UNIVERSIDADE
FEDERAL DO CEARÁ
Engenharia de Computação
Inteligência Computacional
Nome: Stefane Adna dos Santos
Matrícula: 403249
Professor: Jarbas Joaci de Mesquita SA Junior

1. Questão 1

A primeira parte da questão consiste em importar as bibliotecas necessárias, receber o dataset "aerogerador.dat" e configurar os dados lidos, de uma forma que facilite a utilização desses dados.

```
import numpy as np
import os
import re
import matplotlib.pyplot as plt
```

```
file = open("data/aerogerador.dat", "r")
x, y = [], []
x_aux, y_aux = 0,0
for linha in file:
    linha = linha.strip()
    linha = re.sub('\s+','-',linha)
    x_aux, y_aux = linha.split("-")
    y.append(float(y_aux))
    x.append(float(x_aux))
file.close()
Y = np.array(y)
```

A variável 'x' vai receber os dados de entradas e a variável 'y' os dados de saída. Logo em seguida, os dados da variável 'y' foram convertidos para uma lista numpy, para facilitar os cálculos nas próximas etapas da questão. Após isso, o usuário deve inserir o grau do polinômio desejado.

```
k = int(input("Digite o grau do polinomio:"))
```

Segundo a aula ministrada pelo professor, para realizar a regressão, e quando existe apenas uma variável de entrada, deve-se montar uma matriz com os valores de entrada, da seguinte forma:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^k \\ 1 & x_2 & x_2^2 & \cdots & x_2^k \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^k \end{bmatrix}_{n\times(k+1)}$$

Onde n é o número de observações da variável “x” e k é o grau do polinômio escolhido. Para transformar os dados de entrada ‘x’ na matriz X foi criado o seguinte código.

```
#Cria a MATRIZ X
X = np.zeros((len(y),k+1), dtype=np.float64)
for i in range(len(y)):
    for j in range(k+1):
        X[i][j] = x[i]**j
```

Neste código, inicialmente é criado uma matriz de n x (k+1) e depois os valores de observação de entrada são preenchidos na matriz. Após isso, são utilizadas as equações passadas pelo professor no slide, para realizar o cálculo da estimativa dos quadrados mínimos (equação 11) , preditor (equação 12) e resíduo (equação 13).

```
B = np.linalg.inv(X.T@X)@(X.T@y)
y_preditor = X@B
e_residuo = y - y_preditor
```

Para realizar o cálculo do coeficiente de determinação R^2 foi utilizada a equação 48 do slide, e para o cálculo do coeficiente de determinação ajustado foi utilizado a equação 49 do slide. Para isso, foi necessário o cálculo da média dos valores de saída ‘y’.

```
for i in range(len(y)):
    y_media = y_media + y[i]
y_media= y_media/len(y)
```

Para o cálculo do coeficiente de determinação, é necessário o cálculo do SQe e Syy.

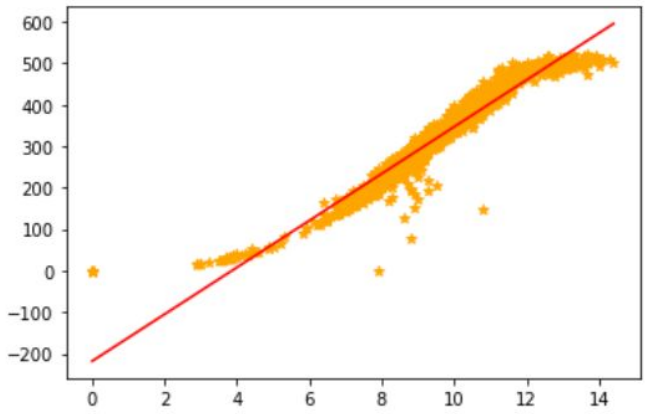
```
for i in range(len(y)):
    sqe = sqe + (y[i]-y_preditor[i])**2

for i in range(len(y)):
    sy = sy + (y[i]-y_media)**2
```

Após isso, foi calculado o coeficiente de determinação R^2 e o coeficiente de determinação ajustado R^2aj.

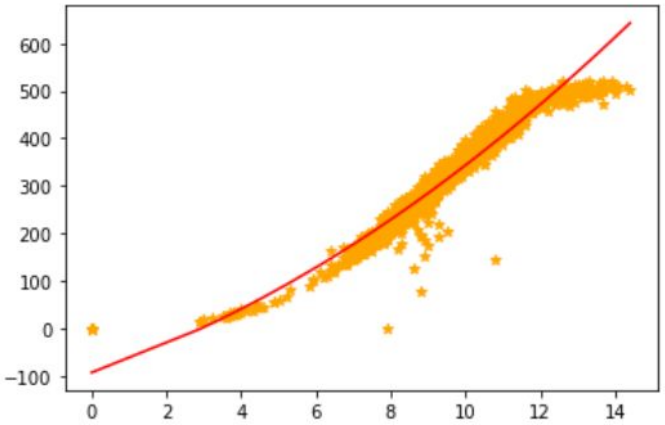
```
r2 = 1 - (sqe/sy)
r2aj = 1 - ((sqe/(len(y)-(k+1))) / (sy/(len(y)-1)) )
```

Conforme foi requisitado na questão, o código deve determinar os modelos de regressão polinomial com graus de 1 a 5. Abaixo, pode-se visualizar as saídas do algoritmo.

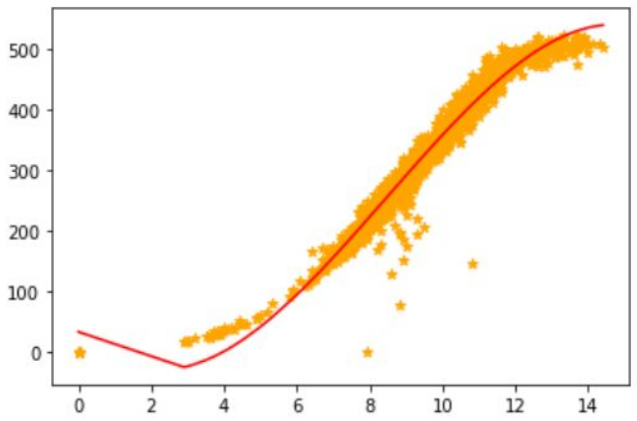
```
GRAU DO POLINOMIO: 1
O coeficiente de determinação é dado por: 0.9291604695978947
O coeficiente de determinação ajustado é dado por: 0.9291289573512745
```



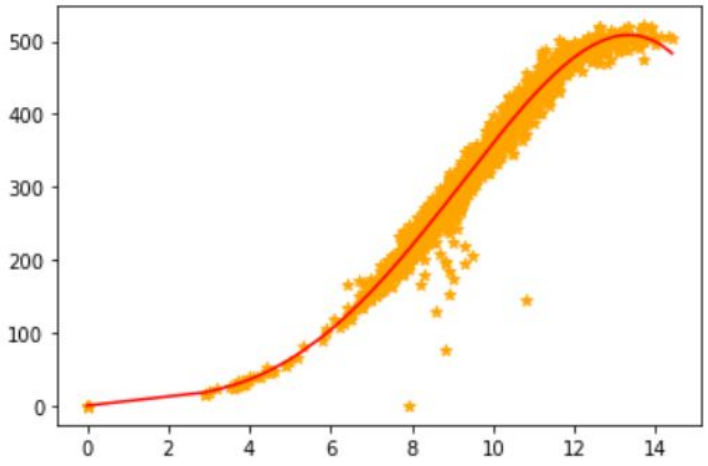

```
GRAU DO POLINOMIO: 2
O coeficiente de determinação é dado por: 0.9434238833776911
O coeficiente de determinação ajustado é dado por: 0.9433735263535502
```



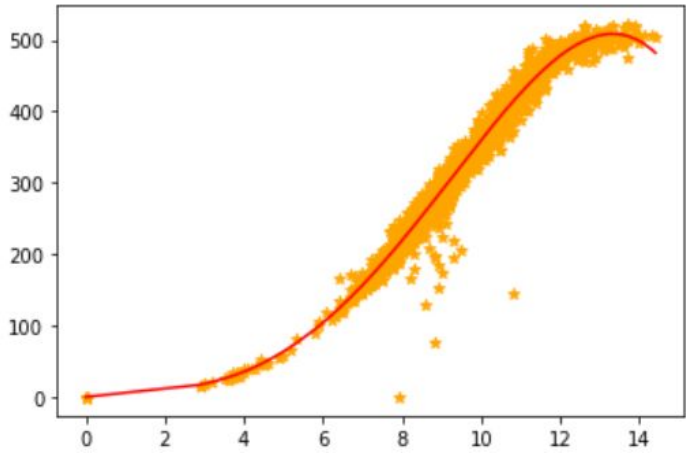

```
GRAU DO POLINOMIO: 3
O coeficiente de determinação é dado por: 0.9690229223762248
O coeficiente de determinação ajustado é dado por: 0.9689815460481432
```



```
GRAU DO POLINOMIO: 4
O coeficiente de determinação é dado por: 0.9737242419030897
O coeficiente de determinação ajustado é dado por: 0.9736774254075942
```




```
GRAU DO POLINOMIO: 5
O coeficiente de determinação é dado por: 0.9737255940945313
O coeficiente de determinação ajustado é dado por: 0.9736670504093586
```



2. QUESTÃO 2

Inicialmente, as bibliotecas necessárias foram importadas e os valores de entrada e saída foram adicionados ao código.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import axes3d
```

```
x1 = [122,114,86,134,146,107,68,117,71,98]
x2 = [139,126,90,144,163,136,61,62,41,120]
y = [0.115,0.120,0.105,0.090,0.100,0.120,0.105,0.080,0.100,0.115]
```

Após isso, a Matriz X foi preenchida utilizando esses valores.

```
#Cria a MATRIZ X
X = np.zeros((len(x1),3), dtype=np.float64)
for i in range(len(x1)):
    X[i][0] = 1
    X[i][1] = x1[i]
    X[i][2] = x2[i]
X
```

```
array([[  1., 122., 139.],
       [  1., 114., 126.],
       [  1.,  86.,  90.],
       [  1., 134., 144.],
       [  1., 146., 163.],
       [  1., 107., 136.],
       [  1.,  68.,  61.],
       [  1., 117.,  62.],
       [  1.,  71.,  41.],
       [  1.,  98., 120.]])
```

Os mesmos passos seguidos na questão 01 foram seguidos para desenvolver o resto da questão.

```
B = np.linalg.inv(X.T@X)@(X.T@y)
y_preditor = X@B
e_residuo = y - y_preditor
```

```
y_media = 0
sqe = 0
sy = 0
for k in range(len(y)):
    y_media = y_media + y[k]
y_media= y_media/len(y)

for n in range(len(y)):
    sqe = sqe + (y[n]-y_preditor[n])**2

for m in range(len(y)):
    sy = sy + (y[m]-y_media)**2

r2 = 1 - (sqe/sy)
r2aj = 1 - ((sqe/(len(y)-(3))) / (sy/(len(y)-1)) )
```

Após isso, foi calculado o coeficiente de determinação R^2 e o coeficiente de determinação ajustado R^2aj. Os valores foram plotados em um plano 3D.

```
print("O valor de B:",B)
print("O coeficiente de determinação é dado por:",r2)
print("O coeficiente de determinação ajustado é dado por:",r2aj)
fig = plt.figure()
axi = fig.add_subplot(111, projection='3d')
axi.scatter(x1,x2,y,cmap='hsv')
axi.set_xlabel('X-label')
axi.set_xlabel('Y-label')
axi.set_xlabel('Z-label')
plt.show()
```

```
O valor de B: [ 0.12737897 -0.00065924  0.00044084]
O coeficiente de determinação é dado por: 0.7238823500872479
O coeficiente de determinação ajustado é dado por: 0.6449915929693186
```