Elementos de Programación y Cálculo Numérico

Clase 1 – Introducción y Presentación del curso

Docentes

- Diego Passarella, diego.passarella@noreste.udelar.edu.uy
- Víctor Viana, victor.viana@noreste.udelar.edu.uy
- Fellipe Lousada, fellipe.lousada@noreste.udelar.edu.uy

Objetivo

Desarrollar competencias en programación (Python), cálculo numérico y análisis y procesamiento de datos, con foco en la resolución de problemáticas lineales y no lineales relevantes en la Ingeniería, incorporando además nociones de inteligencia artificial aplicada al ámbito forestal.

Organización y temario

Clases Semanales (Teórico / Prácticas (~ 2 horas)

Temario:

- Fundamentos de Programación
- Cálculo numérico aplicado
- Ecuaciones diferenciales ordinarias (EDOs)
- Análisis y Minería de datos
- Inteligencia Artificial: Aprendizaje Automático, Aprendizaje Profundo, Modelos Generativos y Modelos de Lenguaje

Metodología y Evaluación

- Trabajo Práctico - Monografía (50%)
- Presentación (50%)
- Para exonerar, se requiere al menos un 60% en el Trabajo Práctico y en la Presentación.

Sugerencias de temas para el Trabajos Final

- **Evaluación numérica de la humedad del suelo y vegetación con índices remotos**
- **Sistema de alerta temprana para plagas forestales basado en datos meteorológicos**
- **Modelado del crecimiento forestal mediante ecuaciones diferenciales y inteligencia artificial**
- **Optimización de rutas de transporte de madera bajo condiciones climáticas variables**
- **Predicción de riesgo de incendios forestales con modelos multivariables**
- **Estimación de biomasa forestal con modelos de regresión.**
- **Clasificación de especies de árboles mediante análisis de imágenes.**

Historia de Python

En 1991 **Guido van Rossum** desarrolló Python mientras trabajaba buscando un lenguaje fácil de aprender y usar, inspirado en la simplicidad pero con capacidades avanzadas como la de C. Nombró el lenguaje en honor al grupo de comedia británico Monty Python, reflejando su deseo de hacerlo divertido y accesible. Guido lideró su desarrollo durante más de 30 años y en 2018, se retiró del liderazgo activo, pero sigue siendo una figura influyente en el mundo de la programación.

Principales virtudes de Python

Simplicidad y legibilidad: Su sintaxis clara y minimalista facilita el aprendizaje

Multiplataforma: Funciona en diversos sistemas operativos como Windows, macOS, Linux, entre otros.

Ecosistema robusto: Posee miles de librerías para análisis de datos, desarrollo web, machine learning, etc.

Lenguaje interpretado: No requiere compilación, lo que permite una rápida ejecución y pruebas interactivas.

Programación multiparadigma: Compatible con programación orientada a objetos, funcional e imperativa.

Gran comunidad: Una comunidad activa que ofrece recursos, documentación y soporte gratuito.

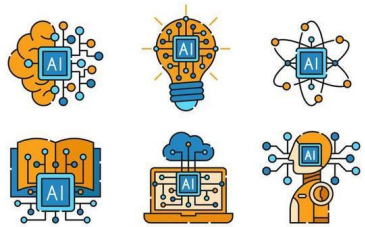
Integración: Fácil de integrar con otros lenguajes y tecnologías (C, C++, Java, REST APIs, etc).

¿Por qué Python (y no R)?

- Sintaxis más sencilla y consistente
- Menos abstracciones
- Mayor flexibilidad (Propósito general y Enorme base de código (paquetes))

Áreas de mayor desarrollo

- **Análisis y Ciencia de Datos**: Con librerías como pandas, numpy, matplotlib y scikit-learn.
- **Inteligencia Artificial (Machine and Deep Learning)**: Usando TensorFlow, PyTorch y Keras.
- **Desarrollo Web**: Con frameworks como Django y Flask.
- **Automatización y Scripting**: Ideal para crear scripts que simplifiquen tareas repetitivas.
- **Desarrollo de software**: Para crear aplicaciones completas y prototipos.
- **Web Scraping**: Para extraer datos de sitios web, utilizando herramientas como BeautifulSoup o Scrapy.



Conceptos básicos de Python

int (Enteros): Números enteros, positivos o negativos, sin decimales (ejemplo: 5, -3).

float (Flotantes): Números con decimales (ejemplo: 3.14, -0.5).

str (Cadenas): Texto, representado entre comillas simples o dobles (ejemplo: "Hola").

bool (Booleanos): Representa valores lógicos: True o False.

list (Listas): Conjuntos ordenados de elementos, que pueden ser de diferentes tipos (ejemplo: [1, "hola", 3.5]).

tuple (Tuplas): Conjuntos ordenados e inmutables de elementos (ejemplo: (1, 2, 3)).

dict (Diccionarios): Pares clave-valor, usados para almacenar datos asociados (ejemplo: {"nombre": "Ana", "edad": 30}).

set (Conjuntos): Colecciones desordenadas de elementos únicos (ejemplo: {1, 2, 3}).

Conceptos básicos de Python

Variable: Una **variable** en programación es un contenedor que almacena un valor o dato en la memoria.

En Python, se definen simplemente asignando un valor con el operador = (por ejemplo, x = 10).

*Python es un lenguaje **débilmente tipado**, lo que significa que no requiere especificar el tipo de dato de una variable al declararla; este se infiere automáticamente en tiempo de ejecución. Esto facilita el desarrollo rápido y flexible, ya que las variables pueden cambiar de tipo durante la ejecución. Sin embargo, esta característica puede generar errores inesperados en comparación con lenguajes fuertemente tipados como Java o C++, donde el tipo de las variables es estricto y debe ser declarado explícitamente.*

Pros

✔ **Facilidad de uso y flexibilidad**: No requiere declarar tipos de variables, lo que acelera el desarrollo y hace que el código sea más.

✔ **Menor curva de aprendizaje**: Ideal para principiantes.

✔ **Prototipado rápido**: Permite iterar rápidamente en el desarrollo de scripts o aplicaciones sin preocuparse por declarar y manejar tipos explícitos.

✔ **Flexibilidad en funciones y estructuras de datos**: Las funciones pueden aceptar y devolver diferentes tipos de datos sin restricciones estrictas.

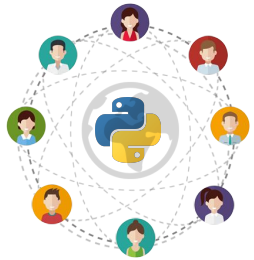
Contras

X **Mayor riesgo de errores en tiempo de ejecución**: Como los tipos no se verifican en tiempo de compilación, los errores relacionados con tipos pueden surgir solo al ejecutar el código.

X **Menor eficiencia en programas grandes**: Los errores relacionados con tipos pueden ser difíciles de detectar y depurar en proyectos grandes.

X **Desempeño**: Las inferencias de tipos de datos pueden hacer que Python sea más lento en comparación con lenguajes de tipado estático.

Ecosistema de Python



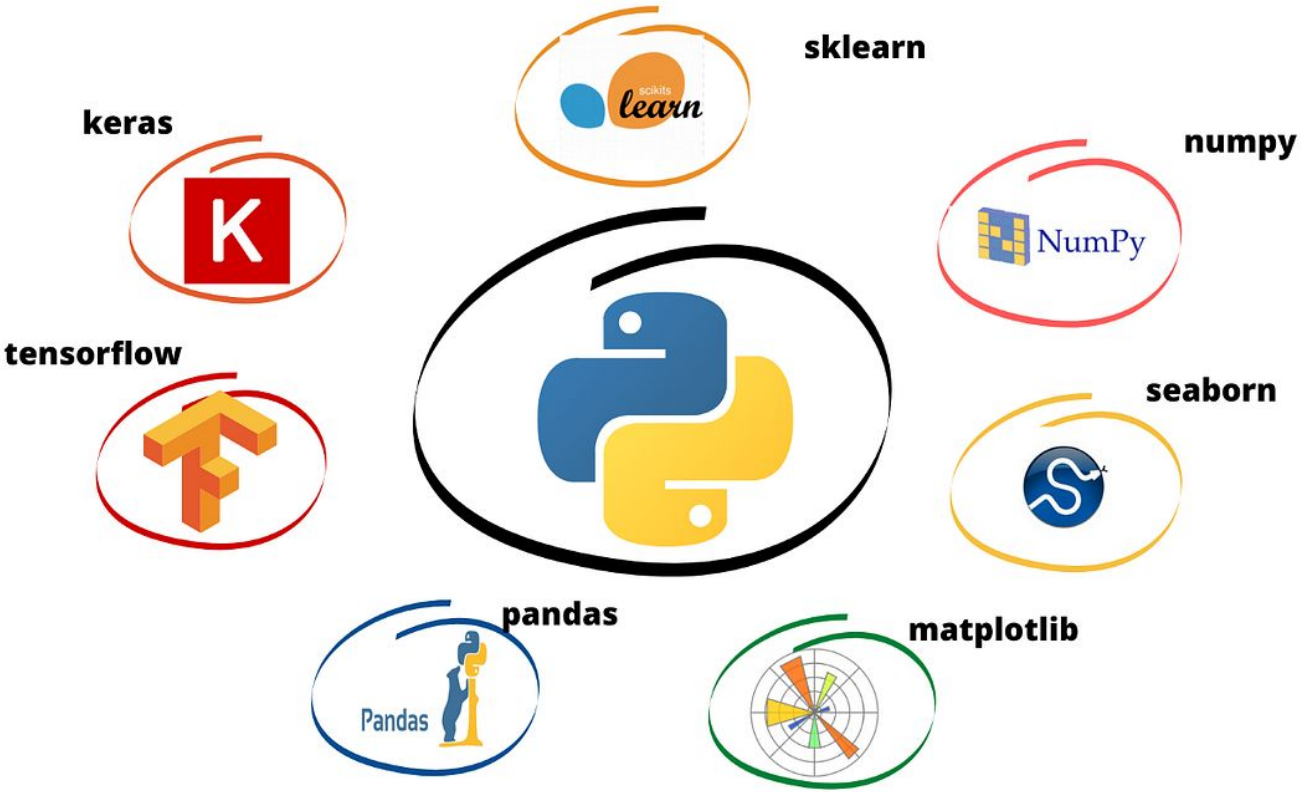
Módulos: Un módulo es un archivo de Python que contiene definiciones y funciones reutilizables. Se importa con **import**. Ejemplo: el módulo **math** ofrece funciones matemáticas.

Paquetes: Un paquete es una colección de módulos organizados en un directorio con un archivo especial __init__.py. Permite estructurar proyectos grandes. Ejemplo: numpy es un paquete con múltiples módulos para cálculo numérico.

Librerías: Una librería es una colección de módulos o paquetes diseñados para resolver problemas específicos. Ejemplo: pandas para análisis de datos, matplotlib para gráficos.

Frameworks: Un framework es una infraestructura más grande que proporciona un conjunto completo de herramientas y reglas para desarrollar aplicaciones completas. Ejemplo: Django para desarrollo web o TensorFlow para machine learning.

Ecosistema Data Science Python



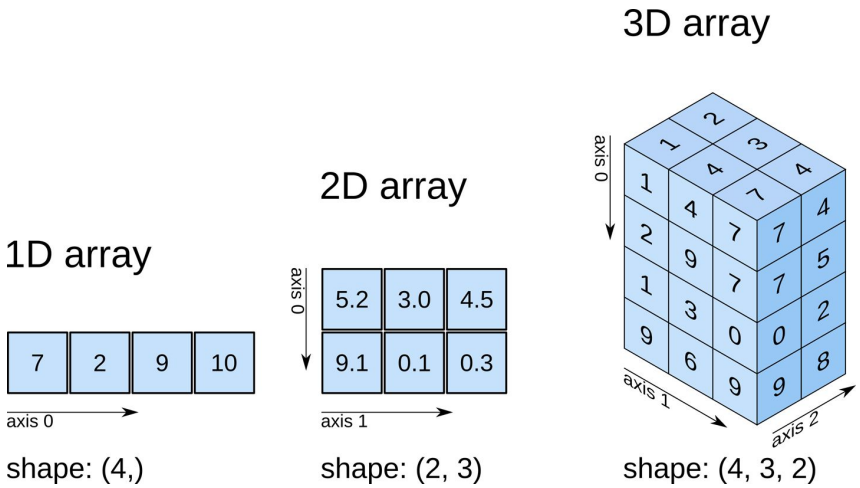
Principales librerías Data Science Python

Numpy

NumPy (Numerical Python) es una biblioteca fundamental en Python para el cálculo numérico y científico.

Proporciona soporte para trabajar con arreglos multidimensionales “**arrays**”, lo que permite manejar datos de forma compacta, eficiente y organizada.

Los arrays de NumPy son más rápidos y consumen menos memoria que las estructuras nativas como listas.



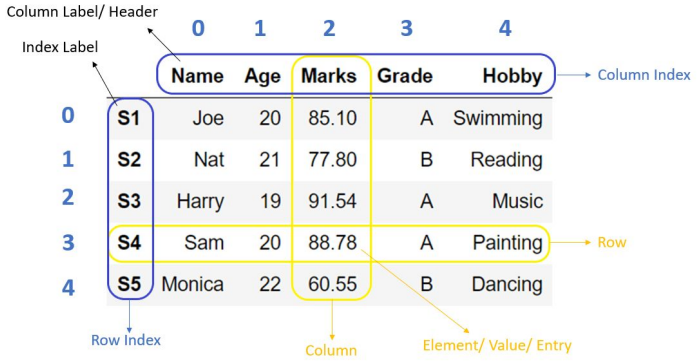
Principales librerías Data Science Python

Pandas

Pandas es una biblioteca de Python diseñada para el análisis y manipulación de datos.

Su estructura principal, el **DataFrame**, permite trabajar con datos tabulares de manera organizada y eficiente, similar a una hoja de cálculo o tabla SQL.

El manejo de datos como DataFrames facilita operaciones complejas, al proporcionar herramientas intuitivas para filtrar, agregar, transformar y analizar información.



Pandas

Principales librerías Data Science Python

Librerías de Inteligencia Artificial (Machine and Deep Learning)

Creado para proporcionar una herramienta accesible y eficiente para aplicar algoritmos de Machine Learning (ML). Su objetivo principal era integrar algoritmos de aprendizaje automático en el ecosistema de **Python científico**.

Desarrollado por **Google** ampliamente utilizado dentro de Google para sus productos, como Google Photos, Google Translate, y YouTube

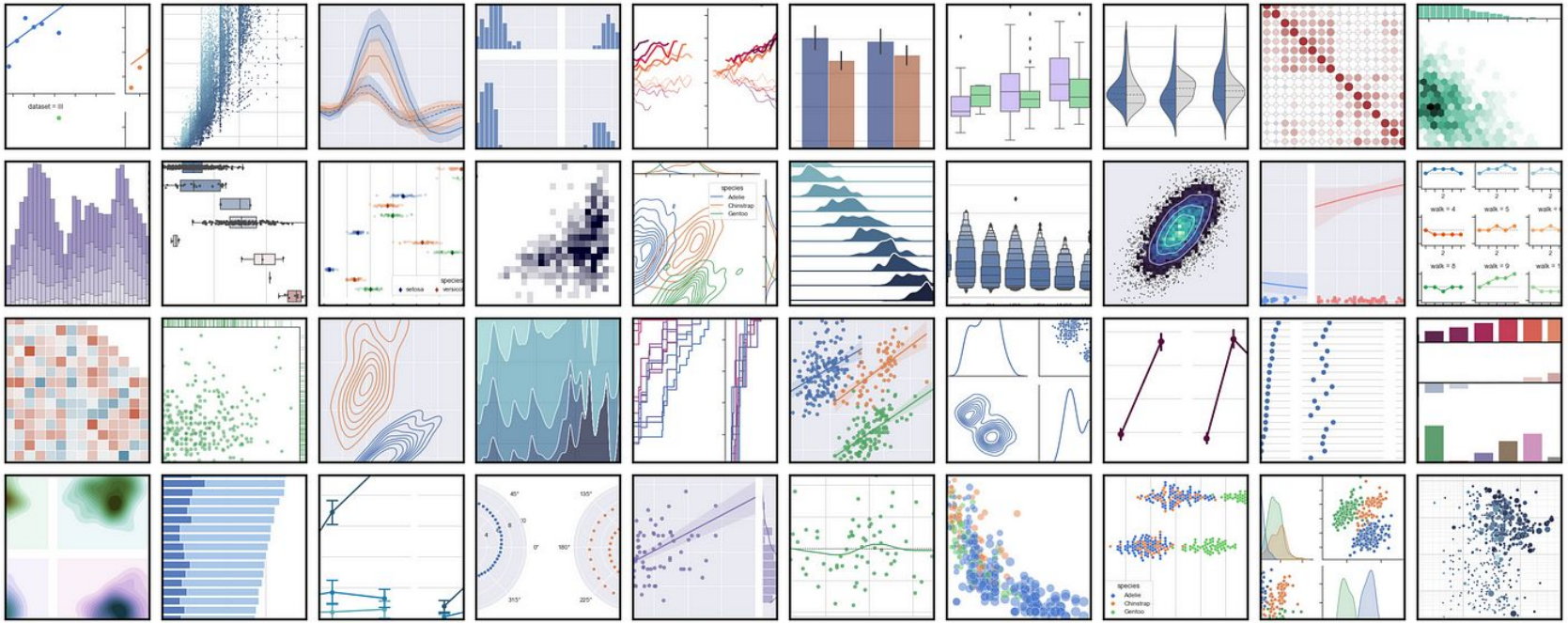
Desarrollado por **Meta** utilizado para algoritmos avanzados en Facebook, Instagram, y WhatsApp, especialmente en sistemas de recomendación y NLP.

Principales librerías de visualización Python

matplotlib

seaborn

plotly



Entorno de Trabajo Python

Google Colaboratory

Google Colab es un entorno de desarrollo interactivo basado en la nube que permite crear y ejecutar cuadernos de **Jupyter**, facilitando el desarrollo y la colaboración en proyectos de programación, análisis de datos y machine learning. Ofrece acceso a recursos computacionales avanzados, como **GPU** (Unidades de Procesamiento Gráfico) y **TPU** (Unidades de Procesamiento Tensorial), directamente desde un navegador web, sin necesidad de configuración local.

Es un entorno sumamente amigable para el desarrollo que además cuenta con Python y R preinstalado por defecto, junto con una gran cantidad de librerías dedicadas a ciencia de datos.

Jupyter Notebooks & Google Colaboratory

Jupyter Notebooks

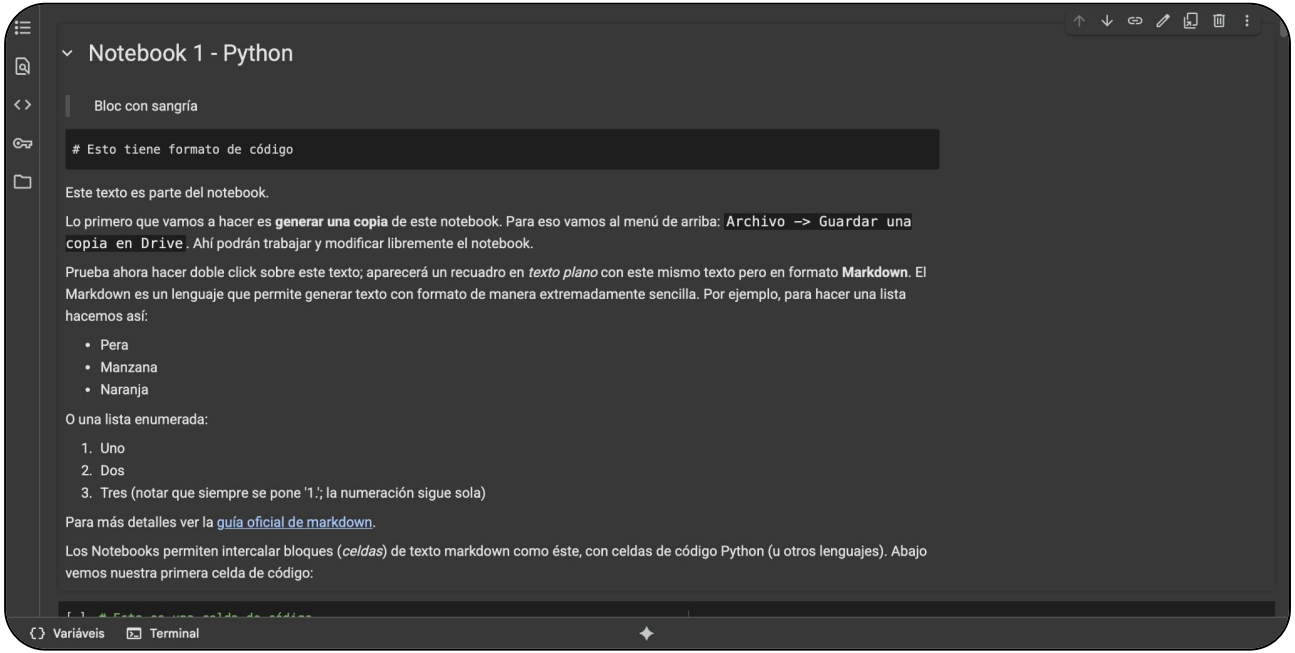
- Entorno interactivo
- Interfaz web (en navegador)
- Permite intercalar código y documentación
- Totalmente gratuito y libre
- Paquetes de Python
- Pueden instalarse en cualquier computadora

Google Colab

- Servicio de Notebooks de Google
- Versión básica gratis
  - En muchos casos alcanza y sobra
  - Suficiente para nuestros objetivos
- Evita tener que instalar y configurar entorno en nuestra máquina!
- Corre en cualquier parte
- Se pueden bajar Notebooks generados

Ejercicios Prácticos

Notebook 1:
Introducción a Python y Colab



Materiales Adicionales

Primeros pasos, consejos y buenas prácticas en Google Colab

Colab

Frequently Asked Questions

Guía para iniciantes

Atajos útiles de Google Colab

Guía rápida

Tutorial de Python

Visualizaciones de datos en Colab de forma sencilla

Tutorial avanzado de Python