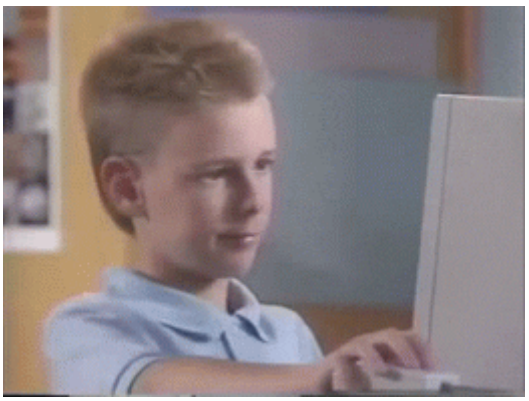
Hacia la utilización de los datos para mejorar la toma de decisiones

> *"The world is one big data problem."*
>
> Andrew McAfee, co-director of the MIT Initiative on the Digital Economy, and the associate director of the Center for Digital Business at the MIT Sloan School of Management.

Los primeros pasos en Data Science

¡Bienvenido/a a tu primera toolbox! Aquí encontrarás disponibles los conceptos clave y las herramientas que serán necesarias para prepararte para cada encuentro con tu equipo docente y compañeros/as. Además, motivaremos tu acceso a los canales de comunidad: te propondremos interactuar con otros/as Data Scientists para disipar dudas, hacer consultas y, por qué no, conocer y recomendar cosas interesantes sobre los temas que trabajaremos.

El primer Sprint de Aprendizaje (que consta de 16 toolboxes para 16 encuentros), tendrá por objetivo familiarizarnos con las herramientas fundamentales de Data Science, lo que nos permitirá comenzar a pensar como Data Scientists.



En particular, en este Sprint trabajaremos sobre dos áreas importantes: Análisis Exploratorio de Datos e Introducción al Aprendizaje Automático. El primero, es el

nombre que se le suele dar a la primera aproximación que hacemos a un dataset: buscamos patrones, visualizamos y comenzamos a entender de qué se tratan esos datos con los que vamos a trabajar. El Aprendizaje Automático (Machine Learning) es la disciplina que permitirá a nuestras computadoras - y a nosotros/as - aprender de ellos.

Para lograr estos objetivos, estudiaremos:

- Un lenguaje de programación (Python) junto con las librerías propias del análisis de datos.
- Probabilidad y Estadística: repasaremos y estudiaremos las herramientas de estas disciplinas que son fundamentales para el análisis de datos.

¿Qué es Data Science (o Ciencia de Datos)?

Nuestro tiempo de ocio, que muchas veces es poco pero no por eso menospreciado, queremos aprovecharlo al máximo. Sentarnos en el sillón a mirar una película o una serie en Netflix puede que sea todo lo que necesitamos un sábado a la noche después de tanta actividad. Pero a veces nos pasa que la plataforma cuenta con tantas opciones que lentamente comenzamos a perdernos dentro de los cientos de títulos de los cuales podemos elegir. Entrar en la paradoja de la elección puede hacernos perder mucho tiempo, y hasta sacarnos las ganas de ver algo nuevo para terminar viendo lo mismo de siempre.

Pero, por suerte, (¡o gracias a los/as Data Scientists!) Netflix y otras plataformas de streaming tienen opciones para recomendarnos en base a nuestra interacción con ellas. ¿Cómo lo hacen? Netflix cuenta con una gran cantidad de información de sus usuarios: qué vieron, qué les gustó y qué no, qué cosas dejaron sin terminar, etc. Con esa información, puede decidir qué recomendarnos en base a nuestro historial como usuarios, pero también a partir de otros/as usuarios/as que vieron contenido igual o similar al que vimos nosotros. Cómo hacer esas recomendaciones es un problema de Data Science.

Data Science es un campo **interdisciplinario** tanto en sus objetivos como en sus metodologías que, por medio de un proceso **iterativo,** busca:

DEFINIR

Partimos de preguntas que queremos responder. ¿Cuáles son los datos que necesitamos para responder las preguntas que nos hacemos?

INVESTIGAR

Se obtienen los datos, se “limpian” y se procede a explorarlos.

ANALIZAR

Los datos obtenidos se analizan con modelos (estadísticos, Machine Learning, etc.). Interpretamos los resultados y transformamos datos en información.

PRESENTAR

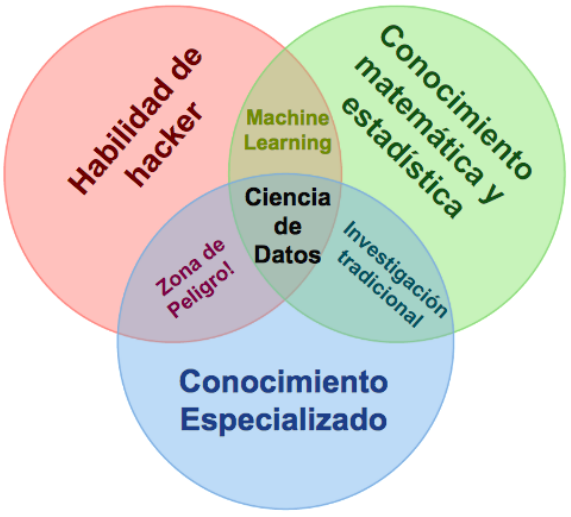
Presentamos los resultados obtenidos y las conclusiones a las que llegamos.

¿Qué hace un/a Data Scientist?

Esta podría ser una primera aproximación a la respuesta: **un/a data scientist (científico/a de datos) es alguien que, se hace tres preguntas:**

- *Dado un problema, ¿qué datos nos pueden ayudar a abordarlo y resolverlo?*
- *¿Dónde - y cómo - podemos conseguir datos?*
- *Dado un conjunto de datos, ¿qué problemas interesantes podemos trabajar con él?*

En este sentido, el diagrama de Venn de Drew Conway ubica a la tarea de un Data Scientist en la intersección entre los siguientes conocimientos y



competencias:

* Podríamos agregar las habilidades para comunicar los resultados.

La descripción que aporta Favio Vázquez resulta esclarecedora:

> *"un Científico de Datos es una persona (...) encargada de analizar problemas de negocios/organizaciones y ofrecer una solución estructurada que comienza convirtiendo este problema en una pregunta válida y completa. Luego, utilizando el método científico, la programación y las herramientas computacionales desarrollar códigos que preparan, limpian y analizan datos para crear modelos y responder la pregunta inicial".*

¿Por qué programar si elegimos trabajar con datos?

"Piensa en tu conjunto de datos como un grupo de fotos en negativo que las encontraste en un cuarto oscuro de revelado. La extracción de datos consiste en utilizar los equipos para revelar las fotos lo más rápido posible, para que puedas ver si hay algo inspirador o interesante en ellas. Al igual que con las fotos, recuerda no tomarte en serio lo que ves. Tú no tomaste las fotos, así que no sabes mucho sobre las historias que hay detrás de ellas. La regla de oro de la minería de datos es: ***enfocarse en lo que está aquí****"* (Cassie Kozyrkov — Head of Decision Intelligence, Google).

Siguiendo el ejemplo de Cassie Kozyrkov, los lenguajes de programación -como Python o R-, serían la herramienta de revelado. ¡Con calma! Si bien vamos a usar un lenguaje y aprenderemos a programar, esta NO es una carrera de programación. Aprenderemos lo necesario para resolver lo que un/a data scientist necesita. Los grandes volúmenes de datos que existen actualmente y los métodos utilizados para analizarlos nos exigen trabajar de manera eficiente pero flexible. Esto sólo es posible hacerlo utilizando un lenguaje de programación.

¿Qué es programar?

Cuando buscamos definiciones acerca de qué es la programación, nos encontramos con respuestas tales como “dar instrucciones al ordenador” o bien “crear software usando un lenguaje de programación”. Según wikipedia, es el proceso por el cual una persona desarrolla un programa valiéndose de una herramienta que le permita escribir el código **(nosotros usaremos Python)** y de otra que sea capaz de “traducirlo” a lo que se conoce como lenguaje de máquina, que puede comprender el ordenador.

¿Por qué Python?

Python es un lenguaje de programación de código abierto (open source) con una sintaxis simplificada y flexible que lo convierte en una de las mejores opciones para utilizar. Existen muchos motivos para elegir Python:

- Es fácil de usar, sobre todo cuando lo comparamos con otros lenguajes de programación.
- Es rápido y eficiente.
- Hay una gran comunidad usándolo, en particular en Data Science, Aprendizaje Automático e Inteligencia Artificial.
- Cuenta con una amplia cantidad de **librerías** específicas (¡pronto veremos qué son!).

Primeros Pasos con Python

Hay muchas formas de agrupar a los lenguajes de programación. Un poco de la jerga que puedes encontrar es:

- **Bajo o Alto nivel**: los lenguajes de alto nivel son más comprensibles para los/as usuarios/as y programadores/as, mientras que los de bajo nivel están más próximos a la arquitectura del hardware. En general, existe un compromiso entre Alto y Bajo nivel y la velocidad de ejecución, su eficiencia, etc.

- **Compilados o Interpretados**: cuando programamos, escribimos instrucciones que la computadora deberá ejecutar. Estas instrucciones, escritas en un lenguaje que el humano comprende, deben ser “traducidas”

> al lenguaje de máquina, es decir, instrucciones que la computadora entienda (en el fondo, 0 y 1). Un lenguaje compilado requiere un paso adicional antes de ser ejecutado —la compilación— para convertir el código a lenguaje de máquina. Un lenguaje interpretado, en cambio, es convertido a lenguaje de máquina a medida que es ejecutado.

Python es un lenguaje de **alto nivel** (¡aunque no lo creas!) e **interpretado**. En general, en Data Science se utilizan lenguajes interpretados (R también es un lenguaje de programación interpretado).

Sigamos avanzando. Si programar es "dar instrucciones al ordenador", de alguna forma debemos dar esas instrucciones. En la programación tradicional, esas instrucciones están escritas en uno o varios archivos (*scripts*), que juntos forman un programa. Para escribir esas instrucciones, el código es escrito usando un *entorno de desarrollo*, que es simplemente la interfaz con la computadora que usamos para, precisamente, escribir código. Para darte una idea, un entorno de desarrollo puede ser el Bloc de Notas de Windows, aunque en general usamos entornos más avanzados con muchas funcionalidades (colores específicos para instrucciones o palabras clave, autocompletar, compilar o ejecutar el código a medida que lo vamos desarrollando, etc.). Qué entorno de desarrollo utilizaremos dependerá del lenguaje de programación y, también, de la tarea a desarrollar.

En Data Science el entregable no suele ser un programa, pero tampoco un informe. Sino, más bien, lo que se llama un notebook.

Un notebook es la combinación de código con informe. Incluye el código utilizado para analizar los datos o generar figuras, intercalado con texto en lenguaje natural mediante el que explicamos qué estamos haciendo, los resultados alcanzados, las conclusiones y hasta ecuaciones matemáticas. Aquí va un ejemplo:

Notebook: DS_Toolbox_01_Ejemplo

Con el fin de generar estos notebooks, contamos con varios entornos de desarrollo. El más conocido es Jupyter Notebook, que instalaremos en nuestras computadoras. Cuando deseemos trabajar en la nube, usaremos Google Colab (que también corre sobre Jupyter Notebook). Sigue las instrucciones que te

dejamos en la sección de “Herramientas”. ¡Lo necesitarás para poner manos a la obra durante la meeting!

¡Prepárate para la próxima meeting!

Challenge de Herramientas (Parte 1)

📝 Instala Python. Asegúrate de estar conectado/a a una red segura y estable, y sigue los pasos a continuación:

Vamos a instalar una distribución particular: Miniconda. Si ya tienes instalado Anaconda está perfecto. Si tienes otra distribución, instalar Miniconda preferiblemente. Dejamos las instrucciones para los Sistemas Operativos más populares. En todos los casos, deben usar Python 3.7, Arquitectura de 64 Bits

WINDOWS

PASO 1: Instalar Python-Miniconda

1) Ir al link y descargar la versión correspondiente a tu sistema operativo. En este caso, Miniconda3 Windows 64-bit.

2) ¡Instalar!

3) Poner que “sí” a las preguntas que les haga

PASO 2: Comprobar que se instaló correctamente

1) Abrir la Anaconda Prompt (que es similar a una terminal). Si no sabes lo que es una terminal, basta con que vayas a la barra de inicio de tu computadora y escribas “terminal”, “prompt” o “cmd” en el buscador. En este caso, abrir “Anaconda Prompt”, que debería figurar entre las opciones.

Así se suele ver la Anaconda Prompt en Windows.

2) Tipear *python* + Enter

3) Fijarse qué versión de Python les aparece

4) Poner *2+5* y apretar Enter. ¿Qué más pueden hacer?

5) Para salir, poner *exit()* + Enter o cerrar la terminal.

Si algún paso falló, puede ser que A) no encuentres la Anaconda Prompt ó B) cuando escribas Python no reconozca esa instrucción. En ese caso, no te preocupes y salteate lo pasos siguientes. Lo pueden consultar durante el primer encuentro.

macOS (Apple)

PASO 1: Instalar Python-Miniconda

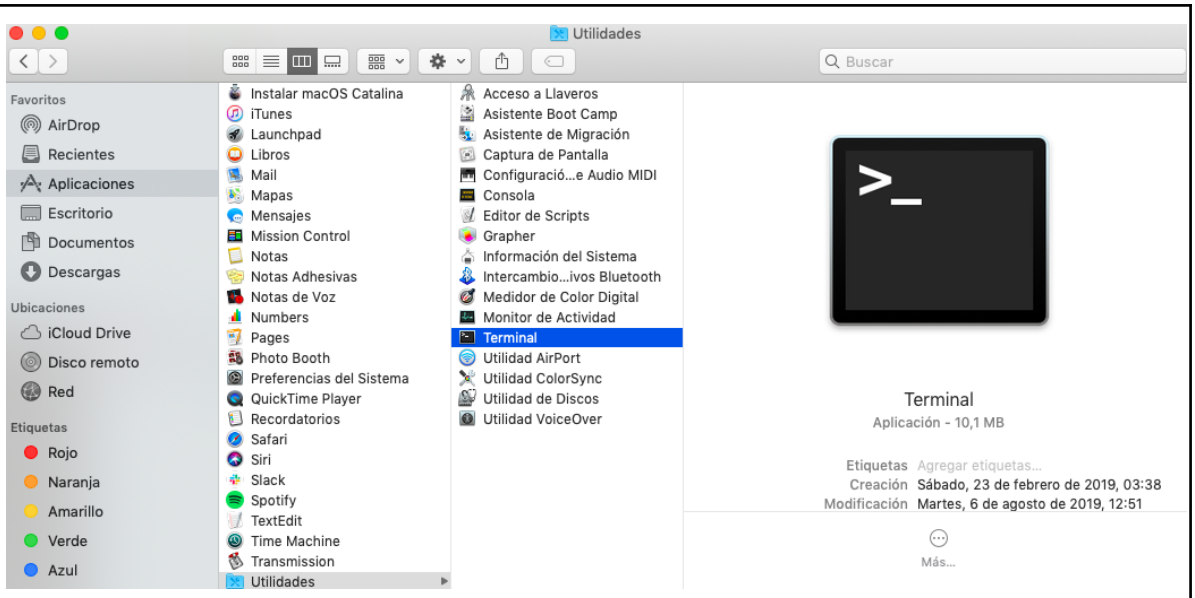
1) Ir al link y descargar la versión correspondiente a tu sistema operativo. En este caso, Miniconda3 MacOSX 64-bit pkg.

2) ¡Instalar!

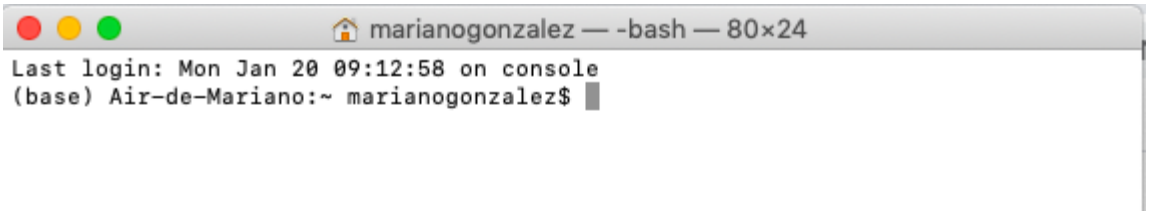
3) Poner que “sí” a las preguntas que les haga

PASO 2: Comprobar que se instaló correctamente

1) Abrir una Terminal (Aplicaciones -> Utilidades -> Terminal).



Así puedes abrir la terminal.



Así se suele ver una terminal en Mac luego de haber instalado Miniconda

2) Tipear *python* + Enter

3) Fijarse qué versión de Python les aparece

4) Poner *2+5* y apretar Enter. ¿Qué más pueden hacer?

5) Para salir, poner *exit()* + Enter o cerrar la terminal.

UBUNTU

PASO 1: Instalar Python-Miniconda

1) Ir al link y descargar la versión correspondiente a tu sistema operativo. En este caso, Miniconda Linux 64-bit.

2) ¡Instalar!

3) Poner que “sí” a las preguntas que les haga

PASO 2: Comprobar que se instaló correctamente

1) Abrir una Terminal (Aplicaciones -> Utilidades -> Terminal).

Así se suele ver una terminal en Ubuntu.

2) Tipear *python* + Enter

3) Fijarse qué versión de Python les aparece

4) Poner *2+5* y apretar Enter. ¿Qué más pueden hacer?

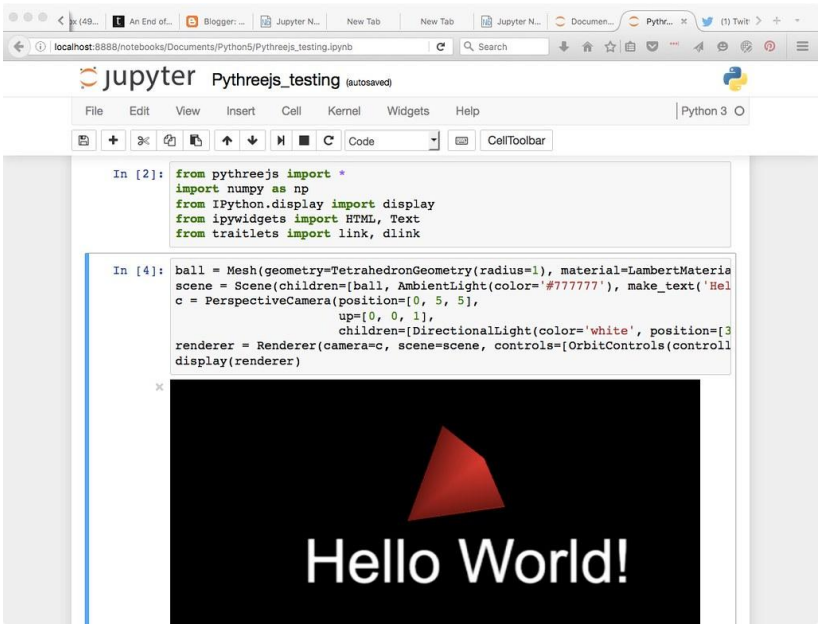
5) Para salir, poner *exit()* + Enter o cerrar la terminal.

Si algún paso falló, puede ser que cuando escribas Python no reconozca esa instrucción. En ese caso, no te preocupes y salteate lo pasos siguientes. Lo pueden consultar durante el primer encuentro.

Challenge de Herramientas (Parte 2)

📝 Instala Jupyter (los siguientes pasos son comunes a todos los sistemas operativos).

Si bien se puede trabajar desde la terminal, la realidad es que no es el modo más cómodo de hacerlo, ya que luego de cada instrucción, debemos apretar Enter para que se ejecute, las instrucciones no quedan guardadas, etc. Por ese motivo durante la cursada nosotros vamos a usar notebooks de Jupyter. como el que les mostramos anteriormente.



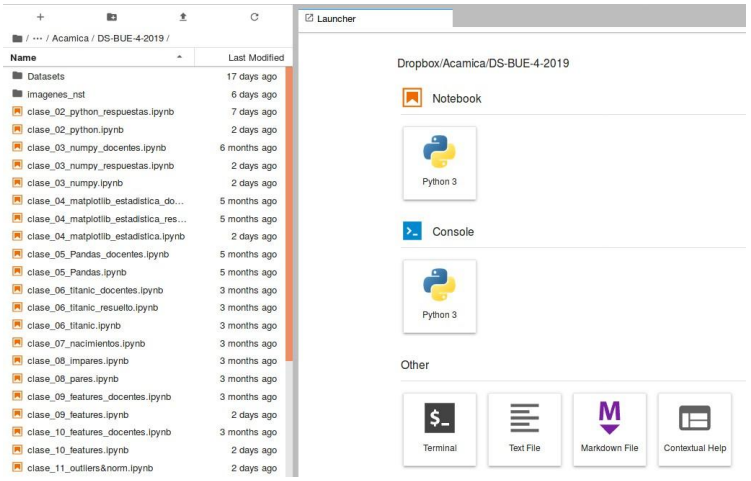
Los notebooks de Jupyter permiten intercalar código con texto, fórmulas matemáticas, imágenes y figuras generadas por el propio código. Es una herramienta muy útil de prototipado y comunicación efectiva entre colegas.

Conda (ver **Glosario**) nos permite crear ambientes (environments) de trabajo e instalar librerías, las cuales vendrían a ser complementos de un lenguaje de programación. No es obligatorio para trabajar con Python crear un ambiente, pero sí conveniente. A medida que vayamos trabajando con Python y vayamos instalando más librerías, van a ir quedando claro las ventajas de trabajar con ambientes, pero recomendamos que miren el siguiente enlace, **Getting started with Python environments (using Conda), o la documentación oficial sobre cómo crear ambientes con Conda.**

1. Abrir una terminal nuevamente y poner *conda create --name datascience*. Les preguntará si proceder, poner que sí (*y*).
2. Activar el ambiente: *conda activate datascience*
3. Instalar las librerías Jupyter, Notebook y JupyterLab: "*conda install jupyter notebook jupyterlab*". Les preguntará si proceder, poner que sí (*y*).
4. Comprobar que anduvo. Poner *jupyter lab*. Debería abrirles una pestaña en su navegador.
5. Descargar el notebook que te mostramos de ejemplo y abrirlo. Para ello,

pueden buscarlo en sus computadoras usando la sección izquierda del *Lab*.

6. Si quieren crear un nuevo notebook vacío, pueden hacerlo desde la pestaña File (¡Usar Python 3!) o clickeando en la opción Python 3 debajo de Notebook en el *Launcher*.



Herramientas (Parte 3)

🔧 Google Colab. Para empezar a conocer esta herramienta, te recomendamos que mires:

- El Notebook de Bienvenida
- El Notebook de Overview of Colaboratory Features

Profundiza

📹 El alcance de ser data scientist

📄Si te interesa conocer más sobre el crecimiento de Python en distintas disciplinas te invitamos a leer la siguiente nota: Aprender a programar en Python: por qué muchos programadores están empezando

📹 ¿Te quedaron muchas dudas o no entendiste qué hacen algunas de las herramientas? Te dejamos una serie de videos donde, brevemente, se explica el uso de cada una de ellas. ¡Ninguno dura más de tres minutos!

- Introducción
- Ambientes de trabajo

- Qué es Conda
- Instalando libs en Conda
- Jupyter
- ¿Shell o notebook?
- Markdown y código

Muchas veces vas a ver que dejamos más de un recurso sobre un mismo tema. Si tienes ganas y tiempo, míralos a todos, pero ante la duda elige UNO solo. Es mejor leer uno completo que varios por arriba. Hay mucha información repetida y no te perderás de mucho.

Python

Te dejamos algunos recursos para que mires sobre Python y cómo usarlo. Recomendamos que tengas Jupyter Lab abierto con un Notebook vacío - o, si prefieres, Colab - y pruebes los ejemplos que te van mostrando. ¡No tengas miedo de probar y *romper*! ¡Aprender a programar es aprender de los errores!

- 📄 https://learnxinyminutes.com/docs/python/ - Exclusivo sobre programación en Python, sin mucho contexto y directo al grano. Minimalista, pero detallado. Muy útil si ya tienes una base de programación, pero si no, también. Tal vez el mejor recurso para empezar a probar cosas en tu notebook o en Colab. En algunas cosas profundiza más de lo que vamos a necesitar, pero eso nunca es malo.
- 📄 https://www.tutorialsteacher.com/python - Muy completo y se explaya sobre algunas cuestiones que en esta toolbox mencionamos brevemente. Útil para “tener a mano”, ya que tiene un índice que es cómodo. No le prestes atención a la sección sobre cómo instalar Python.
- 📗 Python Data Science Handbook - Un recurso que utilizaremos mucho, ya que se trata de un libro de acceso gratuito con una explicación de muchas herramientas que veremos durante el programa. Además, está orientado a Ciencia de Datos, como su nombre lo indica. Lo recomendamos fuertemente, pero tal vez conviene que lo empiecen a mirar una vez que ya hayan programado un poco.

Glosario

Python: lenguaje de programación que utilizaremos durante el programa. Es un lenguaje de propósito general, pero tiene una gran comunidad en Ciencia de Datos, Machine Learning y Deep Learning

Miniconda: una distribución de Python, hermana menor - y minimalista - de Anaconda.

Conda: un manejador de paquetes. Es la herramienta que utilizaremos por defecto para instalar librerías en python y crear ambientes. Cuando falle, recurriremos pip.

Jupyter: entorno de desarrollo. Viene en dos sabores, Jupyter Notebook y, más recientemente, Jupyter Lab. Ambos trabajan sobre los mismos notebook y son muy parecidos. La principal diferencia es que Lab tiene una interfaz más amigable y más parecida a un navegador.

Notebook: archivo en el cual escribimos código, visualizamos resultados e imágenes, escribimos texto con preguntas, descripciones, conclusiones, etc. Su extensión es *.ipynb*.

Google Colab: entorno de desarrollo online, orientado a Ciencia de Datos, basado en Jupyter, en el cual podemos desarrollar o correr notebooks en la nube. Gran recurso si no queremos instalar todas las librerías en nuestras computadoras.