МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет по лабораторной работе № 1
«Работа с IPython и Jupyter Notebook»

по дисциплине «Технологии распознавания образов»

Выполнила:
Первых Дарья Александровна,2 курс, группа ПИЖ-б-о-20-1,
Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г.

ВЫПОЛНЕНИЕ

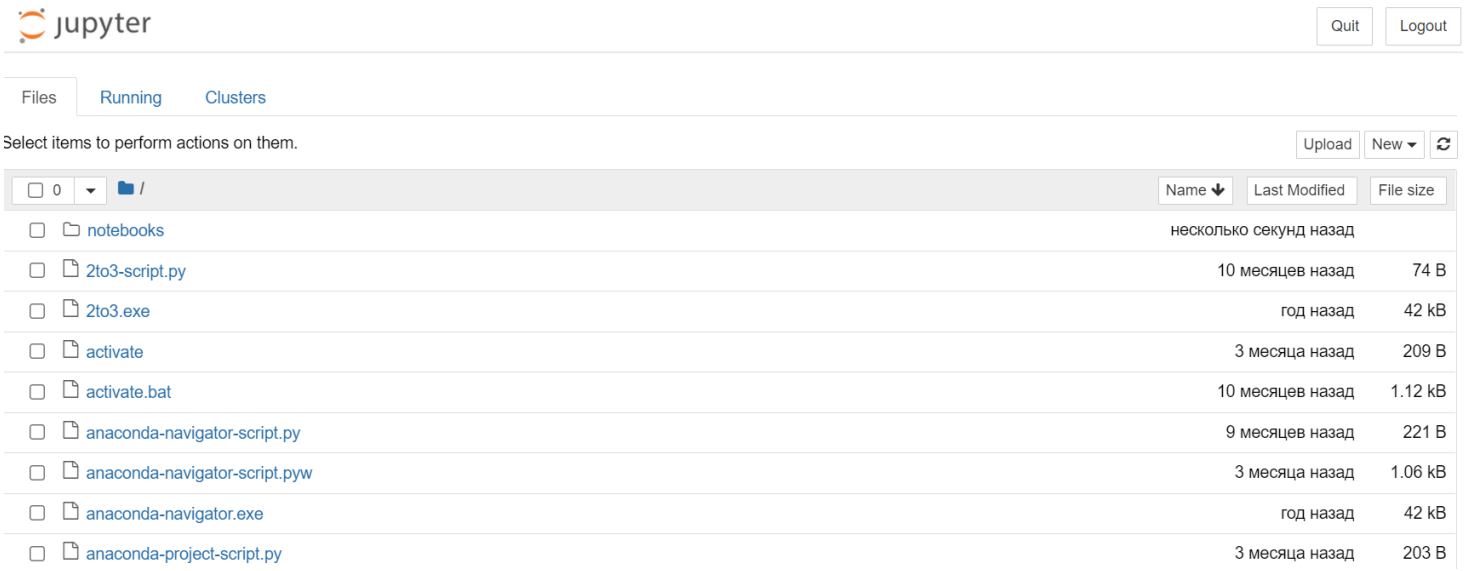

Рисунок 1 – Пример установки и запуска ноутбука

```
In [1]: 2 + 3
Out[1]: 5

In [2]: a = 5
        b = 7
        print(a + b)
        12

In [3]: n = 7
        for i in range(n):
            print(i*10)
        0
        10
        20
        30
        40
        50
        60

In [4]: i = 0
        while True:
            i += 1
            if i > 5:
                break
            print("Test while")
        Test while
        Test while
        Test while
        Test while
        Test while
```

Рисунок 2 – Пример сложения

```
In [2]: a = 5
        b = 7
        print(a + b)

        12
```

Рисунок 3 – Пример сложения переменных и вывод результата

```
In [5]: n = 7
        for i in range(n):
            print(i*10)

        0
        10
        20
        30
        40
        50
        60
```

Рисунок 4 – Пример работы с циклом

```
In [6]: i = 0
        while True:
            i += 1
            if i > 5:
                break
            print("Test while")

        Test while
        Test while
        Test while
        Test while
        Test while
```

Рисунок 5 – Пример вывода текста через цикл

File Edit View Insert Cell Kernel Help

Рисунок 6 – Пример элементов интерфейса

Рисунок 7 – Пример панели инструментов

```
In [ ]:
```

Рисунок 8 – Пример рабочего поля с ячейками

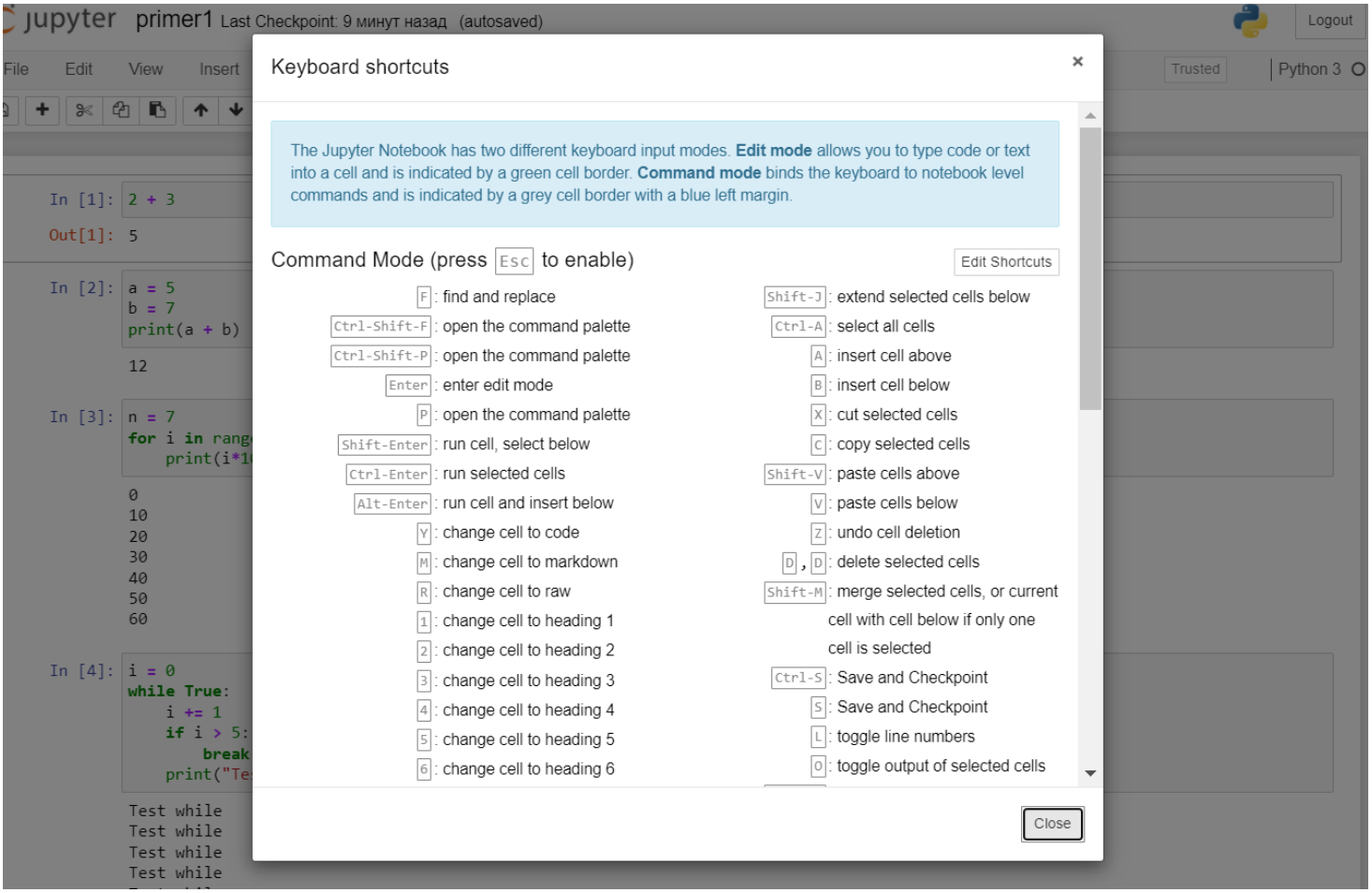

Рисунок 9 – Пример Keyboard Shortcuts

```
In [8]: from matplotlib import pylab as plt
        %matplotlib inline

In [9]: x = [i for i in range(50)]
        y = [i**2 for i in range (50)]
        plt.plot(x, y)

Out[9]: [<matplotlib.lines.Line2D at 0x24110e92c70>]
```

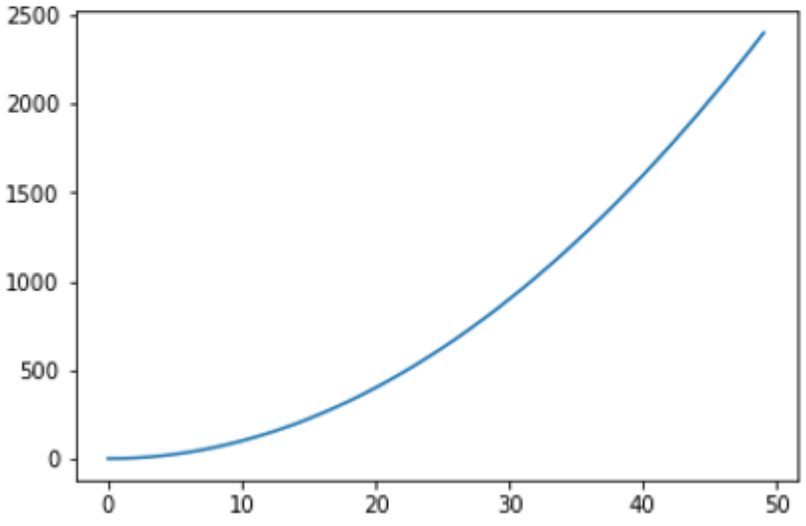

Рисунок 10 – Пример вывода графика

```
In [15]: %lsmagic
Out[15]: Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave
%bookmark  %cd  %clear  %cls  %colors  %conda  %config  %connect_inf
o  %copy  %ddir  %debug  %dhist  %dirs  %doctest_mode  %echo  %ed  %
edit  %env  %gui  %hist  %history  %killbgscripts  %ldir  %less  %lo
ad  %load_ext  %loadpy  %logoff  %logon  %logstart  %logstate  %logs
top  %ls  %lsmagic  %macro  %magic  %matplotlib  %mkdir  %more  %not
ebook  %page  %pastebin  %pdb  %pdef  %pdoc  %pfile  %pinfo  %pinfo2
%pip  %popd  %pprint  %precision  %prun  %psearch  %psource  %pushd
%pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx  %rel
oad_ext  %ren  %rep  %rerun  %reset  %reset_selective  %rmdir  %run
%save  %sc  %set_env  %store  %sx  %system  %tb  %time  %timeit  %un
alias  %unload_ext  %who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%capture  %%cmd  %%debug  %%file  %%htm
l  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun  %%pypy
%%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx
%%system  %%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

Рисунок 11 – Пример использования %ismagic

```
In [6]: %env TEST = 5
env: TEST=5
```

Рисунок 12 – Пример использования %env

```
In [9]: %%time
import time
for i in range(50):
    time.sleep(0.1)
Wall time: 5.49 s

In [10]: %timeit X = [(i**10) for i in range(10)]
7.08 µs ± 419 ns per loop (mean ± std. dev. of 7 runs, 100000 loops
each)
```

Рисунок 13 – Пример использования %%time и %timeit

```
In [1]: ticket = int(input("Номер билета: "))
        Номер билета: 123321

In [2]: sum1 = ticket // 100000 + ticket // 10000 % 10 + ticket // 1000 % 10

In [3]: print(sum1)
        6

In [4]: sum2 = ticket % 1000 // 100 + ticket % 100 // 10 + ticket % 10

In [5]: print(sum2)
        6

In [6]: if sum1 == sum2:
            print("Счастливый билет")
        else:
            print("Не счастливый билет")
        Счастливый билет

In [7]: ticket = int(input("Номер билета: "))
        Номер билета: 123451

In [8]: sum1 = ticket // 100000 + ticket // 10000 % 10 + ticket // 1000 % 10

In [9]: print(sum1)
        6

In [10]: sum2 = ticket % 1000 // 100 + ticket % 100 // 10 + ticket % 10

In [11]: print(sum2)
         10

In [12]: if sum1 == sum2:
             print("Счастливый билет")
         else:
             print("Не счастливый билет")
         Не счастливый билет
```

Рисунок 14 – Решение задания №1

In [14]:
```
password = input("Enter your pasword ")
name = input("What is your name? ")
```

```
Enter your pasword an13drei
What is your name? andrei
```

In [15]:
```
if password.isalpha() or password.isdigit():
    print("weak")
    exit(-1)
if password.islower() or password.isupper():
    print("weak")
    exit(-1)
```

```
weak
```

In [16]:
```
unic = set(password)
if len(unic) < 4:
    print("weak")
    exit(-1)
```

In [17]:
```
if name.lower() in password.lower():
    print("weak")
    exit(-1)
else:
    print("strong")
```

```
strong
```

Рисунок 15 – Решение задания №2

In [6]:
```
amount = int(input("Amount? "))
```

```
Amount? 7
```

In [7]:
```
a, b = 0, 1
for i in range(amount):
        sum = a + b
        a = b
        b = sum
        print(sum)
```

```
1
2
3
5
8
13
21
```

Рисунок 16 – Решение задания №3

```
import csv
from math import sqrt


with open('sp.csv', 'r', newline='') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    ch_t = []
    ch_o = []
    for row in data:
        if row[0] == "China":
            ch_t.append(float(row[3]))
            ch_o.append(float(row[6]))
```

```
total = sum(ch_t) / len(ch_t)
ozon = sum(ch_o) / len(ch_o)
print("Среднее значение коэффициента загрязнения воздуха в Китае "
      f": {total}"
)
print("Среднее значение коэффициента загрязнения воздуха озоном в "
      "Китае  "
      f": {ozon}"
)
```

```
Среднее значение коэффициента загрязнения воздуха в Китае : 121.992275447679
Среднее значение коэффициента загрязнения воздуха озоном в Китае  : 21.64167907440399
```

```
v1 = sum((elem-total)**2 for elem in ch_t) / len(ch_t)
st_t = sqrt(v1)
v2 = sum((elem-ozon)**2 for elem in ch_o) / len(ch_o)
st_o = sqrt(v2)
print("Стандартное отклонение коэффициента общего загрязнения воздуха: "
      f"{st_t}"
)
print("Стандартное отклонение коэффициента загрязнения воздуха озоном: "
      f"{st_o}"
)
```

```
Стандартное отклонение коэффициента общего загрязнения воздуха: 35.43897212905132
Стандартное отклонение коэффициента загрязнения воздуха озоном: 8.347006809336012
```

```
sum_ab = 0
sum_square = 0
for idx, elem in enumerate(ch_t):
    sum_ab += elem * ch_o[idx]
    sum_square += elem**2
size = len(ch_t)
k_lin = (size * sum_ab - sum(ch_t) * sum(ch_o))/(size * sum_square - sum(ch_t)**2)
b_lin = ozon - total * k_lin
func_val = []
for elem in ch_t:
    func_val.append(k_lin * elem + b_lin)
print(f"Уравнение линейной зависимости: y = {k_lin}x + {b_lin}")
print("Значения функции на кривой методом наименьших квадратов: ")
for val in func_val:
    print(val)
```

```
Уравнение линейной зависимости: y = 0.23504576503393373x + -7.032088636426096
Значения функции на кривой методом наименьших квадратов:
36.313454080979696
35.69939699313493
34.6376395348382
33.458560756956444
32.24004865765768
30.74304612961879
29.490684411441464
28.150365762587647
27.01232071848073
25.451643521712214
24.261125396094496
23.075864927894337
22.348132163167616
21.716834820734032
20.703590364450655
19.138354116290287
17.1655191635613
```

```
cor_chisl = 0
for idx, elem in enumerate(ch_t):
    cor_chisl += (elem - total)*(ch_o[idx] - ozon)
sqr_diff_total = sum((elem-total)**2 for elem in ch_t)
sqr_diff_ozone = sum((elem-ozon)**2 for elem in ch_o)
r_xy = cor_chisl / sqrt(sqr_diff_total * sqr_diff_ozone)
print(f"Коэффициент парной корреляции: {r_xy}")
```

```
Коэффициент парной корреляции: 0.9979362071170657
```

Рисунок 17 – Провождение исследования

Условие

15-го января планируется взять кредит в банке на несколько месяцев. Условия его возврата таковы:

— 1-го числа каждого месяца долг возрастает на 5% по сравнению с концом предыдущего месяца;

— со 2-го по 14-е число месяца необходимо выплатить часть долга;

— 15-го числа каждого месяца долг должен быть на одну и ту же сумму меньше долга на 15-е число предыдущего месяца.

На сколько месяцев можно взять кредит, если известно, что общая сумма выплат после полного погашения кредита на 25% больше суммы, взятой в кредит.

Решение

По формуле для переплаты П при выплате суммы кредита S дифференцированными платежами имеем:

$$\Pi = \frac{n+1}{200} rS,$$

где n — искомое число месяцев, а r = 5 — величина платежной ставки в процентах (см. Гущин Д. Д. «Встречи с финансовой математикой»; для получения полного балла доказательство этих формул необходимо приводить на экзамене). По условию, переплата П равна 0,25S, тогда:

$$0,25S = \frac{n+1}{2} \cdot 0,05S,$$

откуда n = 9.

In [10]:
```
S = 1
r = 0.05
n = 0.25*S*2/r*S - 1
print(f"Клиент выплатит кредит за {n} месяцев")
```

```
Клиент выплатит кредит за 9.0 месяцев
```

Рисунок 18 – Задача

ВОПРОСЫ

1. Как осуществляется запуск Jupyter notebook?

Для запуска Jupyter Notebook перейдите в папку Scripts (она находится внутри каталога, в котором установлена Anaconda) и в командной строке наберите: ipython notebook

2. Какие существуют типы ячеек в Jupyter notebook?

Code
Markdown
Raw NBConvert
Heading

3. Как осуществляется работа с ячейками в Jupyter notebook?

Перед первой строкой написано In []. Это ключевое слово значит, что дальше будет ввод. Попробуйте написать простое выражение вывода.

Вывод должен отобразиться прямо в notebook. Это и позволяет заниматься программированием в интерактивном формате, имея возможность отслеживать вывод каждого шага.

Также обратите внимание на то, что In [] изменилась и вместе нее теперь In [1]. Число в скобках означает порядок, в котором эта ячейка будет запущена. В первой цифра 1, потому что она была первой запущенной ячейкой. Каждую ячейку можно запускать индивидуально и цифры в скобках будут менять соответственно.

Если есть несколько ячеек, то между ними можно делиться переменными и импортами. Это позволяет проще разбивать весь код на связанные блоки, не создавая переменную каждый раз. Главное убедиться в запуске ячеек в правильном порядке, чтобы переменные не использовались до того, как были созданы.

4. Что такое "магические" команды Jupyter notebook? Какие "магические" команды Вы знаете?

%ismagic – список доступных магических команд

%env – для работы с переменными окружения

%run – для запуска файлов с расширением «.py»

%%time – позволяет получить информацию о времени работы кода в рамках одной ячейки

%timeit – запускает переданный ей код 1000000 (по умолчанию) и выводит информацию среднем значении трёх наиболее быстрых прогонах

5. Самостоятельно изучите работу с Jupyter notebook и IDE PyCharm и Visual Studio Code. Приведите основные этапы работы с Jupyter notebook в IDE PyCharm и Visual Studio Code.

IDE, которая играет важную роль при разговоре о Python, — это Jupyter Notebook. Ранее известный как IPython Notebook, Jupyter Notebook особенно важен для придания формы тому, что Дональд Кнут, ученый-компьютерщик из Стэнфорда, назвал «грамотным программированием».

Грамотное программирование — это стандартная форма программирования, ориентированная на удобочитаемость кода. Это позволяет программистам придавать форму логическим единицам своего кода, значению этих единиц кода иих результатам. Скомпилированный блокнот представляет код как законченный ипонятный мыслительный процесс и его технологическое воплощение.

Для поддержки грамотного программирования в Jupyter Notebook есть множество доступных инструментов, которые обеспечивают полную свободу редактирования кода с его соответствующей поддерживающей прозой.

Начиная с базового уровня, записные книжки (файлы, в которых написан код) могут разделять код на «ячейки». Ячейки позволяют легко различать определенные функции.

Помимо ячеек кода, доступны ячейки разметки, в которых легко ввести описание кода, значение или результаты. Возможности редактирования ячеек разметки безграничны; вы можете поиграть с текстовыми форматами, изображениями и даже математическими уравнениями и диаграммами.

Обширная поддержка интеграции Jupyter Notebook в PyCharm позволяет разработчикам создавать, выполнять и отлаживать исходные коды, одновременно изучая их выходные данные.

PyCharm позволяет вносить изменения в исходный документ разными способами. Это включает:

- Редактирование и предварительный просмотр.
- Использование записной книжки как исходного кода с определениями в виде текстов.
- Предоставление предварительных просмотров в реальном времени вместе с отладкой.
- Параметры автосохранения вашего кода.
- Выделение всех типов синтаксических ошибок и ошибок.
- Возможность добавлять комментарии к строкам.
- Возможность одновременного выполнения и предварительного просмотра результатов.
- Разрешения на использование специального отладчика Jupyter Notebook Debugger.

- Распознавайте файлы.ipynb по значку.

Для работы с Python в записных книжках Jupyter необходимо активировать среду Anaconda в VS Code или другую среду Python, в которой установлен пакет Jupyter. Для выбора среды используйте команду Python: Select Interpreter из командной палитры (Ctrl+Shift+P).

После активации соответствующей среды можно создать и открыть записную книжку Jupyter, подключиться к удаленному серверу Jupyter для запуска ячеек кода и экспортировать записную книжку Jupyter в виде файла Python.