INTRODUZIONE
A
PANDAS & MATPLOTLIB

Cos’è pandas?

- Libreria Python open-source e facile da utilizzare
- Fornisce strumenti di analisi dati e strutture dati ad alte prestazioni
- Molto utilizzata in AI/ML per pre-processamento, analisi e selezione dei dati

Strutture Dati Principali:

- **Series:** array monodimensionale con etichette
- **DataFrame:** tabella bidimensionale con etichette su righe e colonne

Integrazione con Altri Tools:

- Compatibile con molte altre librerie Python (come NumPy, SciPy)
- Può leggere e scrivere dati da vari formati: CSV, Excel, SQL databases, etc.

pandas: panoramica di utilizzo

Importazione:

```
import pandas as pd
```

Creazione di Strutture Dati:

- **Series**
- `s = pd.Series([1, 2, 3, 4])`
- **DataFrame da lista**
- `df = pd.DataFrame([[1, 'A'], [2, 'B']], columns=['Numero', 'Lettera'])`
- **DataFrame da Excel oppure CSV:**
- `df = pd.read_excel('file.xlsx')`
- `df = pd.read_csv('file.csv')`

Il DataFrame (df) sarà la struttura dati principale che userete in AI.
Potete immaginarla come una tabella di dati.

pandas: operazioni sui DataFrame

Visione e Selezione dei Dati da DataFrame

- **Primi n righe:** `df.head(n)` **(Prime 5 righe:** `df.head()`**)**
- **Ultime n righe:** `df.tail(n)` **(Ultime 5 righe:** `df.tail()`**)**
- **Dimensione:** `df.shape` **(ritorna una coppia (m, n) del numero di righe e di colonne)**
- **Selezione di colonne:** `df['Nome_Colonna']` **oppure** `df[['Colonna_1', 'Colonna_2']]`
- **Selezione basata su condizioni:** `df[df['Nome_Colonna'] > valore]`

Operazioni su DataFrame

- **Media di una colonna:** `df['Nome_Colonna'].mean()`
- **Frequenza di elementi distinti di una colonna:** `df['Nome_Colonna'].value_counts()`
- **Gruppo e aggregazione:** `df.groupby('Nome_Colonna').mean()`

Manipolazione dei Dati:

- **Modifica di colonne:** `df['Nuova_Colonna'] = df['Vecchia_Colonna'] * 2`
- **Rimozione di colonne:** `df.drop('Nome_Colonna', axis=1)`
- **Gestione valori mancanti:** `data.fillna(valore)` **oppure** `data.dropna()`

Cos’è matplotlib?

Cos'è Matplotlib?

- Una libreria di visualizzazione in Python.
- Permette di creare una vasta gamma di grafici statici, animati e interattivi.

Componenti principali:

- `Figure`: L'intera immagine o finestra del grafico. **`plt.figure()`** crea una figura
- **`plt.figure().close()`** chiude una figura
- **`plt.figure().clear()`** pulisce (resetta) una figura
- `Axes`: La "parte" del grafico con dati, una singola istanza in una figura che può avere titoli, etichette, ecc.
- **`plt.show()`**: Visualizza il grafico.

`pyplot` è il modulo che offre gli strumenti della libreria per creare i grafici:

Importazione

```
import matplotlib.pyplot as plt
```

matplotlib - Grafico a torta

- Rappresenta le proporzioni di categorie rispetto al totale.
- Utilizzato per mostrare una distribuzione percentuale.

Codice di esempio:

```
etichette = 'Categoria1', 'Categoria2'
valori = [valore1, valore2]
plt.pie(etichette, labels=etichette, autopct='%1.1f%%', startangle=90)
plt.title("Titolo del Grafico")
```

Parametri utilizzati:

- `sizes`: valori per ogni segmento
- `labels`: nomi per ogni segmento
- `autopct`: formato per mostrare la percentuale
- `startangle:` angolo di inizio del primo segmento

matplotlib - Grafico a barre

Rappresenta le quantità per categorie discrete.

Codice di esempio:

```
plt.bar(categorie, valori, align='center', alpha=0.7)
plt.xticks(categorie, ["Nome1", "Nome2", "Nome3"])
```

Parametri utilizzati:

- `sizes`: valori per ogni segmento
- `labels`: nomi per ogni segmento
- `autopct`: formato per mostrare la percentuale
- `startangle:` angolo di inizio del primo segmento

matplotlib - Serie di dati e curve

Si possono sovrapporre sulla stessa figura più plot, tipicamente di serie temporali, curve o funzioni, per analizzare possibili trend o interazioni tra più dati in evoluzione.

Codice di esempio:

```
plt.figure()
plt.plot(x1, y1, label="Legenda1")
plt.plot(x2, y2, label="Legenda2")
# x ed y sono tipicamente liste o colonne di DF
plt.legend()  # Mostra la legenda sul grafico
plt.show()
```

Personalizzazione:

- `color`: Specifica il colore della linea
- `linestyle`: Specifica il tipo di linea (es. '`-`', '`--`', '`-.`')
- `marker`: Specifica il marcatore (per dati sparsi) (es. '`*`', '`+`', '`o`')

Es: `plt.plot(x, y, marker='s', linestyle='--', color='r', label='Titolo')`