DESCUBRIENDO EL PODER DE LA PROGRAMACION

CURSO INICIAL DE PYTHON

Jimmy Chung
Alexander Solis

CONTENIDO DEL CURSO

1. Introducción, Instalación y Conceptos básicos
2. Variables, Expresiones, Funciones y Operadores
3. Condicionales y Ciclos – Patrones de sintaxis válidos
4. Listas, tuplas y diccionarios
5. Errores y Excepciones
6. Objetos, Clases y funciones en Python
7. Guardar y Recuperar datos de archivos
8. Pandas y Matplotlib

PIP y las librerías externas

pip es el sistema de gestión de paquetes estándar para Python. Es una herramienta que facilita la instalación, actualización y gestión de librerías y paquetes de software en Python. Pip se utiliza para descargar paquetes desde el Python Package Index (PyPI).

Antes de utilizar pip, primero debemos verificar que está instalado, para ello utilizamos el comando pip --versión en una terminal del sistema, este comando nos mostrará la versión actual de pip.

- Instalar una librería: pip install nombre_librería.
- Actualizar una librería: pip install --upgrade nombre_librería
- Desinstalar una librería: pip uninstall nombre_libreria

Instalación de Python y **pip**: https://www.youtube.com/watch?v=GLcbD-mCAO8

PIP y las librerías externas

Librerías de Python más importantes en diferentes áreas

Análisis de datos y ciencia de datos:

- Pandas: Librería para el análisis y manipulación de datos estructurados.
- NumPy: Librería para la computación científica y manipulación de matrices y arreglos.
- Matplotlib: Librería para la visualización de datos en gráficos y gráficas.
- SciPy: Librería para la computación científica y resolución de problemas matemáticos.

Aprendizaje automático (Machine Learning):

- Scikit-learn: Librería para el aprendizaje automático y minería de datos.
- TensorFlow: Librería de aprendizaje automático y redes neuronales de código abierto.
- Keras: Librería de alto nivel para la construcción y entrenamiento de modelos de aprendizaje profundo.
- PyTorch: Librería de aprendizaje automático basada en tensores y redes neuronales.

PIP y las librerías externas

Librerías de Python más importantes en diferentes áreas

Procesamiento de imágenes y visión por computadora:

- OpenCV: Librería para el procesamiento de imágenes y visión por computadora.
- Pillow: Librería para el procesamiento de imágenes y manipulación de formatos de imagen.
- scikit-image: Librería para el procesamiento de imágenes y extracción de características.

Desarrollo web:

- Flask: Microframework para el desarrollo de aplicaciones web.
- Django: Framework web completo y de alto nivel.
- Requests: Librería para realizar peticiones HTTP.

Pandas: Analítica de datos

Primer paso instalar pandas: pip install pandas

Aspectos mas importantes a tener en cuenta al comenzar a trabajar con pandas y explorar sus funcionalidades:

DataFrame: El DataFrame es la estructura de datos fundamental en pandas. Es similar a una tabla en una base de datos o una hoja de cálculo de Excel. Un DataFrame consta de filas y columnas, donde cada columna puede contener diferentes tipos de datos. Es importante entender cómo trabajar con los DataFrames, ya que te permitirán manipular y analizar datos de manera eficiente.

Pandas: Analítica de datos

Lectura y escritura de archivos: Pandas ofrece funciones para leer y escribir datos desde y hacia diversos formatos, como archivos CSV, Excel, SQL, etc. Es crucial comprender cómo utilizar estas funciones para importar datos a un DataFrame desde diferentes fuentes y exportar los resultados de tus análisis.

Exploración de datos: Antes de realizar cualquier análisis o manipulación de datos, es esencial explorar y comprender la estructura y los tipos de datos en tu DataFrame. Pandas proporciona funciones útiles para obtener información sobre el DataFrame, como info(), head(), tail(), describe(), etc. Estas funciones te permitirán obtener una vista previa de tus datos y comprender su distribución y características.

Pandas: Analítica de datos

Manipulación de datos: Una vez que hayas importado tus datos en un DataFrame, tendrás varias herramientas para manipularlos. Pandas te permite seleccionar y filtrar filas y columnas específicas, realizar operaciones aritméticas y lógicas en los datos, eliminar duplicados, reemplazar valores, etc. Aprender a utilizar estas técnicas de manipulación de datos te permitirá transformar y limpiar tus conjuntos de datos de manera efectiva.

Operaciones estadísticas y agregaciones: Pandas ofrece numerosas funciones para realizar cálculos estadísticos y agregaciones en tus datos. Puedes calcular la media, mediana, desviación estándar, y realizar otras operaciones como agrupar datos por categorías y aplicar funciones de agregación. Estas operaciones te ayudarán a obtener información valiosa y resumida sobre tus datos.

Pandas: Analítica de datos

Visualización de datos: Aunque pandas no es una biblioteca de visualización en sí misma, se integra bien con otras bibliotecas populares de visualización de datos, como Matplotlib y Seaborn. Esto te permite crear gráficos y visualizaciones atractivas directamente desde tus DataFrames. La capacidad de visualizar datos es esencial para comprender patrones, tendencias y relaciones en tus conjuntos de datos.

Práctica y experimentación: Al igual que con cualquier nuevo concepto, la práctica es clave para comprender y dominar pandas. A medida que avances en tu curso de pandas, te recomendaría realizar ejercicios prácticos y experimentar con diferentes operaciones y técnicas de manipulación de datos. Esto te ayudará a consolidar tus conocimientos y familiarizarte con las capacidades de pandas.

Pandas: DataFrames

Como mencionamos al principio, DataFrame es la estructura de datos fundamental en pandas, y es como tener una hoja de Excel.

Podemos crear un DataFrame a partir de un Diccionario o una Lista, sin necesidad de cargar los datos desde un archivo de Excel, CSV, o una Base de

```python
import pandas as pd

datos = [["Carlos", "Abogado", 40],
         ["Pedro", "Contador", 35],
         ["Maria", "Ingeniero", 30],
         ["Rosa", "Arquitecto", 25]]

personas = pd.DataFrame(datos, columns=["Nombre", "Profesion", "Edad"])

print(personas)
```

```
   Nombre   Profesion  Edad
0  Carlos     Abogado    40
1   Pedro    Contador    35
2   Maria   Ingeniero    30
3    Rosa  Arquitecto    25
```

```python
import pandas as pd

datos = {"Nombre": ["Carlos", "Pedro", "Maria", "Rosa"],
         "Profesion": ["Abogado", "Contador", "Ingeniero", "Arquitecto"],
         "Edad": [40, 35, 30, 25]}

personas = pd.DataFrame(datos)

print(personas)
```

```
   Nombre   Profesion  Edad
0  Carlos     Abogado    40
1   Pedro    Contador    35
2   Maria   Ingeniero    30
3    Rosa  Arquitecto    25
```

Pandas: Series

Una Serie en Pandas es una estructura de datos unidimensional que puede contener cualquier tipo de datos, como números, cadenas de texto o fechas. Se puede pensar en una Serie como una columna en una hoja.

```python
import pandas as pd

# Crear una Serie
serie = pd.Series([10, 20, 30, 40, 50])

# Crear una Serie
serie_nombres = pd.Series(["Carlos", "Pedro", "Maria", "Rosa"])

print(serie)

print(serie_nombres)
```

```
0    10
1    20
2    30
3    40
4    50
dtype: int64
0    Carlos
1     Pedro
2     Maria
3      Rosa
dtype: object
```

Pandas: Archivos Excel

```python
import pandas as pd

# Cargar el archivo de Excel
df = pd.read_excel('ventas.xlsx')

# Obtener ventas totales por región y mes
ventas_totales = df.groupby(['Región', 'Mes'])['Ventas'].sum()

# Imprimir el resumen de ventas totales por región y mes
print("Resumen de ventas totales por región y mes:")
print(ventas_totales)

# Obtener ventas totales y promedio por producto
ventas_productos = df.groupby('Producto')['Ventas'].agg(['sum', 'mean'])

# Imprimir el resumen de ventas totales y promedio por producto
print("\nResumen de ventas totales y promedio por producto:")
print(ventas_productos)
```

Pandas: Archivos CSV

```python
import pandas as pd

# Cargar el archivo CSV
df = pd.read_csv('ventas.csv')

# Obtener ventas totales por región y mes
ventas_totales = df.groupby(['Región', 'Mes'])['Ventas'].sum()

# Imprimir el resumen de ventas totales por región y mes
print("Resumen de ventas totales por región y mes:")
print(ventas_totales)

# Obtener ventas totales y promedio por producto
ventas_productos = df.groupby('Producto')['Ventas'].agg(['sum', 'mean'])

# Imprimir el resumen de ventas totales y promedio por producto
print("\nResumen de ventas totales y promedio por producto:")
print(ventas_productos)
```

Graficar datos con Matplotlib

Matplotlib es una biblioteca poderosa y flexible para la creación de gráficos en Python. Proporciona una amplia gama de opciones de personalización y soporte para diversos tipos de gráficos, lo que la convierte en una herramienta esencial para el análisis y la visualización de datos.

Matplotlib se integra de manera transparente con otras bibliotecas populares de análisis de datos en Python, como NumPy y Pandas. Esto permite combinar las capacidades de estas bibliotecas para realizar análisis y visualizaciones más sofisticadas.

Ejercicio 01

Ventas

Imagina que eres el analista de datos de una empresa de comercio electrónico y se te ha asignado la tarea de analizar los datos de ventas de productos en diferentes regiones. Tienes un archivo de datos llamado "ventas.xlsx" que contiene la información de las ventas con las siguientes columnas: "Producto", "Región", "Mes" y "Ventas".

- Calcular las ventas totales por región y mes.
- Determinar los productos más vendidos en cada región.
- Identificar los meses con mayores ventas.
- Comparar las ventas entre diferentes regiones utilizando gráficos.

Referencias bibliográficas

Documentación oficial Python.

https://docs.python.org/es/3/tutorial/index.html

https://docs.python.org/es/3/library/exceptions.html#exception-context

https://docs.python.org/es/3/library/exceptions.html#concrete-exceptions

https://docs.python.org/es/3/library/functions.html#open

https://docs.python.org/es/3/glossary.html#term-file-object

Referencias bibliográficas

Librerías Python.

Pandas: https://pandas.pydata.org/

Matplotlib: https://pandas.pydata.org/