Obtención y Preparación

de Datos

Sesión 3

Leyendo y Escribiendo Archivos CSV y Excel con Pandas

En ciencia de datos, la obtención y manipulación de datos es fundamental.

- Pandas permite trabajar con archivos CSV y Excel de forma eficiente.
- Base: Construida sobre NumPy, optimizada para datos tabulares.

Aprenderemos a:

- Leer archivos CSV y Excel.
- Modificar y limpiar datos.
- Guardar los datos en nuevos archivos.

¿Qué es un Archivo CSV?

CSV (Comma-Separated Values) almacena datos en formato tabular.
Características:

✅ Ligero y compatible con hojas de cálculo y bases de datos.

✅ Filas separadas por nueva línea (\n).

✅ Valores separados por comas ",", punto y coma ";" u otros delimitadores.

✅ No admite estilos ni fórmulas.

Ejemplo de archivo CSV

```
Nombre,Edad,Puntaje
Juan,25,85
María,30,92
Pedro,27,78
```

Leyendo un Archivo CSV con Pandas

La función read_csv() de Pandas permite cargar datos desde un archivo CSV en un DataFrame, facilitando la manipulación

Ejemplo básico de lectura de un CSV:

```python
import pandas as pd

# Leer el archivo CSV
df = pd.read_csv("datos.csv")

# Mostrar las primeras filas
print(df.head())
```

Personalizando la Lectura de un CSV

Parámetros Comunes de read_csv():

- sep=",": Define el delimitador.
- header=0: Define la fila del encabezado.
- index_col=0: Usa una columna como índice.
- na_values=["NA", "NULL"]: Trata ciertos valores como nulos.

Ejemplo con parámetros personalizados:

```
df = pd.read_csv("datos.csv", sep=";", index_col=0, na_values=["?", "NULL"])
```

Escribiendo un Archivo CSV con Pandas

Una vez modificado un DataFrame en Pandas, se puede exportar nuevamente a un archivo CSV usando to_csv().

Ejemplo básico de escritura de un CSV:

```
df.to_csv('datos_guardados.csv', index=False)
```

Escribiendo un Archivo CSV con Pandas

Opciones útiles en to_csv():

- index=False evita guardar el índice.
- sep=";" cambia el delimitador.
- na_rep="Desconocido" reemplaza valores nulos.

Ejemplo con más parámetros:

```
df.to_csv("nuevo_archivo.csv", index=False, sep=";", na_rep="Desconocido")
```

Archivos Excel en Pandas

Excel es uno de los formatos más utilizados en análisis de datos. Pandas facilita la lectura y escritura de archivos Excel sin necesidad de abrirlos manualmente.

¿Qué es XLRD?

- XLRD es una librería de Python que permite leer archivos de Excel con extensión .xls (formato antiguo de Excel).
- Desde Pandas 1.2 en adelante, se recomienda usar openpyxl en lugar de XLRD para archivos .xlsx.

Instalación de openpyxl (para leer archivos .xlsx)

```
pip install openpyxl
```

Archivos Excel en Pandas

Leyendo un archivo Excel

🔹 La función read_excel() permite cargar datos desde una hoja de cálculo de Excel en un DataFrame.

Ejemplo básico de lectura de Excel:

```
df = pd.read_excel("datos.xlsx")
print(df.head())
```

Archivos Excel en Pandas

Leyendo un archivo Excel

Parámetros Comunes de read_excel()

- sheet_name="Hoja1": Especifica la hoja a leer (por defecto es la primera).
- index_col=0: Define qué columna se usará como índice del DataFrame.
- usecols=["Nombre", "Edad"]: Permite seleccionar solo ciertas columnas.

Ejemplo con parámetros personalizados:

```
df = pd.read_excel("datos.xlsx", sheet_name="Ventas", usecols=["Cliente", "Total"], index_col="Cliente")
```

Archivos Excel en Pandas

Escribiendo un archivo Excel:

- Para exportar datos de Pandas a Excel, se usa to_excel().

Ejemplo básico de escritura en Excel:

```
df.to_excel("nuevo_archivo.xlsx", index=False)
```

Archivos Excel en Pandas

Escribiendo un archivo Excel:

Opciones útiles en to_excel():

- sheet_name="Resultados": Define el nombre de la hoja donde se guardarán los datos.
- na_rep="Sin datos": Define cómo se guardarán los valores nulos.

Ejemplo con parámetros personalizados

```
df.to_excel("resultados.xlsx", sheet_name="Datos Analizados", index=False, na_rep="Sin datos")
```

Extrayendo Tablas de la Web

Pandas también permite extraer datos de tablas HTML en páginas web usando read_html(), lo que facilita la recopilación de información desde la web.

¿Cómo funciona read_html()?

La función busca automáticamente todas las tablas de una página web y las convierte en una lista de DataFrames.

Ejemplo de extracción de datos web:

```
url = "https://es.wikipedia.org/wiki/Anexo:Países_y_territorios_dependientes_por_población"
tablas = pd.read_html(url)
df_poblacion = tablas [0] # Selecciona la primera tabla

# Mostrar la tabla encontrada
print(df_poblacion.head())
```

Preguntas

Sección de preguntas

Obtención y Preparación

de Datos

Continúe con las actividades