Introducción a Pandas: El Poder del Análisis de Datos en Python

Pandas es una biblioteca de Python de código abierto. Está diseñada para el análisis y la manipulación de datos. Creada por Wes McKinney en 2008, ahora la mantiene una comunidad. Está construida sobre NumPy, ofreciendo estructuras de datos flexibles y eficientes. Facilita la limpieza, transformación y análisis de datos tabulares y series temporales. Es ampliamente utilizada en ciencia de datos, finanzas, economía e ingeniería.

by Juan Luis Cueto Morelo

Componentes Clave de Pandas: Series y DataFrames

Series

Un array unidimensional etiquetado, similar a un array de NumPy pero con índices. Puede contener datos de cualquier tipo. Los índices explícitos permiten un acceso y manipulación intuitivos.

```
pd.Series([10, 20, 30, 40], index=['a', 'b', 'c', 'd']`
```

DataFrames

Una estructura tabular bidimensional, similar a una hoja de cálculo o tabla SQL. Compuesta por Series alineadas, compartiendo el mismo índice. Permite almacenar y manipular datos heterogéneos.

```
pd.DataFrame({'col1': [1, 2], 'col2': ['A', 'B']})
```

Funcionalidades Esenciales de Pandas

1 Lectura y Escritura de Datos

Soporta diversos formatos como CSV, Excel, SQL, JSON. Funciones como **read_csv()** y **to_excel()** son clave. Permiten la importación y exportación fácil de datos.

```
df =
pd.read_csv('datos.csv')
df.to_excel('salida.xlsx
')
```

2 Indexación y Selección de Datos

Selecciona por etiqueta (**.loc**) o por posición (**.iloc**). Permite un filtrado condicional preciso. Facilita la extracción de subconjuntos de datos específicos.

```
df.loc[0, 'columna']
df.iloc[0, 0]
df[df['valor'] > 10]
```

3 Manejo de Datos Faltantes

Identifica valores NaN con **isna()**. Elimina filas o columnas con **dropna()**. Imputa valores faltantes con **fillna()**, usando media o mediana.

```
df.isna()
df.dropna()
df.fillna(df['col'].mean
())
```

Manipulación de Datos con Pandas

Transformación de Datos

Aplica funciones con **.apply()** y **.map()**. Crea nuevas columnas basadas en cálculos. Permite manipular datos para análisis posteriores.

Agrupamiento de Datos

Agrupa por una o varias columnas con **groupby()**. Calcula estadísticas descriptivas por grupo. Revela patrones y tendencias en subconjuntos.

Unión y Combinación

Concatena DataFrames con **concat()**. Combina DataFrames con **merge()**, similar a SQL joins. Facilita la integración de conjuntos de datos.

Limpieza de Datos con Pandas

Manejo de Duplicados

Identifica filas duplicadas con **duplicated()**. Elimina entradas redundantes con **drop_duplicates()**. Asegura la unicidad de los datos.

Corrección de Tipos de Datos

Convierte tipos con **astype()**. Maneja fechas y horas con **to_datetime()**. Garantiza la consistencia de los formatos.

Eliminación de Outliers

Identifica outliers con Z-score o IQR. Filtra filas con valores atípicos. Mejora la robustez del análisis estadístico.

Análisis Exploratorio de Datos (EDA) con Pandas

Estadísticas Descriptivas

Calcula media, mediana, desviación estándar y más con **describe()**. Analiza distribuciones con histogramas y diagramas de caja. Proporciona una visión general de los datos.

Correlación

Calcula la matriz de correlación con **corr()**. Visualiza la correlación con mapas de calor. Revela relaciones entre variables.

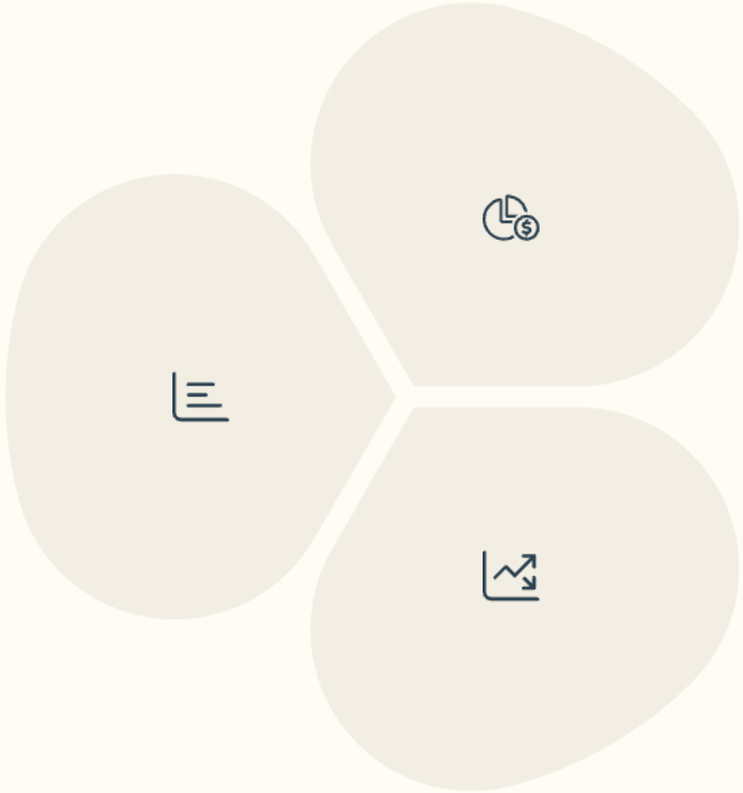
Tablas de Contingencia

Crea tablas de contingencia con **crosstab()**. Analiza relaciones entre variables categóricas. Facilita la comprensión de dependencias.

Visualización de Datos con Pandas

Integración con Matplotlib

Pandas ofrece métodos directos para gráficos básicos. Mayor flexibilidad con Matplotlib y Seaborn. Permite personalización avanzada.



Tipos de Gráficos

Histogramas, diagramas de dispersión, gráficos de barras. También diagramas de caja y más. Visualiza tendencias y distribuciones.

Ejemplos de Uso

df['columna'].hist() para histogramas. **df.plot.scatter(x='columna1', y='columna2')** para dispersión. Crea visualizaciones rápidamente.

Características Avanzadas y Optimización

MultiIndex

Índices jerárquicos para datos multidimensionales. Permite una organización de datos más compleja.

Panel Data

Estructura de datos tridimensional, hoy menos usada. Reemplazada por alternativas más eficientes.

Categorical Data

Tipo de dato eficiente para variables categóricas. Optimiza el uso de memoria y rendimiento.

Optimización con Dask

Uso de **Dask** para datasets grandes. Permite trabajar con datos que no caben en memoria. Escala el análisis de datos.

Resumen Integral de Pandas

Fundamentos y Manipulación

Pandas es una herramienta esencial para el análisis de datos. Sus Series y DataFrames son clave. Permite una manipulación de datos muy eficiente.

Limpieza y Análisis Avanzado

Simplifica la limpieza y preparación de datos. Facilita el análisis exploratorio detallado. Además, ofrece visualización y optimizaciones para escala.