Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины «Искусственный интеллект и машинное обучение»

Выполнил:
Неутолимов Дмитрий Сергеевич
2 курс, группа ИВТ-б-о-23-2,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Доцент департамента цифровых, робототехнических систем и электроники института перспективной инженерии Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2025 г

Тема: Работа с Jupyter Notebook, JupyterLab и Google Colab.

Цель: исследовать базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка программирования Python.

Ссылка на репозиторий: https://github.com/SiXTeeN100/AIaML

Порядок выполнения работы:

Задание 1. Работа с ячейками Markdown

Создал новую Markdown-ячейку, в которой написал заголовок, добавил жирный и курсивный текст, создал нумерованный и маркированный списки, вставил формулу согласно индивидуального задания №9 и вставил изображение.

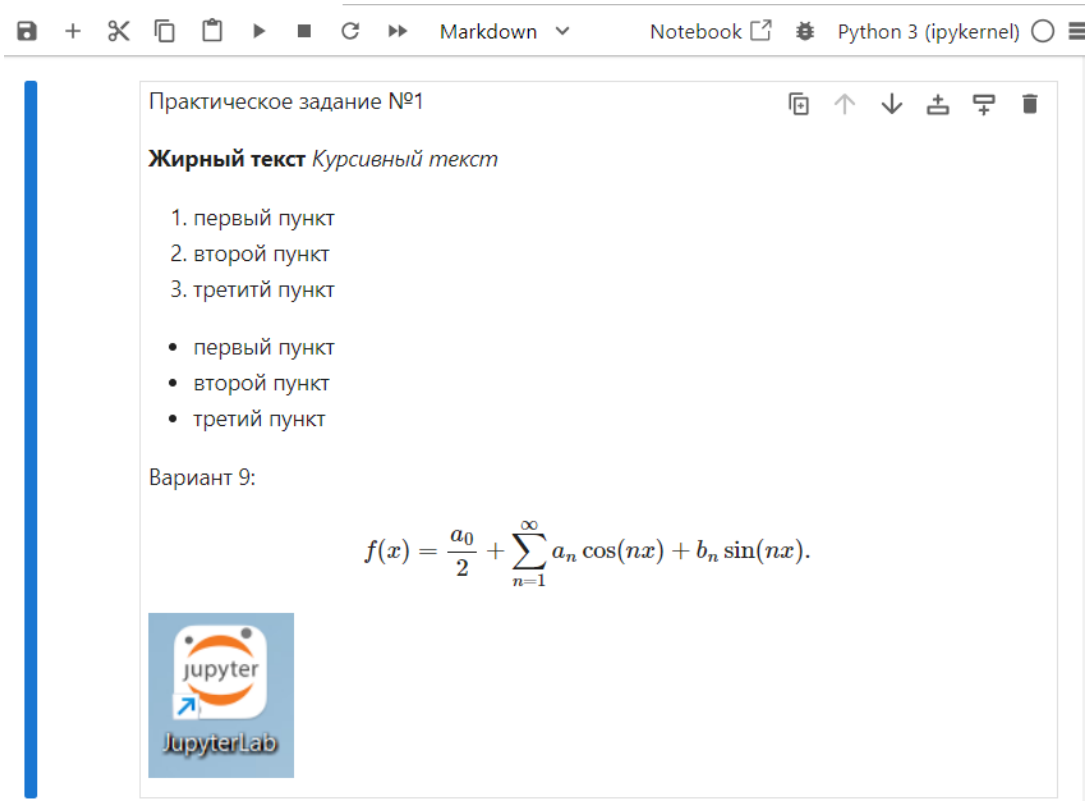

Рисунок 1 – Ячейка Markdown

Создал ячейку Python-кода, где запросил у пользователя его имя и вывел приветствие.

```
3]: name = input("Введите имя: ")
    print(f"Привет, {name}! Добро пожаловать в JupyterLab!")

Введите имя:  Дмитрий
Привет, Дмитрий! Добро пожаловать в JupyterLab!
```

Рисунок 2 – Ячейка Python-кода

Задание 2. Работа с файлами

Создал и сохранил текстовый файл с помощью open(), записал в него несколько строк текста, закрыл файл и затем открыл его снова, считав содержимое и выведя на экран. Проверил, существует ли файл, используя os.path.exists(). Удалил файл с помощью модуля os.

```
import os

with open("example.txt", "w") as f:
    f.write("Первая строка\n")
    f.write("2-ая строка\n")
    f.write("Third line\n")

with open("example.txt", "r") as f:
    content = f.read()
    print("Содержимое файла:\n")
    print(content)

print("Файл существует:", os.path.exists("example.txt"))

os.remove("example.txt")
print("Файл удален.")
```

```
Содержимое файла:

Первая строка
2-ая строка
Third line

Файл существует: True
Файл удален.
```

Рисунок 3 – Работа с текстовым файлом

Задание 3. Магические команды Jupyter

Вывел список всех доступных магических команд с помощью %lsmagic. Использовал %time и %%timeit для измерения времени выполнения кода. Создал Python-скрипт в Jupyter с помощью %%writefile script.py и выполню его через !python script.py. Вывел список файлов в текущей директории с помощью %ls. Использовал %history для просмотра истории команд.

[2]:
```
print("Список магических команд:")
%lsmagic
```

```
Список магических команд:
```

[2]: root
- line
- cell

Find...

[4]:
```
print("Измерение времени выполнения кода:")

print("Время выполнения одной строки кода:")
%time sum(range(1000))
```

```
Измерение времени выполнения кода:
Время выполнения одной строки кода:
CPU times: total: 0 ns
Wall time: 0 ns
```

[4]:
```
499500
```

[5]:
```
%%timeit
total = 0
for i in range(1000):
    total += i
```

```
21.7 µs ± 308 ns per loop (mean ± std. dev. of 7 runs, 10,000 loops each)
```

*[11]:
```
%%writefile test_script.py
for i in range(3):
    print(f"Итерация {i}")

!python test_script.py
```

```
Overwriting test_script.py
```

[12]:
```
print("Список файлов в текущей директории:")
%ls
```

Рисунок 4 – Магические команды Jupyter

```
Список файлов в текущей директории:
 ’®¬ ў гбва®®бвўЃ С Ё¬ЃЃв ¬Гв€r Windows
 ‘ГаЁ®л® ®¬Ґа в®¬ : 52C7-6EF6

 ‘®нГа¦Ё¬®Г Ї ЇЄЁ C:\Users\„¬ЁваЁ®

27.02.2025  23:12    <DIR>          .
26.02.2025  15:19    <DIR>          ..
27.02.2025  21:49    <DIR>          .anaconda
27.02.2025  21:58    <DIR>          .conda
27.02.2025  21:49               146 .condarc
27.02.2025  21:48    <DIR>          .continuum
07.09.2023  22:48    <DIR>          .dotnet
27.02.2025  23:00    <DIR>          .ipynb_checkpoints
27.02.2025  22:13    <DIR>          .ipython
27.02.2025  21:57    <DIR>          .jupyter
21.11.2024  13:46    <DIR>          .matplotlib
22.12.2023  22:04               192 .packettracer
07.09.2023  22:56    <DIR>          .templateengine
01.12.2024  00:57    <DIR>          .VirtualBox
04.09.2024  20:57    <DIR>          .vscode
22.12.2023  22:07    <DIR>          Cisco Packet Tracer 8.2.1
26.02.2025  15:25    <DIR>          Contacts
27.02.2025  22:21    <DIR>          Desktop
26.02.2025  15:25    <DIR>          Documents
27.02.2025  21:00    <DIR>          Downloads
26.02.2025  15:25    <DIR>          Favorites
26.02.2025  15:25    <DIR>          Links
26.02.2025  15:25    <DIR>          Music
27.02.2025  21:33    <DIR>          OneDrive
26.02.2025  15:25    <DIR>          Pictures
26.02.2025  15:25    <DIR>          Saved Games
27.02.2025  23:10                60 script.py
26.02.2025  15:25    <DIR>          Searches
08.09.2023  12:44    <DIR>          source
12.09.2023  14:10                 0 Sti_Trace.log
27.02.2025  22:31            11я671 Task1.ipynb
27.02.2025  23:12             8я357 Task2.ipynb
27.02.2025  23:12               105 test_script.py
```

Рисунок 5 – Магические команды Jupyter

```
27.02.2025  22:53             1я692 Untitled.ipynb
26.02.2025  15:50    <DIR>          Videos
23.09.2024  20:48    <DIR>          VirtualBox VMs
               8 д ®«®ў         22я223 Ў ®в
              28 Ї Ї®Є  45я411я401я728 Ў ®в бў®ў®д®
```

```
[13]: print("История команд:")
      %history
```

```
История команд:
print("Список магических команд:")
%lsmagic

print("Измерение времени выполнения кода:")

print("Время выполнения одной строки кода:")
%time sum(range(1000))

print("Время выполнения блока кода:")
%%timeit
total = 0
for i in range(1000):
    total += i

print("Создание и выполнение скрипта:")
%%writefile script.py
print("Привет из script.py!")

!python script.py

print("Список файлов в текущей директории:")
%ls

print("История команд:")
%history
print("Список магических команд:")
%lsmagic
print("Измерение времени выполнения кода:")
```

Рисунок 6 – Магические команды Jupyter

```
for i in range(1000):
    total += i
print("Создание и выполнение скрипта:")
%%writefile script.py
print("Привет из script.py!")

!python script.py
%%writefile script.py
print("Привет из script.py!")

!python script.py
%%writefile test_script.py
for i in range(3):
print(f"Итерация {i}")

!python test_script.py
%%writefile test_script.py
for i in range(3):
    print(f"Итерация {i}")

!python test_script.py
%%writefile test_script.py
for i in range(3):
    print(f"Итерация {i}")

!python test_script.py
%%writefile test_script.py
for i in range(3):
    print(f"Итерация {i}")

!python test_script.py
print(test_script.py)
print("Список файлов в текущей директории:")
%ls
print("История команд:")
%history
```

Рисунок 7 – Магические команды Jupyter

Задание 4. Взаимодействие с оболочкой системы

Вывел список файлов в текущей директории с помощью !ls. Проверил, какой Python используется, с помощью !which python. Создал папку test_folder с помощью !mkdir test_folder и убедился, что она появилась. Переместил файл в новую папку и удалил его. Очистил вывод в ячейке с помощью !clear.

```
print("Список файлов в текущей директории:")
%ls
```

```
Список файлов в текущей директории:
 ’®¬ ў гбва®©бвўЃ С Ё¬ЃЃв ¬ЃвЄг Windows
 ‘ГаЁ©л© ®¬Ґа в®¬ : 52C7-6EF6

 ‘®¤Ґа¦Ё¬®Ґ Ї ЇЄЁ C:\Users\„¬ЁваЁ©

05.03.2025  18:14    <DIR>          .
26.02.2025  15:19    <DIR>          ..
27.02.2025  21:49    <DIR>          .anaconda
27.02.2025  21:58    <DIR>          .conda
27.02.2025  21:49               146 .condarc
27.02.2025  21:48    <DIR>          .continuum
07.09.2023  22:48    <DIR>          .dotnet
27.02.2025  23:47    <DIR>          .ipynb_checkpoints
27.02.2025  22:13    <DIR>          .ipython
27.02.2025  21:57    <DIR>          .jupyter
21.11.2024  13:46    <DIR>          .matplotlib
22.12.2023  22:04               192 .packettracer
07.09.2023  22:56    <DIR>          .templateengine
01.12.2024  00:57    <DIR>          .VirtualBox
04.09.2024  20:57    <DIR>          .vscode
04.03.2025  22:00    <DIR>          BrawlhallaReplays
22.12.2023  22:07    <DIR>          Cisco Packet Tracer 8.2.1
26.02.2025  15:25    <DIR>          Contacts
04.03.2025  23:56    <DIR>          Desktop
26.02.2025  15:25    <DIR>          Documents
02.03.2025  17:52    <DIR>          Downloads
26.02.2025  15:25    <DIR>          Favorites
26.02.2025  15:25    <DIR>          Links
26.02.2025  15:25    <DIR>          Music
05.03.2025  14:59    <DIR>          OneDrive
26.02.2025  15:25    <DIR>          Pictures
```

Рисунок 8 – Вывод списка файлов с помощью %ls

```python
import sys
print(sys.executable)
```

```
C:\Users\Дмитрий\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\python.exe
```

```python
import os

if not os.path.exists("test_folder"):
    os.makedirs("test_folder")
    print(f"Папка '{folder_name}' создана.")
else:
    print(f"Папка '{folder_name}' уже существует.")
```

```
Папка 'test_folder' уже существует.
```

```python
import shutil
import os


file = "test_file.txt"
folder = "test_folder"

if os.path.exists(file):
    shutil.move(file, folder)
    print(f"Файл {file} перемещён в {folder}")
else:
    print(f"Файл {file} не найден!")

# Удаление файла после перемещения
moved_file_path = os.path.join(folder, "test_file.txt")
if os.path.exists(moved_file_path):
    os.remove(moved_file_path)
    print(f"Файл {moved_file_path} удалён.")
else:
    print(f"Файл {moved_file_path} не найден для удаления.")
```

```
Файл test_file.txt перемещён в test_folder
Файл test_folder\test_file.txt удалён.
```

Рисунок 9 – Проверка версии Python, создание папки, перемещение и удаление файла

Задание 5. Работа с Google Drive в Google Colab

Подключил Google Drive к Colab, создал и сохранил текстовый файл в Google Drive, прочитал файл из Google Drive, создал и сохранил CSV-файл вручную, используя Microsoft Excel.

```python
from google.colab import drive
drive.mount('/content/drive')

file_path = "/content/drive/MyDrive/my_text_file.txt"
with open(file_path, "w") as f:
  f.write("Это тестовый файл, сохраненный в Google Drive.\n")
  f.write("Вторая строка текста.")
print("Файл успешно сохранен в Google Drive.")

with open(file_path, "r") as f:
  content = f.read()
print("Содержимое файла:\n", content)

students = [
  ["Иванов И.И.", 20, "ИВТ-101"],
  ["Петров П.П.", 22, "ИВТ-102"],
  ["Сидорова А.А.", 21, "ИВТ-103"]
]
csv_path = "/content/drive/MyDrive/students.csv"
with open(csv_path, "w") as f:
  for student in students:
    f.write(",".join(map(str, student)) + "\n")
print("Файл students.csv успешно сохранен в Google Drive.")
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
Файл успешно сохранен в Google Drive.
Содержимое файла:
 Это тестовый файл, сохраненный в Google Drive.
Вторая строка текста.
Файл students.csv успешно сохранен в Google Drive.
```

Рисунок 10 – Работа с Google Drive в Google Colab

Рисунок 11 – Созданный CSV-файл

Ответы на контрольные вопросы:

1. Какие основные отличия JupyterLab от Jupyter Notebook?

JupyterLab и Jupyter Notebook — это инструменты для работы с интерактивными тетрадями, но JupyterLab является их более продвинутой версией. Вот ключевые различия:

1. Интерфейс и организация работы:
   - Jupyter Notebook: представляет собой отдельные веб-страницы с линейной последовательностью ячеек. Окружение ограничено одной тетрадью.
   - JupyterLab: это полноценная интегрированная среда (IDE) с вкладками, панелями, файловым менеджером, терминалом и редактором кода. Можно открывать и редактировать несколько файлов одновременно.
2. Работа с файлами:
   - В Jupyter Notebook можно работать только с .ipynb-файлами.

- В JupyterLab можно работать с разными типами файлов: .ipynb, .py, .csv, .md и даже .json, .yaml и .txt.

3. Многозадачность:
   - В Jupyter Notebook приходится открывать несколько вкладок браузера для работы с разными тетрадями.
   - В JupyterLab можно работать с несколькими тетрадями в одном окне благодаря системе вкладок и разделению экрана.
4. Поддержка расширений:
   - Jupyter Notebook поддерживает расширения, но их сложнее настраивать.
   - JupyterLab имеет встроенный менеджер расширений, что делает добавление новых функций (например, поддержки дополнительных языков программирования, тем и плагинов) более удобным.
5. Гибкость интерфейса:
   - В Jupyter Notebook фиксированный интерфейс.
   - В JupyterLab можно изменять расположение панелей, настраивать вид и работать в нескольких окнах.
6. Поддержка терминала:
   - В Jupyter Notebook запуск терминала требует отдельных шагов.
   - В JupyterLab есть встроенный терминал, позволяющий выполнять команды прямо в интерфейсе.
7. Производительность:
   - JupyterLab может потреблять больше ресурсов, особенно если открыто много вкладок.
   - Jupyter Notebook проще и легче, но менее функционален.

2. Как создать новую рабочую среду (ноутбук) в JupyterLab?

1. В меню выбрать File → New → Notebook
2. Выбрать доступно ядро (обычно Python)
3. Откроется новая тетрадь, состоящая из ячеек (Cells)

3. Какие типы ячеек поддерживаются в JupyterLab и как их переключать?

Код (Code) – для написания и выполнения программного кода.

Текст (Markdown) – используется для оформления пояснений, форматированного текста и математических формул на основе LaTeX.

Вывод (Raw) – предназначен для хранения необработанного текста, например, для экспорта в другие форматы.

4. Как выполнить код в ячейке и какие горячие клавиши для этого используются?

Ctrl+Enter или Shift+Enter. В первом случае введенный вами код будет выполнен интерпретатором Python, во втором – будет выполнен код и создана новая ячейка, которая расположится уровнем ниже.

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

Терминал доступен через File → New → Terminal, используется для выполнения команд оболочки.

6. Какие инструменты JupyterLab позволяют работать с файлами и структурами каталогов?

Используется файловый браузер для навигации по каталогам и управления файлами. Также доступны команды терминала для работы с файловой системой.

7. Как можно управлять ядрами (kernels) в JupyterLab?

В меню "Kernel" можно перезапустить, остановить или сменить ядро. Также можно управлять ядрами через терминал.

8. Каковы основные возможности системы вкладок и окон в интерфейсе JupyterLab?

JupyterLab позволяет открывать несколько вкладок и окон для одновременной работы с разными файлами и инструментами. Вкладки можно перетаскивать и организовывать по своему усмотрению.

9. Какие магические команды можно использовать в JupyterLab для измерения времени выполнения кода? Приведите примеры

Можно использовать %%time и %timeit.

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

%timeit запускает переданный ей код 100000 раз (по умолчанию) и выводит информацию о среднем значении трех наиболее быстрых прогонах.

10. Какие магические команды позволяют запускать код на других языках программирования в JupyterLab?

Можно использовать %% с указанием языка, например, %%javascript.

11. Какие основные отличия Google Colab от JupyterLab?

Google Colab работает в облаке и предоставляет доступ к GPU и TPU бесплатно.

Colab интегрирован с Google Drive и другими сервисами Google.

JupyterLab требует локальной установки или сервера.

12. Как создать новый ноутбук в Google Colab?

Перейти в Файл → Новый ноутбук. Откроется рабочая область с первой ячейкой.

13. Какие типы ячеек доступны в Google Colab, и как их переключать?

Код (Code) – для написания и выполнения Python-кода.

Текст (Markdown) – используется для оформления документации, пояснений и формул (LaTeX).

14. Как выполнить код в ячейке Google Colab и какие горячие клавиши для этого используются?

Запустить ячейку	Shift + Enter
Добавить новую ячейку ниже	Ctrl + M B
Удалить текущую ячейку	Ctrl + M D
Изменить тип ячейки на Markdown	Ctrl + M N

Изменить тип ячейки на код Ctrl + M Y

15. Какие способы загрузки и сохранения файлов поддерживает Google Colab?

Файлы можно загружать вручную через боковую панель (Файлы → Загрузить файлы) или с помощью Python. Команда files.upload() для загрузки файлов с локального компьютера. Для сохранения файлов можно использовать команду files.download().

16. Как можно подключить Google Drive к Google Colab и работать с файлами?

Google Colab позволяет работать с файлами на Google Диске и загружать файлы в локальное окружение.

from google.colab import drive

drive.mount('/content/drive')

После выполнения появится ссылка, по которой нужно авторизоваться.

17. Какие команды используются для загрузки файлов в Google Colab из локального компьютера?

Файлы можно загружать вручную через боковую панель

from google.colab import files

uploaded = files.upload()

18. Как посмотреть список файлов, хранящихся в среде Google Colab?

Можно использовать команду !ls в ячейке кода для просмотра файлов в текущем каталоге.

19. Какие магические команды можно использовать в Google Colab для измерения выполнения кода? Приведите примеры

Можно использовать %timeit для измерения времени выполнения кода и %%time для измерения времени выполнения всей ячейки.

%timeit sum(range(1000))

%%time

total = sum(range(10**6))

20. Как можно изменить аппаратные ресурсы в Google Colab (например, переключиться на GPU)?

Перейти в Среда выполнения → Изменить среду выполнения.

В поле Аппаратный ускоритель выбрать:

GPU (для графического ускорения)

TPU (для ускорения в TensorFlow)

Нажать Сохранить.

Вывод: в ходе лабораторной работы были исследованы базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка программирования Python. Была произведена работа с ячейками Markdown, файлами, с Google Drive в Google Colab. Были изучены магические команды Jupyter, а также рассмотрено взаимодействие с оболочкой системы.