Introducción a Python para ML

Día 1: Fundamentos de Python

EAE Business School Barcelona

2 de febrero de 2026

Sobre el Instructor

Alberto Cámara

- PhD en Matemáticas, Freelancer
- Experiencia liderando equipos de datos
- Organizador en **PyBCN**
- **Heavy user** de Python
- Frecuentemente doy clases de **Data Analysis, Machine Learning y más**

Temas organizativos

Horario del Curso

Hora	Actividad	Duración
9:00	**Primera Parte**	2h
11:00	☕ **Descanso**	30 min
11:30	**Segunda Parte**	2h
13:30	**Evaluación / Preguntas**	30 min

8 días (2-11 febrero, saltando el 12)

Algunas reglas

- Las clases empiezan a las 09:00, se ruega puntualidad
- Curso presencial
- Haced preguntas a menudo
- Tendremos un enfoque muy práctico.
  - Iremos del ejemplo a la teoría
- Equivocarse a menudo es parte del juego
- La evaluación se hará a partir de los ejercicios del curso y de los cuestionarios finales
- Todos los materiales del curso serán compartidos: diapositivas y notebooks

Conozcámonos 🙋

Antes de comenzar, me gustaría saber más sobre **vosotros**:

1. **¿Experiencia previa con programación?**
2. **¿Herramientas de análisis / programación que usáis?**
3. **¿Qué esperáis aprender de este curso?**

Objetivo del Curso

- **Pregunta**: ¿Cuál es la misión de un analista de datos?
- **Respuesta**: Proporcionar evidencia respaldada por datos para apoyar la **toma de decisiones**
- **Nuestro objetivo**: Ayudar a esta misión a partir del uso básico de Python y sus librerías

Qué es Python?

- Lenguaje de programación creado por Guido van Rossum en 1991.
- Lenguaje **interpretado** y de **tipado dinámico**
- De alto nivel: diseñado para la **legibilidad** y **simplicidad**

¿Por qué Python?

Python es:

- **Fácil de leer y aprender** - sintaxis clara
- **Un ecosistema rico** - pandas, numpy, scikit-learn, plotly
- **Valorado por el mercado** - buena skill laboral
- **Una comunidad activa** - abundantes recursos y ayuda
- **Versátil** - permite desarrollar software de calidad y desplegarlo en producción
- Es el lenguaje en el que están escritos los frameworks más usados para **ML** e **IA**.

Herramientas que Usaremos

Google Colab - Notebooks en la nube

- No requiere instalación
- Ejecuta código Python en el navegador
- Gratis y potente
- Ideal para aprender y compartir

👉 **Por favor, abrid ahora: colab.research.google.com**

Plan del Día

Primera Parte (9:00-11:00)

- Introducción

1. Variables y tipos de datos
2. Operadores básicos
3. Primer ejemplo completo

Pausa (11:00-11:30)

Segunda Parte (11:30-13:30)

1. Strings y formateo (f-strings)
2. Control de flujo (if/else, bucles)
3. Funciones

Evaluación (13:30-14:00)

- Quiz corto de conceptos básicos

Primeros Pasos con Python

Imaginad que trabajáis en una empresa y necesitáis analizar ventas.

```python
# Ventas de la semana
ventas = [1200, 1500, 1800, 1350, 2000, 2200, 1900]
dias = ['Lun', 'Mar', 'Mié', 'Jue', 'Vie', 'Sáb', 'Dom']

# Cálculos automáticos
promedio = sum(ventas) / len(ventas)
maximo = max(ventas)
dia_maximo = dias[ventas.index(maximo)]

print(f"Promedio semanal: €{promedio:.0f}")
print(f"Mejor día: {dia_maximo} con €{maximo}")
```

Resultado:

```
Promedio semanal: €1707
Mejor día: Sáb con €2200
```

Variables: Contenedores de Información

Las **Variables** guardan información:

```
# Crear variables
precio = 150
producto = "Laptop"
en_stock = True
descuento = 0.15
```

Tipos de Datos Básicos

Tipo	Ejemplo	Uso
`int`	`42`, `1000`	Números enteros
`float`	`3.14`, `19.99`	Números decimales
`str`	`"Hola"`, `'Python'`	Texto
`bool`	`True`, `False`	Verdadero/Falso

Ejemplo

```
edad = 28                    # int
precio = 99.99               # float
nombre = "Ana"               # str (string)
es_activo = True             # bool (boolean)

# Ver el tipo
print(type(edad))            # <class 'int'>
```

Operadores

```
# Operaciones básicas
suma = 10 + 5            # 15
resta = 10 - 5           # 5
multiplicacion = 10 * 5 # 50
division = 10 / 5        # 2.0 (siempre devuelve float)

# Operaciones adicionales
division_entera = 10 // 3  # 3 (sin decimales)
resto = 10 % 3             # 1 (módulo - el resto)
potencia = 2 ** 3          # 8 (2 elevado a 3)
```

Orden de operaciones: igual que en matemáticas (PEMDAS)

```
resultado = 2 + 3 * 4       # 14 (no 20)
resultado = (2 + 3) * 4     # 20 (con paréntesis)
```

Operadores de Comparación

Comparar valores devuelve `True` o `False`:

```
# Igualdad y desigualdad
5 == 5      # True  (igual a)
5 != 3      # True  (no igual a)

# Mayor y menor
10 > 5      # True
10 < 5      # False
10 >= 10    # True (mayor o igual)
5 <= 10     # True (menor o igual)
```

Ahora al notebook

Ejercicio Variables y Operadores (5 minutos)

Trabajando con Texto

Strings (cadenas de texto) se definen con comillas:

```
nombre = "Ana"
apellido = 'García'           # Comillas simples o dobles
mensaje = "Hola, ¿cómo estás?"

# Concatenación (unir strings)
nombre_completo = nombre + " " + apellido  # "Ana García"

# Repetición
separador = "-" * 20          # "--------------------"
```

Métodos útiles de strings:

```
# Longitud
len(nombre)                    # 3

nombre.upper()                 # "ANA"
nombre.lower()                 # "ana"
"  espacios  ".strip()         # "espacios" (quita espacios)
```

Formateo de texto

Podemos concatenar strings usando `+`:

```
nombre = "Carlos"
edad = 35
sueldo = 45000.50

mensaje = "Me llamo " + nombre + " y tengo " + str(edad) + " años"
```

También podemos usar una f-string e interpolarla:

```
mensaje = f"Me llamo {nombre} y tengo {edad} años"

# Con formato
print(f"Sueldo: €{sueldo:,.2f}")  # "Sueldo: €45,000.50"
print(f"Porcentaje: {0.156:.1%}") # "Porcentaje: 15.6%"
```

En general, **f-strings** son más claras y menos propensas a errores

🔬 Ahora al notebook

Ejercicio Strings y F-Strings (7 minutos)

Descanso de 30 minutos

Nos vemos a las 11:30 para la segunda parte.

Control de Flujo

Operadores Lógicos: combinan condiciones **bool** con **and**, **or**, **not**:

```
edad = 25
tiene_licencia = True

# AND: ambas deben ser True
puede_conducir = edad >= 18 and tiene_licencia  # True

# OR: al menos una debe ser True
es_fin_de_semana = dia == "sábado" or dia == "domingo"

# NOT: invierte el valor
no_es_lunes = not (dia == "lunes")
```

if/else

Sintaxis básica:

```
if condicion:
    # Código si condición es True
    print("Se cumple")
else:
    # Código si condición es False
    print("No se cumple")
```

Ejemplo:

```
precio = 150
descuento_aplicado = 0

if precio > 100:
    descuento_aplicado = precio * 0.10
    precio_final = precio - descuento_aplicado
    print(f"Descuento de €{descuento_aplicado:.2f}")
else:
    precio_final = precio
    print("Sin descuento")

print(f"Precio final: €{precio_final:.2f}")
```

Cuando hay más de dos opciones:

```
nota = 75

if nota >= 85:
    calificacion = "Excelente"
elif nota >= 70:
    calificacion = "Notable"
elif nota >= 50:
    calificacion = "Aprobado"
else:
    calificacion = "Suspenso"

print(f"Tu calificación: {calificacion}")
```

Ahora al notebook Ejercicio If/Elif/Else

Bucle for: iterar sobre secuencias

```
# Iterar sobre una lista de nombres
equipos = ["Marketing", "Ventas", "IT", "Finanzas"]

for equipo in equipos:
    print(f"Departamento: {equipo}")
```

Output:

```
Departamento: Marketing
Departamento: Ventas
Departamento: IT
Departamento: Finanzas
```

Iterar sobre números con range():

```python
# range(inicio, fin) - fin no se incluye
for i in range(1, 6):
    print(f"Mes {i}")
# Output: Mes 1, Mes 2, Mes 3, Mes 4, Mes 5
```

Bucle while

while repite mientras la condición sea True:

```
# Contador simple
contador = 1
while contador <= 5:
    print(f"Vuelta número {contador}")
    contador += 1    # Equivale a: contador = contador + 1

print("Bucle terminado")
```

⚠️ Cuidado con bucles infinitos:

```python
# MAL: bucle infinito
while True:
    print("Esto nunca termina")  # ¡No ejecutes esto!

# BIEN: con condición de salida
respuesta = ""
while respuesta != "si":
    respuesta = input("¿Continuar? (si/no): ")
```

🔬 Ahora al notebook

Ejercicio Bucles (10 minutos)

Funciones

Revisitemos el ejemplo. Nos gustaría poder calcular el precio final sin copiar/pegar.

```
precio = 150

if precio > 100:
    descuento_aplicado = precio * 0.10
    precio_final = precio - descuento_aplicado
else:
    precio_final = precio
```

Las **funciones** son la solución a este problema:

```
def calcular_precio_final(precio):
    if precio > 100:
        descuento_aplicado = precio * 0.10
        precio_final = precio - descuento_aplicado
    else:
        precio_final = precio
    return precio_final

calcular_precio_final(150)  # Aplica descuento
calcular_precio_final(90)   # No aplica descuento
```

¿Qué son las Funciones?

Funciones son como "recetas" reutilizables:

- **Encapsulan** código que hace una tarea específica
- **Reutilizables** - escribir una vez, usar muchas veces (*DRY*)
- **Organizan** el código en piezas (*unidades*) manejables
- **Facilitan** el testing y debugging

Definir una Función

Sintaxis básica:

```
def nombre_funcion(parametros):
    """Documentación opcional de qué hace la función"""
    # Código de la función
    return resultado
```

Funciones con Múltiples Parámetros

Las funciones pueden aceptar varios parámetros - incluso con valores por defecto:

Sintaxis:

```
def calcular_precio_final(precio, descuento_porcentaje, iva=0.21):
    """Calcula precio final aplicando descuento e IVA"""
    descuento = precio * (descuento_porcentaje / 100)
    precio_con_descuento = precio - descuento
    precio_final = precio_con_descuento * (1 + iva)
    return precio_final
```

🔬 Ahora al notebook

Ejercicio Funciones (10 minutos)

Usando Funciones con Parámetros

Llamadas con diferentes argumentos:

```
# Usar IVA por defecto (21%)
precio = calcular_precio_final(100, 10)
print(f"Precio: €{precio:.2f}")  # €108.90

# Especificar IVA diferente (10%)
precio2 = calcular_precio_final(100, 10, 0.10)
print(f"Precio: €{precio2:.2f}")  # €99.00
```

No todas las funciones necesitan `return`.

```
def imprimir_reporte(ventas, mes):
    print(f"REPORTE DE VENTAS - {mes.upper()}")
    print("=" * 40)
    print(f"Total de ventas: €{sum(ventas):,.2f}")
    print(f"Venta promedio: €{sum(ventas)/len(ventas):,.2f}")
    print(f"Venta máxima: €{max(ventas):,.2f}")
    print("=" * 40)

# Usar la función
ventas_enero = [1200, 1500, 1800, 1350, 2000]
imprimir_reporte(ventas_enero, "Enero")
```

Buenas Prácticas con Funciones

✅ DO:

- Nombres descriptivos: `calcular_promedio()` no `cp()`
- Una función, una tarea
- Mantener funciones cortas (< 20 líneas idealmente)
- Usar parámetros por defecto cuando tenga sentido

❌ DON'T:

- Funciones gigantes que hacen mil cosas
- Nombres vagos: `procesar()`, `hacer_algo()`
- Modificar variables globales dentro de funciones
- Más de 5 parámetros (difícil de recordar)

💡 **Si copias y pegas código → es momento de hacer una función**

🔬 Ahora al notebook

Ejercicio Integrador (15 minutos)

Resumen del Día

Hoy aprendimos:

1. ✅ Variables y tipos de datos (int, float, str, bool)
2. ✅ Operadores (aritméticos, comparación, lógicos)
3. ✅ Strings y f-strings para formateo
4. ✅ Control de flujo (if/elif/else)
5. ✅ Bucles (for, while)
6. ✅ Funciones (definir y usar)

Mañana

- Estructuras de datos (listas, diccionarios)
- Introducción a pandas
- Trabajar con datos reales

Evaluación (13:30-14:00)

Quiz de 5 preguntas de opción múltiple

Temas cubiertos:

- Tipos de datos en Python
- Operadores básicos
- Sintaxis de control de flujo
- Definición de funciones

👉 **Link al quiz:** [se compartirá en el chat]

Gracias