u0392V^(T)")
378             print()
379
380             print("SEM O TERMO INDEPENDENTE: ")
381             print("R = ", R)
382             print()

```

```

381         print("V = ",autovetores)
382         print()
383         print('\u039B = ',matrizDiagonal)
384         print()
385         print("V^T = ",np.transpose(autovetores))
386
387         print()
388         print()
389
390         print("COM O TERMO INDEPENDENTE: ")
391         print("R = ", R1)
392         print()
393         print("V = ", autovetores1)
394         print()
395         print('\u039B = ', matrizDiagonal1)
396         print()
397         print("V^T = ", np.transpose(autovetores1))
398
399         print()
400
401     if (resultado == 3):
402
403         while (tipo_iris != 4):
404             print('##### MENU IRIS #####')
405             print("Digite qual especie voce deseja fazer o SVD: ")
406             print("1 = Iris-Setosa")
407             print("2 = Iris-Versicolor")
408             print("3 = Iris Virginica")
409             print("4 = VOLTAR AO MENU PRINCIPAL")
410             print('#####')
411             print()
412
413             tipo_iris = int(input())
414
415             if (tipo_iris == 4):
416                 menu()
417
418             else:
419
420                 dados = pegarDados (tipo_iris)
421                 R,p,R1,p1 = construir_equacao_normal(dados)
422                 U, s, VT = contruir_svd(R)
423                 U1, s1, VT1 = contruir_svd(R1)
424
425                 if (tipo_iris == 1):
426                     print("Iris-Setosa\n")
427                 elif (tipo_iris == 2):
428                     print("Iris-Versicolor\n")
429                 elif (tipo_iris == 3):
430                     print("Iris-Virginica\n")
431
432                 print("R = U\u03A3V^(T)")

```

```

433         print()
434
435         print("SEM O TERMO INDEPENDENTE: ")
436         print("R = ", R)
437         print()
438         print("U = ", U)
439         print()
440         print('\u03A3 = ', np.diag(s))
441         print()
442         print("V^T = ", VT)
443         print()
444         print()
445
446         print("COM O TERMO INDEPENDENTE: ")
447         print("R = ", R1)
448         print()
449         print("U = ", U1)
450         print()
451         print('\u03A3 = ', np.diag(s1))
452         print()
453         print("V^T = ", VT1)
454
455         print()
456
457
458
459     if (resultado == 4):
460         A = [5.0,2.3,3.3,1.0]
461         B = [4.6,3.2,1.4,0.2]
462         C = [5.0,3.3,1.4,0.2]
463         D = [6.1,3.0,4.6,1.4]
464         E = [5.9,3.0,5.1,1.8]
465
466         print("SEM O TERMO INDEPENDENTE: ")
467         dados = pegarDados (1)
468         R,p,R1,p1 = construir_equacao_normal(dados)
469         w_setosa = PLU(R,p)
470
471         dados = pegarDados (2)
472         R,p,R1,p1 = construir_equacao_normal(dados)
473         w_versicolor = PLU(R,p)
474
475         dados = pegarDados (3)
476         R,p,R1,p1 = construir_equacao_normal(dados)
477         w_virginica = PLU(R,p)
478
479         c_independente = False
480         estimativa = estimar_amstras(A,w_setosa,w_versicolor,w_virginica,
481                                     c_independente)
481         print("A = ", estimativa)
482         estimativa = estimar_amstras(B,w_setosa,w_versicolor,w_virginica,
483                                     c_independente)

```

```

483     print("B = ", estimativa)
484     estimativa = estimar_amstras(C,w_setosa,w_versicolor,w_virginica,
        c_independente)
485     print("C = ", estimativa)
486     estimativa = estimar_amstras(D,w_setosa,w_versicolor,w_virginica,
        c_independente)
487     print("D = ", estimativa)
488     estimativa = estimar_amstras(E,w_setosa,w_versicolor,w_virginica,
        c_independente)
489     print("E = ", estimativa)
490
491     print()
492     print("COM O TERMO INDEPENDENTE: ")
493     dados = pegarDados (1)
494     R,p,R1,p1 = construir_equacao_normal(dados)
495     w1_setosa = PLU(R1,p1)
496
497     dados = pegarDados (2)
498     R,p,R1,p1 = construir_equacao_normal(dados)
499     w1_versicolor = PLU(R1,p1)
500
501     dados = pegarDados (3)
502     R,p,R1,p1 = construir_equacao_normal(dados)
503     w1_virginica = PLU(R1,p1)
504
505     c_independente = True
506     estimativa1 = estimar_amstras(A,w1_setosa,w1_versicolor,
        w1_virginica,c_independente)
507     print("A = ", estimativa1)
508     estimativa1 = estimar_amstras(B,w1_setosa,w1_versicolor,
        w1_virginica,c_independente)
509     print("B = ", estimativa1)
510     estimativa1 = estimar_amstras(C,w1_setosa,w1_versicolor,
        w1_virginica,c_independente)
511     print("C = ", estimativa1)
512     estimativa1 = estimar_amstras(D,w1_setosa,w1_versicolor,
        w1_virginica,c_independente)
513     print("D = ", estimativa1)
514     estimativa1 = estimar_amstras(E,w1_setosa,w1_versicolor,
        w1_virginica,c_independente)
515     print("E = ", estimativa1)
516
517     menu()
518
519     ##### chamada ao menu
520     menu()

```

7 Bibliografia

- <https://algebralinearufcg.github.io/jup-not/prog02-learning-numpy.html>

- <https://machinelearningmastery.com/singular-value-decomposition-for-machine-learning/>
- <https://pt.coredump.biz/questions/34007632/how-to-remove-a-column-in-a-numpy-array>
- <https://pythonforundergradengineers.com/unicode-characters-in-python.html>
- <https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.svd.html>
- <https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.diagsvd.html#scipy.linalg.diagsvd>
- https://www.geeksforgeeks.org/python-exit-commands-quit-exit-sys-exit-and-os_exit/