Министерство образования Республики Беларусь

Учреждение образования

“Брестский государственный университет”

Кафедра ИИТ

Лабораторная работа №7

По дисциплине: “ЯП”

Тема: “NumPy”

Выполнил:

Студент 2-го курса

Группы ПО-7

Качан В.В.

Проверила:

Дряпко А.В.

Брест, 2021

Задание 1.

Код программы:

import numpy as np  
import scipy  
import scipy.spatial  
import matplotlib.pyplot as plt  
from math import sqrt, dist  
import scipy.sparse as sp  
import scipy.sparse.linalg as spln  
from scipy.stats import multivariate\_normal  
  
arr\_pth = [[1, 0, 1], [2, 0, 2], [3, 0, 3], [4, 4, 4]]  
  
def task\_1\_pth(arr):  
 comp = 1  
 for i in range(len(arr[0])):  
 element = arr[i][i]  
 if element:  
 comp \*= element  
 return comp  
  
  
def task\_1\_np(arr):  
 dig = np.diag(arr)  
 return np.prod(dig[dig != 0])  
  
  
print("Python:", end=" ")  
%timeit task\_1\_pth(arr\_pth)  
print("NumPy:", end=" ")  
%timeit task\_1\_np(arr\_pth)

Результат выполнения:

Python: 651 ns ± 108 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

NumPy: 14.8 µs ± 1.06 µs per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Задание 2:

Код программы:

arr\_pth2 = [[9, 4, 2], [6, 0, 0], [9, 9, 3]]  
i\_1 = [1, 2, 1]  
j\_1 = [1, 0, 1]  
  
arr\_np2 = np.array(arr\_pth2)  
i\_2 = np.array(i\_1)  
j\_2 = np.array(j\_1)  
  
  
def task\_2\_1(arr, i, j):  
 result\_array = []  
 for index in range(len(i)):  
 result\_array.append(arr[i[index]][j[index]])  
 return result\_array  
  
  
def task\_2\_2(arr, i, j):  
 return arr[i, j]  
  
  
print("Python:", end=" ")  
%timeit task\_2\_1(arr\_pth2, i\_1, j\_1)  
print("NumPy:", end=" ")  
%timeit task\_2\_2(arr\_np2, i\_2, j\_2)

Результат выполнения:

Python: 838 ns ± 89.4 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

NumPy: 3.72 µs ± 342 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Задание 3:

Код программы:

x\_1 = [1, 2, 2, 4]  
y\_1 = [4, 2, 1, 2]  
  
x\_2 = np.array(x\_1)  
y\_2 = np.array(y\_1)  
  
  
def task\_3\_1(arr1, arr2):  
 return sorted(arr1) == sorted(arr2)  
  
  
def task\_3\_2(arr1, arr2):  
 return np.array\_equal(np.bincount(arr1), np.bincount(arr2))  
  
  
print("Python:", end=" ")  
%timeit task\_3\_1(x\_1, y\_1)  
print("NumPy:", end=" ")  
%timeit task\_3\_2(x\_2, x\_2)

Результат выполнения:

Python: 516 ns ± 64 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

NumPy: 7.48 µs ± 697 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Задание 4:

Код программы:

x\_1 = [6, 2, 0, 3, 0, 0, 5, 7, 0]  
x\_2 = np.array(x\_1)  
  
  
def task\_4\_1(arr):  
 counter = 0  
 for i in range(1, len(arr)):  
 if not arr[i-1]:  
 continue  
 new\_element = arr[i]  
 if new\_element > counter:  
 counter = new\_element  
 return counter  
  
  
def task\_4\_2(arr):  
 zero = arr == 0  
 return arr[1:][zero[:-1]].max()  
  
  
print("Python:", end=" ")  
%timeit task\_4\_1(x\_1)  
print("NumPy:", end=" ")  
%timeit task\_4\_2(x\_2)

Результат выполнения:

Python: 832 ns ± 63.9 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

NumPy: 4.7 µs ± 521 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Задание 5:

Код программы:

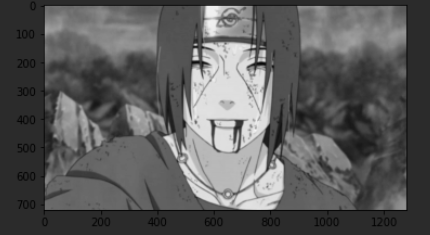
photo\_arr\_2 = plt.imread("D:\pythonlabs\lab7\image.png")  
photo\_arr\_1 = photo\_arr\_2.tolist()  
rgb\_ration\_1 = [0.298, 0.587, 0.114]  
rgb\_ration\_2 = np.array(rgb\_ration\_1)  
  
  
def task\_5\_1(arr, rgb\_ration):  
 result\_array = []  
 for height in arr:  
 new\_width = []  
 for width in height:  
 color = 0  
 for i in range(3):  
 color += width[i] \* rgb\_ration[i]  
 new\_width.append(color)  
 result\_array.append(new\_width)  
 return result\_array  
  
  
def task\_5\_2(arr, rgb\_ration):  
 return np.dot(arr[..., :3], rgb\_ration)  
  
arr\_py = task\_5\_2(photo\_arr\_2, rgb\_ration\_2)  
plt.imshow(arr\_py, cmap='gray')  
plt.show()  
  
  
print("Python:", end=" ")  
%timeit task\_5\_1(photo\_arr\_1, rgb\_ration\_1)  
print("NumPy:", end=" ")  
%timeit task\_5\_2(photo\_arr\_2, rgb\_ration\_2)

Результат выполнения:

Python: 568 ms ± 64 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)

NumPy: 27.8 ms ± 1.76 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)

Image:



Задание 6:

Код программы:

def task\_6\_1(arr):  
 bin\_arr = [0 for \_ in range(max(arr)+1)]  
 for item in arr:  
 bin\_arr[item] += 1  
  
 arr1, arr2 = [], []  
 for i in range(len(bin\_arr)):  
 element = bin\_arr[i]  
 if not element:  
 continue  
 arr1.append(i)  
 arr2.append(element)  
 return arr1, arr2  
  
  
def task\_6\_2(arr):  
 bin\_count = np.bincount(arr)  
 non\_zero\_indexes = np.nonzero(bin\_count)[0]  
 return non\_zero\_indexes, bin\_count[non\_zero\_indexes]  
  
  
print("Python:", end=" ")  
%timeit task\_6\_1(arr\_pth)  
print("NumPy:", end=" ")  
%timeit task\_6\_2(arr\_np)

Результат выполнения:

Python: 2.31 µs ± 283 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

NumPy: 3.85 µs ± 1.52 µs per loop (mean ± std. dev. of 7 runs, 100000 loops eac

Задание 7:

Код программы:

arr\_x\_1 = [2, 7, 6, 6, 9, 6, 3, 4, 9]  
arr\_y\_1 = [1, 0, 0, 7, 2, 2, 4, 3, 0]  
  
arr\_x\_2 = np.array(arr\_x\_1)  
arr\_y\_2 = np.array(arr\_y\_1)  
  
  
def task\_7\_2(arr\_x, arr\_y):  
 return np.linalg.norm(arr\_x - arr\_y)  
  
  
print("Python:", end=" ")  
%timeit dist(arr\_x\_1, arr\_y\_1)  
print("NumPy:", end=" ")  
%timeit task\_7\_2(arr\_x\_2, arr\_y\_2)  
print("Scipy:", end=" ")  
%timeit scipy.spatial.distance.euclidean(arr\_x\_2, arr\_y\_2)

Результат выполнения:

Python: 340 ns ± 51.6 ns per loop (mean ± std. dev. of 7 runs, 1000000 loops each)

NumPy: 6.86 µs ± 522 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Scipy: 9.68 µs ± 830 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)

Задание 8:

Код программы:

sigma = np.array([  
 [2.3, 0, 0, 0],  
 [0, 1.5, 0, 0],  
 [0, 0, 1.7, 0],  
 [0, 0, 0, 2]  
])  
  
mu = np.array([2, 3, 8, 10])  
x = np.array([2.1, 3.5, 8, 9.5])  
  
  
def lognormpdf(x, mu, S):  
 norm\_ratio = len(S) \* np.log(2\*np.pi) + np.linalg.slogdet(S)[1]  
 err = x - mu  
 numerator = spln.spsolve(S, err).T.dot(err)  
 return -0.5 \* (norm\_ratio + numerator)  
  
  
print("NumPy:", end=" ")  
%timeit lognormpdf(x, mu, sigma)  
res1 = lognormpdf(x, mu, sigma)  
print("Value NumPy =", res1)  
  
print("Scipy:", end=" ")  
%timeit multivariate\_normal(mu, sigma).logpdf(x)  
res2 = multivariate\_normal(mu, sigma).logpdf(x)  
print("Value Scipy =", res2)

Результат выполнения:

NumPy: 374 µs ± 185 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)

Value NumPy = -5.054836210528194

Scipy: 242 µs ± 68.3 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)

Value Scipy = -5.054836210528194

Вывод: Научился работать с NumPy. Массивы NumPy более компактны, чем списки Python. Доступ к чтению и записи элементов также выполняется быстрее с помощью NumPy.