AC – Aprendizagem Computacional / Machine Learning

P0a – NumPy

Jorge Henriques
jh@dei.uc.pt

Departamento de Engenharia Informática
Faculdade de Ciências e Tecnologia

- *We assume knowledge of basic programming principles*

    - *Cycles*
    - *Selection (if..then...else)*
    - *File access*
    - *Variables and data type*
    - *Vectors/matrices*

    - *Two libraries*
        - ***Numpy*** *numerical calculus*
        - ***Matplotlib*** *plot*

python

NumPy matplotlib

Contents

- 1| Directories
- 2| Numpy
  - Vectors and Matrices
  - Matrix operations
  - Statistics
  - Linear regression
- 3| Matplotlib

ANACONDA

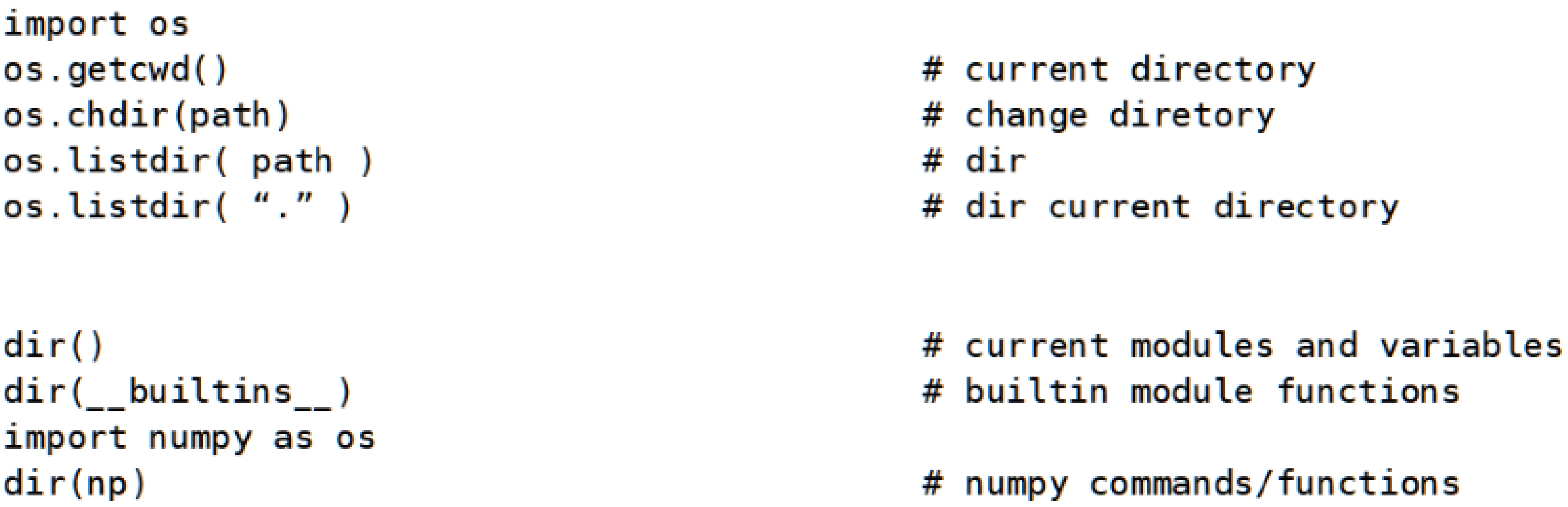
https://www.anaconda.com/download

Python Environment

spyder

Editor

1| Directories



```
import os
os.getcwd()                                  # current directory
os.chdir(path)                               # change diretory
os.listdir( path )                           # dir
os.listdir( “.” )                            # dir current directory


dir()                                        # current modules and variables
dir(__builtins__)                            # builtin module functions
import numpy as os
dir(np)                                      # numpy commands/functions
```

Exercise 0
>> print(os.listdir(“.”))

- **Tabular data**
  - **Numerical Py**
    - Supports processing of arrays (vectors) and matrices
    - Provides mathematical functions to operate on these matrices

ID	Name	Gender	Age	Weight	Height
1	GromencyMaria	F	23	90	173
2	Hasdrubal of the Incarnation	M	22	45	156
3	Idalécio Caroço	M	38	88	188
4	Virgolino Botija	M	99	78	167
…					
99	Joaquina Marreca	F	67	65	166
100	Anastácia Sardinha	F	77	56	123

numpy

Importar numpy

```
> import numpy as np
```

Vectors and Matrices

Create, access, modify

- Create from a file

```
> A= np.loadtxt(“dados.txt”)
```

- Create explicitly

```
> a = np.array([1, 4, 5, 8], float)
```

```
  [ 1., 4., 5., 8.]
```

```
> np.linspace(0,2,10)
```

```
  [0.        , 0.22222222, 0.44444444, 0.66666667, 0.88888889,
          1.11111111, 1.33333333, 1.55555556, 1.77777778, 2.        ]
```

2| numpy

values.txt

```
12.20
13.10
14.30
8.01
9.22
10.20
4.01
7.01
18.20
9.19
```

Access to values

```
> a = np.array([1, 4, 5, 8], float)
> a[3]
  8.0
> a[:2]
  [ 1., 4.]
```

Modify values

```
> a = np.array([1, 4, 5, 8], float)
> a[0] = 5.
> a
  [ 5., 4., 5., 8.]
> a.fill(0)
  [ 0., 0., 0., 0.]
```

Exercise 1 – numerical values from a file – values.txt

- Access an element or multiple elements
- values[3] ?
- values[3:4] ?
- values[2:5] ?

Create a Matrix - bidimensional

```
> a = np.array([[1, 2, 3], [4, 5, 6]], float)
> a
  [[ 1., 2., 3.],
  [ 4., 5., 6.]]
> a[0,0]
  1.0
> a[0,1]
  2.0

> np.ones((N,M))
> np.zeros((N,M))
> np.zeros_like(a)
> np.ones_like(a)
> np.identity(4, dtype=float)
> np.eye(4, k=1, dtype=float)
```

```
# matriz de uns dimensão (N
# matriz de zeros dimensão
# matriz de zeros com a dir
# matriz de uns com a dimer
# matriz identidade
# matriz identidade, diagonal valores= k
```

```
> a.shape
> a.size
```

```
> X=np.ones((N,M))
> Y=np.zeros((N,M))
> Z=np.concatenate((X,Y), axis=0)
> Z=np.concatenate((X,Y), axis=1)
```

WH.txt

```
65 167
52 145
89 189
98 198
75 175
74 174
77 180
89 155
89 157
98 201
```

Exercise 2 – numeric matrices – WH.txt

- Access an element
- Access all weights (column)
- Access a student's data (row)
- How many columns?
- How many lines?

Some functions

```
> a = np.array([2, 4, 3], float)
> a.sum()
  9.0
> a.prod()
  24.0
```

```
> a = np.array([2, 1, 9], float)
> a.mean()                                  # mean
  4.0
> a.var()                                   # variance
  12.666666666666666
> a.std()                                   # standard deviation
  3.5590260840104371
```

```
> a = np.array([2, 1, 9], float)
> a.min()
  1.0
> a.max()
  9.0
```

```
> a = np.array([6, 2, 5, -1, 0], float)
> a.sort()
```

values.txt

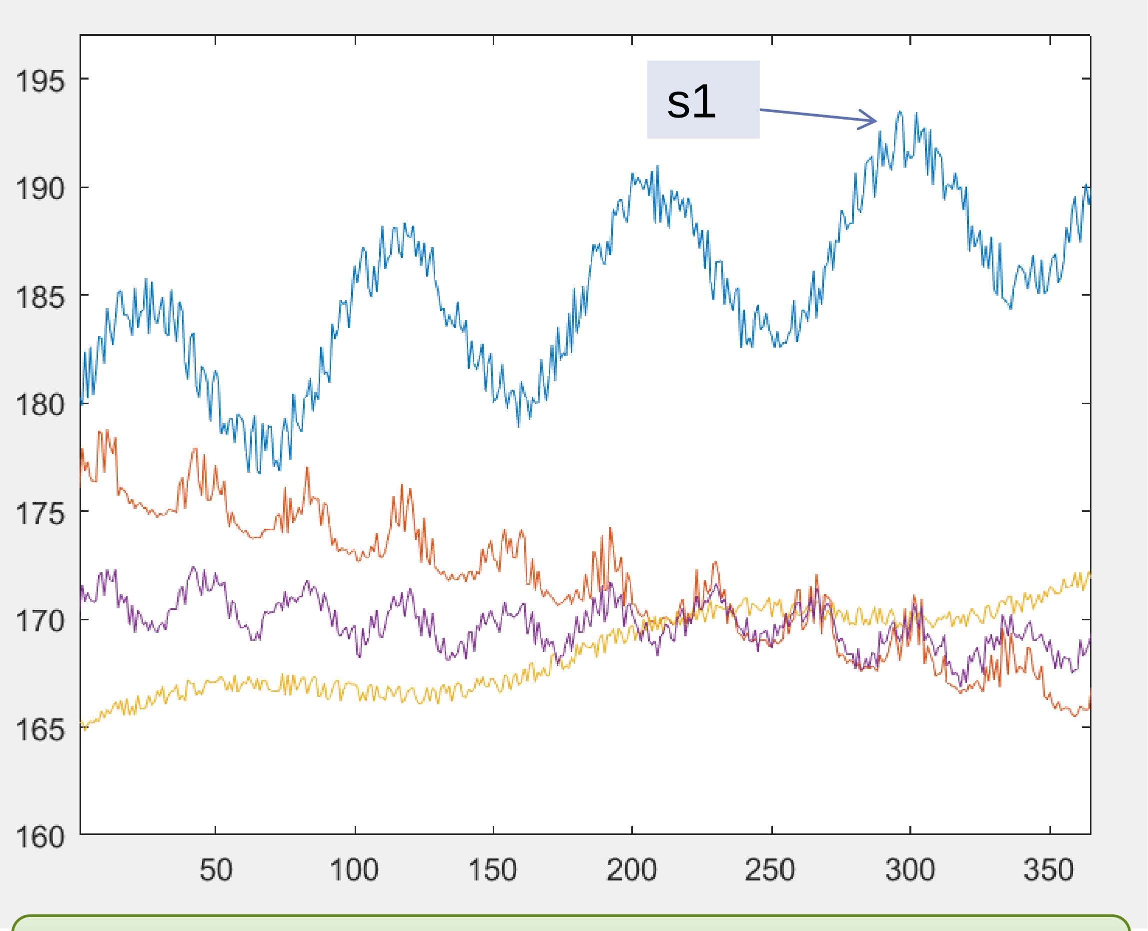
```
12.20
13.10
14.30
8.01
9.22
10.20
4.01
7.01
18.20
9.19
```

Cycles

Repetition

Exercise 3 – average and sum of values

- Stock market shares



Exercise 4 – maximum value of share value (s1)

Shares.txt

Day s1 s2 s3 s4
.....

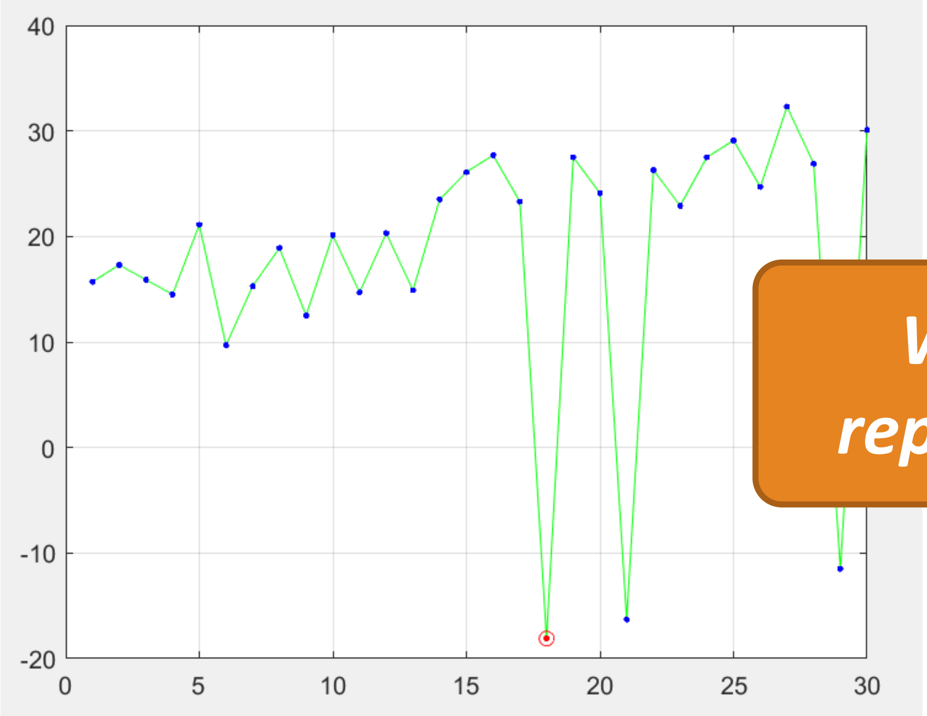
365 lines (one year)

Calculate

- What is the maximum value?
- What day?

If..then..else Selection

- Exercise 5 – Alarm!!
  - Faulty sensor



While repetition

- Day?
- What is the maximum value?

Calculate

- First day it is not working

temperatures.txt

```
15.7000
17.3000
15.9000
14.5000
14.7000
20.3000
14.9000
23.5000
26.1000
27.7000
23.3000
 -18.1000
 21.1000
9.7000
15.3000
18.9000
12.5000
20.1000
```

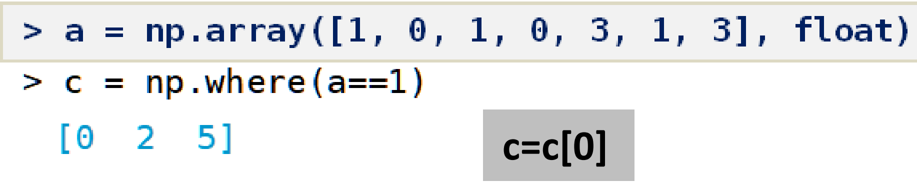
2| numpy

```
> a = np.array([6, 2, 5, -1, 0], float)
> a.clip(0, 5)                          # between two values
  [ 5., 2., 5., 0., 0.]
```

```
> a = np.array([1, 1, 4, 5, 5, 5, 7], float)
> np.unique(a)                          # unique values
  [ 1., 4., 5., 7.]
```

```
> a = np.array([1, 3, 0], float)
> b = np.array([0, 3, 2], float)
> a > b
  [ True, False, False]
> a == b
  [False, True, False]
> a <= b
  [False, True, True]
```

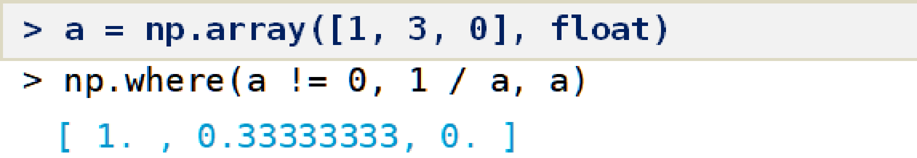
2| numpy



```
> a = np.array([1, 0, 1, 0, 3, 1, 3], float)
> c = np.where(a==1)
  [0  2  5]
```

c=c[0]

# find indexes



```
> a = np.array([1, 3, 0], float)
> np.where(a != 0, 1 / a, a)
  [ 1. , 0.33333333, 0. ]
```

# find indexes and replace

- Exercise 6 – Missing values!
  - Weight / Height /ShoeSize/ Gender
  - Gender = {0,1} = {Fem, Male}

Students.txt

```
177.69 80.88 43.00     0
188.84 79.68 44.00     0
180.30 75.58 42.00     1
184.40 72.90 43.00     1
168.63 86.85 45.00     1
178.01 69.29 46.00     0
191.10 67.28 41.00     0
 185.08 73.30 -1.00    0
 162.62 70.35 44.00    1
169.32 69.09 40.00     0
 168.90 74.83 -1.00    1
170.15 68.34 41.00     0
177.30 57.56 37.00     1
187.97 62.90 41.00     0
174.89 68.56 42.00     0
……
```

- Replace ShoeSize missing
  - 1| For 40
  - 2| By the average of valid sizes
  - 3|
    - 42 if Height>175
    - 39 if Height <=175

Matrix operations

Multiplication, determinant, Eigenvalues and eigenvectors, Inverse, pseudoinverse
The dot function is used for inner product and for multiplication of matrices

```
> a = np.array([1, 2, 3], float)
> b = np.array([0, 1, 1], float)
>a+b
  [ 1., 3., 5.],
>a-b
  [ 1., 1., 2.],
>a*b
  [ 0., 2., 3.],
> dist = numpy.linalg.norm(a-b)
  2,4494897427
```

# product element by element

# distance between two vectors

2| numpy

```
> a = np.array([1, 2, 3], float)
> b = np.array([0, 1, 1], float)
> np.dot(a, b)
  5.0
```

# dot product

```
> a = np.array([[4, 2, 0], [9, 3, 7], [1, 2, 1]], float)
> b = np.array([[1, 1], [0, -1], [1, 0]], float)
> np.dot(a,b)
    [ 4.,  2.],
    [16.,  6.],
    [ 2., -1.]])
```

# matrix multiplication

2| numpy

```
> np.linalg.det(a)                                  # determinant
  -53.999999999999993
> vals, vecs = np.linalg.eig(a)                     # eigenvalues
> vals
  [ 9.  , 2.44948974, -2.44948974]
> vecs
  [[-0.3538921 , -0.56786837, 0.27843404],
  [-0.88473024, 0.44024287, -0.89787873],
  [-0.30333608, 0.69549388, 0.34101066]]

> vals, vecs = np.linalg.eig(a)[0]                  # eigenvalues
> vals, vecs = np.linalg.eig(a)[1]                  # eigenvectores

> c = np.linalg.inv(a)
  [[ 0.14814815, 0.07407407, -0.25925926],
  [ 0.2037037 , -0.14814815, 0.51851852],
  [-0.27777778, 0.11111111, 0.11111111]]

> c = np.linalg.pinv(b)                             # pseudo-inverse
  [[ 0.33333333  0.33333333  0.66666667]
  [ 0.33333333 -0.66666667 -0.33333333]]
```

Statistics

Median, mean, variance, standard deviation, correlation coefficient, covariance

```
> a = np.array([1, 4, 3, 8, 9, 2, 3], float)
> np.median(a)
  3.0
> a.mean()
  4.285714285714286
> a.var()
  7.918367346938775
> a.std()
  2.813959371941744
```

```
> a = np.array([1, 2, 3], float)
> b = np.array([0, 1, 1], float)
> cc= np.corrcoef(a, b) )
  [[1.        0.8660254]
   [0.8660254 1.        ]]
```

# correlation

```
> a = np.array([[1, 2, 1, 3], [5, 3, 1, 8]], float)
> np.cov(a)
  [[ 0.91666667, 2.08333333],
  [ 2.08333333, 8.91666667]])
```

# co-variance

Linear Regression

Assume data is read from a file of size N,M
Input = Column 1 to M-1
Output = Column M
Yp = q0 + q1 col1 + q2 col2 + … + q(M-1) col(M-1)

```
> D=np.loadtxt("dados.txt")                  # D=(N,m)
> n,m=D.shape
> X=D[:,0:m-1];
> Y=D[:,m-1];
> Z= np.ones((n,1))
> X= np.concatenate((Z,X), axis=1)

> RESULT = np.linalg.lstsq(X,Y,rcond=None)   # Model
> par=RESULT[0]
> sumErro=RESULT[1]

> YP=np.dot(X,par)                           # Estimate values and error
> erro=Y-YP
```

matplotlib

matplotlib

Importar numpy

```
> import matplotlib.pyplot as plt
```

- Exercise 6 – Missing values!
  - Weight / Height /ShoeSize/ Gender
  - Gender = {0,1} = {Fem, Male}

Students.txt

```
177.69 80.88 43.00    0
188.84 79.68 44.00    0
180.30 75.58 42.00    1
184.40 72.90 43.00    1
168.63 86.85 45.00    1
178.01 69.29 46.00    0
191.10 67.28 41.00    0
 185.08 73.30 -1.00   0
 162.62 70.35 44.00   1
169.32 69.09 40.00    0
 168.90 74.83 -1.00   1
170.15 68.34 41.00    0
177.30 57.56 37.00    1
187.97 62.90 41.00    0
174.89 68.56 42.00    0
……
```

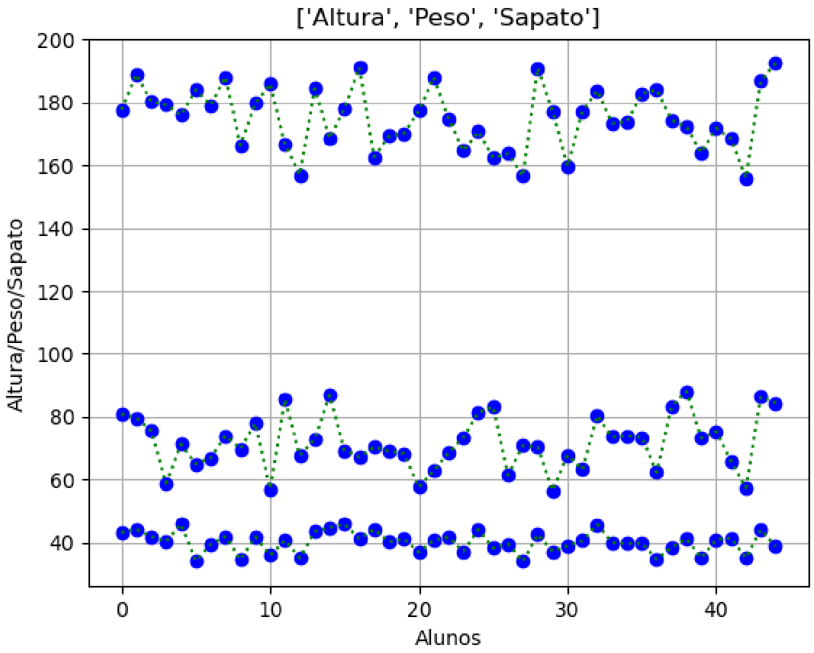
- Replace ShoeSize missing
  - 1| For 40
  - 2| By the average of valid sizes
  - 3|
    - 42 if Height>175
    - 39 if Height <=175

Exemplo 1

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

D=np.loadtxt("P0_ALUNOS.txt")
n,m=D.shape
X=D[:,0:m-1];
Y=D[:,m-1];
t=range(0,n)
valores = ['Altura','Peso','Sapato','Genero']
```

```
plt.figure(1)
plt.plot(t, X,'bo',t, X,'g:')
plt.title(str(valores[0:3]) )
plt.xlabel('Alunos')
plt.ylabel('Altura/Peso/Sapato')
plt.grid()
plt.show()
```



- **plot (mark color)**

b	blue
g	green
r	red
c	cyan
m	magenta
y	yellow
k	black
w	white

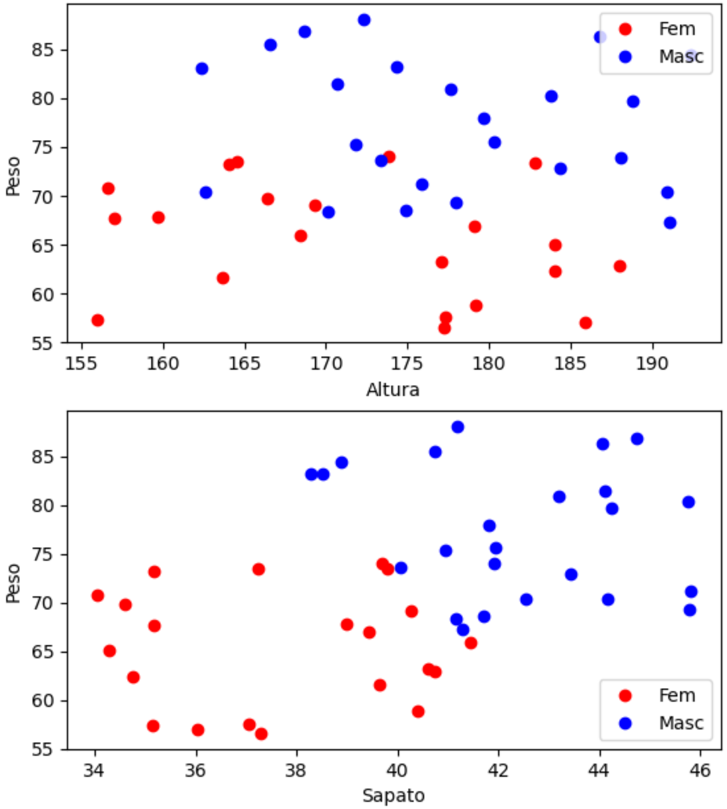
.	point
o	circle
x	x-mark
+	plus
*	star
s	square
d	diamond
v	triangle (down)
^	triangle (up)
<	triangle (left)
>	triangle (right)
p	pentagram
h	hexagram
-	solid
:	dotted
-:	dashdot
--	dashed

Exemplo 2

```
id0 = np.where(Y==0)
id0 = id0[0]
id1 = np.where(Y==1)
id1 = id1[0]

plt.figure(3)
plt.subplot(2, 1, 1)
plt.plot(altura[id0],peso[id0],'ro',altura[id1],peso[id1],'bo' )
plt.legend(["Fem", "Masc"], loc ="upper right")
plt.xlabel('Altura')
plt.ylabel('Peso')

plt.subplot(2, 1, 2)
plt.plot(sapato[id0],peso[id0],'ro',sapato[id1],peso[id1],'bo' )
plt.legend(["Fem", "Masc"], loc ="lower right")
plt.xlabel('Sapato')
plt.ylabel('Peso')
plt.show()
```



- Exercises
  - *importyou*
  - *import numpyto thenp*
  - *import matplotlib.pyplotto theplt*

  - *#exercise0 -boards*
  - *#exercise1 –vectors- “notes.txt”*
  - *#exercise2 – matrices – “PA.txt" students*
  - *#exercise3 –functions/maximum/plot– “scholarship.txt”*
  - *#exercise4 –functions/ envelope – “scholarship.txt” (column 3 and column 4)*
  - *#exercise5 – faulty sensor “temperatures,txt”*

  - *#exercise6 – Betwhat elsegrew (difference between the last and first value) – stock market*
  - *#exercise7 – Missing values “students.txt”thirdcolumnn~shoe(-1)*