ESCUELA DE
INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

ECYS | DTT
DESARROLLO DE TRANSFERENCIA TECNOLÓGICA

Día, Fecha	Lunes, 26/08/2024
Hora de inicio:	17:20

Seminario de Sistemas 2 [A]

Jose Fernando Alvarez Morales

Laboratorio

Seminario de Sistemas 2 Sección “A”

Agenda

- Anuncios
- Clase 6
  - Introducción de Python para análisis de datos.
  - Manipulación de datos con Pandas
- Ejemplo Practico

Clase 6

Python

- Python es un lenguaje de programación de alto nivel, interpretado y multiplataforma. Fue creado a principios de la década de 1990 por Guido van Rossum en el Centro para las Matemáticas y la Informática (CWI) en los Países Bajos.

Python

Algunas de las principales características y ventajas de Python son:

1. Sintaxis limpia e intuitiva: Python tiene una sintaxis muy legible y concisa que enfatiza la escritura de código claro y fácil de mantener.
2. Multiplataforma: Python se puede ejecutar en múltiples sistemas operativos, incluyendo Windows, macOS y las diferentes distribuciones de Linux.
3. Interpretado: Python es un lenguaje interpretado, lo que significa que no necesita compilarse antes de ejecutarse. Esto lo hace ideal para el desarrollo rápido de aplicaciones.

Python

4. Tipado dinámico: Python no requiere declarar el tipo de datos de las variables explícitamente, lo que brinda mayor flexibilidad.
5. Biblioteca estándar grande: Python viene con una extensa biblioteca estándar que cubre áreas como desarrollo web, procesamiento de archivos, pruebas, etc.
6. Open Source: Python es de código abierto y tiene una gran comunidad de desarrollo activa que contribuye con nuevos módulos, bibliotecas y mejoras.

Python

Python se utiliza ampliamente en diversas áreas, como desarrollo web (Django, Flask), análisis de datos (NumPy, Pandas), inteligencia artificial (TensorFlow, Scikit-learn), scripting de sistema, automatización, aplicaciones de escritorio y mucho más. Es un lenguaje muy versátil y popular tanto para principiantes como para desarrolladores experimentados.

Python para Análisis de Datos

Python para análisis de datos

Python se ha convertido en uno de los lenguajes de programación más populares y potentes para el análisis de datos debido a su amplio conjunto de bibliotecas y herramientas especializadas.

1. Carga y manipulación de datos:
    - Pandas: Esta biblioteca proporciona estructuras de datos y herramientas de análisis de datos altamente eficientes. Permite cargar datos desde diversos formatos (CSV, Excel, bases de datos, etc.), manipular y limpiar los datos, y realizar operaciones como filtrado, agrupación y combinación.
    - NumPy: Ofrece soporte para operaciones numéricas eficientes con arrays multidimensionales, lo que es fundamental para el procesamiento de datos numéricos

Python para análisis de datos

2. Limpieza y preparación de datos:
    - Python proporciona herramientas para la limpieza y transformación de datos, incluyendo el manejo de valores faltantes, la eliminación de duplicados, la normalización y el ajuste de tipos de datos.
    - Bibliotecas como Pandas, NumPy y SciPy facilitan estas tareas.
3. Exploración y visualización de datos:
    - Matplotlib y Seaborn: Estas bibliotecas permiten crear una amplia variedad de visualizaciones de datos, como gráficos de líneas, barras, histogramas, diagramas de dispersión y más.
    - Pandas también tiene capacidades de visualización integradas para un análisis rápido.

Python para análisis de datos

4. Análisis estadístico:
    - SciPy: Proporciona funciones para cálculos estadísticos, como medidas de tendencia central, pruebas estadísticas, distribuciones de probabilidad y más.
    - StatsModels: Una biblioteca para el modelado estadístico, como regresión lineal, análisis de series de tiempo y modelos estadísticos avanzados.
5. Aprendizaje automático y minería de datos:
    - Scikit-learn: Una biblioteca integral para el aprendizaje automático, que incluye algoritmos para clasificación, regresión, agrupamiento, reducción de dimensionalidad y más.
    - TensorFlow, PyTorch, Keras: Bibliotecas populares para el aprendizaje profundo (deep learning) y la construcción de redes neuronales.

Python para análisis de datos

6. Procesamiento de datos a gran escala:
    - Apache Spark con PySpark: Un framework de código abierto para el procesamiento de grandes conjuntos de datos utilizando programación en Python.
    - Dask: Una biblioteca para computación paralela y escalable en Python, especialmente útil para datos que no caben en la memoria.
7. Automatización y scripting:
    - Python se puede utilizar para automatizar tareas repetitivas y flujos de trabajo de análisis de datos a través de scripts y programas personalizados.
    - Bibliotecas como OS, Sys y Subprocess facilitan la interacción con el sistema operativo y la ejecución de comandos.

Python para análisis de datos

Además de las bibliotecas mencionadas, el ecosistema de Python para análisis de datos también incluye herramientas como Jupyter Notebook (para programación interactiva y documentación), entornos de desarrollo integrados (IDEs) como PyCharm y Spyder, y frameworks web como Django y Flask para el despliegue de aplicaciones de análisis de datos.

Pandas

Pandas

¿Qué significa Pandas?
PANEL DATA

Es una libreria de Python especializada en el manejo y análisis de estructura de datos.
Nos ayuda manipular, modela, analizar y preparar los datos.
Deben estar centralizados.

Pandas

Pandas es una biblioteca de código abierto que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento. Fue desarrollada por Wes McKinney y lanzada por primera vez en 2008. Pandas está construida sobre NumPy y es ampliamente utilizada en el ecosistema de ciencia de datos de Python.

Estructuras de datos principales:

- Series: Es un array unidimensional etiquetado, similar a una columna en una hoja de cálculo. Puede contener cualquier tipo de dato.
- DataFrame: Es una estructura de datos tabular bidimensional, similar a una hoja de cálculo, donde los datos se organizan en filas y columnas. Es la estructura de datos más utilizada en Pandas.

Pandas

Pandas es una biblioteca de código abierto que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento. Fue desarrollada por Wes McKinney y lanzada por primera vez en 2008. Pandas está construida sobre NumPy y es ampliamente utilizada en el ecosistema de ciencia de datos de Python.

Estructuras de datos principales:

- Series: Es un array unidimensional etiquetado, similar a una columna en una hoja de cálculo. Puede contener cualquier tipo de dato.
- DataFrame: Es una estructura de datos tabular bidimensional, similar a una hoja de cálculo, donde los datos se organizan en filas y columnas. Es la estructura de datos más utilizada en Pandas.

Pandas

Ventajas de Pandas:

1. Manejo eficiente de datos faltantes: Pandas facilita el manejo de datos faltantes mediante operaciones y métodos dedicados.
2. Integración y limpieza de datos: Pandas permite cargar, combinar, transformar y limpiar datos de diversos orígenes (CSV, Excel, bases de datos, etc.) de manera sencilla.
3. Operaciones de indexación y selección flexibles: Pandas ofrece poderosas capacidades para seleccionar e indexar datos de acuerdo con etiquetas, rangos, condiciones booleanas y más.
4. Operaciones de agrupación y manipulación de datos: Pandas facilita la agrupación, agregación y transformación de datos con operaciones como groupby, merge, pivot y melt.
5. Integración con otras bibliotecas: Pandas se integra perfectamente con otras bibliotecas populares de Python, como NumPy, Matplotlib y Scikit-learn.
6. Rendimiento optimizado: Pandas está optimizado para operaciones vectorizadas y utiliza eficientemente la memoria, lo que lo hace rápido para conjuntos de datos de gran tamaño.

Pandas

Desventajas de Pandas:

1. Curva de aprendizaje inicial: Aunque Pandas es relativamente fácil de aprender, su amplio conjunto de funciones y métodos puede ser intimidante para principiantes.
2. Consumo de memoria: Pandas puede consumir mucha memoria para conjuntos de datos muy grandes, especialmente si hay muchas columnas de cadenas de caracteres.
3. Rendimiento para operaciones complejas: Aunque Pandas es rápido para operaciones básicas, algunas operaciones más complejas pueden ser lentas en comparación con alternativas como bases de datos.
4. Falta de escalabilidad para big data: Pandas no está diseñado para procesar conjuntos de datos masivos que no caben en la memoria. Para estos casos, se requieren herramientas como Apache Spark o Dask.

Pandas

Usos comunes de Pandas:

- Análisis exploratorio de datos (EDA)
- Limpieza y transformación de datos
- Manipulación y análisis de datos estructurados
- Carga y preparación de datos para tareas de aprendizaje automático
- Análisis financiero y cuantitativo
- Procesamiento de datos en investigación científica y académica

Pandas

Usos comunes de Pandas:

- Análisis exploratorio de datos (EDA)
- Limpieza y transformación de datos
- Manipulación y análisis de datos estructurados
- Carga y preparación de datos para tareas de aprendizaje automático
- Análisis financiero y cuantitativo
- Procesamiento de datos en investigación científica y académica

Numpy

Numpy

NumPy es una biblioteca de código abierto que proporciona soporte para crear y manipular arrays y matrices multidimensionales de datos numéricos. Fue creada a mediados de los años 90 por Jim Hugunin y se ha convertido en un componente fundamental de la stack de Python para computación científica.

Características principales:

- Arrays N-dimensionales: NumPy introduce el objeto array, una estructura de datos eficiente para almacenar y operar con colecciones ordenadas de datos homogéneos.
- Operaciones vectorizadas: NumPy permite realizar operaciones matemáticas de manera vectorizada sobre arrays completos, lo que resulta en un gran aumento del rendimiento en comparación con las operaciones tradicionales en Python.
- Funciones matemáticas: Proporciona una gran colección de funciones matemáticas que operan sobre arrays, como álgebra lineal, transformadas de Fourier, estadísticas, etc.

Numpy

Ventajas de NumPy:

1. Eficiencia computacional: NumPy está optimizado en C, lo que lo hace extremadamente rápido en operaciones numéricas en comparación con Python puro.
2. Vectorización: Las operaciones vectorizadas en NumPy permiten expresar cálculos complejos de manera concisa y eficiente.
3. Integración con otras bibliotecas: NumPy es la base de muchas otras bibliotecas populares de Python, como Pandas, SciPy, Matplotlib, scikit-learn, entre otras.
4. Memoria eficiente: Los arrays de NumPy son densos y compactos en memoria, lo que los hace ideales para trabajar con grandes conjuntos de datos numéricos.
5. Multiplataforma: NumPy es multiplataforma y funciona de manera consistente en diferentes sistemas operativos.

Numpy

Desventajas de NumPy:

1. Solo datos homogéneos: Los arrays de NumPy solo pueden contener datos del mismo tipo, lo que puede ser una limitación en ciertos casos.
2. Curva de aprendizaje inicial: La sintaxis y las convenciones de NumPy pueden ser confusas para los principiantes.
3. Falta de estructuras de datos avanzadas: NumPy carece de estructuras de datos más avanzadas como DataFrames, que se encuentran en bibliotecas como Pandas.
4. Consumo de memoria: Los arrays de NumPy pueden consumir mucha memoria si no se manejan correctamente, especialmente al trabajar con matrices grandes.

Numpy

Usos comunes de NumPy:

- Procesamiento de imágenes y señales
- Álgebra lineal y cálculo numérico
- Simulaciones físicas y modelado
- Aprendizaje automático y análisis de datos
- Computación científica y técnica en general

DUDAS?
Ejemplo Práctico