Introducción a Pandas

J.M. Marín Ramírez

Métodos computacionales

¿Qué es Pandas?

- **Pandas** es una **libreria** de Python utilizada para el análisis y manipulación de datos.
- Proporciona estructuras de datos rápidas y flexibles como **Series** y **DataFrames**.
- Facilita el manejo de grandes volúmenes de datos de manera eficiente.

Importancia de Pandas

- **Fácil manipulación de datos**: Permite limpiar, filtrar, transformar y analizar datos de manera sencilla.
- **Integración con otras bibliotecas**: Se integra bien con bibliotecas populares como **NumPy**, **Matplotlib** y **scikit-learn**.
- **Soporte para datos estructurados**: Maneja datos provenientes de archivos CSV, Excel, SQL y más.
- **Alta eficiencia**: Optimizaciones que permiten trabajar con grandes conjuntos de datos en memoria.

Importar Pandas

Antes de usar Pandas, debemos importarlo.

```
import pandas as pd
```

Estructuras de datos

- **Series**: Estructura unidimensional similar a un array de NumPy.
- **DataFrame**: Estructura bidimensional, similar a una tabla de bases de datos o una hoja de cálculo.

Crear una Serie

Una Serie es como una columna de datos.

```
import pandas as pd

# Crear una Serie
serie = pd.Series([1, 2, 3, 4, 5])
print(serie)
```

Salida:

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

Crear un DataFrame

Un DataFrame es como una tabla de datos. Puedes crear un DataFrame a partir de un diccionario:

```
data = {
    'Nombre': ['Ana', 'Luis', 'Juan'],
    'Edad': [25, 30, 22],
    'Ciudad': ['Madrid', 'Barcelona', 'Sevilla']
}

df = pd.DataFrame(data)
print(df)
```

Salida:

```
  Nombre  Edad      Ciudad
0    Ana    25      Madrid
1   Luis    30   Barcelona
2   Juan    22     Sevilla
```

Leer archivos CSV y TXT

Leer un archivo CSV

Para leer un archivo CSV utilizando Pandas:

```
# Leer un archivo CSV
df_csv = pd.read_csv('datos.csv')
print(df_csv.head())  # Muestra las primeras 5 filas
```

Leer un archivo TXT

Para leer un archivo de texto con separadores por tabulación:

```
# Leer un archivo TXT (separado por tabulaciones)
df_txt = pd.read_csv('datos.txt', sep='\t')
print(df_txt.head())  # Muestra las primeras 5 filas
```

Funciones básicas

Algunas funciones comunes para trabajar con DataFrames:

```
# Obtener información del DataFrame
df.info()

# Describir estadísticas básicas
df.describe()

# Filtrar filas por una condición
df_filtrado = df[df['Edad'] > 25]
```

Limpieza de datos con Pandas

Pandas tiene varias funciones útiles para limpiar datos faltantes o duplicados:

```
# Eliminar filas con valores NaN
df_limpio = df.dropna()

# Rellenar valores faltantes con un valor específico
df_rellenado = df.fillna(0)

# Eliminar filas duplicadas
df_sin_duplicados = df.drop_duplicates()

# Reemplazar valores en una columna
df['Ciudad'] = df['Ciudad'].replace('Desconocida', 'Sin especificar')
```

Manipulación de columnas con Pandas

Aquí hay algunos ejemplos de cómo puedes manipular columnas en un DataFrame:

```
# Renombrar una columna
df_renombrado = df.rename(columns={'Nombre': 'Nombre completo'})

# Crear una nueva columna basada en otra
df['Edad en meses'] = df['Edad'] * 12

# Eliminar una columna
df_sin_columna = df.drop('Ciudad', axis=1)

# Reordenar las columnas
df_reordenado = df[['Nombre completo', 'Edad en meses', 'Ciudad']]
```

Añadir filas con append()

El método append() añade filas de un DataFrame a otro:

```
# Agregar filas de df2 a df1
df_append = df1.append(df2, ignore_index=True)
```

ignore_index=True reindexa el DataFrame resultante para evitar duplicaciones en los índices.

Unión basada en columnas con `merge()`

La función `merge()` permite combinar DataFrames usando columnas como claves, similar a un join en bases de datos.

```
# Unión de dos DataFrames usando la columna 'ID' como cla
df_merged = pd.merge(df1, df2, on='ID', how='outer')
```

Puedes controlar el tipo de unión con el parámetro `how`:

- `'inner'`: Unión interna (solo valores coincidentes).
- `'outer'`: Unión externa (todos los valores, con NaN donde no coinciden).
- `'left'`: Unión izquierda.
- `'right'`: Unión derecha.

Concatenación y unión de DataFrames

Pandas ofrece varias funciones para combinar DataFrames:

concat()

Une múltiples DataFrames de manera horizontal o vertical.

```
# Concatenar dos DataFrames
df_concatenado = pd.concat([df1, df2], axis=0)  # Vertical (filas)
df_concatenado_horizontal = pd.concat([df1, df2], axis=1)  # Horizontal (columna
```

join()

Realiza una unión entre DataFrames utilizando el índice.

```
# Unir dos DataFrames por el índice
df_unido = df1.join(df2, how='inner')  # Unión interna
```