

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 3.3
по дисциплине «Технологии распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1

Трушева В. О. .«__»_____2023г.

Подпись студента_____

Работа защищена « __ »_____20__г.

Проверила Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: исследовать базовые возможности библиотеки NumPy языка программирования Python.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.

2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и выбранный Вами язык программирования (выбор языка программирования будет доступен после установки флажка Add .gitignore).

Owner * Repository name *

pynikcmd / 3.3_LR_TRO ✓

Great repository names are short and memorable. Need inspiration? How about [upgraded-broccoli?](#)

Description (optional)

☐ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python ▼

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License ▼

This will set `main` as the default branch. Change the default name in your [settings](#).

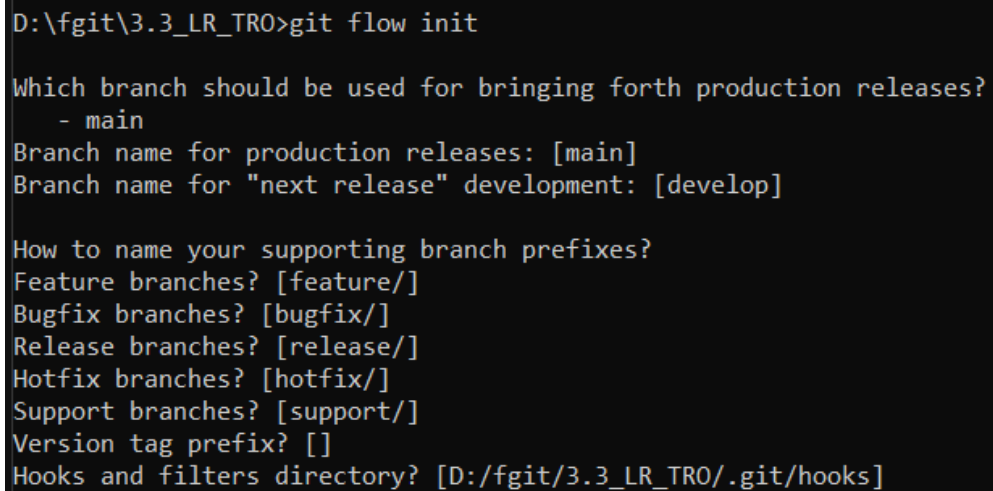
i You are creating a public repository in your personal account.

Create repository

Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория на рабочий компьютер.

4. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.



```
D:\fgit\3.3_LR_TRO>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/fgit/3.3_LR_TRO/.git/hooks]
```

Рисунок 2 – git-flow

5. Дополните файл .gitignore необходимыми правилами для выбранного языка программирования, интерактивной оболочки Jupyter notebook и интегрированной среды разработки.

6. Проработать примеры лабораторной работы.

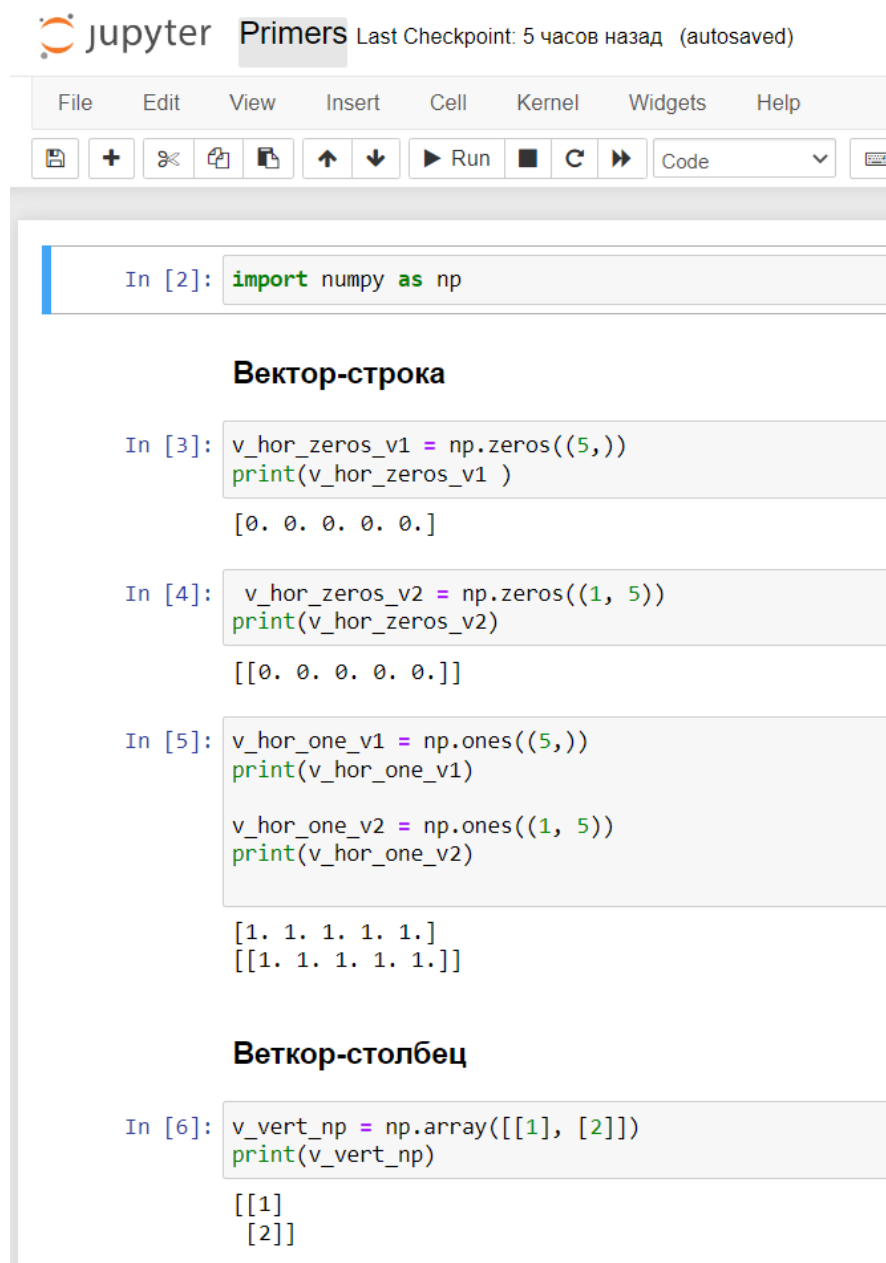


Рисунок 3 – Проработка примеров

7. Создать ноутбук, в котором будут приведены собственные примеры на языке Python для каждого из представленных свойств матричных вычислений.

Jupyter 7_Task Last Checkpoint: 5 часов назад (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run

```
In [2]: import numpy as np
```

Транспонирование матрицы

```
In [6]: A = np.random.randint(-20,25,(3,4))
print(A)

[[ 10 -16  17   1]
 [   1   0   8   0]
 [ 18  10  19  22]]
```

```
In [7]: A_T = A.T
print("Транспонированная матрица:\n",A_T)

Транспонированная матрица:
[[ 10   1  18]
 [-16   0  10]
 [ 17   8  19]
 [   1   0  22]]
```

Свойство 1. Дважды транспонированная матрица равна исходной матрице:

```
In [8]: A = np.random.randint(-20,25,(3,4))
print(A)
print((A.T).T)

[[ -2   9  -2   6]
 [  2  10   5 -12]
 [ 13 -13  11   2]]
[[ -2   9  -2   6]
 [  2  10   5 -12]
 [ 13 -13  11   2]]
```

Рисунок 4 – Выполнения заданий

8. Создать ноутбук, в котором будут приведены собственные примеры решения систем линейных уравнений матричным методом и методом Крамера.

jupyter 8_Task Last Checkpoint: a minute ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run

Матричный метод

Матричный метод может применяться в решении систем линейных уравнений, в которых число неизвестных равно числу уравнений, то есть систем линейных уравнений с квадратной матрицей коэффициентов при неизвестных. Решение системы линейных алгебраических уравнений по матричному методу определяется равенством:

$$X = A^{-1} * B$$

Где

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

- матрица системы,

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}$$

- столбец неизвестных,

$$B = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{pmatrix}$$

- столбец свободных коэффициентов.

Рисунок 5 – Выполнение заданий

9. Зафиксируйте сделанные изменения в репозитории.

10. Выполните слияние ветки для разработки с веткой main (master).

Ответы на вопросы

1. Приведите основные виды матриц и векторов. Опишите способы их создания в языке Python.

Вектором называется матрица, у которой есть только один столбец или одна строка.

Вектор-строка имеет следующую математическую запись.

```
v_hor_np = np.array([1, 2])
```

Вектор-столбец имеет следующую математическую запись.

```
v_vert_np = np.array([[1], [2]])
```

Довольно часто, на практике, приходится работать с квадратными матрицами. Квадратной называется матрица, у которой количество столбцов и строк совпадает.

```
m_sqr_arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

```
m_sqr_mx = np.matrix([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

```
m_sqr_mx = np.matrix('1 2 3; 4 5 6; 7 8 9')
```

Особым видом квадратной матрицы является диагональная – это такая матрица, у которой все элементы, кроме тех, что расположены на главной диагонали, равны нулю.

```
m_diag = [[1, 0, 0], [0, 5, 0], [0, 0, 9]]
```

```
m_diag_np = np.matrix(m_diag)
```

```
diag = np.diag(m_sqr_mx)
```

Единичной матрицей называют такую квадратную матрицу, у которой элементы главной диагонали равны единицы, а все остальные нулю.

```
m_e = [[1, 0, 0], [0, 1, 0], [0, 0, 1]]
```

```
m_e_np = np.matrix(m_e)
```

```
m_eye = np.eye(3)
```

В качестве аргумента функции передается размерность матрицы, в нашем примере – это матрица 3 3. Тот же результат можно получить с помощью функции `identity()`.

```
m_idnt = np.identity(3)
```

У нулевой матрицы все элементы равны нулю.

```
m_zeros = np.zeros((3, 3))
```

2. Как выполняется транспонирование матриц?

Транспонирование матрицы – это процесс замены строк матрицы на ее столбцы, а столбцов соответственно на строки. Полученная в результате матрица называется транспонированной. Символ операции транспонирования – буква Т.

3. Приведите свойства операции транспонирования матриц.

Свойство 1. Дважды транспонированная матрица равна исходной матрице.

Свойство 2. Транспонирование суммы матриц равно сумме транспонированных матриц.

Свойство 3. Транспонирование произведения матриц равно произведению транспонированных матриц, расставленных в обратном порядке.

Свойство 4. Транспонирование произведения матрицы на число равно произведению этого числа на транспонированную матрицу.

Свойство 5. Определители исходной и транспонированной матрицы совпадают.

4. Какие имеются средства в библиотеке NumPy для выполнения транспонирования матриц?

Функция `transpose()` или `A.T`

5. Какие существуют основные действия над матрицами?

Умножение матрицы на число

Сложение матриц

Умножение матриц

Определитель матрицы

Обратная матрица

Ранг матрицы

6. Как осуществляется умножение матрицы на число?

При умножении матрицы на число, все элементы матрицы умножаются на это число.

7. Какие свойства операции умножения матрицы на число?

Свойство 1. Произведение единицы и любой заданной матрицы равно заданной матрице.

Свойство 2. Произведение нуля и любой матрицы равно нулевой матрице, размерность которой равна исходной матрицы.

Свойство 3. Произведение матрицы на сумму чисел равно сумме произведений матрицы на каждое из этих чисел.

Свойство 4. Произведение матрицы на произведение двух чисел равно произведению второго числа и заданной матрицы, умноженному на первое число.

Свойство 5. Произведение суммы матриц на число равно сумме произведений этих матриц на заданное число.

8. Как осуществляется операции сложения и вычитания матриц?

```
>>> A = np.matrix('1 2; 3 4')
>>> B = np.matrix('5 6; 7 8')
>>> L = A + B
>>> R = B + A
>>> print(L)
[[ 6  8]
 [10 12]]
>>> print(R)
[[ 6  8]
 [10 12]]
```

9. Каковы свойства операций сложения и вычитания матриц?

Свойство 1. Коммутативность сложения. От перестановки матриц их сумма не изменяется.

Свойство 2. Ассоциативность сложения. Результат сложения трех и более матриц не зависит от порядка, в котором эта операция будет выполняться.

Свойство 3. Для любой матрицы существует противоположная ей, такая, что их сумма является нулевой матрицей.

10. Какие имеются средства в библиотеке NumPy для выполнения операций сложения и вычитания матриц?

Знак +

11. Как осуществляется операция умножения матриц?

Каждый элемент c_{ij} новой матрицы является суммой произведений элементов i -ой строки первой матрицы и j -го столбца второй матрицы.

12. Каковы свойства операции умножения матриц?

Свойство 1. Ассоциативность умножения. Результат умножения матриц не зависит от порядка, в котором будет выполняться эта операция.

Свойство 2. Дистрибутивность умножения. Произведение матрицы на сумму матриц равно сумме произведений матриц.

Свойство 3. Умножение матриц в общем виде не коммутативно. Это означает, что для матриц не выполняется правило независимости произведения от перестановки множителей.

Свойство 4. Произведение заданной матрицы на единичную равно исходной матрице.

Свойство 5. Произведение заданной матрицы на нулевую матрицу равно нулевой матрице.

13. Какие имеются средства в библиотеке NumPy для выполнения операции умножения матриц?

Функция `dot()`

14. Что такое определитель матрицы? Каковы свойства определителя матрицы?

Определитель матрицы размера (n-го порядка) является одной из ее численных характеристик. Определитель матрицы A обозначается как $|A|$ или $\det(A)$, его также называют детерминантом.

Свойство 1. Определитель матрицы остается неизменным при ее транспонировании.

Свойство 2. Если у матрицы есть строка или столбец, состоящие из нулей, то определитель такой матрицы равен нулю.

Свойство 3. При перестановке строк матрицы знак ее определителя меняется на противоположный.

Свойство 4. Если у матрицы есть две одинаковые строки, то ее определитель равен нулю.

Свойство 5. Если все элементы строки или столбца матрицы умножить на какое-то число, то и определитель будет умножен на это число.

Свойство 6. Если все элементы строки или столбца можно представить как сумму двух слагаемых, то определитель такой матрицы равен сумме определителей двух соответствующих матриц.

Свойство 7. Если к элементам одной строки прибавить элементы другой строки, умноженные на одно и тоже число, то определитель матрицы не изменится.

Свойство 8. Если строка или столбец матрицы является линейной комбинацией других строк (столбцов), то определитель такой матрицы равен нулю.

Свойство 9. Если матрица содержит пропорциональные строки, то ее определитель равен нулю.

15. Какие имеются средства в библиотеке NumPy для нахождения значения определителя матрицы?

Функция `det()` из пакета `linalg`.

16. Что такое обратная матрица? Какой алгоритм нахождения обратной матрицы?

Обратной матрицей матрицы называют матрицу, удовлетворяющую следующему равенству: $A * A^{-1} = E$

где – E это единичная матрица.

Для того, чтобы у квадратной матрицы A была обратная матрица необходимо и достаточно чтобы определитель |A| был не равен нулю. Введем понятие союзной матрицы. Союзная матрица строится на базе исходной A путем замены всех элементов матрицы A на их алгебраические дополнения.

Исходная матрица:

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}.$$

Союзная ей матрица A:

$$A^* = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{pmatrix}.$$

Транспонируя матрицу A, мы получим так называемую присоединенную матрицу A^T :

$$A^{*T} = \begin{pmatrix} A_{11} & A_{21} & \dots & A_{n1} \\ A_{12} & A_{22} & \dots & A_{n2} \\ \dots & \dots & \dots & \dots \\ A_{1n} & A_{2n} & \dots & A_{nn} \end{pmatrix}.$$

Транспонируя союзную, мы получим так называемую присоединенную матрицу.

17. Каковы свойства обратной матрицы?

Свойство 1. Обратная матрица обратной матрицы есть исходная матрица

Свойство 2. Обратная матрица транспонированной матрицы равна транспонированной матрице от обратной матрицы

Свойство 3. Обратная матрица произведения матриц равна произведению обратных матриц.

18. Какие имеются средства в библиотеке NumPy для нахождения обратной матрицы?

Функция `inv()`

19. Самостоятельно изучите метод Крамера для решения систем линейных уравнений.

Приведите алгоритм решения системы линейных уравнений методом Крамера средствами библиотеки NumPy.

Создаём матрицу СЛАУ и матрицу свободных коэффициентов, находим главный определитель матрицы (`det()`), в цикле от 1 до n меняем столбцы матрицы СЛАУ на столбец из матрицы свободных коэффициентов, находим определители полученных матриц, находим корни уравнения путём деления определителей полученных матриц на главный определитель.

20. Самостоятельно изучите матричный метод для решения систем линейных уравнений.

Приведите алгоритм решения системы линейных уравнений матричным методом средствами библиотеки NumPy. Создаём матрицу СЛАУ и матрицу свободных коэффициентов. Для решения применяется средство `numpy.linalg.solve()`.