Wizualizacja danych
semestr letni 2024

Dr Anna Muranova
UWM w Olsztynie

Wykład 9

Biblioteka `matplotlib`

`https://matplotlib.org/`

Zazwyczaj importuje się główny moduł tej biblioteki, czyli 'pyplot'. Zawiera on niezbędne funkcje do generowania wykresów.

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

Szczególnie wydajne jest połączenie możliwości tej biblioteki z innymi bibliotekami naukowymi, takimi jak NumPy, Pandas czy SciPy.

`https://www.w3schools.com/python/matplotlib_intro.asp`

Stylu

Sama biblioteka korzysta z domyślnych stylów podczas „upiększania" i modyfikacji wykresów. Są to np. modyfikacje kolorystyczne, czcionki, dodawanie dodatkowych informacji, legend itd. Jeżeli chcesz zapoznać się z różnymi stylami, które oferuje Matplotlib, to zajrzyj do dokumentacji lub skorzystaj z komendy 'print'

```
print(plt.style.available)
```

W celu użycia konkretnego stylu w twoim projekcie skorzystaj z funkcji

```
plt.style.use('nazwa_stylu')  # np. 'seaborn-v0_8'
```

Wykres liniowy

```
import matplotlib.pyplot as plt
import numpy as np

#plt.style.use('seaborn-v0_8')

# Przygotowanie danych
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y)  # Rysowanie wykresu liniowego
plt.show()  # Wyświetlenie wykresu
```

Wykres liniowy: podpis

```
import matplotlib.pyplot as plt
import numpy as np

#plt.style.use('seaborn-v0_8')

# Przygotowanie danych
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y, label='sin(x)')
       #Dodanie etykiety dla serii danych (do legendy)
plt.xlabel('X')  # Etykieta osi X
plt.ylabel('Y')  # Etykieta osi Y
plt.title('Wykres funkcji sinus')  # Tytuł wykresu
plt.legend()  # Dodanie legendy
plt.grid() #siatka
plt.show()  # Wyświetlenie wykresu
```

Wykres liniowy: opcje zaawansowane

```
import matplotlib.pyplot as plt
import numpy as np


#plt.style.use('seaborn-v0_8')

# Przygotowanie danych
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y, label='sin(x)', color='green', linewidth=3)
       #Dodanie etykiety dla serii danych (do legendy)
plt.xlabel('X')  # Etykieta osi X
plt.ylabel('Y')  # Etykieta osi Y
plt.title('Wykres funkcji sinus')  # Tytuł wykresu
plt.legend()  # Dodanie legendy
plt.grid(color='r', linestyle='--', linewidth=0.5) #siatka
plt.show()  # Wyświetlenie wykresu
```

Kilka wykresów w jednym okienku na jednej skale

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-4,4,100)
y = np.sin(2 * x)
y1 = 2*np.sin(x)
y2 = np.sin(x)
plt.plot(x, y2, 'blue', linestyle="-", label="sin x")
plt.plot(x, y1, 'red', linestyle=":", label="2sin(x)")
plt.plot(x, y, 'green', linestyle="--", label="sin(2x)")
plt.legend(title='Wykres')
plt.show()
```

Kilka wykresów w jednym okienku na rożnych skalach

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-4,4,100)
y = np.sin(2 * x)
y1 = 2*np.sin(x)
y2 = np.sin(x)
fig, axs = plt.subplots(3)
fig.suptitle('Vertically stacked subplots')
axs[0].plot(x, y2, 'blue', linestyle="-", label="sin x")
axs[1].plot(x, y1, 'red', linestyle=":", label="2sin(x)")
axs[2].plot(x,y, 'green', linestyle="--", label="sin(2x)")

#plt.ylim(-2,2)
plt.legend(title='Wykres')
plt.show()
```

Drugi sposób na zakresy

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-4,4,100)
y = np.sin(2 * x)
y1 = 2*np.sin(x)
y2 = np.sin(x)
fig, axs = plt.subplots(3)
fig.suptitle('Vertically stacked subplots')
axs[0].plot(x, y2, 'blue', linestyle="-", label="sin x")
axs[0].set_ylim(-2,2)
axs[1].plot(x, y1, 'red', linestyle=":", label="2sin(x)")
axs[2].plot(x,y, 'green', linestyle="--", label="sin(2x)")
axs[2].set_ylim(-2,2)
plt.legend(title='Wykres')
plt.show()
```

Jeszcze przykład 1

```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 2 * np.pi, 400)
y = np.sin(x ** 2)
fig, axs = plt.subplots(2, 2)
axs[0, 0].plot(x, y)
axs[0, 0].set_title('Axis [0, 0]')
axs[0, 1].plot(x, y, 'tab:orange')
axs[0, 1].set_title('Axis [0, 1]')
axs[1, 0].plot(x, -y, 'tab:green')
axs[1, 0].set_title('Axis [1, 0]')
axs[1, 1].plot(x, -y, 'tab:red')
axs[1, 1].set_title('Axis [1, 1]')
for ax in axs.flat:
    ax.set(xlabel='x-label', ylabel='y-label')
# Hide x labels and tick labels for top plots and y ticks for right plo
for ax in axs.flat:
    ax.label_outer()
```

Jeszcze przykład 2

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2 * np.pi, 400)
y = np.sin(x ** 2)

fig = plt.figure()
gs = fig.add_gridspec(3, hspace=0)
axs = gs.subplots(sharex=True, sharey=True)
fig.suptitle('Sharing both axes')
axs[0].plot(x, y ** 2)
axs[1].plot(x, 0.3 * y, 'o')
axs[2].plot(x, y, '+')

# Hide x labels and tick labels for all but bottom plot.
for ax in axs:
    ax.label_outer()
```

Inne wykresy: wykres punktowy

Zbiór punktów, które nie są ze sobą połączone. Używany w wizualizacji korelacji pomiędzy zmiennymi, rozkładu wartości albo określania grup.

```
import numpy as np
import matplotlib.pyplot as plt

# Przykładowe dane
wiek = np.random.randint(18, 65, 100)
zarobki = np.random.normal(50000, 10000, 100) + (wiek - 18) * 1000
plt.scatter(wiek, zarobki)
plt.xlabel('Wiek')
plt.title('Związek między wiekiem a rocznymi zarobkami')

plt.show()
```

Inne wykresy: histogram

Histogram – jest rodzajem wykresu słupkowego, który ilustruje rozkład wartości danych. Wykorzystywany do wizualizacji częstotliwości występowania wartości w wybranym zbiorze danych.

```
import numpy as np
import matplotlib.pyplot as plt

oceny = np.array([2,2,2,2,3,3,3,3,3,3,3,3.5,3.5,3.5,3.5,3.5,3.5,4,4,
                          4,4,4,4,4.5,4.5,5,5,5,5,5,5])
print(oceny)
plt.hist(oceny, label='Oceny uczniów',align='mid')
plt.xlabel('Ocena')
plt.ylabel('Częstotliwość')
plt.title('Rozkład ocen w grupie uczniów')
plt.legend()
plt.show()
```

Inne wykresy: wykres słupkowy

Wykres słupkowy – jeden z najpopularniejszych wykresów, który przedstawia dane za pomocą poziomych lub pionowych słupków. Zazwyczaj ułatwia porównanie danych, które pogrupowano w konkretne kategorie.

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array(["A", "B", "C", "D"])
y = np.array([3, 8, 1, 10])

plt.bar(x,y)
plt.show()
```

Kolory: `https://www.w3schools.com/python/matplotlib_bars.asp`

Inne wykresy: histogram przy pomocy bar

```
import numpy as np
import matplotlib.pyplot as plt

oceny = np.array([2,2,2,2,3,3,3,3,3,3,3,3.5,3.5,3.5,3.5,3.5,3.5,
                            4,4,4,4,4,4,4.5,4.5,5,5,5,5,5,5])
print(oceny)
x = []
y = []
for i in np.unique(oceny):
    x.append(i)
    y.append(len(oceny[oceny==i]))

plt.bar(x,y, align='center',width=0.3,color='r')
plt.xlabel('Ocena')
plt.ylabel('Częstotliwość')
plt.title('Rozkład ocen w grupie uczniów')
plt.legend()
plt.show()
```

Inne wykresy: wykres kołowy

Wykres kołowy – dane znajdują się w sektorach w obrębie koła. Doskonale obrazuje proporcje poszczególnych elementów lub grup względem całości.

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Grapes"]

plt.pie(y, labels = mylabels)
plt.show()
```

Inne wykresy: wykres kołowy, opcje zaawansowane

```
import matplotlib.pyplot as plt
import numpy as np

y = np.array([35, 25, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
mycolors = ["black", "hotpink", "b", "#4CAF50"]
myexplode = [0.2, 0, 0, 0]

plt.pie(y, labels = mylabels, explode = myexplode, shadow = True,
                colors = mycolors)
plt.legend(title = "Four Fruits:", loc = 2)
plt.show()
```

https://www.geeksforgeeks.org/
change-the-legend-position-in-matplotlib/

Inne wykresy: wykres pudełkowy

Wykres pudełkowy (przedstawiony pionowo) tworzy się odkładając na pionowej osi wartości niektórych parametrów rozkładu. Nad osią umieszczony jest prostokąt (pudełko), którego dol jest wyznaczony przez pierwszy kwartyl, zaś górny bok przez trzeci kwartyl. Wysokość pudełka odpowiada wartości rozstępu ćwiartkowego. Wewnątrz prostokąta znajduje się pozioma linia, określająca wartość mediany.
Rysunek pudełka uzupełniamy odcinkami zwanymi wąsami. W najprostszym wariancie dolny koniec dolnego odcinka wyznacza najmniejszą wartość w zbiorze, natomiast górny koniec górnego odcinka to wartość największa.

```
import matplotlib.pyplot as plt
import numpy as np

# Generate some random data
data = [np.random.normal(0, std, 100) for std in range(1, 4)]
print(data)
# Create a box and whisker plot
plt.boxplot(data)

plt.show()
```

Inne wykresy: mapa ciepła

Mapa ciepła (heatmap) – nie jest typowym wykresem znanym np. ze szkoły, ponieważ występuje w postaci macierzy, na której kolory odpowiadają wartościom konkretnych komórek. Dobrze obrazuje korelacje i gradienty wartości oraz wykrywa wzorce w przypadku danych dwuwymiarowych.

```
import matplotlib.pyplot as plt
import numpy as np

a = np.random.random((16, 16))
plt.imshow(a, cmap='hot', interpolation='nearest')
plt.show()
```

Inne wykresy: mapa ciepła

```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-10, 40, 100)
# długość geograficzna dla Europy (w przybliżeniu)
y = np.linspace(35, 70, 100)
# szerokość geograficzna dla Europy (w przybliżeniu)
X, Y = np.meshgrid(x, y)
cisnienie = 1000 + 10 * np.sin(np.sqrt((X - 10)**2 + (Y - 50)**2))

plt.imshow(cisnienie, cmap='coolwarm', interpolation='nearest')
plt.show()
```

Inne wykresy: wykres konturowy

Wykres konturowy – używany jest do obrazowania danych trójwymiarowych w postaci linii poziomicy na powierzchni dwuwymiarowej. Wizualizuje wartości, które są równe dla konkretnej funkcji na danej płaszczyźnie.
Przykładem może być np. wizualizacja danych pogodowych lub wartości ciśnienia atmosferycznego na mapie meteorologicznej.

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(-10, 40, 100)
# długość geograficzna dla Europy (w przybliżeniu)
y = np.linspace(35, 70, 100)
# szerokość geograficzna dla Europy (w przybliżeniu)
X, Y = np.meshgrid(x, y)
cisnienie = 1000 + 10 * np.sin(np.sqrt((X - 10)**2 + (Y - 50)**2))
plt.contour(X, Y, cisnienie, levels=20, cmap='coolwarm')
plt.xlabel('Długość geograficzna')
plt.ylabel('Szerokość geograficzna')
plt.title('Ciśnienie atmosferyczne na mapie meteorologicznej')
plt.colorbar(label='hPa')
plt.show()
```

Inne wykresy: 3D

```
import numpy as np
import matplotlib.pyplot as plt

# Definiowanie danych
x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
x, y = np.meshgrid(x, y)
z = np.sin(np.sqrt(x**2 + y**2))

# Inicjalizacja wykresu 3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
```

Inne wykresy: 3D

```
# Tworzenie wykresu powierzchniowego
surf = ax.plot_surface(x, y, z, cmap='plasma',
                       edgecolors='k', linewidth=0.5)

# Dodanie paska kolorów
cbar = fig.colorbar(surf, pad=0.15, shrink=0.5, aspect=5)
cbar.ax.yaxis.set_ticks_position('right')

# Etykiety osi
ax.set_xlabel('Oś X', labelpad=10)
# labelpad zapobiega nakładaniu się osi
ax.set_ylabel('Oś Y', labelpad=10)
ax.set_zlabel('Oś Z', labelpad=10)

# Tytuł wykresu
ax.set_title('Wykres powierzchniowy')
plt.show()
```