Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

Nuestro programa para esta sesión:

Esta segunda clase será dividida en dos partes:

- ✓ Parte 1 (20 minutos):

  Instalación de Python3 + Anaconda + PyCharm (opcional)

- ✓ Coffee-break (15 minutos):

  Café + galletas + conversación distendida c/r a Python (videos)

- ✓ Parte 2 (45 minutos):

  Revisión de visualización de data usando Matplotlib

Total de la clase: 1 hora 20 minutos

Autor: Jorge Felipe Monardes Pinto

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

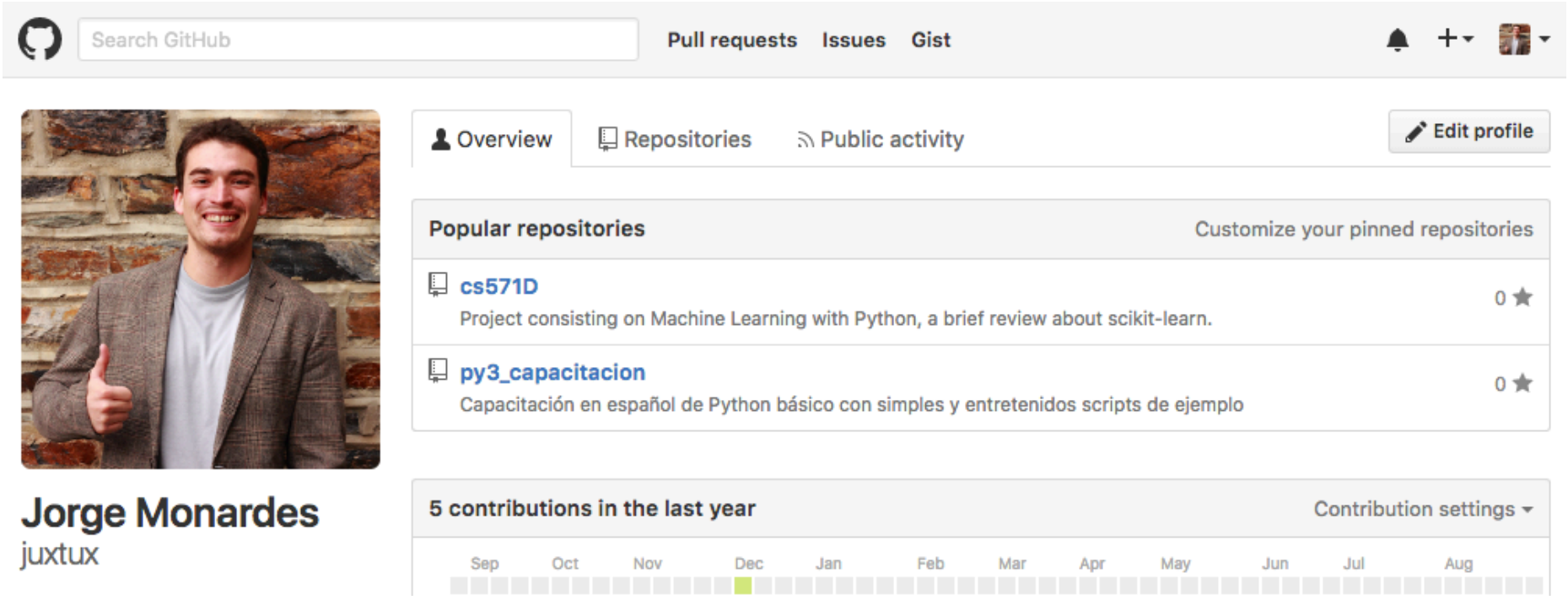
Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

Fuente

Visitar mi repositorio Github : https://github.com/juxtux

Documentación y scripts : https://github.com/juxtux/py3_capacitacion



python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

PARTE I

+

COFFEE-BREAK

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

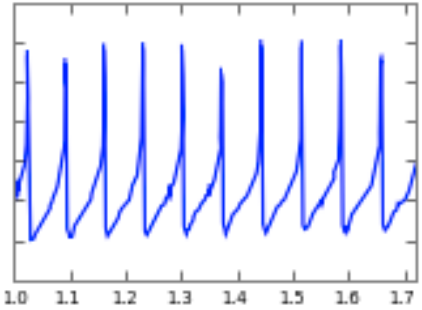
matplotlib

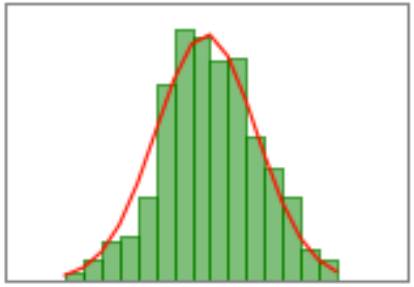
A Fiscally Sponsored Project of
NUMFOCUS
OPEN CODE = BETTER SCIENCE

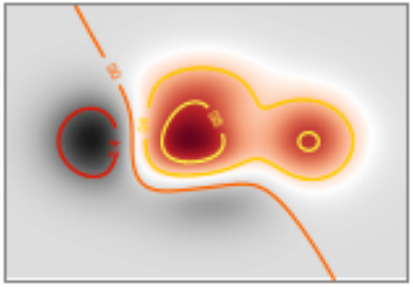
John Hunter (1968-2012)

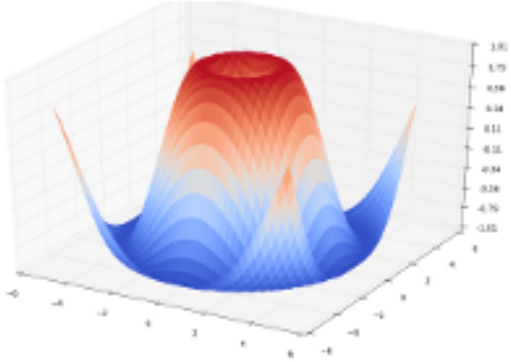
On August 28 2012, John D. Hunter, the creator of matplotlib, died from complications arising from cancer treatment, after a brief but intense battle with this terrible illness. John is survived by his wife Miriam, his three daughters Rahel, Ava and Clara, his sisters Layne and Mary, and his mother Sarah.

If you have benefited from John's many contributions, please say thanks in the way that would matter most to him. Please consider making a donation to the John Hunter Technology Fellowship.









python™

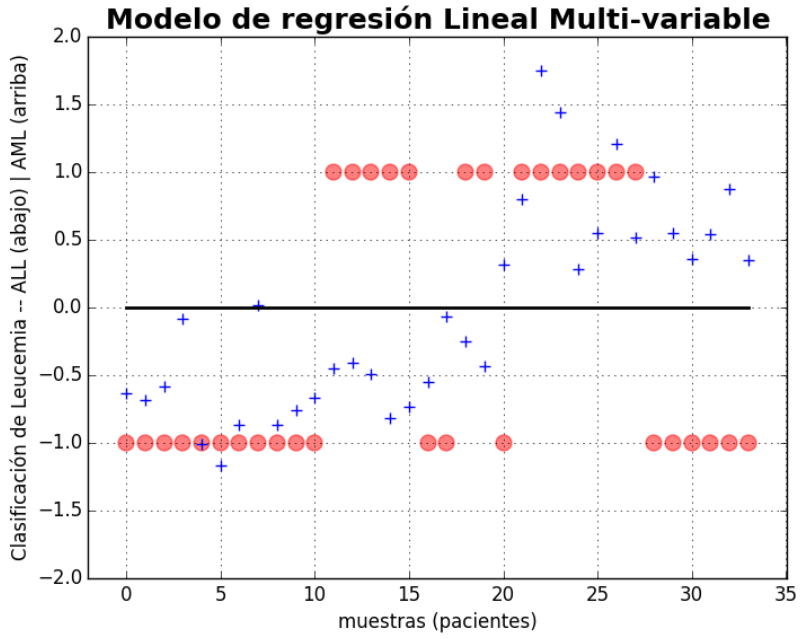
jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencia: http://matplotlib.org/

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```python
__author__ = 'Jorge Monardes'

import numpy as np
import matplotlib.pyplot as plt

test_y = np.random.uniform(-1, 1, 72)
real_y = np.random.uniform(-1, 1, 72)
axis_x = [i for i in range(0,len(test_y))]
sep_y = [0 for i in range(0,len(test_y))]


plt.plot(axis_x, sep_y, color='black',
         linewidth=2)
plt.scatter(axis_x, test_y, color='r',
            alpha=.5, s=100,
            label='y-test data real')
plt.scatter(axis_x, real_y,
            color='b', marker='+', s=60,
            label='y-test predicción del modelo')

plt.xlabel('muestras (pacientes)')
plt.ylabel('Clasificación de Leucemia -- '
           'ALL (abajo) | AML (arriba)')
plt.title('Modelo de regresión Lineal Multi-variable',
          fontsize=18, fontweight='bold')

plt.axis('tight')
plt.grid(True)
plt.ylim(-2, 2)
plt.savefig("myPlot.png")
plt.legend(loc='best')

plt.show()
```



Se recomienda:
http://matplotlib.org/api/pyplot_summary.html

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

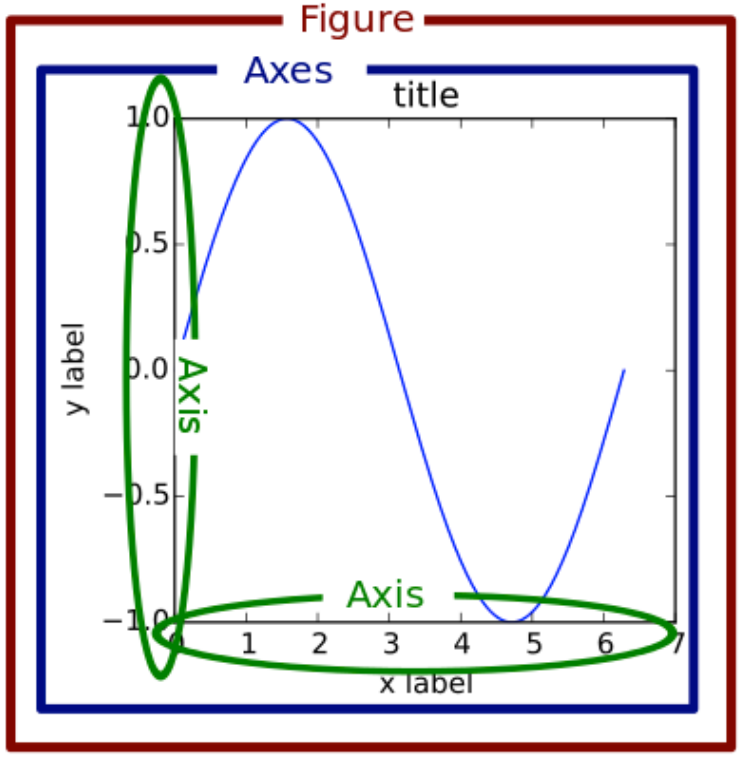
Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13           linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15              alpha=.5, s=100,
16              label='y-test data real')
17 plt.scatter(axis_x, real_y,
18              color='b', marker='+', s=60,
19              label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

…is a collection of command style functions that make matplotlib work like MATLAB.

Todo lo que hagan serán métodos sobre plt



python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencia: http://matplotlib.org/users/pyplot_tutorial.html

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15             alpha=.5, s=100,
16             label='y-test data real')
17 plt.scatter(axis_x, real_y,
18             color='b', marker='+', s=60,
19             label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

Simplemente data. Los datos que correspondan al eje x y al eje y, deben estar cuidadosamente seleccionados en forma vectorial, pero no necesariamente en **tipo**, i.e.,

- List()
- Numpy.array()

Ejemplo Numpy en consola

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencia: http://matplotlib.org/users/pyplot_tutorial.html

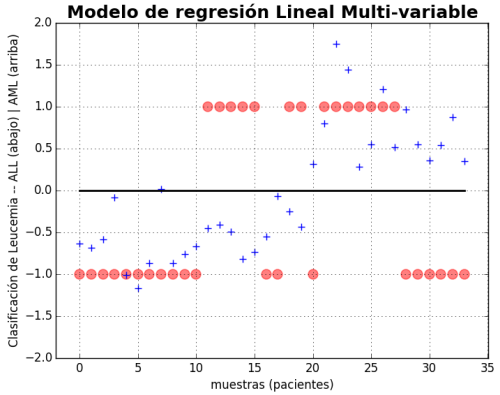
Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13           linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15              alpha=.5, s=100,
16              label='y-test data real')
17 plt.scatter(axis_x, real_y,
18              color='b', marker='+', s=60,
19              label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

plot() se utiliza para líneas y/o curvas continuas

- plot(x, y): vector con valores de eje-x & vector con valores para el eje-y
- plot(x, y, color='black'): color de la línea/curva, también se puede usar c='black'
- plot(x, y, c='black', linewidth=2): espesor de la línea/curva, se puede usar lw=2, el número utilizado puede ser decimal, e.g., lw=1.3
- ¿Donde está en el gráfico?



Referencia: http://matplotlib.org/users/pyplot_tutorial.html

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15             alpha=.5, s=100,
16             label='y-test data real')
17 plt.scatter(axis_x, real_y,
18             color='b', marker='+', s=60,
19             label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

scatter() se utiliza para puntos de dispersión

- scatter(x, y): vector con valores de eje-x & vector con valores para el eje-y
- scatter(x, y, color='r'): color de la línea/curva, también se puede usar c='red' o c='r'
- scatter(x, y, c='r', alpha=.5): grado de transparencia de los puntos de dispersión, el número utilizado debe ser entre 0 (transparente) y 1 (opaco).
- scatter(x, y, c='r', alpha=.5, s=100): se refiere al tamaño que tendrán cada uno de los puntos. Utilizar "size" no aplica y debiese dar errores.
- scatter(x, y, c='r', alpha=.5, s=100, label='texto'): corresponde a la etiqueta que utilicemos para esa serie de datos.

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencia: http://matplotlib.org/users/pyplot_tutorial.html

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15              alpha=.5, s=100,
16              label='y-test data real')
17 plt.scatter(axis_x, real_y,
18              color='b', marker='+', s=60,
19              label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

scatter() IDEM

- scatter(x, y, color='b'): 'b' se refiere a blue, por lo que esta serie será azul.
- scatter(x, y, color='b', marker='+'): tipo de marcador que queramos utilizar. La variedad estándar disponible abajo (la lista de abajo continua, para más revizar link de referencia.)

marker	description
"."	point
","	pixel
"o"	circle
"v"	triangle_down
"^"	triangle_up
"<"	triangle_left
">"	triangle_right
"1"	tri_down
"2"	tri_up
"3"	tri_left
"4"	tri_right
"8"	octagon
"s"	square
"p"	pentagon
"*"	star

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencia: http://matplotlib.org/api/markers_api.html

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15             alpha=.5, s=100,
16             label='y-test data real')
17 plt.scatter(axis_x, real_y,
18             color='b', marker='+', s=60,
19             label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

xlabel() & ylabel()

- xlabel('texto…'): simplemente el texto para el eje-x y eje-y, respectivamente.
- xlabel('texto…', fontsize=12, color='b'): es posible realizar más especificaciones de estos métodos, revisar referencias y Googlear!

Referencia: http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.xlabel

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15              alpha=.5, s=100,
16              label='y-test data real')
17 plt.scatter(axis_x, real_y,
18              color='b', marker='+', s=60,
19              label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

title(): titulo de la figura.

- title('texto…'): simplemente el texto para el gráfico.
- title('texto…', fontsize=18, fontweight='bold'): IDEM a anterior
- title('texto…', fontsize=18, fontweight='bold', loc='center'): pruebenlo, con 'center', 'left', y 'right'.

Referencia: http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.title

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

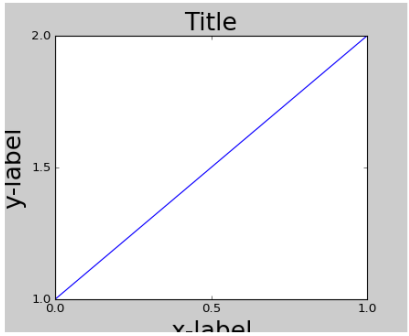
Clase 2: Python Setup & Visualization

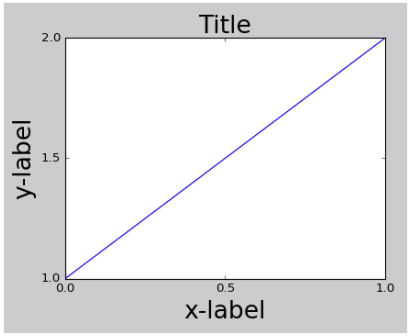
everis
an NTT DATA Company

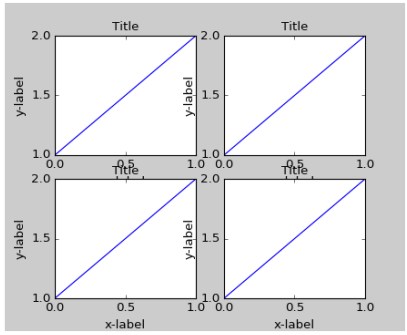
```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15              alpha=.5, s=100,
16              label='y-test data real')
17 plt.scatter(axis_x, real_y,
18              color='b', marker='+', s=60,
19              label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

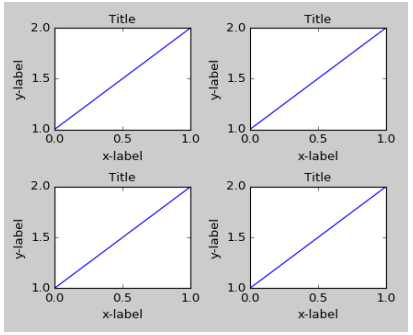
axis(): control de ejes

- axis('tight'):









python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

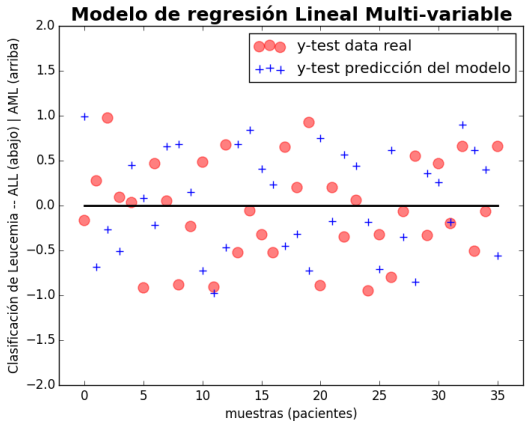
Referencia: http://matplotlib.org/users/tight_layout_guide.html

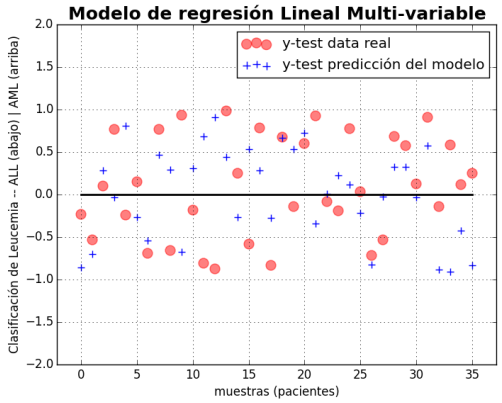
Clase 2: Python Setup & Visualization

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15             alpha=.5, s=100,
16             label='y-test data real')
17 plt.scatter(axis_x, real_y,
18             color='b', marker='+', s=60,
19             label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```

grid(): control de grilla

- grid(False) & grid(True):



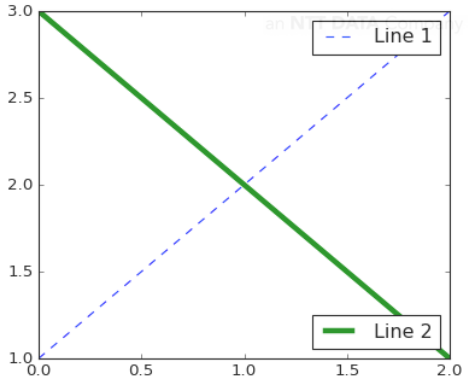


jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

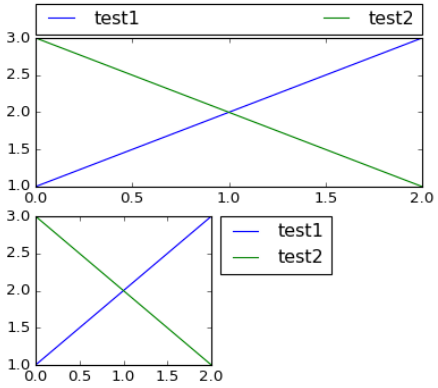
Referencia: http://matplotlib.org/api/axis_api.html

Clase 2: Python Setup & Visualization

```
1  __author__ = 'Jorge Monardes'
2
3  import numpy as np
4  import matplotlib.pyplot as plt
5
6  test_y = np.random.uniform(-1, 1, 72)
7  real_y = np.random.uniform(-1, 1, 72)
8  axis_x = [i for i in range(0,len(test_y))]
9  sep_y = [0 for i in range(0,len(test_y))]
10
11
12 plt.plot(axis_x, sep_y, color='black',
13          linewidth=2)
14 plt.scatter(axis_x, test_y, color='r',
15             alpha=.5, s=100,
16             label='y-test data real')
17 plt.scatter(axis_x, real_y,
18             color='b', marker='+', s=60,
19             label='y-test predicción del modelo')
20
21 plt.xlabel('muestras (pacientes)')
22 plt.ylabel('Clasificación de Leucemia -- '
23            'ALL (abajo) | AML (arriba)')
24 plt.title('Modelo de regresión Lineal Multi-variable',
25           fontsize=18, fontweight='bold')
26
27 plt.axis('tight')
28 plt.grid(True)
29 plt.ylim(-2, 2)
30 plt.savefig("myPlot.png")
31 plt.legend(loc='best')
32
33 plt.show()
```



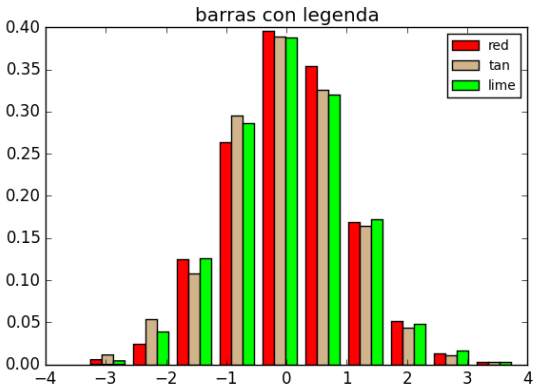
- ylim(-2, 2): límites del eje-y entre -2 y +2
- savefig('myPlot.png'): se guarda imagen en formato PNG del gráfico en el directorio de trabajo.
- legend(loc='best'): de establece necesidad de cuadro de legenda con la "mejor" localización encontrada.
- show(): es un *must have*

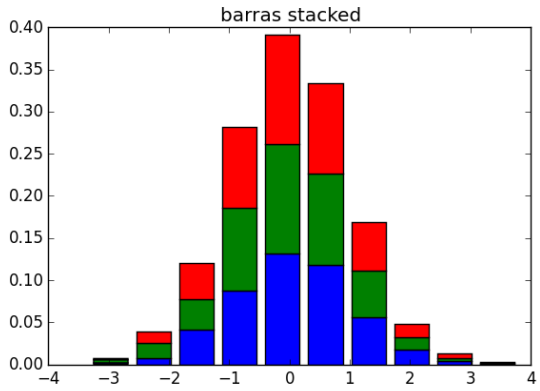


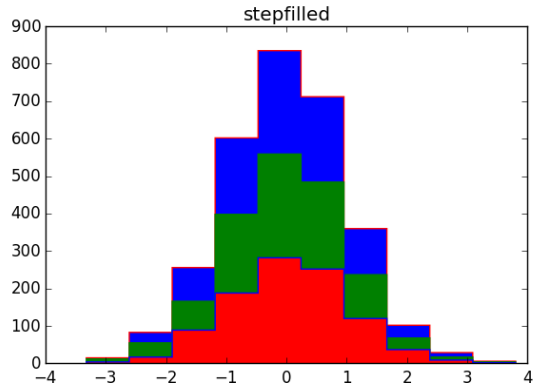
python™

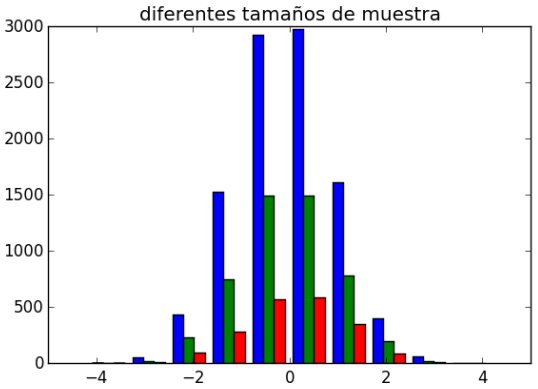
jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization









Subgráficos y objeto Figure

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```python
__author__ = 'Jorge Monardes'

"""
Esta es una adaptación del script original de
http://matplotlib.org/examples/statistics/
histogram_demo_multihist.html
para ilustrar principales componentes de
sub-gráficos en Matplotlib.
"""

import numpy as np
import matplotlib.pyplot as plt

n_bins = 10
x = np.random.randn(1000, 3)

fig, axes = plt.subplots(nrows=2, ncols=2)
ax0, ax1, ax2, ax3 = axes.flat

colors = ['red', 'tan', 'lime']
ax0.hist(x, n_bins, normed=1, histtype='bar',
         color=colors, label=colors)
ax0.legend(prop={'size': 10})
ax0.set_title('barras con legenda')

ax1.hist(x, n_bins, normed=1, histtype='bar',
         stacked=True)
ax1.set_title('barras stacked')

ax2.hist(x, n_bins, histtype='step',
         stacked=True, fill=True)
ax2.set_title('stepfilled')

# Make a multiple-histogram of
# data-sets with different length.
x_multi = [np.random.randn(n) for n in [10000, 5000, 2000]]
ax3.hist(x_multi, n_bins, histtype='bar')
ax3.set_title('diferentes tamaños de muestra')

plt.tight_layout()
plt.show()
```

→ Librerías de siempre

- Queremos diez quantiles (percentiles)
- Data random con distribución normal estandarizada N(0,1). Queremos 3,000 puntos divididos en tres grupos (matriz de mil filas y tres columnas.)

→ subplots(nrows=r, ncols=c): estrega una lista con el objeto figure y una lista de los objetos subgraphs que se generan mediante el número de filas "r" y el número de columnas "c".

En este elemplo plt.subplots(nrow=2, ncols=2):

$$\left[\ \text{plt.figure()}\ ,\ \begin{bmatrix}\text{subplot1()}\ ,\ \text{subplot2()}\\ \text{subplot3()}\ ,\ \text{subplot4()}\end{bmatrix}\right]$$

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```python
__author__ = 'Jorge Monardes'

"""
Esta es una adaptación del script original de
http://matplotlib.org/examples/statistics/
histogram_demo_multihist.html
para ilustrar principales componentes de
sub-gráficos en Matplotlib.
"""

import numpy as np
import matplotlib.pyplot as plt

n_bins = 10
x = np.random.randn(1000, 3)

fig, axes = plt.subplots(nrows=2, ncols=2)
ax0, ax1, ax2, ax3 = axes.flat

colors = ['red', 'tan', 'lime']
ax0.hist(x, n_bins, normed=1, histtype='bar',
         color=colors, label=colors)
ax0.legend(prop={'size': 10})
ax0.set_title('barras con legenda')

ax1.hist(x, n_bins, normed=1, histtype='bar',
         stacked=True)
ax1.set_title('barras stacked')

ax2.hist(x, n_bins, histtype='step',
         stacked=True, fill=True)
ax2.set_title('stepfilled')

# Make a multiple-histogram of
# data-sets with different length.
x_multi = [np.random.randn(n) for n in [10000, 5000, 2000]]
ax3.hist(x_multi, n_bins, histtype='bar')
ax3.set_title('diferentes tamaños de muestra')

plt.tight_layout()
plt.show()
```

fig = plt.figure()

axes = [subplot1() , subplot2()
         subplot3() , subplot4()]

axes.flat = [subplot1() , subplot2(), subplot3(), subplot4()]

ax0 = subplot1()

ax1 = subplot2()

ax2 = subplot3()

ax3 = subplot4()

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Clase 2: Python Setup & Visualization

everis
an NTT DATA Company

```python
__author__ = 'Jorge Monardes'

"""
Esta es una adaptación del script original de
http://matplotlib.org/examples/statistics/
histogram_demo_multihist.html
para ilustrar principales componentes de
sub-gráficos en Matplotlib.
"""

import numpy as np
import matplotlib.pyplot as plt

n_bins = 10
x = np.random.randn(1000, 3)

fig, axes = plt.subplots(nrows=2, ncols=2)
ax0, ax1, ax2, ax3 = axes.flat

colors = ['red', 'tan', 'lime']
ax0.hist(x, n_bins, normed=1, histtype='bar',
         color=colors, label=colors)
ax0.legend(prop={'size': 10})
ax0.set_title('barras con legenda')

ax1.hist(x, n_bins, normed=1, histtype='bar',
         stacked=True)
ax1.set_title('barras stacked')

ax2.hist(x, n_bins, histtype='step',
         stacked=True, fill=True)
ax2.set_title('stepfilled')

# Make a multiple-histogram of
# data-sets with different length.
x_multi = [np.random.randn(n) for n in [10000, 5000, 2000]]
ax3.hist(x_multi, n_bins, histtype='bar')
ax3.set_title('diferentes tamaños de muestra')

plt.tight_layout()
plt.show()
```

Same old story…

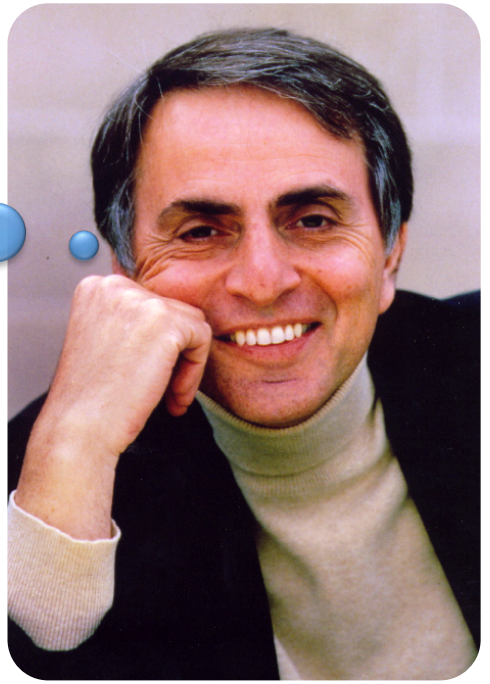
python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Últimas palabras…

everis
an NTT DATA Company

"If you want to make an apple pie from scratch, you must first, invent the universe."



Prof. Carl Sagan
1934 - 1996

python™

jorge.monardes.pinto@everis.com
jorge.monardes@duke.edu

Referencias: http://www.phys.lsu.edu