



Introducción Data Science con Python

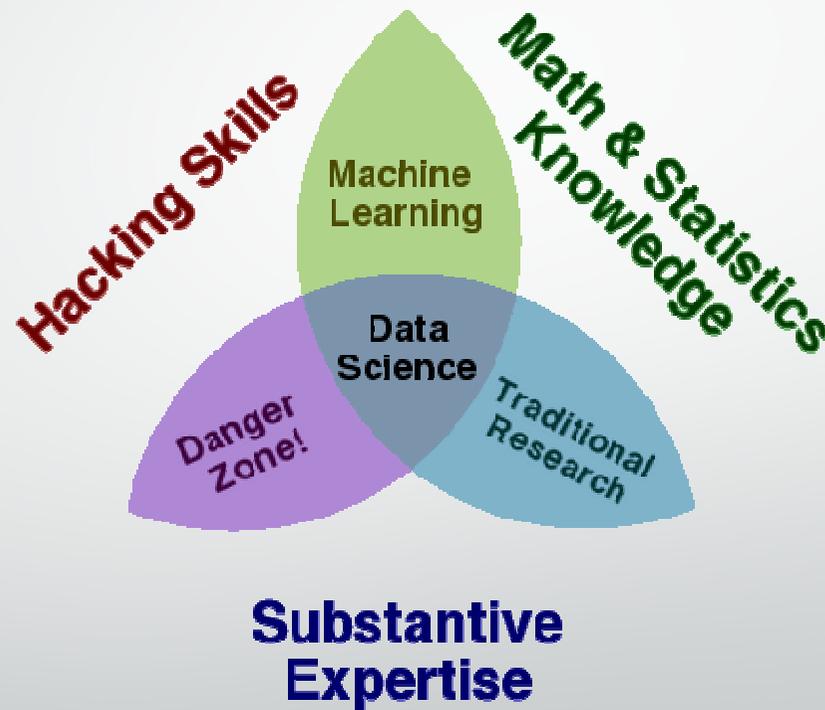
Nestor Castro

Industria 4.0



¿Qué es Data Science?

Data Science es un campo interdisciplinario que aplica técnicas matemáticas, estadísticas y computacionales a diversas áreas: biología, física, economía, sociología entre otras.



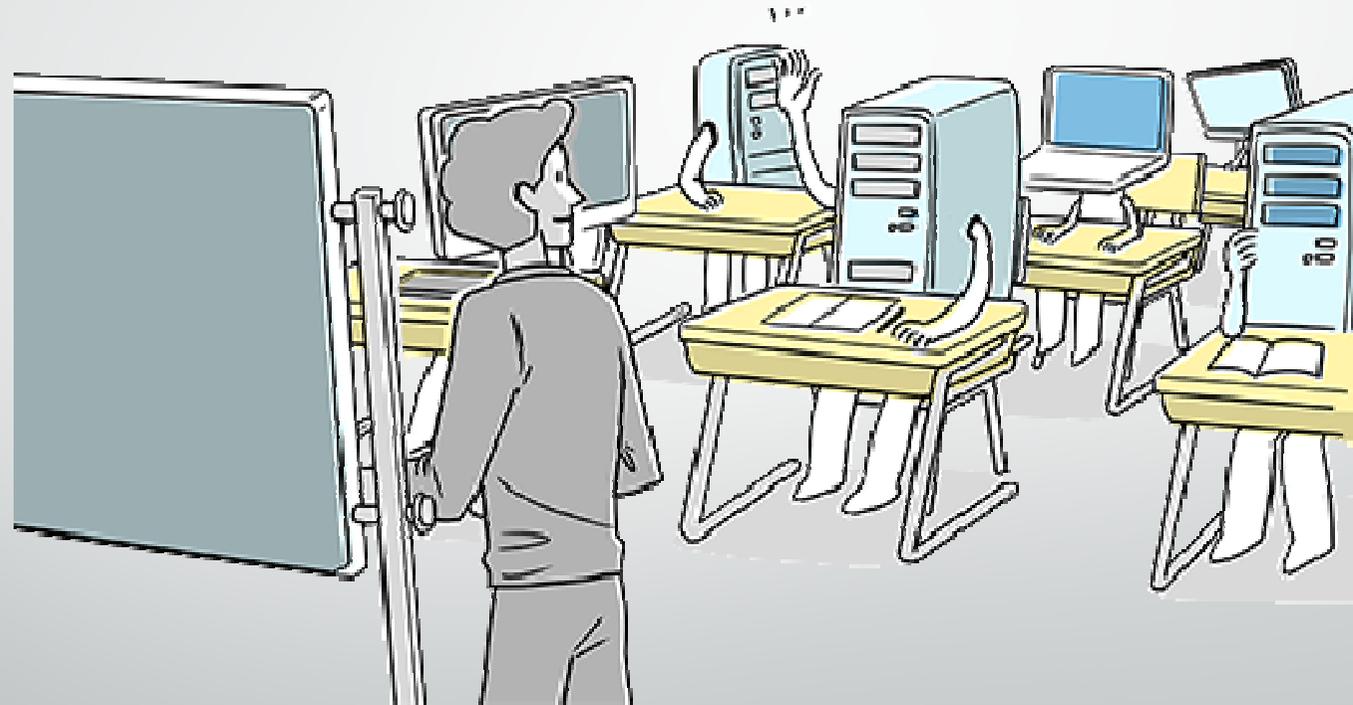
¿Qué es Data Science?

Data Science tiene la misión de modelar, analizar, entender, visualizar y extraer conocimiento a partir de datos.



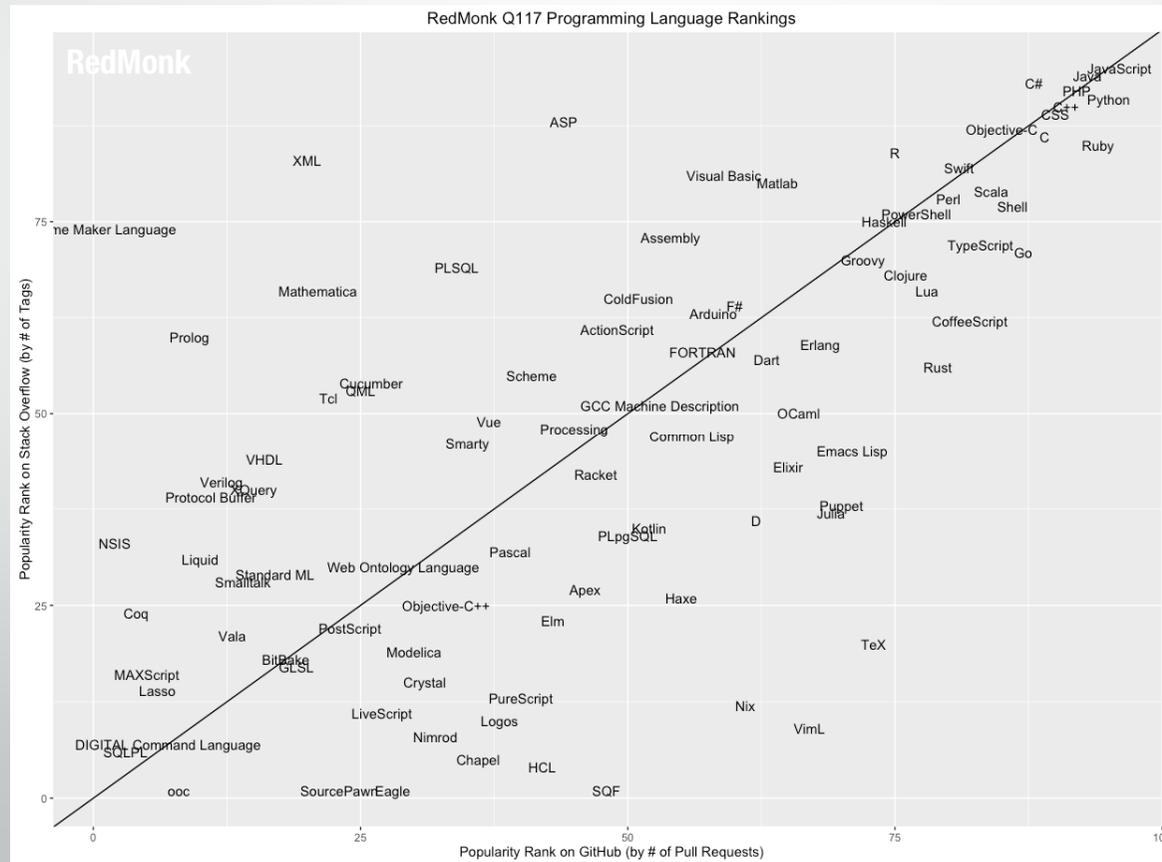
¿Qué es Machine Learning?

Machine Learning es una área cuyo objetivo es desarrollar algoritmos que permitan a los ordenadores aprender.



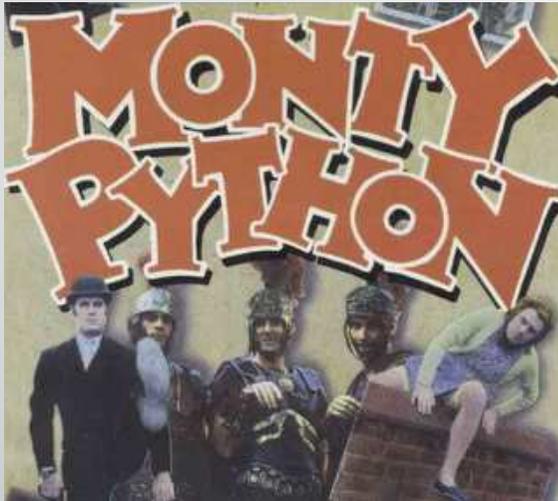
Claves para elegir un lenguaje apropiado

- Documentación y librerías acelera los tiempos de desarrollo
- Robusto
- Eficiente, veloz y escalable
- Contar con librerías diseñadas para Big Data



Python - Historia

- Monty Python (1969) - Grupo Humorista Británico
- Guido Van Rossum (1991)
- Python Software Foundation (2001)
- Python 2.7.13 / 3.6.1



Ecosistema Python

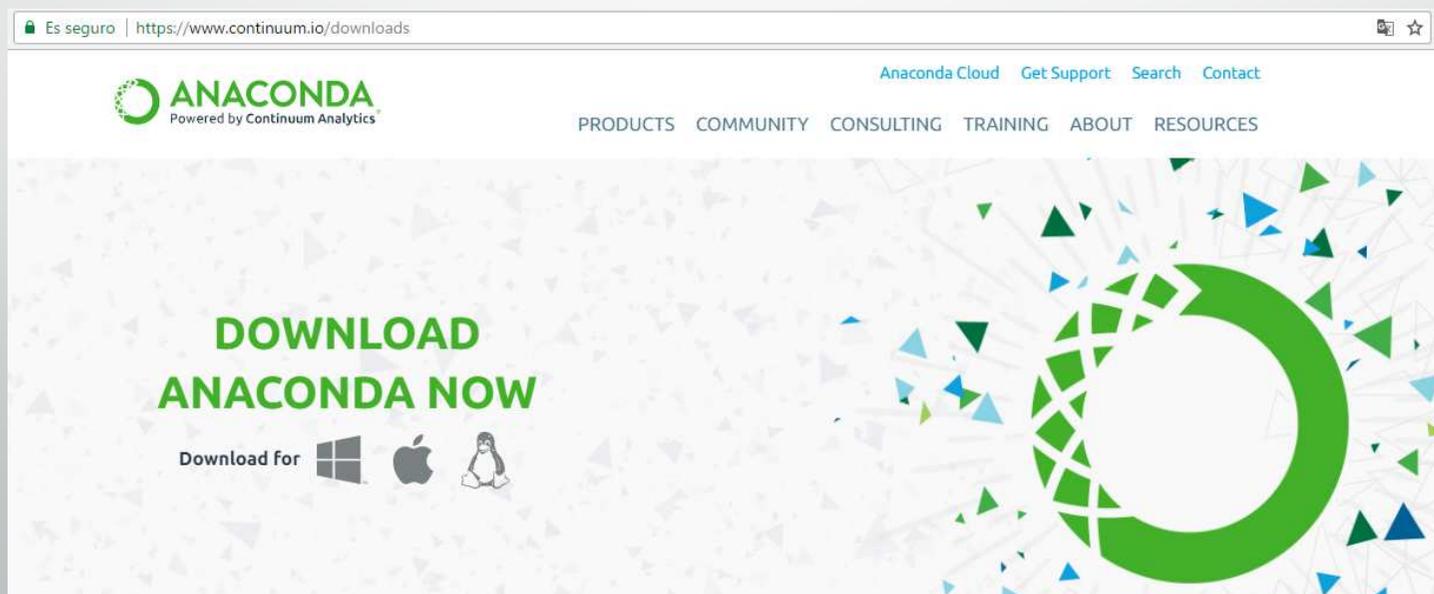
Python reúne las características necesarias para Data Science, además de ser un buen lenguaje de programación de uso general.

Python dispone de un rico ecosistema compuesto de librerías opensource para matemáticas, estadísticas, machine learning y ciencia en general.

	NumPy Base N-dimensional array package		SciPy library Base N-dimensional array package		Matplotlib Comprehensive 2D Plotting
IP[y]: IPython	IPython Enhanced Interactive Console		Sympy Symbolic mathematics		pandas Data structures and analysis

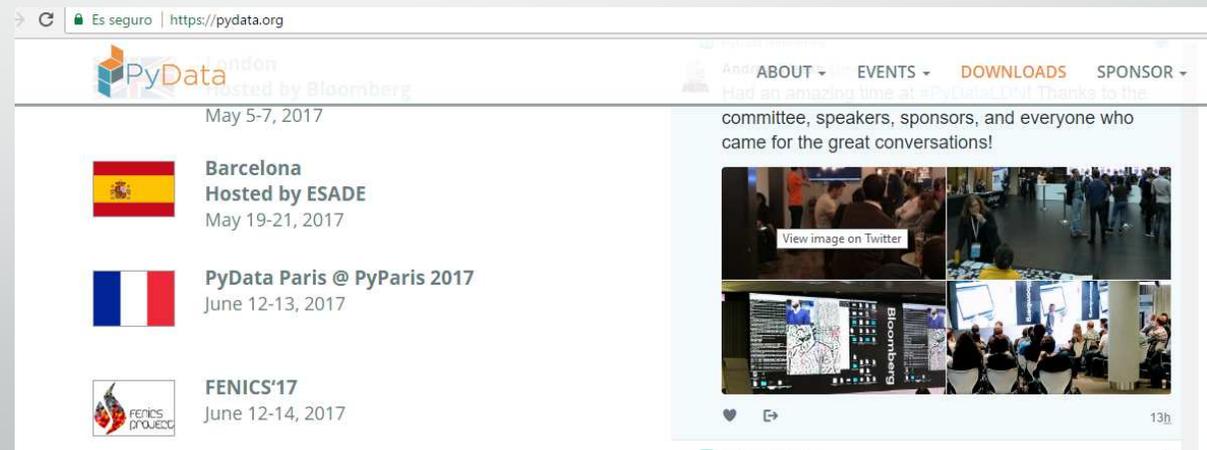
Distribución Python

- Hay muchas distribuciones Python que incluyen todas las librerías.
- Mi preferida es Anaconda de Continuum.
 - Principales características.
 - Gestión de paquetes via Conda o PIP
 - Varias GUI. Spyder, Jupiter, etc.
 - Versiones Linux, Mac y Windows. 32 y 64 bits.



PyData

- Comunidad muy activa, buena gente.
- Pueden ver videos en Youtube
- Conferencias en todo el mundo
- Las libreria fundamentales del ambiente pydata son:
 - Numpy
 - SciPy
 - Pandas
 - Matplotlib
 - Jupyter/ipython (GUI)



The screenshot shows the PyData website at <https://pydata.org>. The page features a navigation menu with links for ABOUT, EVENTS, DOWNLOADS, and SPONSOR. Below the navigation, there are three event listings:

- London** (Trusted by Bloomberg) - May 5-7, 2017
- Barcelona** Hosted by ESADE - May 19-21, 2017
- PyData Paris @ PyParis 2017** - June 12-13, 2017
- FENICS'17** - June 12-14, 2017

On the right side of the screenshot, there is a social media post from PyData. The text of the post reads: "Had an amazing time at aPyData! Thank you to the committee, speakers, sponsors, and everyone who came for the great conversations!". Below the text are two images: one showing a group of people at a conference and another showing a presentation slide titled "Bloomberg". The post has 13 hours of visibility.

Interfaz Jupyter (iPython Notebook)

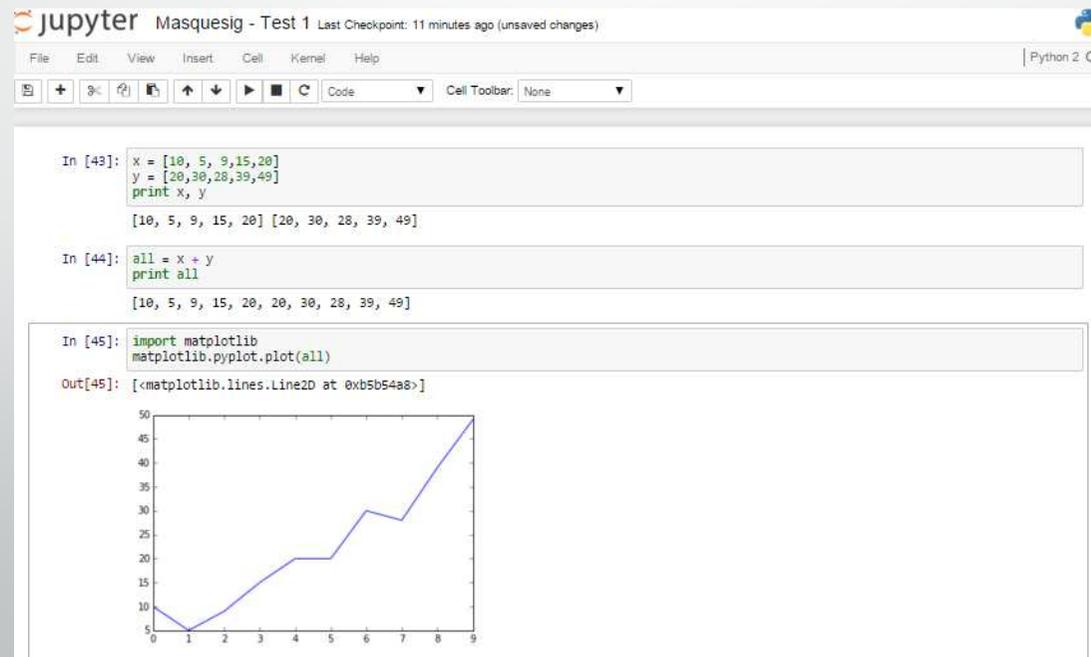
Interfaz Web (IDE), de usuario.

Widgets (para diferentes librerías)

Muy buena para para documentar.

Nos da ayuda poniendo la sentencia seguida de ?. Por ejemplo: `pd.read_csv?`.

Poniendo `!`, delante del comando nos permite ejecutar comando de la shell. Ejemplo: `!dir`, `!ls`



Librerías - Numpy

NumPy es el paquete fundamental para la computación científica con Python. Contiene.

- un poderoso objeto de matriz N-dimensional
- sofisticadas funciones
- herramientas para la integración de C / C ++ y Fortran. Usando las librerías Cython y F2py.
- álgebra lineal útil, transformada de Fourier, y capacidades de números aleatorios
- Funcionalidad semejante a Matlab
- Ejemplo

```
>>> import numpy as np
```

```
>>> x = np.linspace(0, 1, 10)
```

```
>>> x
```

```
array([ 0.        ,  0.11111111,  0.22222222,  0.33333333,  0.44444444,  
        0.55555556,  0.66666667,  0.77777778,  0.88888889,  1.        ])
```

Librerías - Scipy

Librería fundamental se basa en Numpy y es muy amplia.

- Utilidades de muchas disciplinas
- Funciones estadísticas
- Algebra lineal
- Integración numerica
- Interpolación
- Algoritmos espaciales
- Análisis de imágenes

Librerías - Sympy

Librería para manejo simbólico de expresiones.

- Plotea la salida en Latex cuando si usamos Jupyter
- Ideal para resolver sistemas de ecuaciones

Ejemplo:

```
>>> from sympy import symbols
```

```
>>> x, y = symbols('x y')
```

```
>>> expr = x + 2*y
```

```
>>> expr
```

```
x + 2*y
```

Librerías - Pandas

Pandas es una librería de análisis de datos con Python.

Contiene...

- Herramienta para la lectura y escritura de datos: CSV y archivos de texto, Microsoft Excel, bases de datos SQL, etc.
- Estructuras tabulares de datos, llamadas DataFrame. N-Dimensiones.
- Hace más amigable el uso de Numpy
- Facilita el manejo de series temporales
- Alineación inteligente de datos y el manejo integrado de los datos faltantes
- Altamente optimizado para un rendimiento, con rutas de código críticos escritos en C.
- Pandas está en uso en una amplia variedad de ámbitos académicos y comerciales, incluyendo Finanzas, Neurociencia, Economía, Estadística, Publicidad, Web Analytics, y más



Ejemplo – Pandas – Leer fichero

Ejemplo fichero webdaq planta 6

Time,tempSalaEVA(C),tempSalaTECO(C),tempSalaCimel(C),tempSalaBrewer(C),tempEscalera(C),humSalaEVA(%RH)

```
2017/04/02 12:40:58.316154,18.43,16.31,15.75,18.66,15.31,-31.98
2017/04/02 12:41:58.315968,18.68,16.31,15.87,18.54,15.44,-32.04
2017/04/02 12:42:58.315782,18.55,16.44,15.87,18.54,15.31,-31.98
2017/04/02 12:43:58.315596,18.68,16.31,15.50,18.42,15.44,-31.91
2017/04/02 12:44:58.315410,18.68,16.44,15.87,18.66,15.31,-32.04
2017/04/02 12:45:58.315224,18.55,16.44,15.75,18.42,15.31,-31.98
2017/04/02 12:46:58.315038,18.55,16.31,15.62,18.30,15.44,-31.98
2017/04/02 12:47:58.314852,18.43,16.31,15.87,18.66,15.31,-32.04
2017/04/02 12:48:58.314666,18.68,16.44,15.87,18.54,15.31,-31.98
2017/04/02 12:49:58.314480,18.68,16.31,15.75,18.42,15.31,-31.98
2017/04/02 12:50:58.314294,18.55,16.31,15.75,18.42,15.44,-31.98
2017/04/02 12:51:58.314108,18.55,16.31,15.62,18.42,15.44,-32.04
```

Ejemplo – Pandas – Leer fichero

```
# Importamos libreria
```

```
Import pandas as pd
```

```
# Le ponemos nombre a las columnas
```

```
cols=['timestamp', 'tempSalaeva', 'tempSalateco', 'tempSalaCimel', 'tempSalabrewer', 'tempescalera',  
'humsalaeVa']
```

```
#Leemos el fichero
```

```
df = pd.read_csv('/home/webdaq/wd6/_20170403121823.txt',header=0,names=cols)
```

```
>>> df
```

```
timestamp tempSalaeva tempSalateco tempSalaCimel \  
0 2017/04/02 12:40:58.316154 18.43 16.31 15.75  
1 2017/04/02 12:41:58.315968 18.68 16.31 15.87  
2 2017/04/02 12:42:58.315782 18.55 16.44 15.87  
3 2017/04/02 12:43:58.315596 18.68 16.31 15.50  
4 2017/04/02 12:44:58.315410 18.68 16.44 15.87  
5 2017/04/02 12:45:58.315224 18.55 16.44 15.75  
6 2017/04/02 12:46:58.315038 18.55 16.31 15.62
```

Ejemplo – Pandas – Leer de BBDD

```
# Importamos libreria
import pandas as pd
from sqlalchemy import create_engine

# Conexión a la BBDD – PostGreSQL
conn = create_engine('postgresql://user:password@servidor:5432/webdaq')

# Consulta de la BBDD
sqlquery="SELECT * FROM public.wdqplaz where wdqplaz.timestampx between '13/04/2017 00:00:00' AND '13/04/2017 23:59:00'"

# Lanzamos consulta
df = pd.read_sql_query(sqlquery, conn)

>>> df
  id  timestampx  humsalapica  humsalaopt  tempsalapica \
0  1004 2017-04-13 00:00:00    29.27    36.85    17.21
1  1005 2017-04-13 00:01:00    29.08    36.79    17.09
2  1006 2017-04-13 00:02:00    29.02    36.41    17.09
```

Ejemplo – Pandas – Calculo media 10 min

```
# Create un dataframe pivotable, estableciendo como índice la columna timestampx
```

```
df1 = df.set_index(['timestampx'])
```

```
#Resampleamos y calculamos la media cada 10min
```

```
media10m = df1.resample('10min',how='mean')
```

```
>>> df
```

```
id    timestampx  humsalapica  humsalaopt  tempsalapica \
0   1004 2017-04-13 00:00:00    29.27    36.85    17.21
1   1005 2017-04-13 00:01:00    29.08    36.79    17.09
2   1006 2017-04-13 00:02:00    29.02    36.41    17.09
```

```
>>> media10m
```

```
          id  humsalapica  humsalaopt  tempsalapica \
timestampx
2017-04-13 00:00:00  1008.700000  29.176000  36.738000  17.016000
2017-04-13 00:10:00  1019.500000  30.543000  36.687000  16.296000
2017-04-13 00:20:00  1030.500000  29.319000  36.576000  16.860000
2017-04-13 00:30:00  1041.300000  28.979000  36.771000  17.066000
```

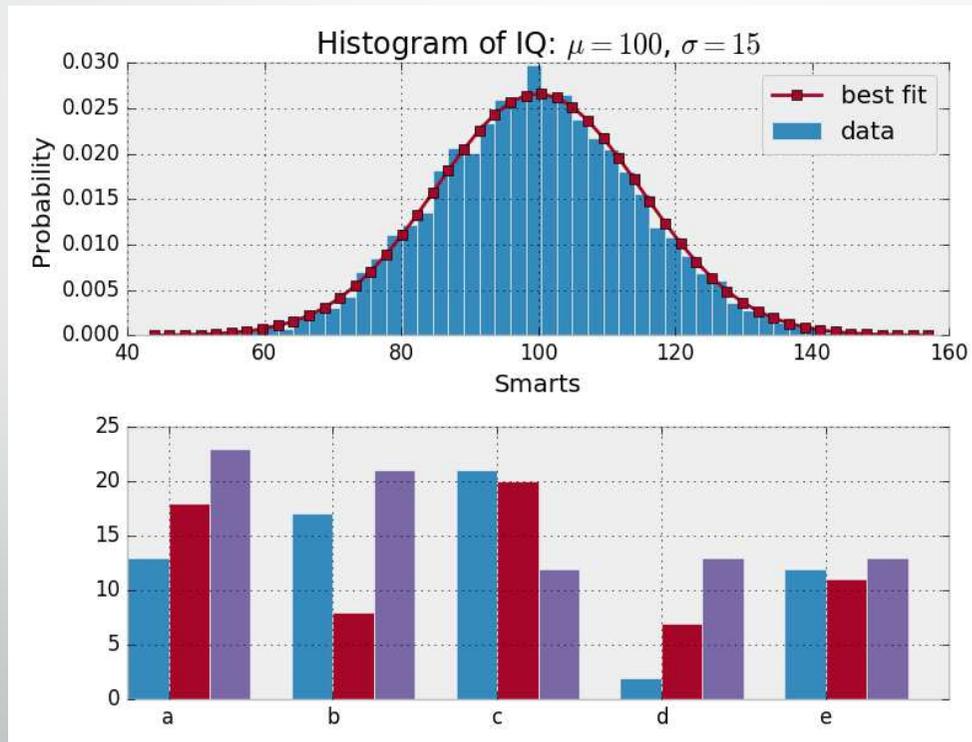
Librerías - Matplotlib

Librería para generación de gráficos con Python

Interfaz funcional estilo Matlab.

Interfaz orientada a objetos para un control más preciso del resultado

Salida de fichero de imagen o INLINE



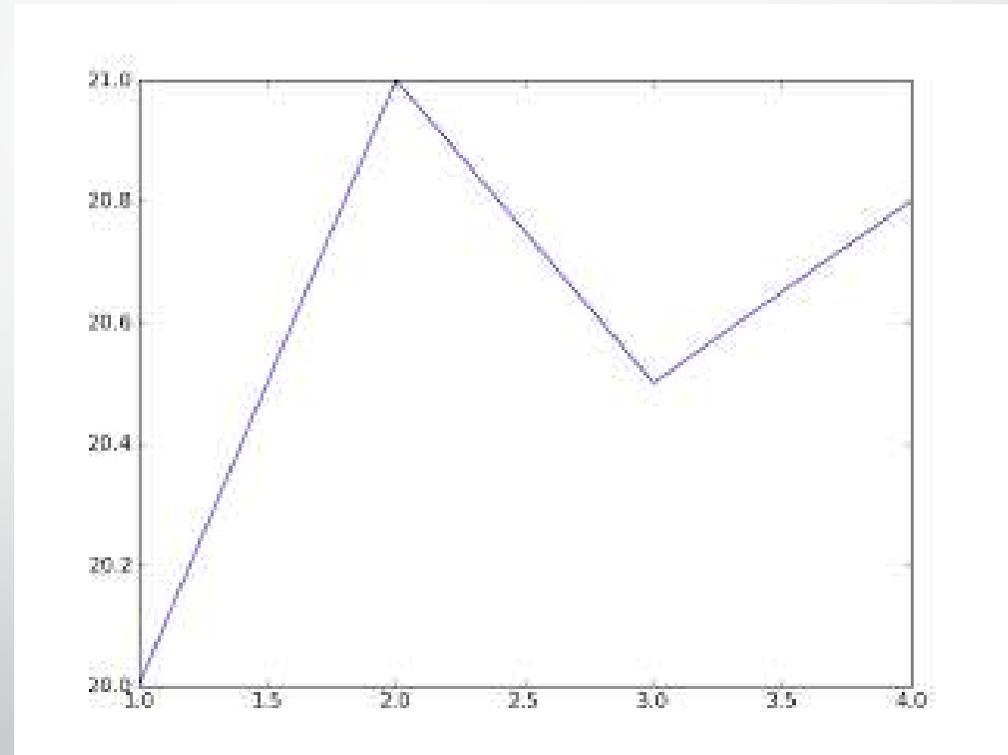
Ejemplo Matplotlib

```
#import matplotlib library  
import matplotlib.pyplot as plt
```

```
#define some data  
x = [1,2,3,4]  
y = [20, 21, 20.5, 20.8]
```

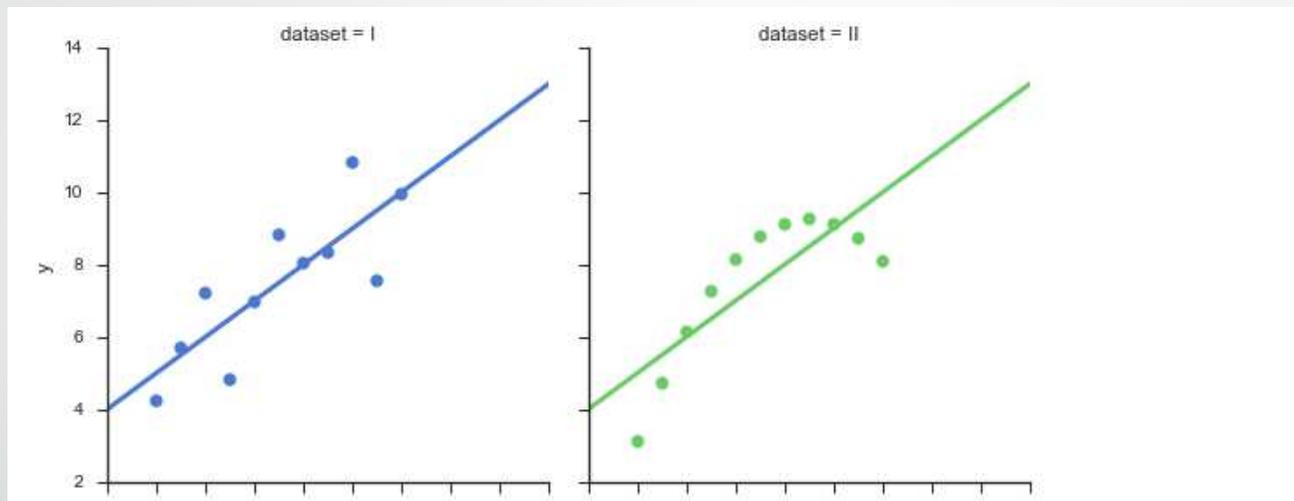
```
#plot data  
plt.plot(x, y)
```

```
#show plot  
plt.show()
```



Librería SeaBorn

Es una evolución de Matplotlib
Ideal para temas estadísticos.



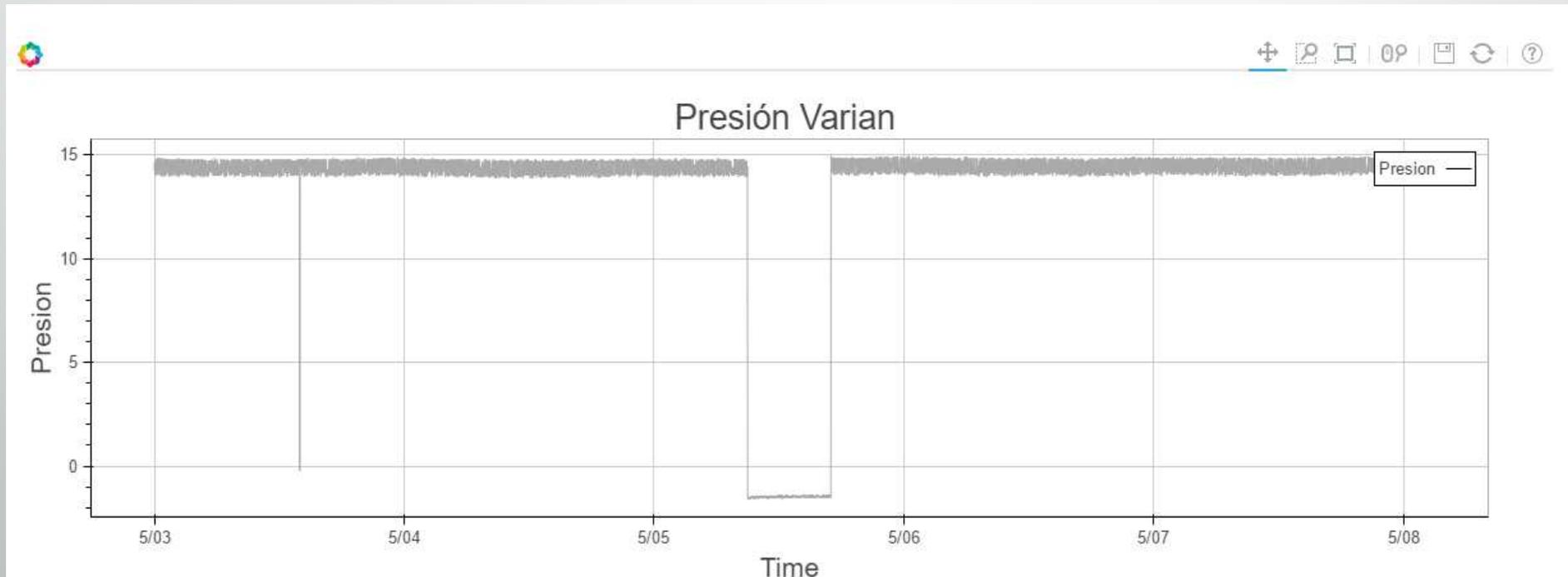
Librería Bokeh

Librería para visualizar gráficos. Permite interactividad.

Compila en CCS+JS+HTML

Genial para analizar datos y buscar errores.

Puede generar ficheros HTML de salida o levantar un mini server web (tornado).



Ejemplo Bokeh

```
from bokeh.charts import Line, show, output_file
import pandas as pd

sqlquery="SELECT * FROM public.wdqpla3 where wdqpla3.timestamp between "'03/05/2017 00:00:00' AND '08/05/2017 23:59:00'"

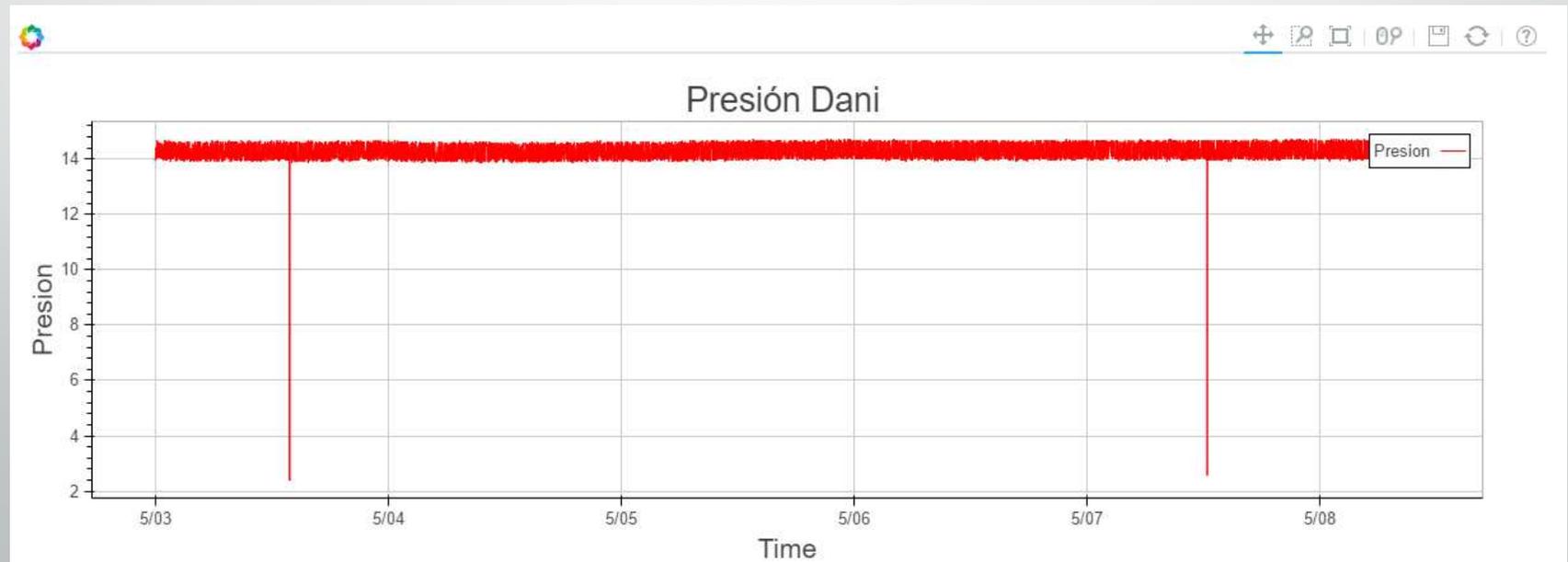
conn = create_engine('postgresql://user:password@servidorip:5432/webdaq')

BBDD = pd.read_sql_query(sqlquery, conn)

p = figure(plot_width=1200, plot_height=400,x_axis_type="datetime")
p.title= "Presión Varian"
p.xaxis.axis_label = 'Time'
p.yaxis.axis_label = 'Presion'
p.line(BBDD['timestamp'], BBDD['pressdani'], line_width=1,legend="Presion",line_color="red")
p.legend.orientation = "top_right"

output_file("pressdani.html", title="Ejemplo")

show (p)
```



HTC

Tenemos diferentes librerías para escalar nuestro código a nivel computacional.

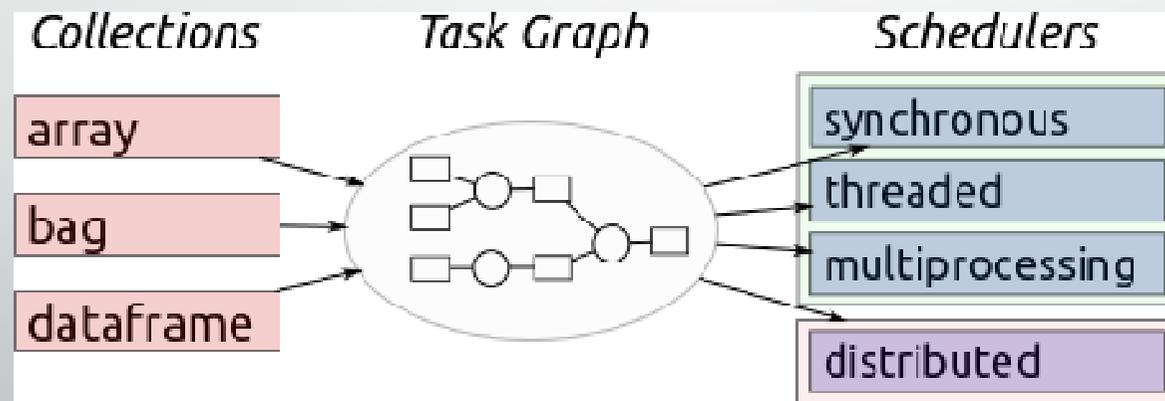
Librería Dask.

- Se integra con objetos de Numpy y Pandas
- Permite la integración con otros proyectos Python
- Muy rápido, muy escalable desde un portátil hasta un cluster de 1000 cores
- Conceptos: clientes, scheduler y workers

Ejemplo Pandas DataFrame:

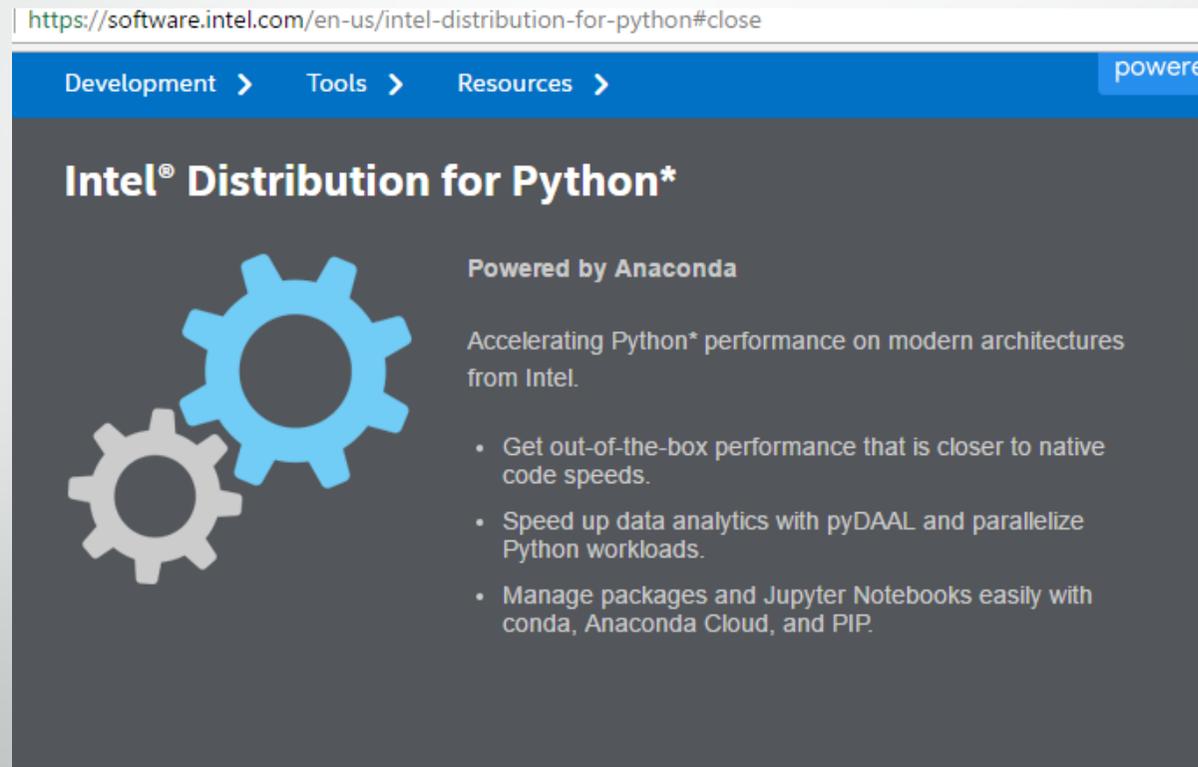
```
import pandas as pd
df = pd.read_csv('2015-01-01.csv')
df.groupby(df.user_id).value.mean()
```

```
import dask.dataframe as dd
df = dd.read_csv('2015-*-*.*.csv')
df.groupby(df.user_id).value.mean().compute()
```



HTC

Intel en 2017 desarrolla una distribución Python para sacar rendimiento a microprocesadores multicore. Soporte Intel KNL y microprocesadores Xeon Phi.
Acuerdo con Continuum, con la integración de Anaconda.
Standalone es Free.



The screenshot shows a web browser window with the URL <https://software.intel.com/en-us/intel-distribution-for-python#close>. The page features a blue navigation bar with links for "Development", "Tools", and "Resources". The main content area has a dark background and is titled "Intel® Distribution for Python*". It includes a graphic of two interlocking gears, one light blue and one grey. Below the graphic, it states "Powered by Anaconda" and "Accelerating Python* performance on modern architectures from Intel." A bulleted list highlights key features: "Get out-of-the-box performance that is closer to native code speeds.", "Speed up data analytics with pyDAAL and parallelize Python workloads.", and "Manage packages and Jupyter Notebooks easily with conda, Anaconda Cloud, and PIP."

<https://software.intel.com/en-us/intel-distribution-for-python#close>

Development > Tools > Resources > powered

Intel® Distribution for Python*



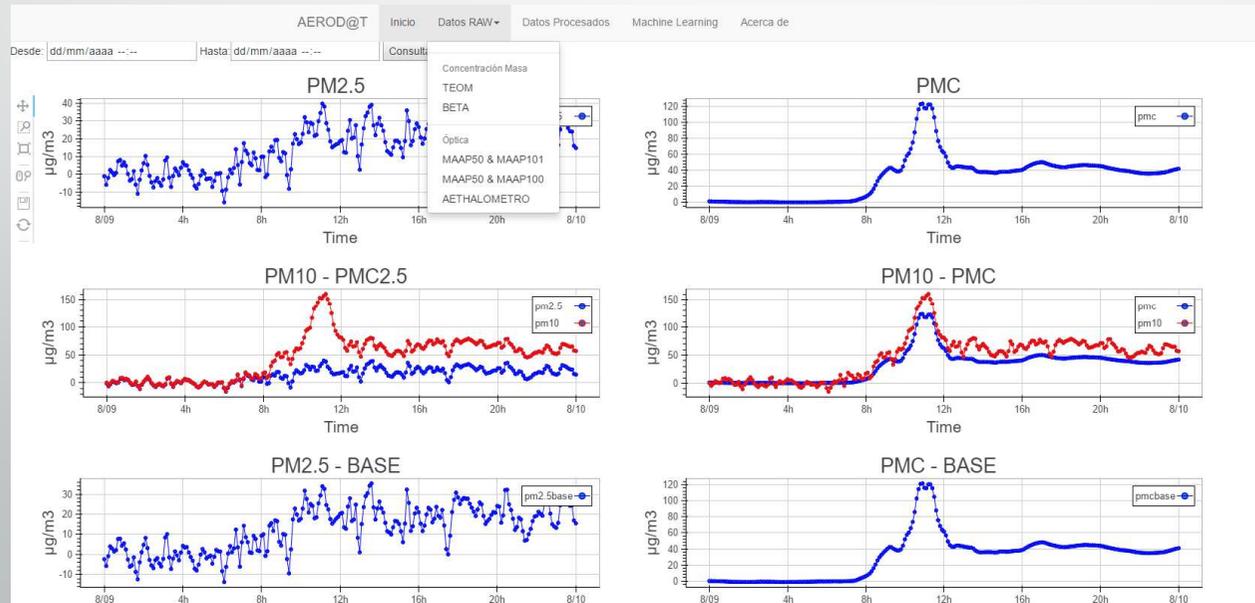
Powered by Anaconda

Accelerating Python* performance on modern architectures from Intel.

- Get out-of-the-box performance that is closer to native code speeds.
- Speed up data analytics with pyDAAL and parallelize Python workloads.
- Manage packages and Jupyter Notebooks easily with conda, Anaconda Cloud, and PIP.

Proyectos usando PyData en CIAI

- Proceso ETL (extraer, transformar y cargar). Los equipos de aerosoles insitu. La BBDD está en PostgreSQL
 - MAAP50 y MAAP100
 - APS
 - TEOM
 - Datos Meteo-Toma
 - NEPH
 - AETHAL
 - BETA
- Tras acuerdo se envía datos del PM10 del TEOM (media horarias) al IAC
- Envío automático de datos al Nilu del NEPH y el MAAP50. <- Javi y Nestor
- Creación web de monitorización y explotación de los datos. <- Rocío, Néstor y Javi



PyAOS - *Python for the Atmospheric and Oceanic Sciences*

Web donde podréis encontrar mucho recursos para temas atmosférico.
<http://pyaos.johnny-lin.com>

PyAOS Python for the Atmospheric and Oceanic Sciences

<http://pyaos.johnny-lin.com>

Home About Mailing Lists Getting Started **Packages** Training Student Awards Legal

Specialized AOS Tools

General

- [atmos](#): “An atmospheric sciences utility library”
- [Climate Data Analysis Tools \(CDAT\)](#)
- [CEWE](#): “A Python Summary Statistics Tool for Massive Data Analysis”
- [Google Earth Engine](#): This is not a Python package but a cloud-based analysis engine and data center with a Python API.
- [Integrated Data Viewer \(IDV\)](#): While not a Python package, this application for analysis and visualizing data can be controlled by [Jython scripting](#).
- [Iris](#): “A Python library for Meteorology and Climatology”
- [obsio](#): “obsio is a Python package that provides a consistent generic interface for accessing weather and climate observations from multiple different data providers.”
- [PyGrADS](#): Python interface to [GrADS](#).
- [PyNGL](#): Python NCAR Graphics Library package.

Search

Categories

- Advanced
- Beginner
- Books
- Climatology
- Code Sprints
- Community
- Conferences
- Data Analysis
- Examples Repository
- Featured Tips
- Functional Programming
- GIS
- Gridding
- I/O
- Installing
- Interfaces
- Jobs
- Linux

Enlaces de interes

Formación.

<https://www.codecademy.com/>

<https://www.udemy.com>

<http://www.edx.com>

Librerías

<https://pydata.org/downloads.html>

HPC

<http://dask.pydata.org>

<https://software.intel.com/en-us/intel-distribution-for-python>



¿¿¿PREGUNTAS???

Los científicos se esfuerzan por hacer posible lo imposible.

Los políticos, por hacer imposible lo posible.

Bertrand Russell