Wizualizacja danych
semestr letni 2024

Dr Anna Muranova
UWM w Olsztynie

Wykład 11

Biblioteka Pandas

Pandas to biblioteka Python, która umożliwia efektywną pracę z danymi w postaci tabelarycznej. Pandas współpracuje z biblioteką Matplotlib i umożliwia szybkie generowanie wykresów.
"the name is derived from the term "panel data", an econometrics term for multidimensional structured data sets."
Ściągawka: `https://github.com/pandas-dev/pandas/blob/master/doc/cheatsheet/Pandas_Cheat_Sheet.pdf`

```
import pandas as pd
```

Dzięki Pandas łatwo jesteśmy w stanie przeprowadzać takie operacje jak: czyszczenie danych, normalizacja danych, wizualizacja danych, analiza statyczna, ładowanie oraz zapisywanie danych i wiele więcej.
Głównym celem biblioteki Pandas jest ułatwienie pracy z danymi, dlatego Pandas wprowadza dwie struktury danych: Series i DataFrame. Zrozumienie tych struktur jest kluczowe do efektywnego korzystania z tej biblioteki.

Series

Series to jednowymiarowa struktura danych, a właściwie tablicy (ndarray), podobna do listy lub kolumny w tabeli. Każdy element (np. liczby całkowite, listy, obiekty, tuple) w Series ma przypisany identyfikator, który nazywany jest indeksem. Series przechowuje dane jednego typu.

```
import pandas as pd

s_int = pd.Series([1, 32, -37, 91, 12, 11, -5],
                        index = ['a','b','c','d','e','f','g'])
s_str = pd.Series(['My', 'name','is', 'Anna', 'Muranova','.'])
print(s_int)
print(s_str)
my_list = [4, 3, 2, 1, 0 ,-1, -2, -3, -4]
s_list = pd.Series(my_list)
print(s_list)
int_list = s_int.tolist()
print(int_list)
s_float = pd.Series([1.5, 32.3, -37.1, 91, 12.9, 11, -5.2],
                index = ['a','b','c','d','e','f','g'])
print(s_float[(s_float < 0)])
```

Data Frame

DataFrame to dwuwymiarowa struktura danych podobna do tabeli w bazie danych lub arkusza kalkulacyjnego Excela. DataFrame składa się z wierszy i kolumn – każda kolumna w DataFrame to Series. Jak pewnie się domyślasz, mimo że dana kolumna zawiera tylko jeden typ danych, to DataFrame może zawierać wiele kolumn, z których każda ma dane innego typu.

```
import pandas as pd

my_list = [1, 32, -37, 91, 12, 11, -5]
df_list = pd.DataFrame (my_list, index = ['a','b','c','d','e','f','g'],
                  columns = ['numbers'])
print(df_list)
df = pd.DataFrame ([[1, 2, 4, 5],[-3, 8, 0.5, 10],[2, -5, 7, 3]],
             index = ['l1','l2','l3'], columns = ['a','b','c','d'])
print(df)
print(df.iloc[1,1:3])
#print(df[1,1:3])
print(df.max())
print(df['a'].max())
#print(df['l2'].max())
```

Wczytywanie danych z pliku

```
import pandas as pd

data = pd.read_csv('penguins.csv', sep=',', index_col=False,
          encoding='UTF-8')
print(data)
```

Przykłady

```
#średnia waga w każdej plecie
print(data.groupby('sex')['body_mass_g'].mean())

#średnia waga w każdej plecie,
print(data.dropna().groupby('sex')['body_mass_g'].mean())
#średnia waga z podziałem na płeć  i gatunek
print(data.dropna().groupby(by=['sex','species'])
['body_mass_g'].mean())

#wszystkie wartości dla pingwinów z najmniejsza waga.
print(data[data['body_mass_g']==data['body_mass_g'].min()])
print(data[data['body_mass_g']==data['body_mass_g'].min()].to_string())

#ilość pingwinów gatunku 'Adelie' na każdej wyspie
print(data[data['species']=='Adelie'].groupby('island').size())

#ilość pingwinów każdego gatunku na każdej wyspie
print(data.groupby(by =['species','island']).size())
```

Wykresy 1

```
import pandas as pd
import matplotlib.pyplot as plt

data = pd.read_csv('penguins.csv', sep=',', index_col=False,
 encoding='UTF-8')

#wykres słupkowy ilości pingwinów w zależności od wyspy.
data.groupby(['island']).size().plot.bar()
plt.show()

#wykres punktowy zależności  szerokości dzióba od długości.
data.plot.scatter(x = 'bill_length_mm',y ='bill_depth_mm')
plt.show()

#w ktorym kolor punktów zależy od plecie, a rozmiar - od wagi.
```

Wykresy 2

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

data = pd.read_csv('penguins.csv', sep=',', index_col=False,
                           encoding='UTF-8')

colors = np.where(data['sex']=='MALE','blue','red')
weight = ((data['body_mass_g']/2000)**5).astype(float)



#wykres punktowy zależności  szerokości dzióba od długości.
#w ktorym kolor punktów zależy od plecie, a rozmiar - od wagi.

data.plot.scatter(x = 'bill_length_mm',y ='bill_depth_mm',
                          c = colors, s = weight)
plt.show()
```

Biblioteka Seaborn

Seaborn, to zgrabna oraz efektywna biblioteka, pozwalająca na szybkie tworzenie atrakcyjnych wykresów, w Python. Została, zbudowana na bazie biblioteki Matplotlib, jednocześnie wzbogacona o dodatkowe typy wykresów.

`https://seaborn.pydata.org/`

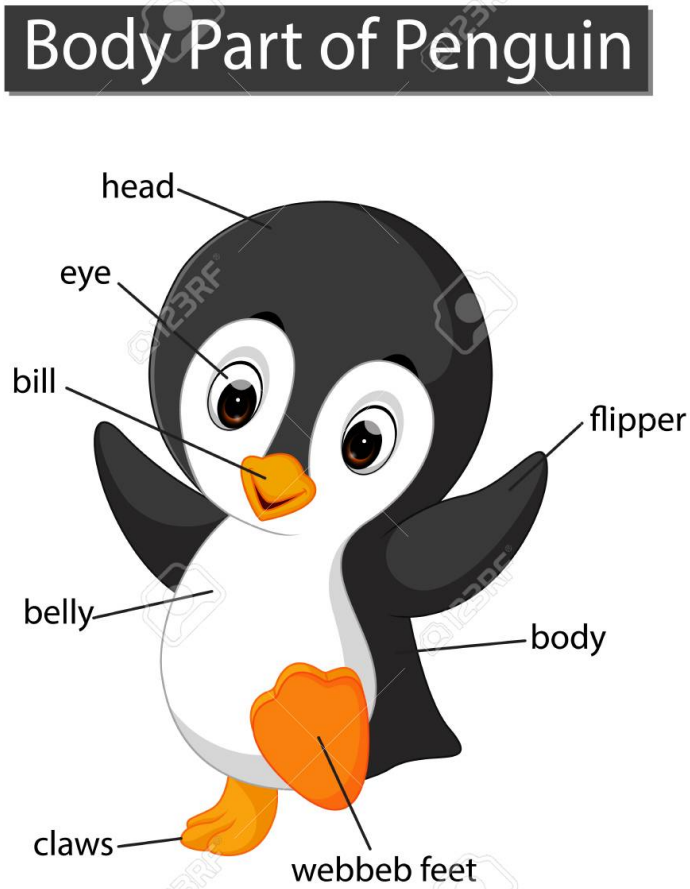
Porównaj:

```
import numpy as np
#import seaborn as sns
import matplotlib.pyplot as plt


def sinplot(flip=1):
    x = np.linspace(0, 14, 100)
    for i in range(1, 5):
        plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)

#sns.set_style("whitegrid")
#sns.set_palette("husl")
sinplot()
#print(sns.axes_style())
plt.show()
```

Pingwiny



https://github.com/mwaskom/seaborn-data

Seaborn: pingwiny

```
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

# Apply the default theme
sns.set_theme()

#penguins = sns.load_dataset("penguins")
penguins = pd.read_csv('penguins.csv', index_col=None,
encoding="UTF-8")

sns.relplot(
    data=penguins,
    x="bill_length_mm", y="bill_depth_mm",
    hue="sex", style="species", size="body_mass_g",)

plt.show()
```

Seaborn: jeszcze pingwiny

```
import matplotlib.pyplot as plt

import seaborn as sns
sns.set_theme(style="ticks")

df = sns.load_dataset("penguins")
sns.pairplot(df, hue="species")
plt.show()
```

4 główne typy wykresów w Seaborn

Seaborn, posiada całkiem rozbudowany arsenał wykresów. Szczegóły dotyczące wszystkich z nich, znajdziemy na stronie produktu. My skupimy się na nauce pakietu oraz na 4 najczęściej stosowanych typach wykresów. Mianowicie

1. Wykresy relacyjne
2. Wykresy z kategoriami
3. Wykresy z regresją
4. Wykresy dystrybucji oraz korelacji (histogram)

Wykresy relacyjne: funkcja relplot()

Podstawowe parametry:

data – wskazujemy zbiór danych
x, y – wskazujemy dane, które mają być umieszczone na osi x oraz y
hue – jeżeli chcemy, aby dane były kolorystycznie różne, w zależności od wartości zmiennej, to tutaj ją podajemy
col, row – w ilu kolumnach i wierszach mają się wyświetlić wykresy
kind – typ wykresu – czy liniowy, czy z punktami
aspect – szerokość wykresu

Wykresy relacyjne: przykład 1

```
import matplotlib.pyplot as plt

import seaborn as sns
sns.set_theme(style="ticks")

tips = sns.load_dataset("tips")
sns.relplot(x="total_bill",
            y="tip",
            aspect=2.5,
            data=tips,
            size='size',
            hue='smoker',
            kind="scatter");
plt.show()#
```

Wykresy relacyjne: przykład 2

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

import seaborn as sns


x0=np.linspace(0,10,100)
print(x0)
sns.relplot(x=x0,
            y=np.sin(x0),
            kind="line");
plt.show()
```

Wykresy z kategoriami

Kolejne, popularne wykresy, to wykresy słupkowe różnego typu. Podstawową funkcją, którą powinniśmy poznać, jest 'catplot()'. Nazwa, od category plot. Podobnie, jak w przypadku funkcji 'relplot()', mamy kilka rodzajów wykresów słupkowych. Count, bar, box itd.

```
import matplotlib.pyplot as plt
import seaborn as sns

df = sns.load_dataset("penguins")
sns.catplot(x="species",
            y="body_mass_g",
            data=df,
            hue = 'sex',
#kind = 'box'
#kind = 'violin'
)
plt.show()
```

Wykresy z kategoriami

```
import matplotlib.pyplot as plt
import seaborn as sns

df = sns.load_dataset("penguins")
sns.catplot(x="species",
            data=df,
            hue = 'sex',
            kind = 'count'
            )
plt.show()
```

Wykresy z regresja

Regresja liniowa:

```
import matplotlib.pyplot as plt
import seaborn as sns

df = sns.load_dataset("penguins")
sns.lmplot(data=df,
           x="bill_length_mm",
           y="bill_depth_mm",
           aspect=2.5,)
plt.show()
```

Histogram

Histogram

```
import matplotlib.pyplot as plt
import seaborn as sns

df = sns.load_dataset("penguins")
sns.histplot(data=df, y="flipper_length_mm")
#sns.histplot(data=df, y="flipper_length_mm",
hue='species', multiple='stack')
plt.show()
```