Wizualizacja danych
semestr letni 2024

Dr Anna Muranova
UWM w Olsztynie

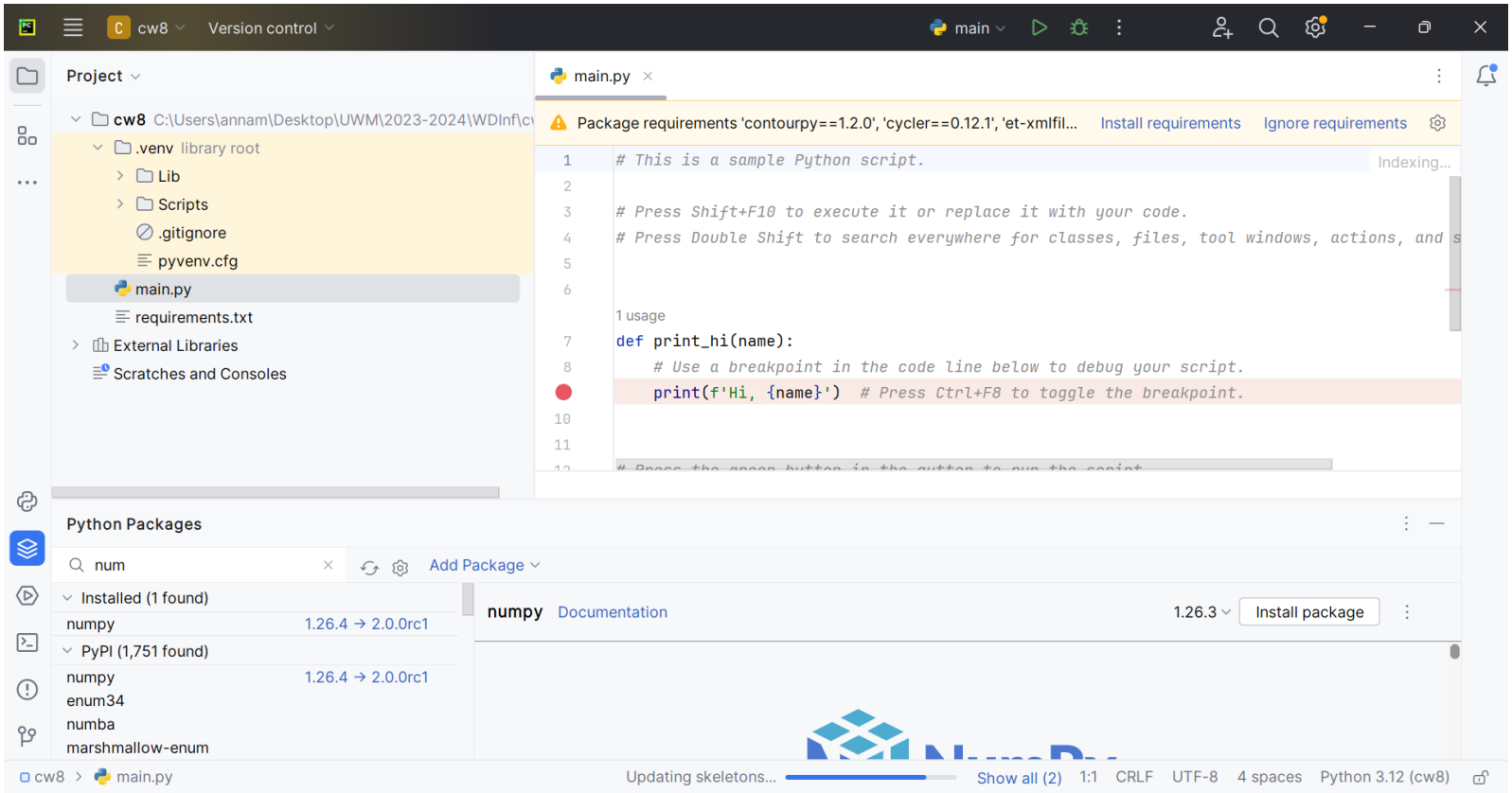
Wykład 7

Biblioteka NumPy

W przypadku metod obliczeniowych jedną z najpopularniejszych bibliotek jest właśnie **NumPy** (Numeric Python). Jak nazwa wskazuje, skupia się ona głównie na wsparciu pracy z danymi numerycznymi i umożliwia efektywne i wydajne dokonywanie operacji na macierzach, tablicach czy wektorach, które zawierają elementy tego samego typu.

Instalowanie pakietów

1 sposób
Python packages –> Wyszukać pakiet **numpy** i zainstalować (lepiej w wersie `1.26.3`)



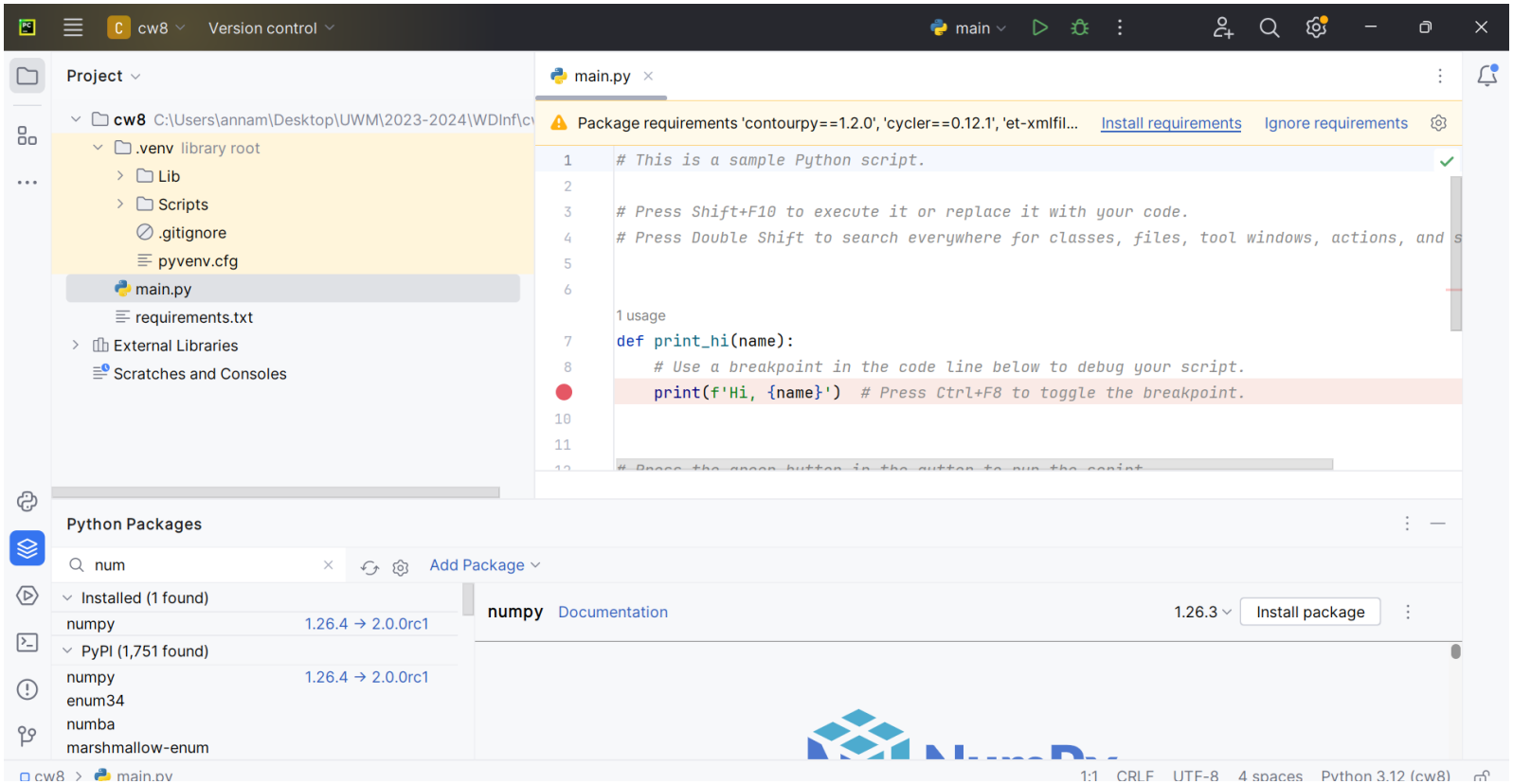
Instalowanie pakietów

2 sposób

Dodać plik `requirements.txt`:

`http://wmii.uwm.edu.pl/~muranova/WDM2024/requirements.txt`

do folderu projektu w przeglądarce plików, otworzyć ten plik w projekcie w **PyCharmie** i na gorze wybrac *Install requirements*



Instalowanie pakietów

2 sposób
Dlaczego może nie być tego banneru na gorze:

- `Reader mode` musi być wyłączony:
  File –> Settings –> Editor –> Reader Mode
- Settings –> Editor –> Inspections –> unsatisfied package requirements musi być zaznaczone

Więcej na ten temat:
`https://www.jetbrains.com/help/pycharm/managing-project-dependencies.html`

Tablicy

Tablica (ang. array) w `NumPy` to struktura o jednym wymiarze lub większej liczbie wymiarów pozwalająca na działania ze zbiorami danych numerycznych, zaczynając od kilkuelementowych po ogromne zbiory przechowywane np. w chmurze. Pamiętajmy, że nie tylko wymiary charakteryzują tablicę – istnieje też kilka innych, równie ważnych cech.

- Tablice są stałej wielkości – nie możemy zmienić rozmiaru po jej utworzeniu.
- Przechowują elementy tego samego typu, np. liczby całkowite lub zmiennoprzecinkowe.
- Wykorzystywane „pod spodem" algorytmy pozwalają na bardzo szybkie operacje i efektywne wykorzystanie pamięci.

Niektórzy porównują tablice do list w Pythonie, ale listy różnią się od tablic, np. możliwością przechowywania elementów różnego typu (listy mogą być heterogeniczne), a także są jednowymiarowe (chociaż możliwe jest przechowywanie jednowymiarowej listy w drugiej liście).

Konwersja listy w tablicę

```
import numpy as np

# jednowymiarowa tablica na podstawie listy
arr_1d = np.array([1, 2, 3])
print(arr_1d)

# dwuwymiarowa tablica na podstawie listy
arr_2d = np.array([[1, 2, 3], [4, 5, 6]])
print(arr_2d)
```

Tworzenie tablic

```
import numpy as np

# jednowymiarowa tablica wypełniona zerami o długości 3
arr_zeros = np.zeros(5)
print(arr_zeros)

# tablica z 10 równo rozłożonymi wartościami w zakresie od 0 do 2
a = np.linspace(0, 2, 10)
print(a)

# dwuwymiarowa tablica wypełniona jedynkami o wymiarach 3x3
arr_2d_ones = np.ones((3, 3))
print(arr_2d_ones)
```

Tablica z wartości losowych.

```
import numpy as np

# jednowymiarowa tablica z losowymi
#wartościami z przedziału [0,1] o długości 3
arr = np.random.rand(3)
print(arr)

# dwuwymiarowa tablica z losowymi
#wartościami z rozkładu normalnego o wymiarach 3x3
arr_2d = np.random.randn(3, 3)
print(arr_2d)
```

Tworzenie tablic: resize vs. reshape

```
import numpy as np

# tworzenie tablicy z 10 elementami
a = np.array([3, 7, 3, 3, 2, 9, 7, 1, 5, 4])
print(np.shape(a))
b = a.reshape(2,5)
print(a)
print(b)
b = a.resize(2,5)
print(a)
print(b)
```

Podstawowe operacje

```
import numpy as np

# tworzenie jednowymiarych tablic a i b
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# dodawanie tablicy b do tablicy a
c = a + b
print(c)

# dodawanie 5 do tablicy a
print(5+a)

# mnożenie tablicy a przez stałą 2
d = 2 * a
print(d)
```

Operacje logiczne

Wynikiem operacji logicznych takich jak AND (koniunkcja), OR (alternatywa) czy NOT (negacja) jest tablica zawierająca wartości logiczne True i False.

```
import numpy as np

# tworzenie jednowymiarych tablic a i b
a = np.array([1, 2, 3])
b = np.array([3, 2, 1])
# sprawdzamy, które elementy w tablicy a są mniejsze od
# odpowiadających im elementów w tablicy b
c = a < b
print(c)

# sprawdzamy które elementy w tablicy a są równe 2 lub 3
d = (a == 2) | (a == 3)
print(d)
```

Operacje redukcyjne

Wynikiem operacji redukcyjnych takich jak suma, minimum, maksimum czy średnia jest skalar.

```
import numpy as np

# tworzenie jednowymiarowej tablicy a
a = np.array([1, 2, 3])

# suma elementów tablicy a
b = np.sum(a)
print(b)

# średnia arytmetyczna dla elementów tablicy a
c = np.mean(a)
print(c)

# największy element w tablicy a
d = np.max(a)
print(d)
```

Operacje algebraiczne

```
import numpy as np

# tworzenie macierzy 3x3
x = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
# transpozycja macierzy a do macierzy b
b = np.transpose(a)
print(a)
print(b)
```

Operacje algebraiczne

Dobrym przykładem jest także mnożenie macierzy i obliczenie wyznacznika.

```
import numpy as np

# tworzenie macierzy a i b
a = np.array([[1, 2], [3, 4]])
b = np.array([[5, 6], [7, 8]])

# operacja mnożenia macierzy a i b
c = np.dot(a, b)
print(c)
# porownaj
print(a*b)

# obliczenie wyznacznika macierzy a
d = np.linalg.det(a)
print(d)
```

Operacje algebraiczne

A także rozwiązanie układu równań liniowych.

```
import numpy as np

# wyznaczenie rozwiązania układu równań liniowych Ax = b
A = np.array([[1, 2], [3, 4]])
b = np.array([5, 6])
x = np.linalg.solve(A, b)
print(x)
print(np.dot(np.linalg.inv(A),b))
```

Inne funkcje

Sortowanie tablicy

```
import numpy as np

# tworzenie tablicy z 10 elementami
a = np.array([3, 7, 3, 3, 2, 9, 7, 1, 5, 4])

# sortowanie tablicy
b = np.sort(a)
print(b)
```

Indeksacja i krojenie

Jak w listach:

```
import numpy as np

data = np.array([1, 2, 3])

print(data[1])
print(data[0:2])
print(data[1:])
print(data[-2:])


b = np.array([[1., 2., 3], [3., 4., 5]])
print(b[1,2])
print(b[:,2])
```

Indeksacja i krojenie

Logiczne:

```
import numpy as np

a = np.array([1,2,3])
print(a[[True, True, False]])

b = np.array([[1 , 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
print(b[b<5])

divisible_by_2 = b[b%2==0]
print(divisible_by_2)

c = b[(b > 2) & (b < 11)]
print(b)
```

https://numpy.org/doc/stable/user/absolute_beginners.html#indexing-and-slicing

Zamiana wartości

```
import numpy as np

a = np.array([[1 , 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
a[2,1] = 0
print(a)

c = a
a[2,:] = 0
print(a)
print(c)

b = np.array([[1 , 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]])
d = b.copy()
b[b>5] = 0
print(b)
print(d)
```

Operacje matematyczne

https://numpy.org/doc/stable/reference/routines.math.html