

# Seminario PYTHON - MCAF

Francisco Gárate Santiago

UCM, 2022

# Sobre mi

Francisco Gárate

- ▶ Licenciado en ADE
- ▶ Licenciado en Ciencias Actuariales y Financieras (CAF)
- ▶ Actuario colegiado
- ▶ Ponente de la EPAF (Escuela de Práctica Actuarial y Financiera) del Instituto de Actuarios Españoles
- ▶ Autor de la librería de python pyliferisk (cálculo actuarial)
- ▶ Responsable de la función actuarial del grupo ASISA

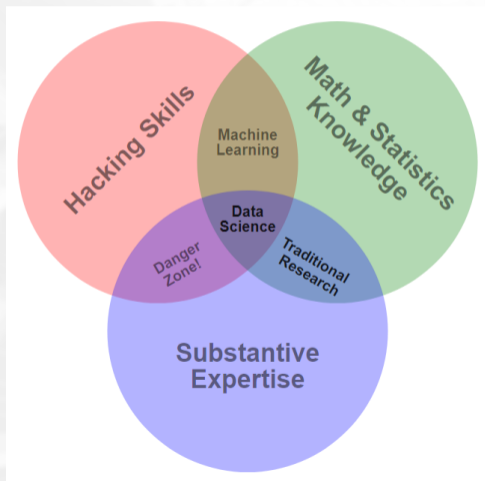
# Objetivo

## Programación en lenguaje Python

- ▶ Conocer la importancia de adquirir conocimientos en el principal lenguaje de programación en el campo de la ciencia de datos, big data y aprendizaje automático.
- ▶ Familiarizarse con el lenguaje de programación python. Ser capaz de crear o modificar una función en python, realizar bucles y condicionales.
- ▶ Conocer las principales funciones y utilidades de las principales librerías estadísticas: numpy, scipy y pandas.
- ▶ Introducción al aprendizaje automático con el lenguaje python.

# ¿Qué es la Ciencia de Datos?

Data Science Venn Diagram - Drew Conway, 2010



- ▶ **Habilidades Hacker.** Ser capaz de manipular archivos de texto en la línea de comandos, comprender operaciones vectorizadas, pensamiento abstracto, identificación de patrones ganadores...
- ▶ **Conocimientos matemáticos y estadísticos:** familiaridad básica con estas herramientas, ejemplo: qué es una regresión de mínimos cuadrados ordinarios y cómo interpretarla.
- ▶ **Experiencia real,** tener toda la información y además saber que hacer con ella, saber enfrentarse a casos reales. (No vale el: *Supongamos una vaca esférica*).

## ¿Por qué Python?

- ▶ Los lenguajes interpretados (como R, Python o Julia) consisten en líneas de instrucciones o scripts que son interpretados en tiempo real, por tanto no requieren ser compilados (como si sucede con C o C++) y pueden ser ejecutados en cualquier sistema operativo e incluso bajo entornos web.
- ▶ Python es uno de los lenguajes más utilizados en la denominada ciencia de datos (*data science*). Es fácil de escribir y de ser entendido sin necesidad de tener conocimientos avanzados de programación.
- ▶ Posee una de las mayores comunidades de desarrolladores (principalmente *open-source*), siendo uno de los lenguajes más populares en las comunidades de soporte (Stackoverflow, Github...), muy por encima de cualquier otro lenguaje.
- ▶ Python tiene soporte nativo en las principales bases de datos (SQL, DB2, Oracle, SAP DB...) y así como aquellas bases de datos utilizadas en Big Data (Azure SQL, Hadoop, Apache Beam, MongoDB o CDH).

# ¿Por qué Python?

<https://bigpanda.cern.ch>, <http://opendata.cern.ch>

- Desarrolladores: CERN, NASA, Federal Reserve Bank, Facebook...

## Python @ CERN

Sebastian Witowski



CERN – home of the Large Hadron Collider that can spit up to **1 petabyte of collision data per second**, impossible for the computing systems to record. The experiments, however, need only a small fraction of “interesting” events. The first-level trigger of ATLAS, for example, selects only **1 in 400 events** (making decisions in ~2.5 millionths of a second). These are sent at **160 GB/second** to the tens of thousands of processor cores which select **1%** of the remaining events for analysis. Even after such drastic data reduction, there are still around **50 petabytes** of data produced at CERN per year.

How does Python fit in this ecosystem? It might not be fast enough to be used for filtering this amount of data, but nevertheless, there are many great projects created with Python at CERN.



Thousands of scientists every day analyse data produced by the experiments at CERN. They need a tool that is based on a flexible and expressive language (like Python), but also works fast with large amount of data. And this is exactly what PyROOT is – a Python module that allows users to interact with ROOT (a data analysis framework written in C++, that is very popular in High Energy Physics community).

PyROOT is more than just a wrapper around ROOT. Python bindings are based on the C++ reflection information stored in the ROOT type system. Thanks to that, Python classes are created dynamically when needed and C++ functions and globals are available automatically in Python.

Users can use PyROOT directly in Python REPL. With all the ROOT classes and functions available out of the box, they can combine them with other modules like NumPy and SciPy to make data analysis easier and faster.



SWAN (Service for Web based Analysis) allows scientists to analyse their data in the CERN cloud. It uses Jupyter notebooks as the interface. The users' sessions are encapsulated in containers, synchronized and stored via CERNBox (file synchronization and sharing service build on top of Owncloud). Large datasets can be accessed on EOS – the disk-based storage system of CERN.



# ¿Por qué es importante saber programar?

Irás más lejos

Katie Bouman, (MIT)

<https://numpy.org/case-studies/blackhole-image/>

<https://analyticsindiamag.com/>

[how-imaging-a-blackhole-gives-us-one-more-reason-to-embrace-python-for-larger-datasets/](#)



# ¿Qué es la Ciencia de Datos?

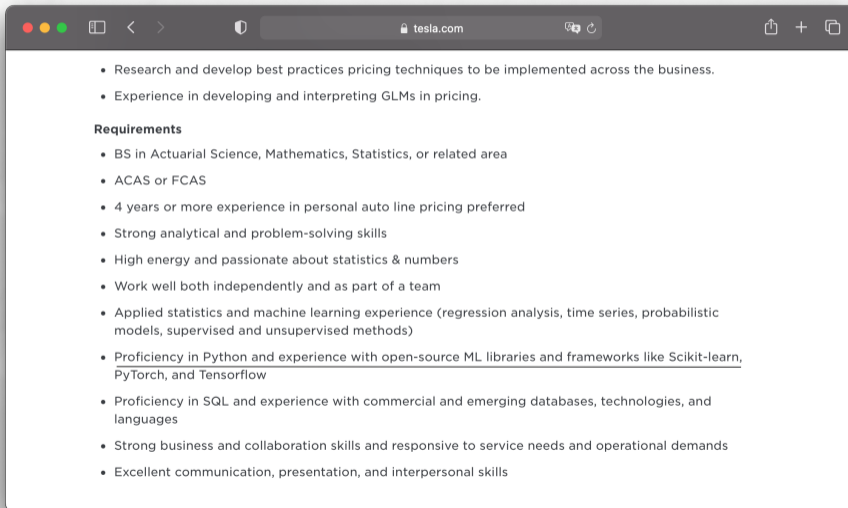
Unicornios



<https://www.tesla.com/careers/search/job/revolutionary-actuary-122171>


# ¿Qué es la Ciencia de Datos?

Unicornios




# ¿Qué es la Ciencia de Datos?

Unicornios



## Data Scientist

BBVA Data & Analytics •  Madrid y alrededores, España

At BBVA Data and Analytics, we are looking for people who want to help us in generating new opportunities and making impact through data science. We would like to embrace your mathematical, analytical, statistical, data-storytelling abilities to our offices in Madrid and Barcelona. If you've been both in academia and business and have a knack for problem solving we want to hear from you.

You like solving real-world problems. You are used to formulating hypotheses, setting up experiments to test these out, reformulate, retest, learn and understand that this is not an endless loop and that things need to converge to some proper balance between mathematical perfection and real-world requirements. You have a background in an analytical discipline and you are comfortable developing and trying out your own code, and can adapt to new working environments and requirements effortlessly.

If you also like storytelling and can use whichever techniques you find that suit best the stories that unfold from the data (visualization tools and libraries, generative narrative and whatnot) it's a plus. Our work sometimes involves communicating what your team just did to an audience that does

### How you match

Criteria provided by job poster

#### Skills

- ☒ Python (Programming Language)
- ☒ SQL
- ☐ Statistics
- ☐ Apache Spark
- ☐ Machine Learning
- ☐ Software Development
- ☐ Programming
- ☐ Data Science
- ☐ Algorithms
- ☐ Analytics

#### Level of education

- ☐ Master's Degree

# Introducción al lenguaje Python

## Características

- ▶ Los lenguajes interpretados (como R, Python o Julia) consisten en líneas de instrucciones o scripts que son interpretados en tiempo real, por tanto no requieren ser compilados (como si sucede con C o C++) y pueden ser ejecutados en cualquier sistema operativo e incluso bajo entornos web.
- ▶ Suele decirse que Python es la navaja suiza de los lenguajes de programación.
- ▶ La instalación de paquetes no puede ser más fácil. Ejemplo: ejecutar en la consola `pip install pandas` para instalar la librería/biblioteca pandas.

# Introducción al lenguaje Python

Guido van Rossum, 1990

El lenguaje Python fue creado por el holandés Guido van Rossum en 1990. La mentalidad con qué fue creado hace que sea muy intuitivo y uno de los lenguajes más fáciles de aprender.

The screenshot shows the Python.org website. At the top is the Python logo and a search bar. Below the logo is a navigation bar with links: About, Downloads, Documentation, Community, Success Stories, News, and Events. On the left side, there is a 'Tweets by @ThePSF' section showing three tweets from Python Software Foundation (@ThePSF). The main content area displays the 'Foreword for "Programming Python" (1st ed.)'. The text of the foreword is as follows:

**Foreword for "Programming Python" (1st ed.)**

This is the foreword I wrote for Mark Lutz' book "Programming Python" (1st ed.), published by O'Reilly. See also my foreword to the 2nd edition.

As Python's creator, I'd like to say a few words about its origins, adding a bit of personal philosophy.

Over six years ago, in December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office (a government-run research lab in Amsterdam) would be closed, but I had a home computer, and not much else on my hands. I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus).

# Introducción al lenguaje Python

## Historia



**Guido van Rossum** ✓

@gvanrossum

Today's Python history lesson: Python took its control and data structures from ABC, its identifiers, strings and %-string formats from C, and its regular expressions from Perl. But its # comments (and #!) and -c command line flag came from the UNIX v7 shell.

Traducido del inglés al [Google](#)

Lección de historia de Python de hoy: Python tomó sus estructuras de control y datos de ABC, sus identificadores, cadenas y formatos de% de cadena de C, y sus expresiones regulares de Perl. Pero sus # comentarios (y #!) y el indicador de línea de comando -c vino del shell de UNIX v7.

7:58 a. m. · 30 ene. 2020 · [Twitter Web App](#)