IMT2200

INTRODUCCIÓN A CIENCIA DE DATOS

2022-2

HERRAMIENTAS PARA CIENCIA DE DATOS

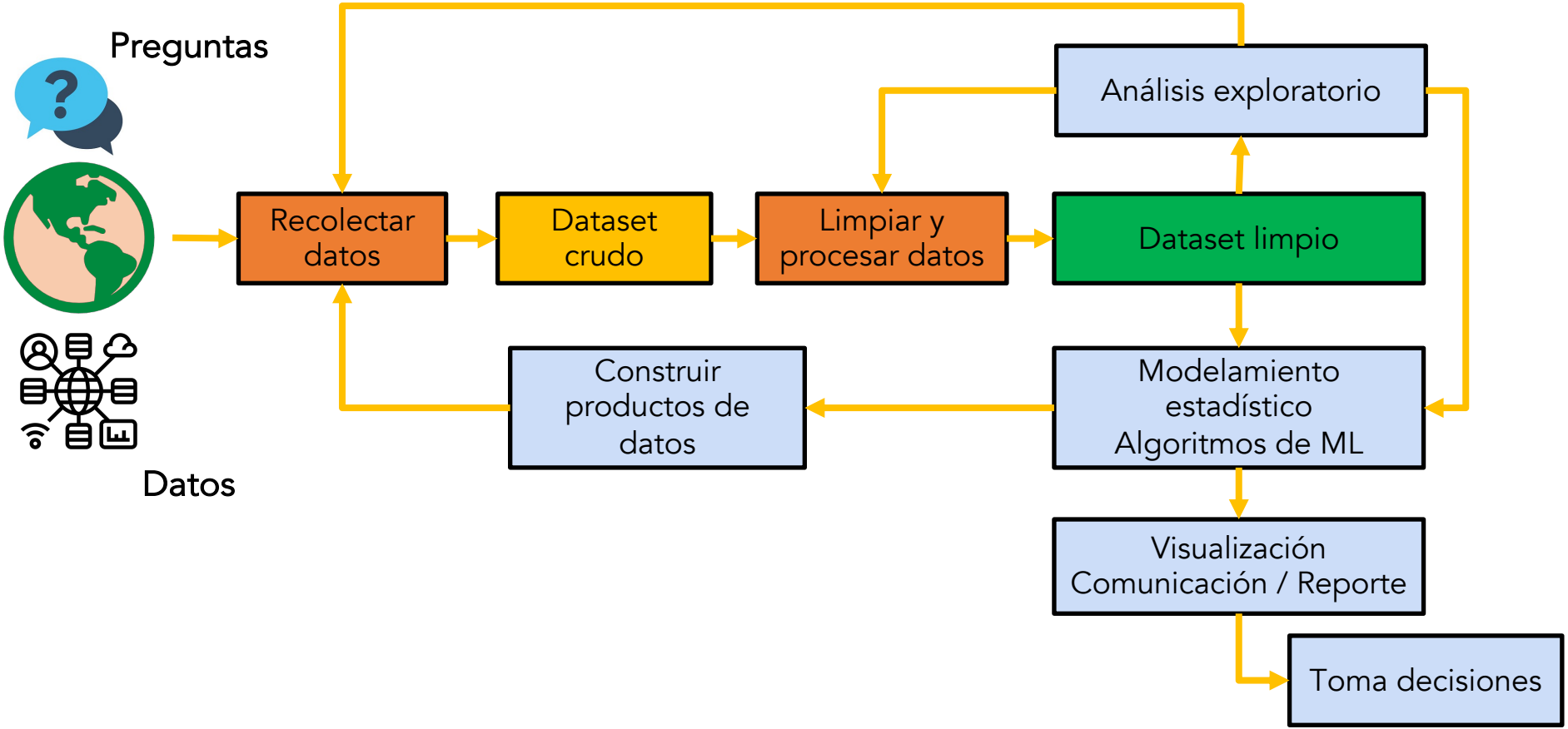
CLASE 2

AGENDA DE HOY

- Repaso Clase 1
- Herramientas computacionales para CD
- Demostración en Jupyter Notebook
- Actividad 1

¿Dudas respecto al programa?

EL PROCESO DE CIENCIA DE DATOS



DATOS vs. INFORMACIÓN

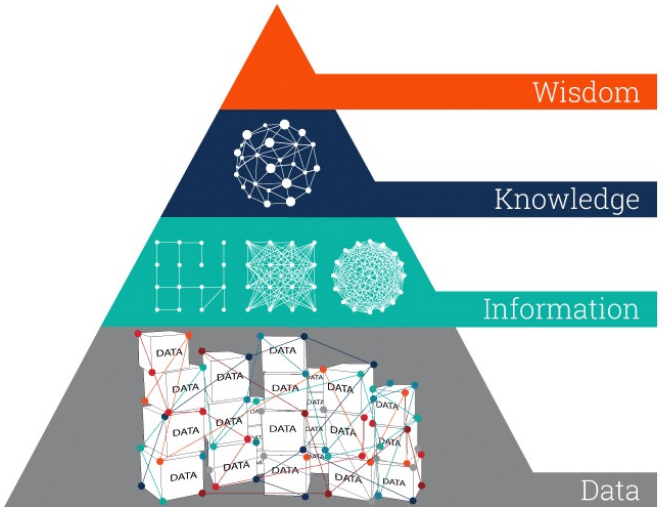
Datos ➔ abstracciones o mediciones obtenidas del mundo real.

Información ➔ datos que han sido procesados, estructurados o contextualizados de manera que son significativos para las personas.

Conocimiento ➔ información que ha sido interpretada y comprendida por una persona, de manera que puede actuar a partir de ella.

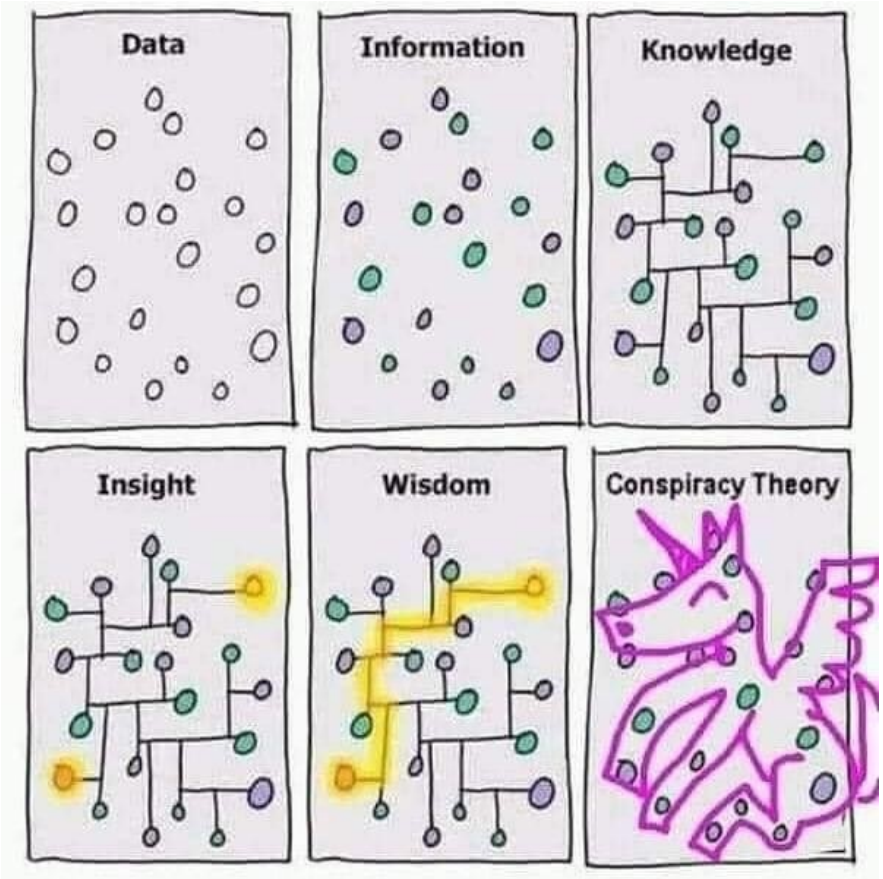
Sabiduría ➔ actuar de forma apropiada a partir del conocimiento

Pirámide DIKW



https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/

DATOS vs. INFORMACIÓN



"The combination of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data"

-John Tuke

"La combinación de algunos datos y un intenso deseo de obtener una respuesta no asegura que sea posible extraer una respuesta razonable de un conjunto dado de datos"

Herramientas Computacionales para Ciencia de Datos

REPRODUCIBILIDAD CIENTÍFICA

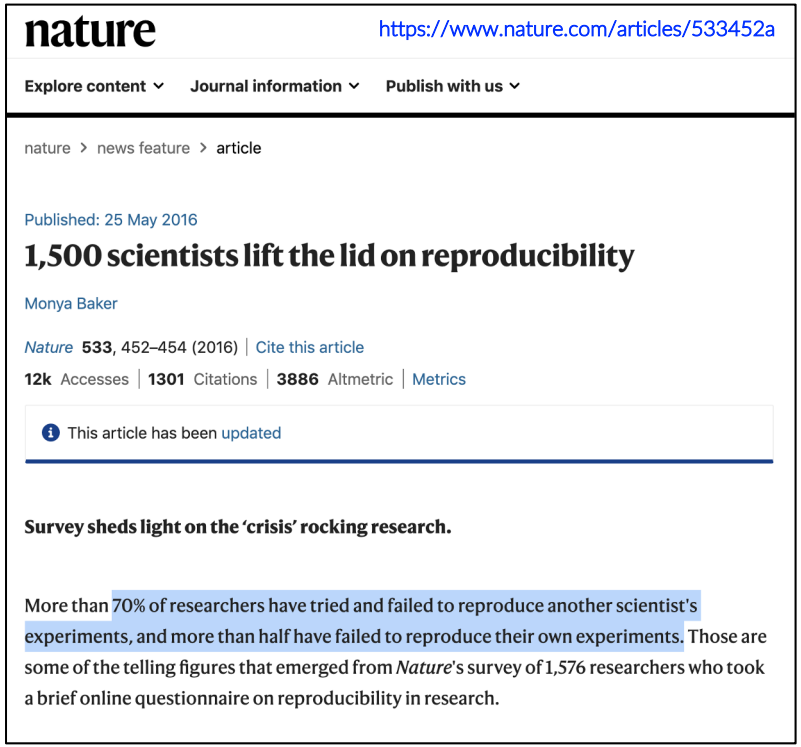
Reproducibilidad:

- Capacidad de reproducir procedimientos científicos (experimentos, análisis, resultados).
- Clave para el desarrollo, colaboración y transparencia científica y profesional

¿Por qué es importante?

- Para repetir un análisis con distintos parámetros.
- Para escalar un análisis ➔ entrega a equipo de producción
- Para crear confianza en los resultados
- Para compartir el conocimiento
- Para ahorrar tiempo y stress ☺️

https://towardsdatascience.com/creating-reproducible-data-science-projects-1fa446369386



nature

https://www.nature.com/articles/533452a

Explore content | Journal information | Publish with us

nature › news feature › article

Published: 25 May 2016

1,500 scientists lift the lid on reproducibility

Monya Baker

Nature **533**, 452–454 (2016) | Cite this article

12k Accesses | **1301** Citations | **3886** Altmetric | Metrics

This article has been updated

Survey sheds light on the 'crisis' rocking research.

More than 70% of researchers have tried and failed to reproduce another scientist's experiments, and more than half have failed to reproduce their own experiments. Those are some of the telling figures that emerged from *Nature*'s survey of 1,576 researchers who took a brief online questionnaire on reproducibility in research.

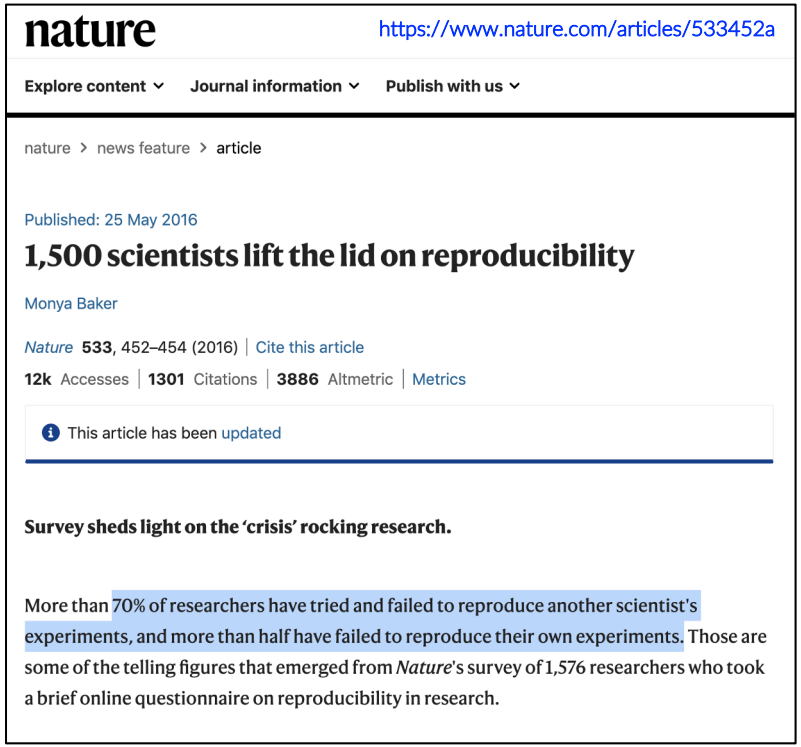
Digital innovators from Alibaba to Google to Facebook to Netflix to Microsoft to Amazon have already actively embraced large-scale, rapid experimentation as integral to their innovation efforts. But every organization seeking authentic insight from experiment needs to be wary of the problems that have made scientific research unreliable.

https://towardsdatascience.com/why-your-company-needs-reproducible-research-d4a08f978d39

REPRODUCIBILIDAD CIENTÍFICA

¿Cómo lo hacemos?

- Disponibilización de **datos**.
- **Automatización** de procesos, sin pasos "manuales".
- Desarrollando procesos computacionales **transparentes**.
- Con suficiente **detalle** para reproducir todo el análisis.
- Usando modelos y herramientas eficientes de **reusabilidad**.
  - Software libre
  - Paquetes de código abierto
  - "Notebooks" computacionales
  - Repositorios y versionamiento



nature https://www.nature.com/articles/533452a

Explore content Journal information Publish with us

nature › news feature › article

Published: 25 May 2016

1,500 scientists lift the lid on reproducibility

Monya Baker

Nature **533**, 452–454 (2016) | Cite this article

12k Accesses | **1301** Citations | **3886** Altmetric | Metrics

This article has been updated

Survey sheds light on the 'crisis' rocking research.

More than 70% of researchers have tried and failed to reproduce another scientist's experiments, and more than half have failed to reproduce their own experiments. Those are some of the telling figures that emerged from *Nature*'s survey of 1,576 researchers who took a brief online questionnaire on reproducibility in research.

Digital innovators from Alibaba to Google to Facebook to Netflix to Microsoft to Amazon have already actively embraced large-scale, rapid experimentation as integral to their innovation efforts. But every organization seeking authentic insight from experiment needs to be wary of the problems that have made scientific research unreliable.

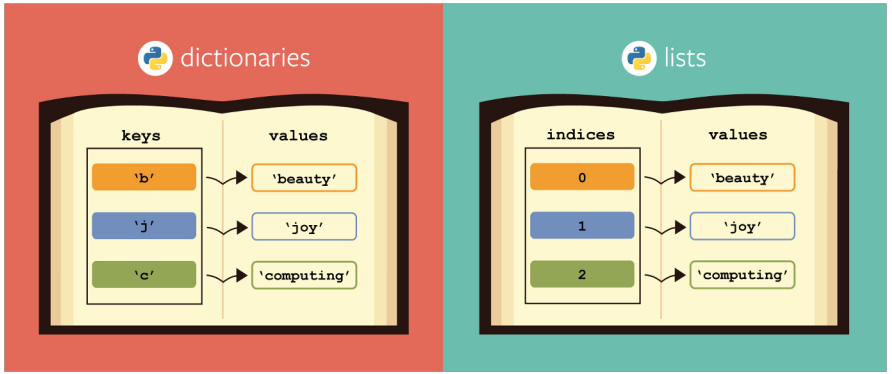
https://towardsdatascience.com/why-your-company-needs-reproducible-research-d4a08f978d39

PYTHON www.python.org

- Lenguaje interpretado de programación: los comandos o instrucciones se ejecutan paso por paso, sin compilación previa.
- Gran y activa comunidad de computación científica y ciencia de datos desarrollada en torno a Python.
- Facilidad para integrar códigos desarrollados en C, C++, FORTRAN, etc. (más eficientes para algunos algoritmos de cálculo)
- Integración de múltiples librerías con distintas funcionalidades y usos (numPy, sciPy, pandas, scikit-learn, …. etc).
- Menos eficiente que otros lenguaje para computación paralela y concurrencias.

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```



PYTHON

Variables

You can name a variable anything as long as it obeys the following rules:

1. It can be only one word.
2. It can use only letters, numbers, and the underscore (`_`) character.
3. It can't begin with a number.
4. Variable name starting with an underscore (`_`) are considered as "unuseful`.

Data Types

Data Type	Examples
Integers	`-2, -1, 0, 1, 2, 3, 4, 5`
Floating-point numbers	`-1.25, -1.0, --0.5, 0.0, 0.5, 1.0, 1.25`
Strings	`'a', 'aa', 'aaa', 'Hello!', '11 cats'`

Comparison Operators

Operator	Meaning
`==`	Equal to
`!=`	Not equal to
`<`	Less than
`>`	Greater Than
`<=`	Less than or Equal to
`>=`	Greater than or Equal to

Functions

```
>>> def hello(name):
>>>     print('Hello {}'.format(name))
>>>
>>> hello('Alice')
>>> hello('Bob')
Hello Alice
Hello Bob
```

if Statements

```
if name == 'Alice':
    print('Hi, Alice.')
```

else Statements

```
name = 'Bob'
if name == 'Alice':
    print('Hi, Alice.')
else:
    print('Hello, stranger.')
```

elif Statements

```
name = 'Bob'
age = 5
if name == 'Alice':
    print('Hi, Alice.')
elif age < 12:
    print('You are not Alice, kiddo.')
```

Dictionaries and Structuring Data

Example Dictionary:

```
myCat = {'size': 'fat', 'color': 'gray', 'disposition': 'loud'}
```

The keys(), values(), and items() Methods

values():

```
>>> spam = {'color': 'red', 'age': 42}
>>> for v in spam.values():
>>>     print(v)
red
42
```

keys():

```
>>> for k in spam.keys():
>>>     print(k)
color
age
```

Lists

```
>>> spam = ['cat', 'bat', 'rat', 'elephant']

>>> spam
['cat', 'bat', 'rat', 'elephant']
```

Getting Individual Values in a List with Indexes

```
>>> spam = ['cat', 'bat', 'rat', 'elephant']
>>> spam[0]
'cat'
```

LIBRERÍAS EN PYTHON

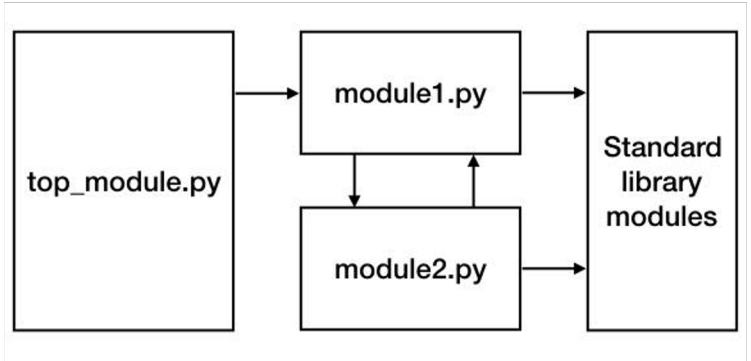
- Script: archivo de Python que es ejecutado directamente para hacer "algo".

- Módulo: archivo Python que es importado por scripts u otros módulos. Define clases, funciones y variables a ser usados por otros módulos.

- Paquete: colección de módulos relacionados que proveen una cierta funcionalidad.

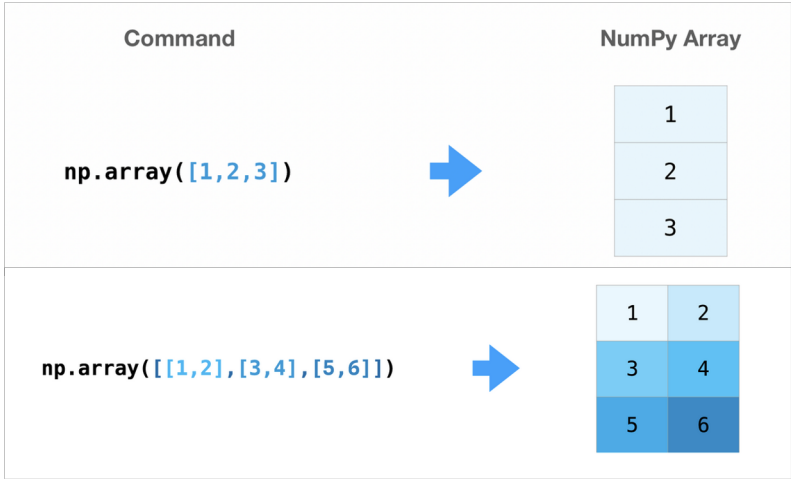
- Librería: colección de códigos reutilizable. Puede estar formadas por muchos módulos o paquetes.

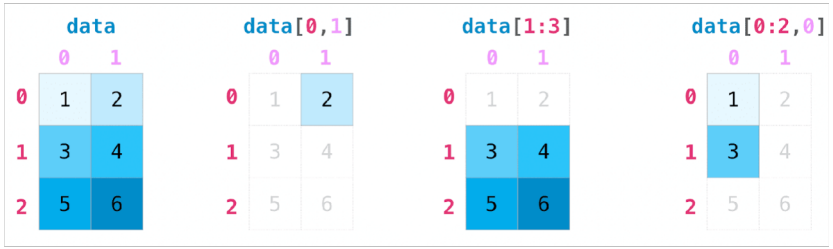
- Python Package Index (PyPi): repositorio de software en lenguaje Python.
  - Permite encontrar e instalar software desarrollado y compartido por la comunidad Python.



NumPy https://numpy.org/

- Provee las estructuras, algoritmos y funciones para la mayoría de las aplicaciones científicas relacionadas con datos numéricos.
- Arreglos multi-dimensionales (ndarray): eficientes para almacenar y manipular data.
- Funciones para realizar operaciones matemáticas entre arreglos ➔ evita iteraciones
  - Álgebra lineal, transformadas de Fourier, números aleatorios, etc.
- Interoperable: estructura de data primaria para muchas otras herramientas de Python.
- Eficiente: basada en códigos en C bien optimizados.
- Herramientas para leer y escribir datos/arreglos al disco/archivos.





NumPy https://numpy.org/

NumPy

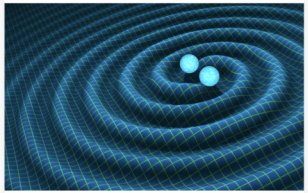
The fundamental package for scientific computing with Python

CASE STUDIES

FIRST IMAGE OF A BLACK HOLE

How NumPy, together with libraries like SciPy and Matplotlib that depend on NumPy, enabled the Event Horizon Telescope to produce the first ever image of a black hole

DETECTION OF GRAVITATIONAL WAVES



In 1916, Albert Einstein predicted gravitational waves; 100 years later their existence was confirmed by LIGO scientists using NumPy.

SPORTS ANALYTICS

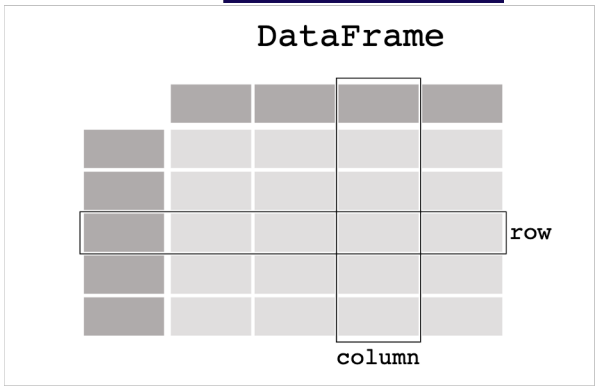
Cricket Analytics is changing the game by improving player and team performance through statistical modelling and predictive analytics. NumPy enables many of these analyses.

POSE ESTIMATION USING DEEP LEARNING

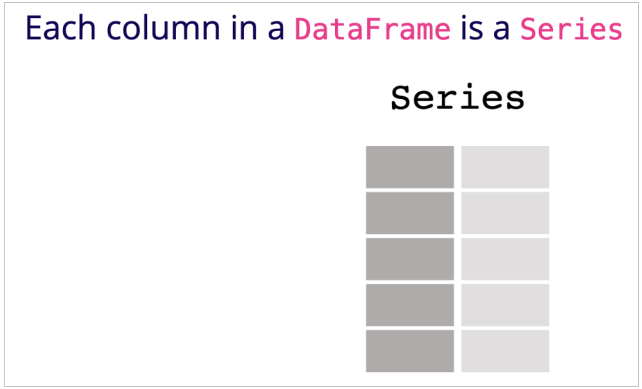
DeepLabCut uses NumPy for accelerating scientific studies that involve observing animal behavior for better understanding of motor control, across species and timescales.

PANDAS https://pandas.pydata.org/

https://pandas.pydata.org/



- Provee las estructuras y funciones de alto nivel para trabajar con datos estructurados y tablas en forma rápida, fácil y expresiva.
  - **DataFrame:** estructura de datos tabular, orientada a columnas, con etiquetas de filas y columnas.
  - **Series**: objeto tipo arreglo 1-D, con etiqueta.
- Combina la eficiencia de cálculo de NumPy, con capacidades de manipulación de datos típicas de hojas de cálculo (e.g. Excel) y bases de datos relacionales.
- Provee funcionalidades de indexación para reformatear (reshape) , seccionar (slice), agregar, filtrar, seleccionar subconjuntos de datos.



MATPLOTLIB https://matplotlib.org

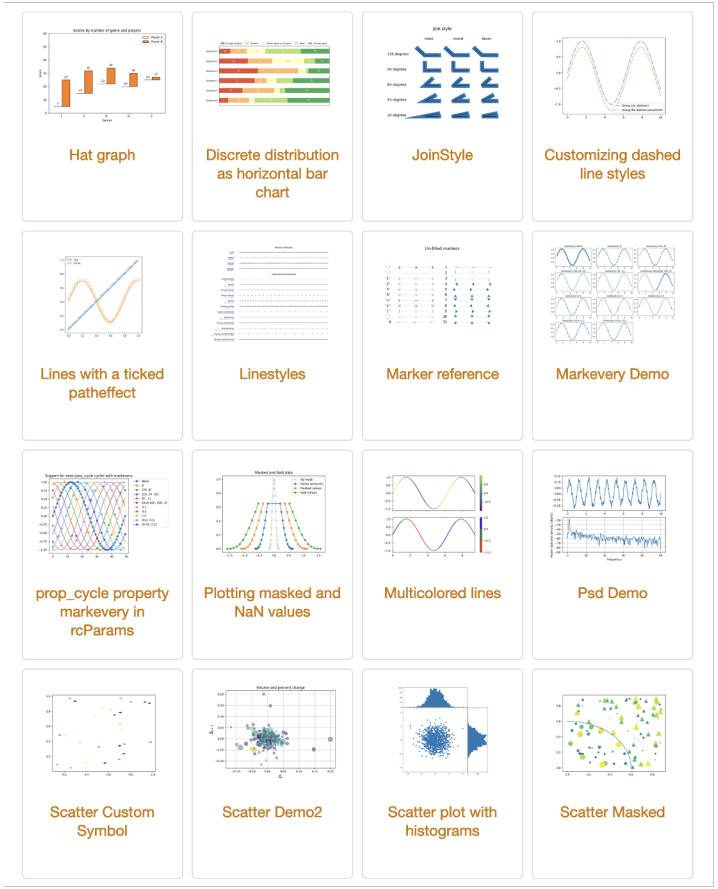
- Librería de Python más popular para producción de gráficos y otras visualizaciones 2D/3D estáticas, animadas e interactivas.
- Permite controlar y personalizar todos los detalles de una figura.

- Existen varios paquetes que extienden y construyen sobre las funcionalidades de Matplotlib, incluyendo algunas interfaces de ploteo de alto nivel:
  - Seaborn: interfaz para generación de gráficos estadísticos atractivos e informativos. Ayuda a explorar y entender la data.
  - Cartopy: paquete diseñado para el procesamiento de data geoespacial, para producir mapas y otros análisis espaciales

- Galerías:
  - https://matplotlib.org/stable/gallery/index.html
  - https://www.python-graph-gallery.com/

- Cheatsheets:

  https://github.com/matplotlib/cheatsheets#cheatsheets



matplotlib

Cheat sheet — Version 3.2

Quick start

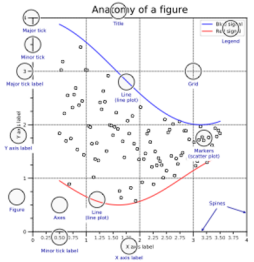
```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt


X = np.linspace(0, 2*np.pi, 100)
Y = np.cos(X)

fig, ax = plt.subplots()
ax.plot(X,Y,color='C1')

fig.savefig("figure.pdf")
fig.show()
```

Anatomy of a figure



Subplots layout

subplot[s](cols,rows,…)
fig, axs = plt.subplots(3,3)

G = **gridspec**(cols,rows,…)
ax = G[0,:]

ax.**inset_axes**(extent)

d=**make_axes_locatable**(ax)
ax=d.new_horizontal('10%')

Getting help

- matplotlib.org
- github.com/matplotlib/matplotlib/issues
- discourse.matplotlib.org
- stackoverflow.com/matplotlib
- gitter.im/matplotlib
- twitter.com/matplotlib
- Matplotlib users mailing list

Basic plots

plot([X],Y,[fmt],…)
X, **Y**, fmt, color, marker, linestyle

scatter(X,Y,…)
X, **Y**, [s]izes, [c]olors, markers, cmap

bar[h](x,height,…)
x, **height**, width, bottom, align, color

imshow(Z,[cmap],…)
Z, cmap, interpolation, extent, origin

contour[f]([X],[Y],Z„…)
X, Y, **Z**, levels, colors, extent, origin

quiver([X],[Y],U,V,…)
X, Y, **U**, **V**, C, units, angles

pie(X,[explode],…)
Z, explode, labels, colors, radius

text(x,y,text,…)
x, **y**, **text**, va, ha, size, weight, transform

fill[_between][x](…)
X, Y1, Y2, color, where

Advanced plots

step(X,Y,[fmt],…)
X, **Y**, fmt, color, marker, where

boxplot(X,…)
X, notch, sym, bootstrap, widths

errorbar(X,Y,xerr,yerr,…)
X, **Y**, xerr, yerr, fmt

hist(X, bins, …)
X, bins, range, density, weights

violinplot(D,…)
D, positions, widths, vert

barbs([X],[Y], U, V, …)
X, Y, **U**, **V**, C, length, pivot, sizes

eventplot(positions,…)
positions, orientation, lineoffsets

hexbin(X,Y,C,…)
X, **Y**, C, gridsize, bins

xcorr(X,Y,…)
X, **Y**, normed, detrend

Scales

ax.**set_[xy]scale**(scale,…)

linear any values
log values > 0
symlog any values
logit 0 < values < 1

Projections

subplot(…,projection=p)

p='polar'
p='3d'

p=Orthographic()
from cartopy.crs import Cartographic

Lines

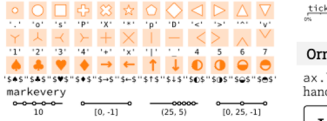
linestyle or ls

"-" ":" "-." "--" (0,(0.01,2))

capstyle or dash_capstyle

"butt" "round" "projecting"

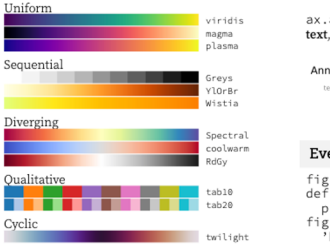
Markers



Colors

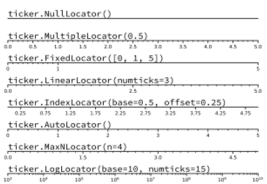
Colormaps

plt.**get_cmap**(name)



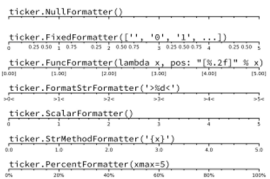
Tick locators

```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_locator(locator)
```



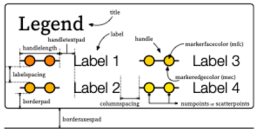
Tick formatters

```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_formatter(formatter)
```



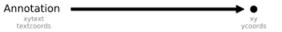
Ornaments

ax.**legend**(…)
handles, labels, loc, title, frameon



ax.**colorbar**(…)
mappable, ax, cax, orientation

ax.**annotate**(…)
text, **xy**, **xytext**, xycoords, textcoords, arrowprops



Event handling

```
fig, ax = plt.subplots()
def on_click(event):
  print(event)
fig.canvas.mpl_connect(
  'button_press_event', on_click)
```

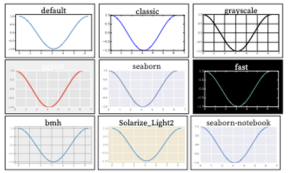
Animation

```
import matplotlib.animation as mpla

T = np.linspace(0,2*np.pi,100)
S = np.sin(T)
line, = plt.plot(T, S)
def animate(i):
  line.set_ydata(np.sin(T+i/50))
anim = mpla.FuncAnimation(
  plt.gcf(), animate, interval=5)
plt.show()
```

Styles

plt.style.use(**style**)



Quick reminder

```
ax.grid()
ax.patch.set_alpha(0)
ax.set_[xy]lim(vmin, vmax)
ax.set_[xy]label(label)
ax.set_[xy]ticks(list)
ax.set_[xy]ticklabels(list)
ax.set_[sup]title(title)
ax.tick_params(width=10, …)
ax.set_axis_[on|off]()

ax.tight_layout()
plt.gcf(), plt.gca()
mpl.rc('axes', linewidth=1, …)
fig.patch.set_alpha(0)
text=r'$\frac{-e^{i\pi}}{2^n}$'
```

Keyboard shortcuts

ctrl + s	Save	ctrl + w	Close plot
r	Reset view	f	Fullscreen 0/1
f	View forward	b	View back
p	Pan view	o	Zoom to rect
x	X pan/zoom	y	Y pan/zoom
g	Minor grid 0/1	G	Major grid 0/1
l	X axis log/linear	L	Y axis log/linear

Ten Simple Rules

READ

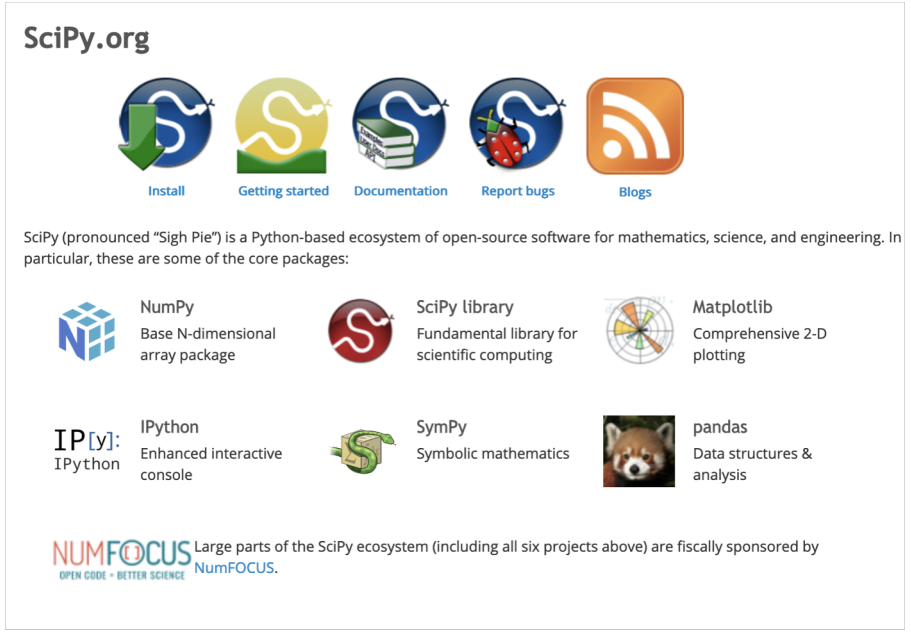
1. Know Your Audience
2. Identify Your Message
3. Adapt the Figure
4. Captions Are Not Optional
5. Do Not Trust the Defaults
6. Use Color Effectively
7. Do Not Mislead the Reader
8. Avoid "Chartjunk"
9. Message Trumps Beauty
10. Get the Right Tool

SCIPY https://www.scipy.org/

- Ecosistema de software de código abierto basado en Python para matemáticas, ciencia e ingeniería.

- La *librería* SciPy provee una colección de paquetes enfocados en distintos problemas o requerimientos propios de computación científica:
  - Integración (scipy.integrate)
  - Álgebra lineal (scipy.linalg)
  - Optimización (scipy.optimize)
  - Análisis de señales (scipy.signal)
  - Estadísticas (scipy.stats)
  - Interpolación (scipy.interpolation)

- Usa arreglos de numpy como estructura básica de datos.

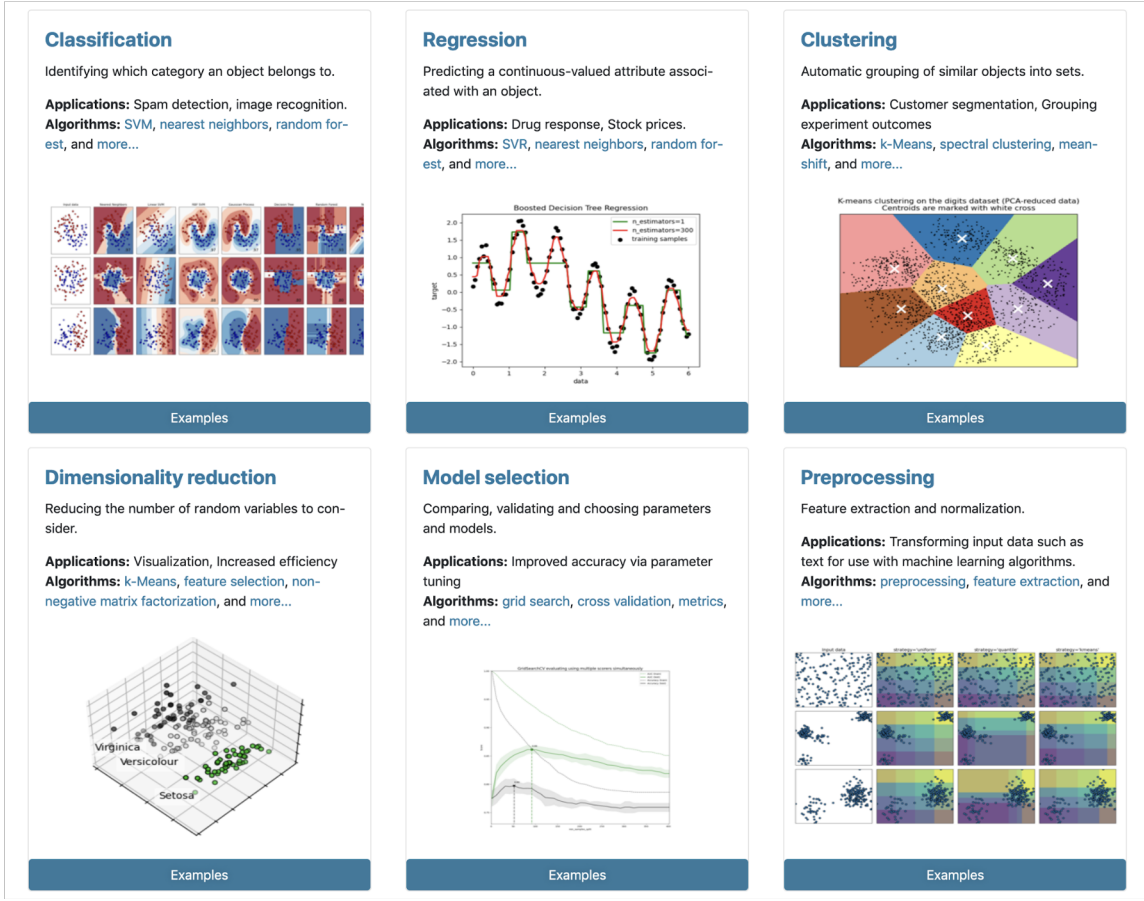


SCIKIT LEARN https://scikit-learn.org/stable/

- Construido sobre NumPy, SciPy y matplotlib.

- Provee herramientas simples y eficientes para **aprendizaje de máquinas:**

  - Preprocesamiento
  - Clasificación
  - Regresión
  - Clustering
  - Reducción de dimensionalidad
  - Selección de modelos



GEOPANDAS

https://geopandas.org/

- Librería que facilita el trabajo con datos geoespaciales en Python.

- Extiende los tipos de datos usados por pandas para permitir manipulación y operaciones espaciales con datos geométricos.
  - DataFrame ➔ GeoDataframe
  - Series ➔ GeoSeries

- Se basa en otras librerías pre-existentes:
  - **fiona** ➔ acceso a formatos de datos geoespaciales
  - **shapely** ➔ operaciones geométricas

```
[27]: data.head()
```

[27]:

	GROUP	CLASS	geometry	area
0	64	32421	POLYGON ((379394.248 6689991.936, 379389.790 6...	76.027392
1	64	32421	POLYGON ((378980.811 6689359.377, 378983.401 6...	2652.054186
2	64	32421	POLYGON ((378804.766 6689256.471, 378817.107 6...	3185.649995
3	64	32421	POLYGON ((379229.695 6685025.111, 379233.366 6...	13075.165279
4	64	32421	POLYGON ((379825.199 6685096.247, 379829.651 6...	3980.682621

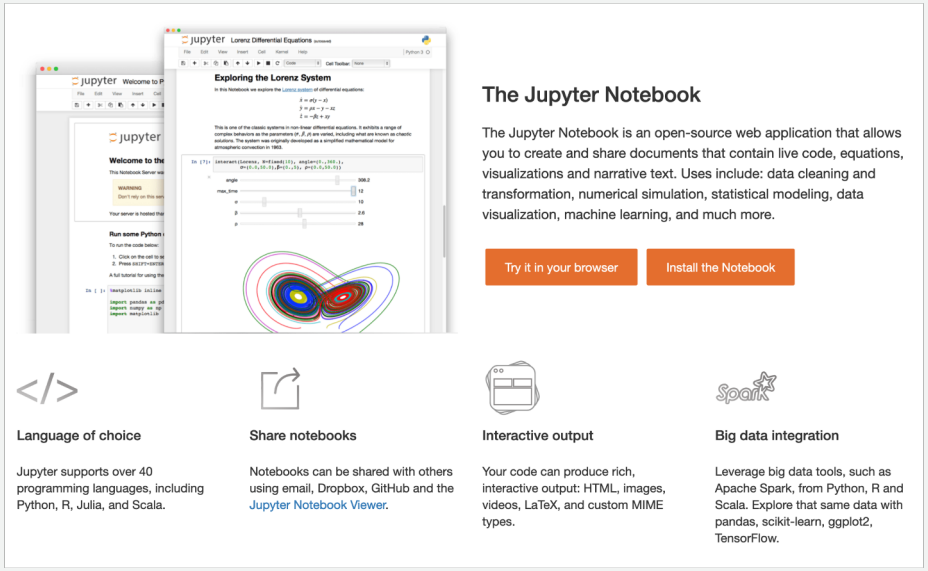
JUPYTER NOTEBOOKS

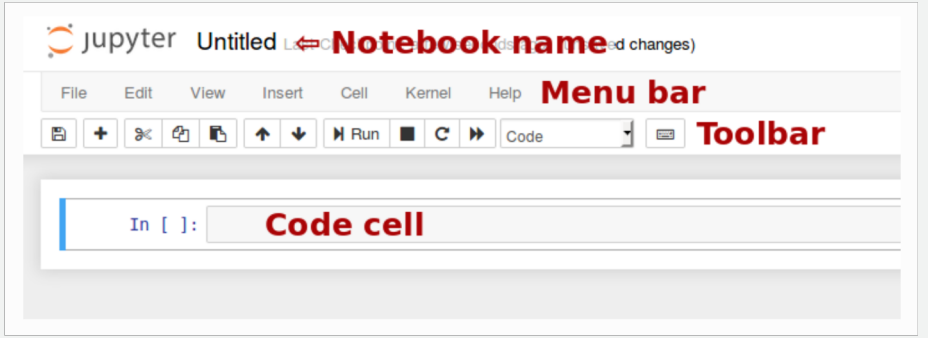
- Aplicación web open source para crear y compartir documentos **(.ipynb)** con:
  - Código ejecutable
  - Resultados del código
  - Texto narrativo e imágenes

- Soporta múltiples lenguajes de programación.
- Notebooks pueden ser exportados a múltiples formatos estáticos.

- Instalación: https://jupyter.org/install.html

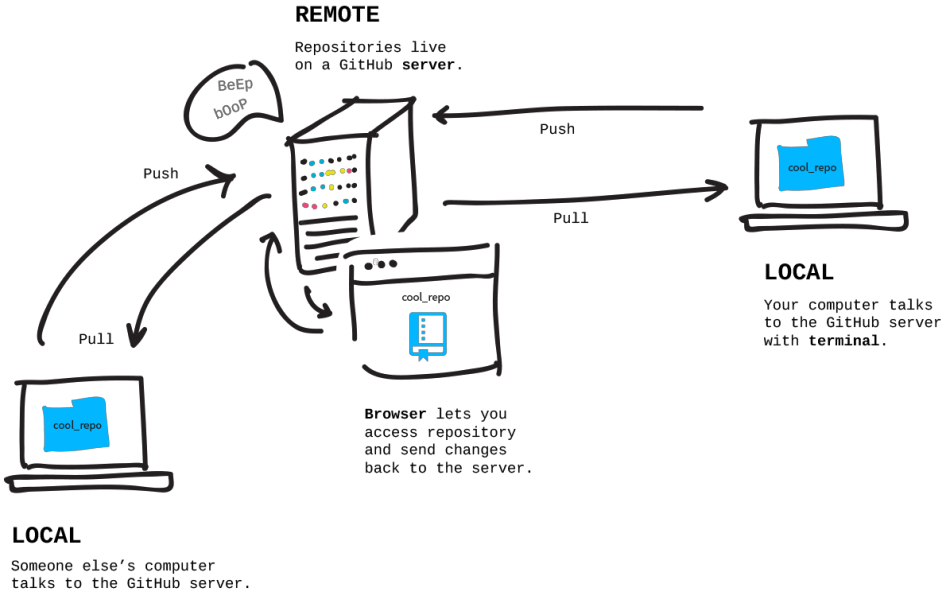
- Documentación:
  https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/examples_index.html





GITHUB

- GitHub es una plataforma de hospedaje de código para el control de versiones y la colaboración.
- Permite que personas trabajen juntos en proyectos desde donde sea.
- Cada proyecto es un repositorio, que todos los archivos de tu proyecto y el historial de revisiones de cada uno de ellos.
  - Puedes debatir y administrar el trabajo de tu proyecto dentro del repositorio.



https://docs.github.com/es/get-started/quickstart/hello-world

GITHUB y PORTFOLIOS

How to Build a Data Science Portfolio

Michael Galarnyk Jul 8, 2018 · 18 min read

A portfolio is one way to show people you are that data science unicorn.

https://towardsdatascience.com/how-to-build-a-data-science-portfolio-5f566517c79c

These data science portfolios will awe and inspire you (mid-2020 edition)

Use these to improve your own data science portfolio, learn new skills or discover new, interesting projects.

https://towardsdatascience.com/these-data-science-portfolios-will-awe-and-inspire-you-mid-2020-edition-728e1021f60

The Promise of Portfolios: Training Modern Data Scientists

by Deborah Nolan and Sara Stoudt

Published on Jul 30, 2021

last released 3 weeks ago

ABSTRACT

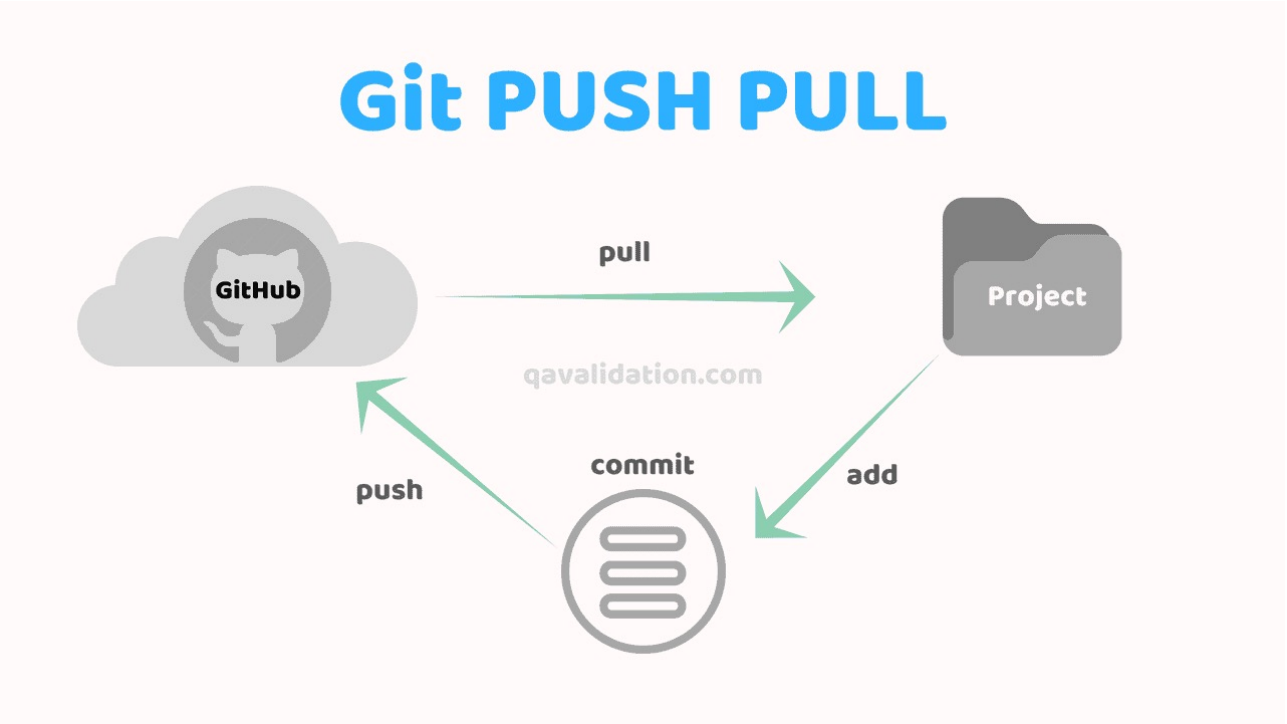
Data scientists rely on many technical skills and the ability to reason about data to solve problems. As educators grapple with how to prepare students in this new field, they are faced with identifying both what a student must know and what a student should be able to do by the end of their data science education, and also how to collect evidence of those abilities.

We present a way to unite and coordinate individual efforts toward training well-rounded data scientists: a data science portfolio that highlights strong communication. This structuring of classroom assignments provides a way to evaluate students' mastery of material in each class and also allows for a student to build a professional portfolio that remains valuable after the class is over.

Data science portfolio pieces broadly include written and visual assignments that give students practice crafting data-driven arguments and narratives for a variety of audiences. This flexible nature of the portfolio gives students a way to demonstrate their abilities in an inclusive, 'choose your own adventure' way.

https://hdsr.mitpress.mit.edu/pub/pm67plq9/release/2

GITHUB


Git PUSH PULL
GitHub
pull
Project
qavalidation.com
push
commit
add