



Taller de Introducción a Python con Google Colab

FACULTAD DE INGENIERÍA Y NEGOCIOS

Osvaldo Yáñez Osses, Ph.D.

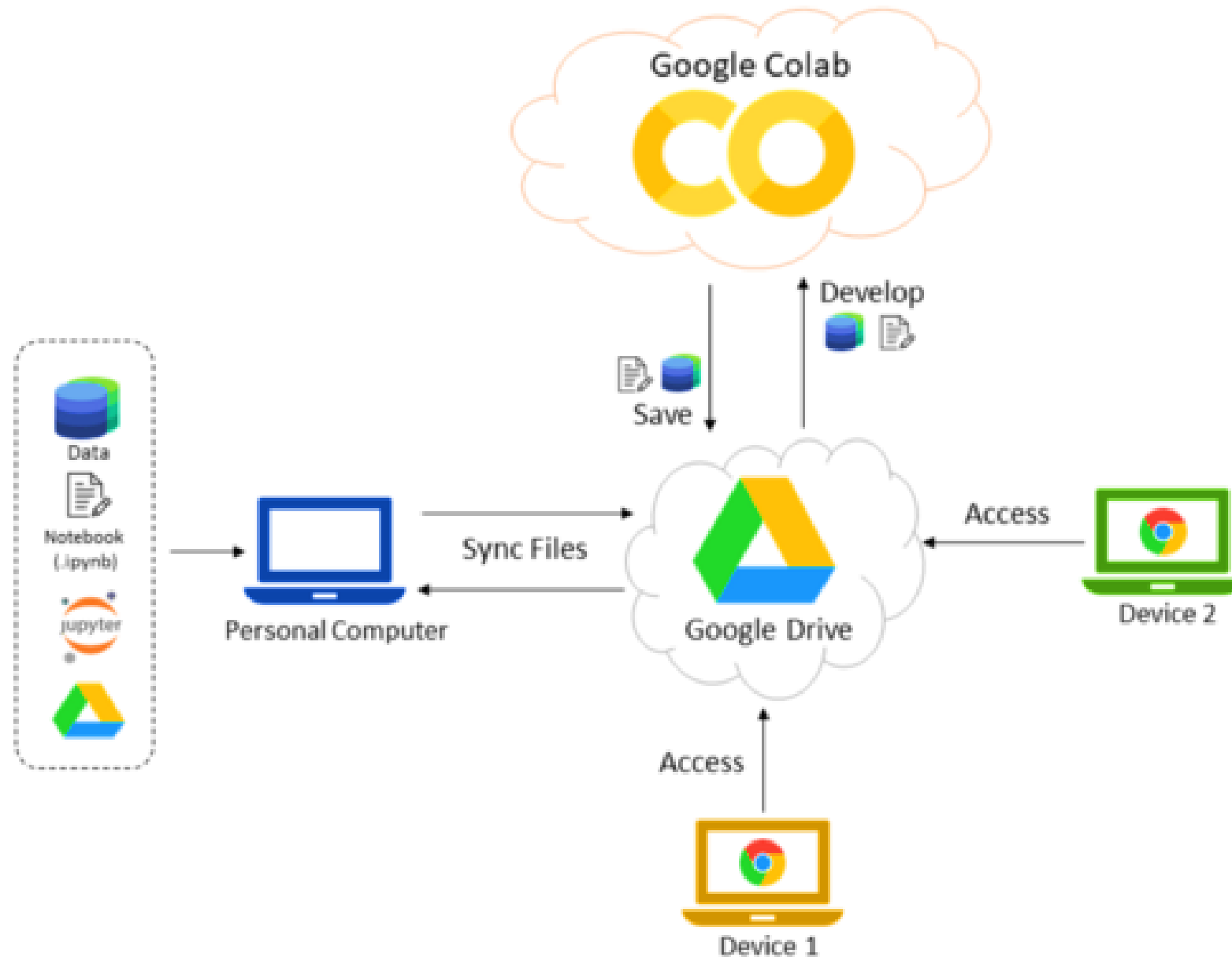
oyanez@udla.cl

Octubre - 2022

Es un servicio o libreta en línea gratuito, conectado con tu cuenta de Google, que te permite programar en python y usar CPU/GPU de una computadora en la nube.

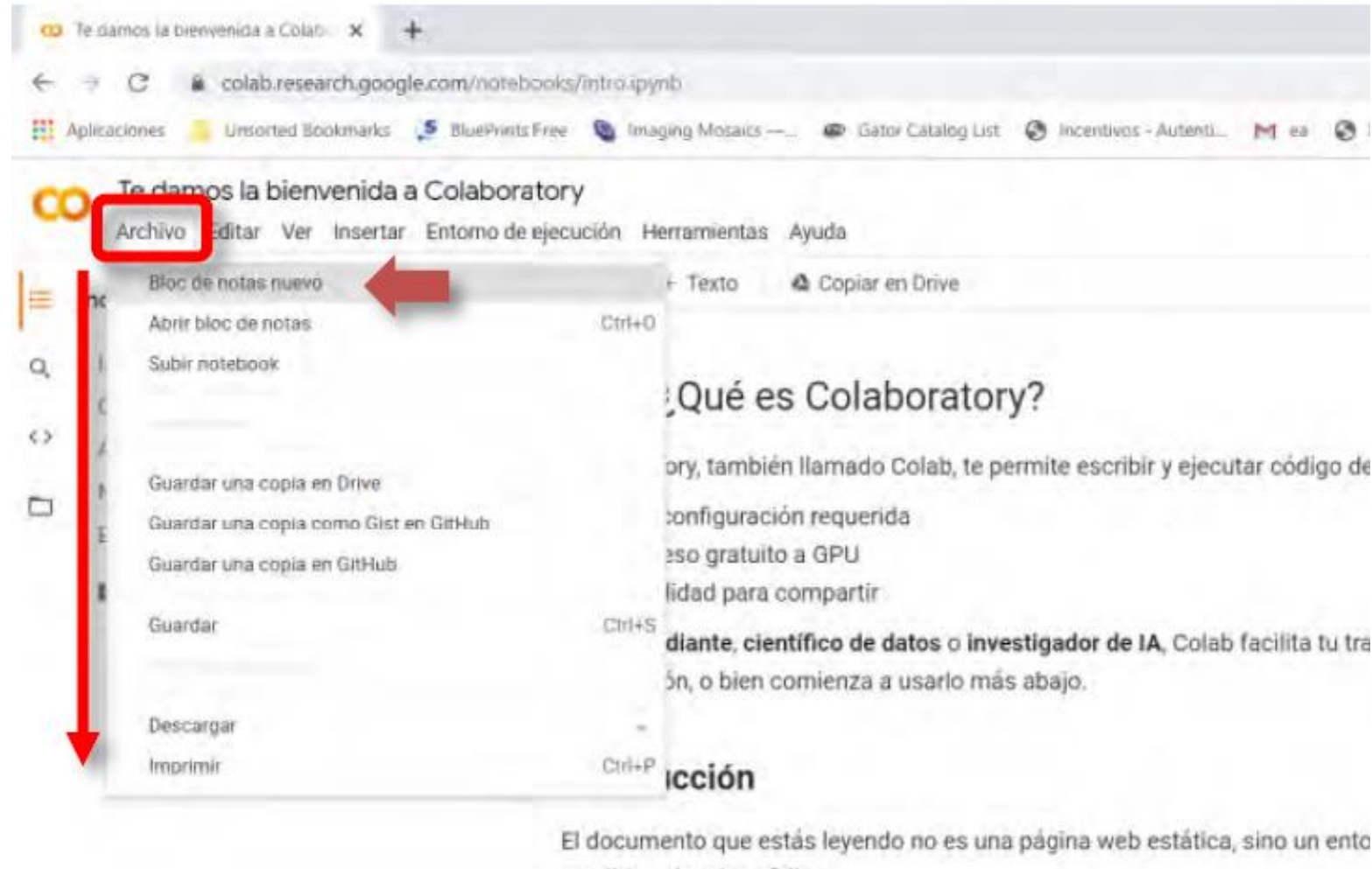
Las ventajas que tiene son :

- No requiere configuración.
- Acceso a GPUs sin coste adicional.
- Permite compartir contenido fácilmente.
- Colab puede facilitar tu trabajo, ya seas **estudiante, científico de datos o investigador de IA.**

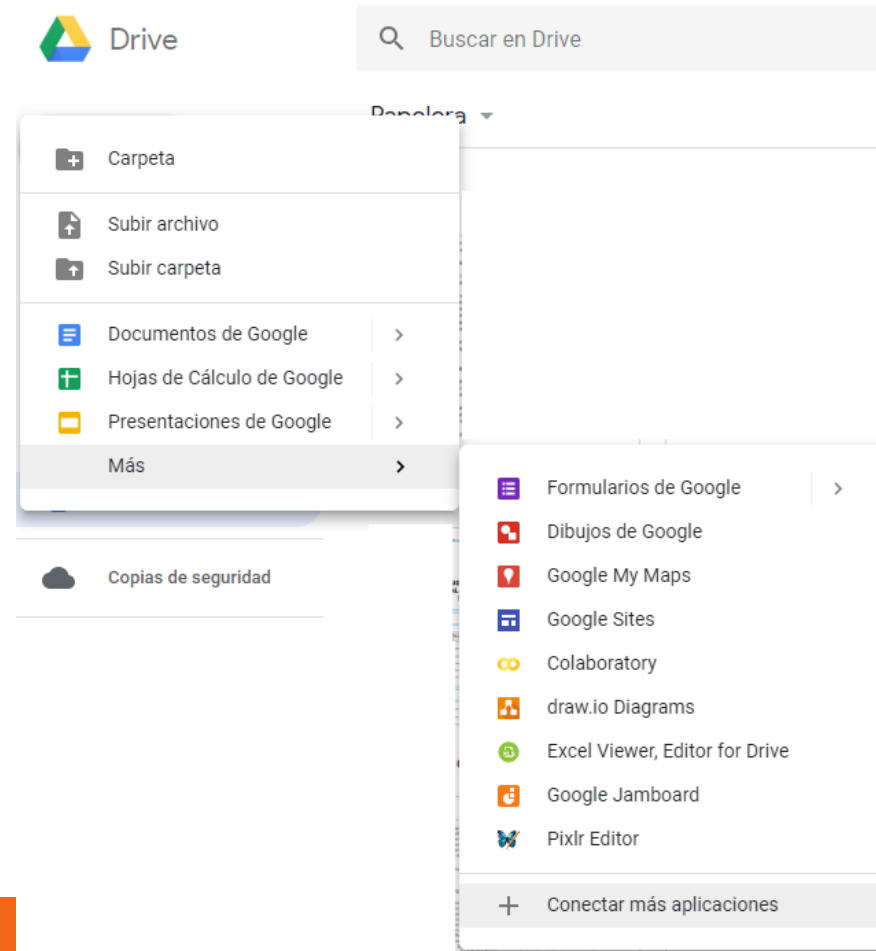


- [Google Colab](#)
- [Binder](#)
- [Kaggle notebooks](#)
- [DataLore](#)
- [Microsoft Azure ML](#)

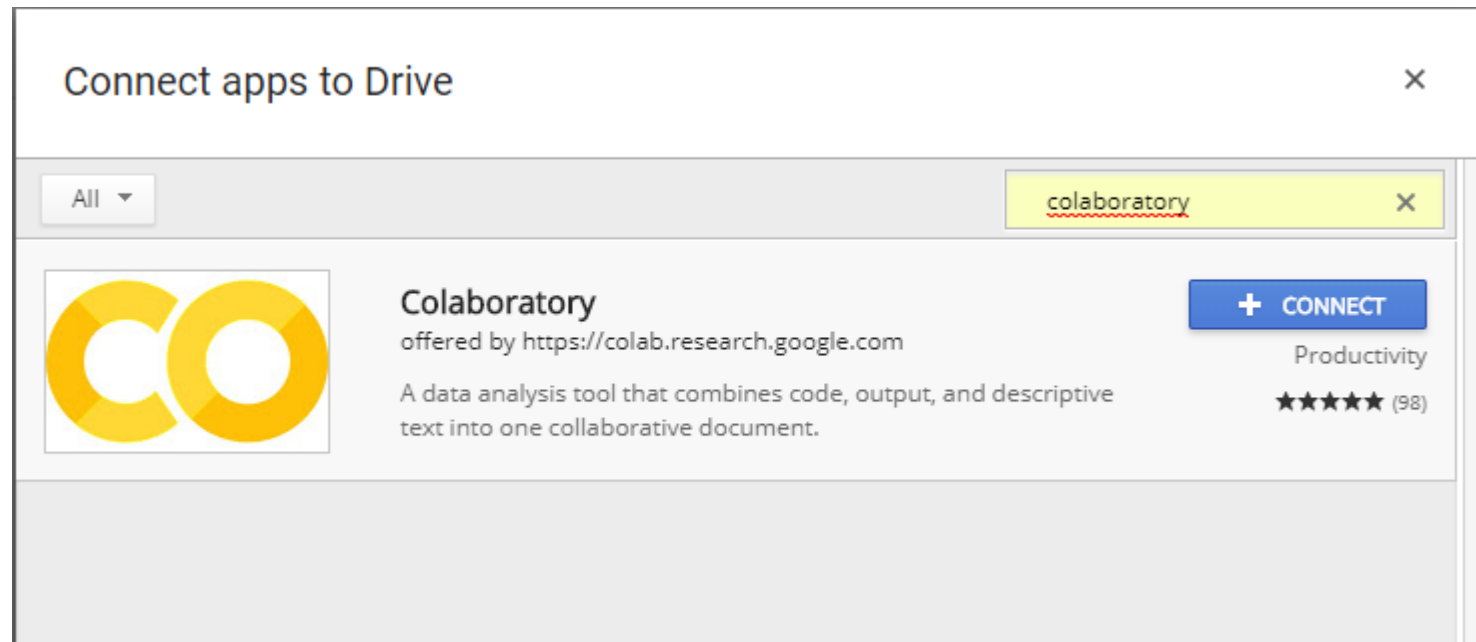
Ir al sitio de “Google Colab” (<https://colab.research.google.com/>), ir a “Archivo” (arriba a la izquierda) y elegir la opción “Block de notas nuevo”



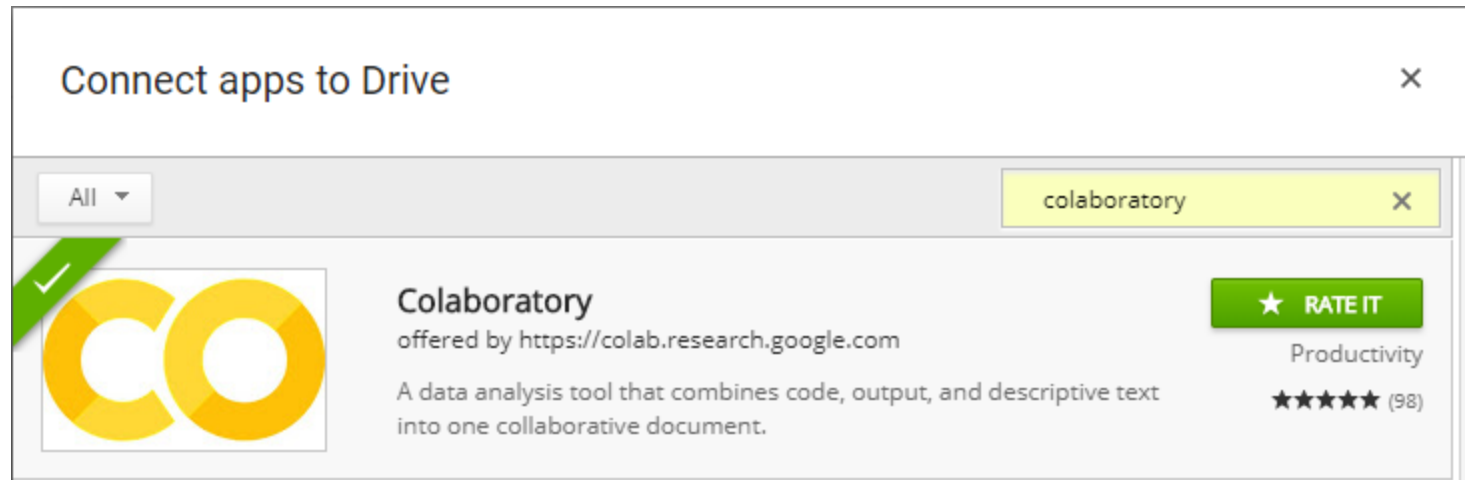
- Google Colab es una aplicación como cualquier otra aplicación (Google Docs, Sheets, etc.) que desee utilizar en Google Drive. Para instalarlo, has clic en el botón + Nuevo -> Más-> Conectar más aplicaciones, como se muestra a continuación. En mi caso ya tengo instalado previamente Google Colab, por eso vas a ver el icono en la figura.



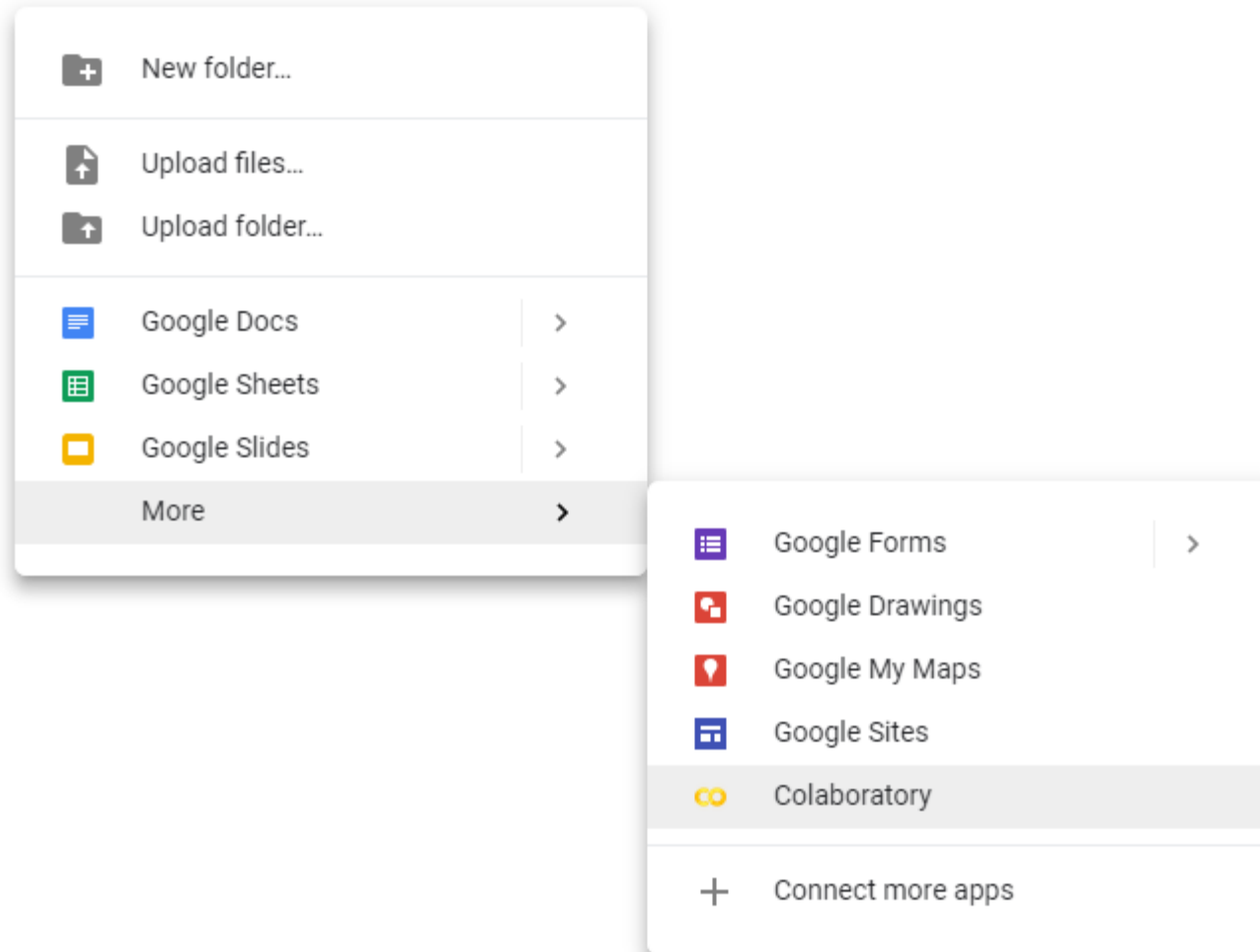
- A partir de ahí, se abrirá una ventana. Todo lo que debes hacer es escribir en la pestaña de la esquina superior derecha “Colaboratory” y aparecerá la aplicación.



- Ahora, has clic en el botón azul de conectar. Una vez instalada, aparecerá un mensaje indicando si permite que la aplicación abra archivos que puede abrir, has clic en Sí. Una vez que este paso se haya completado con éxito, debería verse el logotipo de Google Colab con un visto verde.

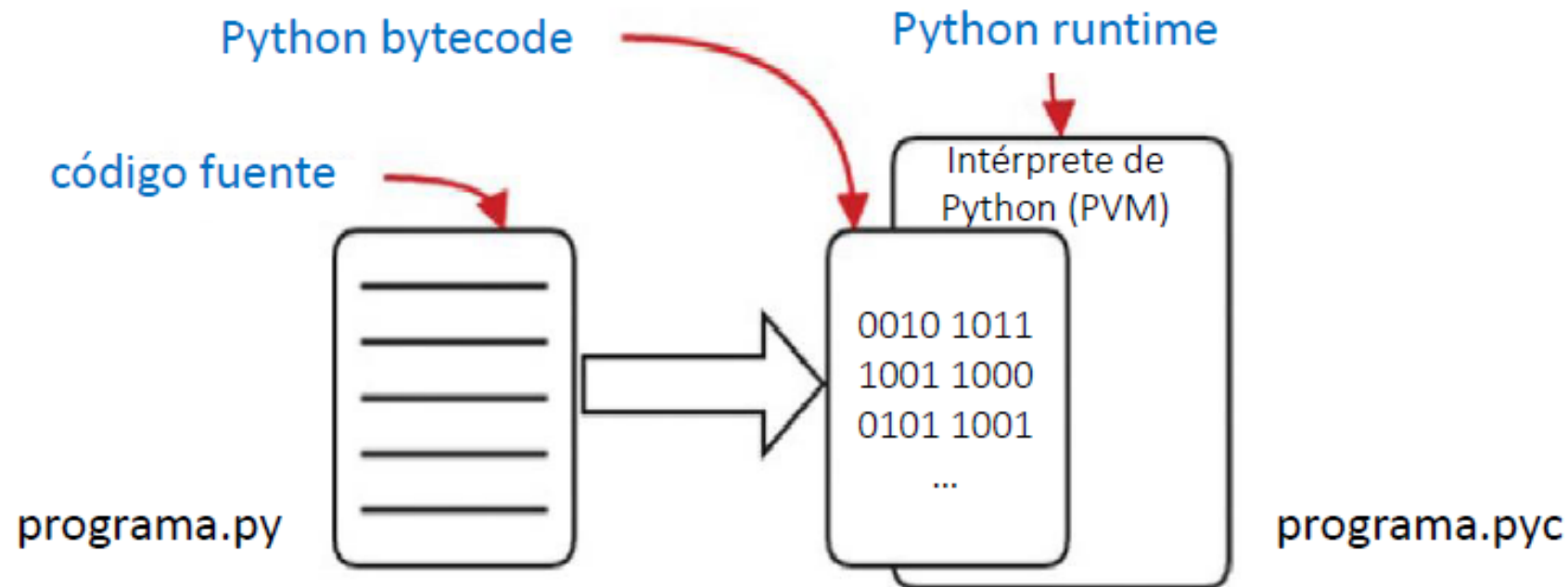


- Ahora, has clic en el botón + Nuevo -> Colaboratory



Esto creará un cuaderno en tu unidad de Google Drive y abrirá una nueva pestaña para el cuaderno.

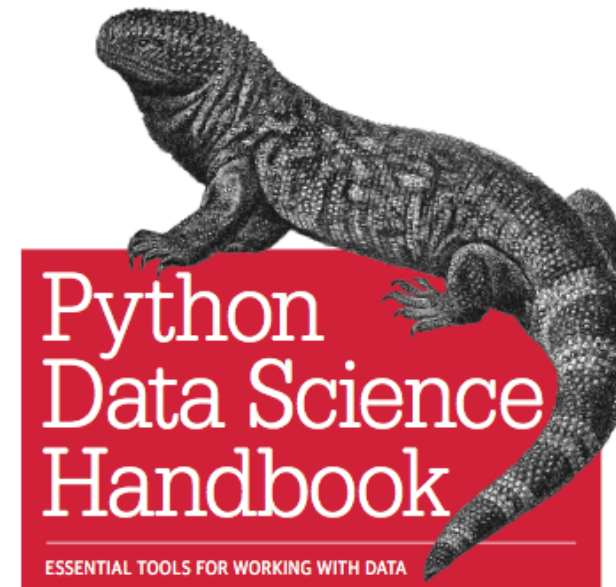
- Python es un lenguaje de programación multiparadigma: combina los paradigmas procedurales, orientado a objetos y funcional.
- Es un lenguaje interpretado: el *intérprete de Python* lee y ejecuta las instrucciones en el programa de usuario.
- El intérprete de Python compila a Python bytecode, un código multiplataforma que ejecuta en una Python Virtual Machine (PVM).



Python Data Science Handbook

Jake VanderPlas

O'REILLY®



<https://jakevdp.github.io/PythonDataScienceHandbook/>

Benefits of Learning Python



Data science



Machine Learning



Deep Learning



Web development



Predictive Analytics



Advanced Analytics

Finance and Trading



System automation



Computer graphics



Game development



Scientific Research



Security Testing



@pythonclcoding

@clcoding



Python Coding

PYTHON LIBRARIES AND FRAMEWORKS

Machine Learning

- Numpy
- Keras
- Theano
- Pandas
- PyTorch
- TensorFlow
- Scikit-Learn
- Matplotlib
- Scipy
- Seaborn

Web Development

- Django
- Flask
- Bottle
- CherryPy
- Pyramid
- Web2Py
- TurboGears
- CubicWeb
- Dash
- Falcon

Automation Testing

- Splinter
- Robot
- Behave
- PyUnit
- PyTest

Game Development

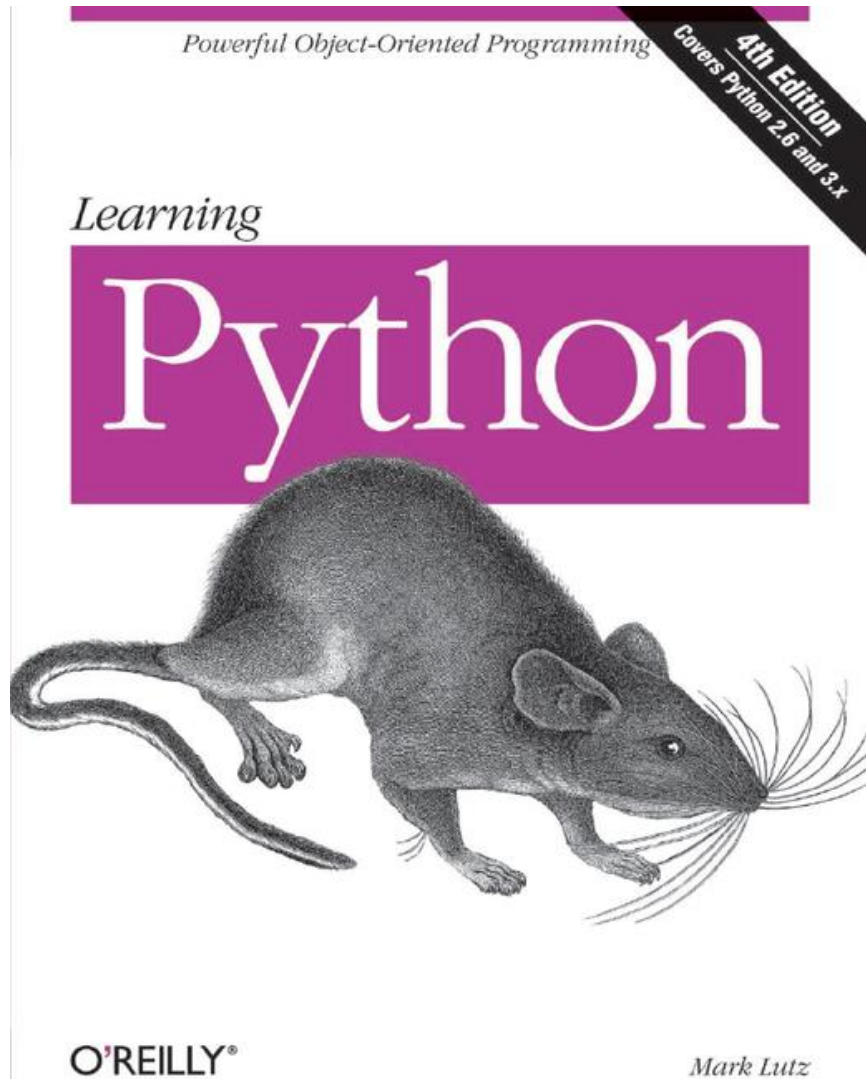
- PyGame
- PyGlet
- PyOpenGL
- Arcade
- Panda3D

Image Processing

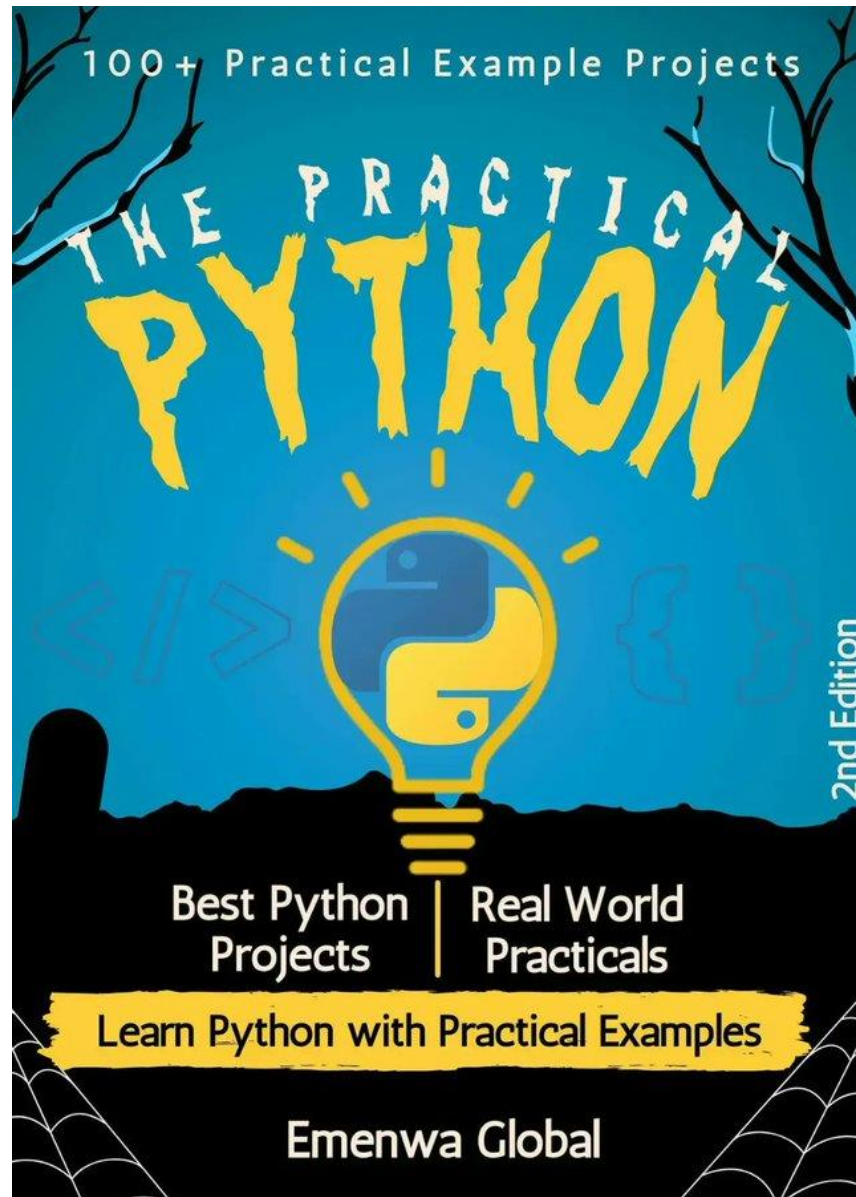
- OpenCV
- Mahotas
- Scikit-Image
- Pgmagick
- SimpleITK

Web Scraping

- Requests
- BeautifulSoup
- Scrapy
- Selenium
- lxml



https://www.dropbox.com/s/3gxyaaz362pb4no/Learning_Python.pdf?dl=0



<https://www.goodreads.com/book/show/59823784-the-practical-python>

110+ Data Science Projects with Python

(Solved and Explained for free)



<https://python.plainenglish.io/85-data-science-projects-c03c8750599e>



Top Python libraries a Data Scientist need to know

01

Pandas

02

NumPy

03

SciPy

04

Scrapy

05

Matplotlib



Seaborn

06

Scikit-Learn

07

TensorFlow

08

Scikit-Image

09

Librosa

10

Machine Learning Algorithms For Beginners



DECISION TREE

1. WHAT IS A DECISION TREE?

A type of supervised learning algorithm that models decisions in a hierarchical structure. It splits the data into subsets based on feature values, creating a tree-like structure. It is used for both classification and regression tasks.

MULTIPLE LINEAR REGRESSION

Multiple linear regression attempts to model the relationship between two or more features and a response by fitting a linear equation to observed data. The slope is defined by multiple linear regression coefficients, which are used to predict the response. The difference between the predicted and actual values is the error. You can use it to find out which factor has the highest impact on the predicted output and how different variables relate to each other.

$$y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$$

ASSUMPTIONS

1. Linearity: The relationship between dependent and independent variables should be linear.
2. Homoscedasticity: Constant variance of the residuals.
3. Independence: The residuals should be independent.

DUMMY VARIABLES

Variables that are used to represent categorical data in a regression model. They are typically binary (0 or 1) and are used to indicate the presence or absence of a certain feature.

RANDOM FOREST

AN INTUITION TO RANDOM FOREST

WHAT IS THE RANDOM FOREST ALGORITHM?

Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

Ensemble learning models aggregate multiple machine learning models, allowing for overall better performance.

SIMPLE LINEAR REGRESSION

Regression is a type of supervised learning algorithm that models the relationship between a single feature and a response variable. It is used for regression tasks.

How to find the best fit line?

In the regression model, we are trying to find the best fit line that passes through the data points. The equation for the best fit line is:

$$y = b_0 + b_1x_1$$

K-Means Clustering

UNSUPERVISED LEARNING

1. WHAT IS UNSUPERVISED LEARNING?

Unsupervised learning is a type of machine learning where the model is given data without any labels or target values. The goal is to find patterns and structure in the data.

2. CLUSTERING ALGORITHMS

Clustering algorithms are used to group data points into clusters based on their similarity. The most common clustering algorithm is K-Means.

K NEAREST NEIGHBOURS

An Intuition to K-NN Classification Algorithm

What is k-NN?

k-Nearest Neighbour algorithm is a simple yet most used classification algorithm. It can also be used for regression.

k-NN is non-parametric (means that it does not make any assumptions on the underlying data distribution), instance-based (means that our algorithm doesn't explicitly learn a model), instead, it classifies the new data by looking at the training instances.) and used in a large range of machine learning settings.

How Does k-NN Algorithm work?

k-NN is a lazy learning algorithm because it does not build a model during the training phase. It only stores the training data and waits until it is needed to make a prediction.

LOGISTIC REGRESSION

WHAT IS LOGISTIC REGRESSION

Logistic regression is used for a different class of problems known as classification problems. Here the aim is to predict the group to which the current object under observation belongs to. It gives you a discrete binary outcome between 0 and 1. A simple example would be whether a person will vote or not in upcoming elections.

How Does It Work?

Logistic Regression measures the relationship between the dependent variable (our label, what we want to predict) and the one or more independent variables (our features), by estimating probabilities using its underlying logistic function.

Sigmoid Function

The Sigmoid Function is an S-shaped curve that can take any real-valued number and map it into a value between the range of 0 and 1, but never exactly at those limits.

HIERARCHICAL CLUSTERING

Hierarchical clustering is a type of unsupervised learning, where the aim is to build up a hierarchy of clusters. It is used for clustering tasks.

<https://www.theinsaneapp.com/2021/11/machine-learning-algorithms-for-beginners.html>

integer, float, boolean, string, bytes

```

int 783 0 -192 0b010 0o642 0xF3
float 9.23 0.0 -1.7e-6
bool True False
str "One\nTwo"
bytes b"toto\xfe\775"

```

hexadecimal octal binary octal hexa

Multiline string:
"X\ty\tz"
"1\t2\t3"
escaped new line
'I\''
escaped
escaped tab

immutable

Base Types

ordered sequences, fast index access, repeatable values

```

list [1,5,9]
tuple (1,5,9)
str bytes

```

Non modifiable values (immutables) expression with only commas → tuple

key containers, no a priori order, fast key access, each key is unique

```

dict {"key": "value"}
dict (a=3, b=4, k="v")
set {"key1", "key2"}
frozenset immutable set

```

dictionary (key/value associations) {1: "one", 3: "three", 2: "two", 3.14: "n"}

collection {1, 9, 3, 0}

empty

Container Types

for variables, functions, modules, classes... names

a..zA..Z_ followed by a..zA..Z_0..9

- diacritics allowed but should be avoided
- language keywords forbidden
- lower/UPPER case discrimination

a toto x7 y_max BigOne
by and for

Identifiers

=

assignment → binding of a name with a value

1) evaluation of right side expression value

2) assignment in order with left side names

```

x=1.2+8*sin(y)
a=b=c=0
y,z,r=9.2,-7.6,0
a,b=b,a
a,*b=seq
*a,b=seq
x+=3
x-=2
x=None
del x

```

assignment to same value

multiple assignments

values swap

unpacking of sequence in item and list

increment → x=x+3

decrement → x=x-2

undefined → constant value

remove name x

and

or

not

...

Variables assignment

int("15") → 15

int("3f", 16) → 63

int(15.56) → 15

float("-11.24e8") → -1124000000.0

round(15.56, 1) → 15.6

bool(x) False for null x, empty container x, None or False x; True for other x

str(x) → "..." representation string of x for display (cf. formatting on the back)

chr(64) → '@' ord('@') → 64

repr(x) → "..." literal representation string of x

bytes([72, 9, 64]) → b'H\te'

list("abc") → ['a', 'b', 'c']

dict([(3, "three"), (1, "one")]) → {1: 'one', 3: 'three'}

set(["one", "two"]) → {'one', 'two'}

separator str and sequence of str → assembled str

str splitted on whitespaces → list of str

"words with spaces".split() → ['words', 'with', 'spaces']

str splitted on separator str → list of str

"1,4,8,2".split(",") → ['1', '4', '8', '2']

sequence of one type → list of another type (via list comprehension)

[int(x) for x in ('1', '29', '-3')] → [1, 29, -3]

Conversions

negative index

positive index

positive slice

negative slice

Access to sub-sequences via lst [start slice: end slice: step]

lst[: -1] → [10, 20, 30, 40]

lst[1: -1] → [20, 30, 40]

lst[: :2] → [10, 30, 50]

lst[1: -1] → [50, 40, 30, 20, 10]

lst[: : -2] → [50, 30, 10]

lst[: :2] → [10, 20, 30, 40, 50]

Missing slice indication → from start / up to end.

On mutable sequences (lst), remove with del lst[3:5] and modify with assignment lst[1:4]=[15, 25]

Sequence Containers Indexing

Comparisons: < > <= >= == !=

(boolean results)

a and b logical and

a or b logical or

not a logical not

True False

True and False constants

Boolean Logic

parent statement:

statement block 1...

parent statement:

statement block 2...

next statement after block 1

configure editor to insert 4 spaces in place of an indentation tab.

Statements Blocks

module true: file true.py

from monmod import nom1, nom2 as fct

import monmod

modules and packages searched in python path (cf sys.path)

Modules/Names Imports

if logical condition:

statements block

Can go with several elif, elif... and only one final else. Only the block of first true condition is executed.

if age <= 18:

state="Kid"

elif age >= 65:

state="Retired"

else:

state="Active"

Conditional Statement

Signaling an error:

raise Exception(...)

Errors processing:

try:

normal processing block

except Exception as e:

error processing block

finally block for final processing in all cases

Exceptions on Errors

floating numbers... approximated values

Operators: + - * / // % **

Priority (...)

@ → matrix × python 3.5 + numpy

(1+5.3)*2+12.6

abs(-3.2)+3.2

round(3.57, 1)+3.6

pow(4, 3)+64.0

usual order of operations

Maths

angles in radians

from math import sin, pi...

sin(pi/4) → 0.707...

cos(2*pi/3) → -0.4999...

sqrt(81) → 9.0

log(e**2) → 2.0

ceil(12.5) → 13

floor(12.5) → 12

modules math, statistics, random, decimal, fractions, numpy, etc (cf. doc)

Maths