

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 ПРАКТИЧЕСКОЙ РАБОТЕ №1
дисциплины
«Искусственный интеллект и машинное обучение»
Вариант 14

Выполнил:
Степанов Артем 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.

1. Выполнил примеры из практической работы по Jupyter Notebook.

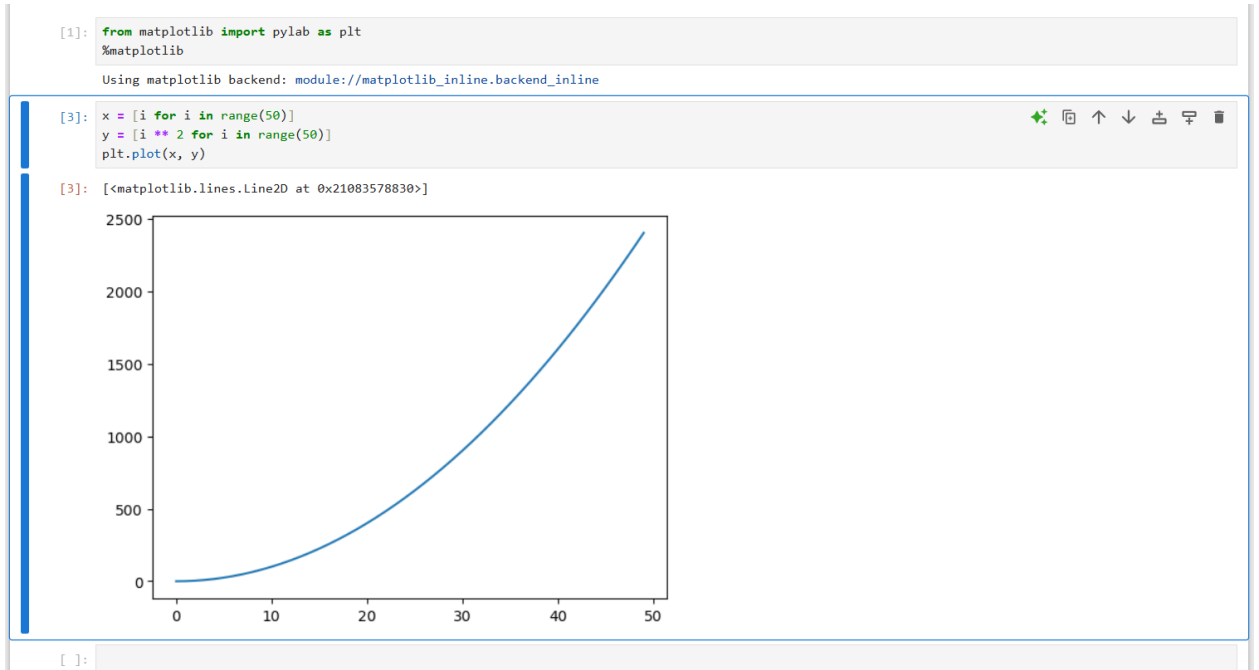


Рисунок 1 – Пример построения графика с использованием библиотеки matplotlib

The screenshot shows a Jupyter Notebook interface. The first cell contains the code: `%%time`, `i = 0`, `while i < 100000:`, `i += 1`, and `print('VSE')`. The output of the first cell is: `VSE`, `CPU times: total: 15.6 ms`, and `Wall time: 11 ms`.

Рисунок 2 – Пример использования команды %%time

2. Выполнил примеры из практической работы по Jupyter Lab.

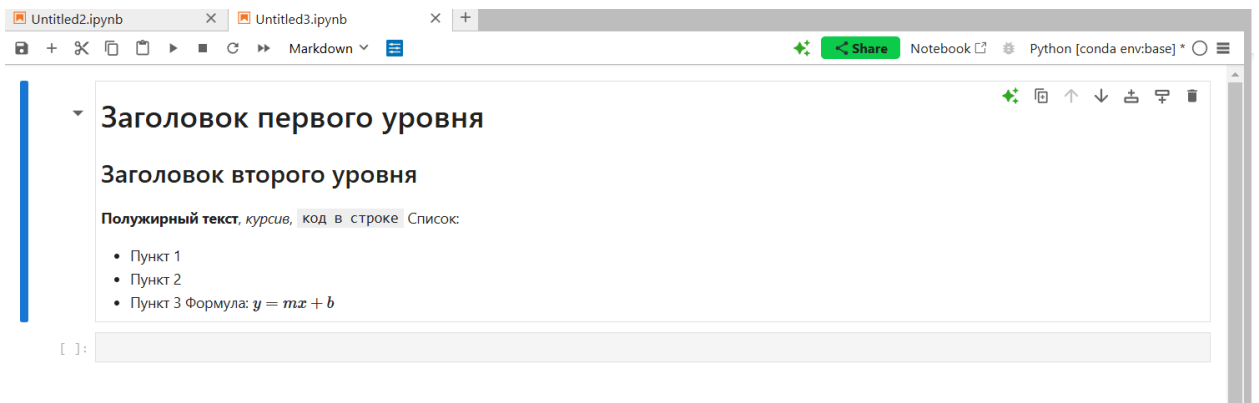


Рисунок 3 – Пример работы с markdown ячейками

```
[2]: import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 10, 100)
y = np.sin(x)
plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.title("График синусоиды")
plt.show()
```

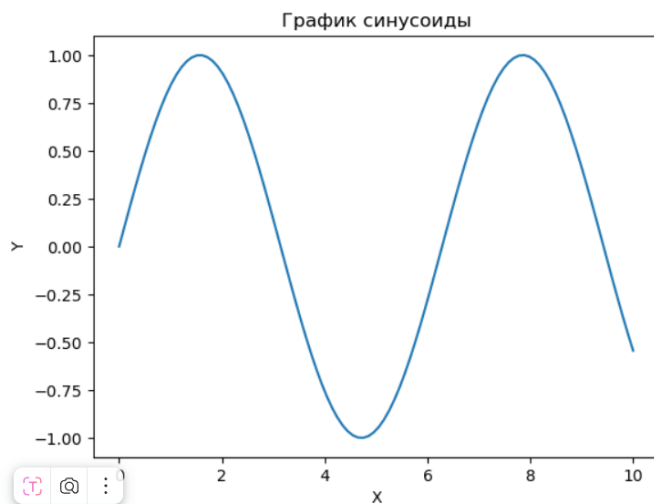


Рисунок 4 – Пример работы с библиотекой matplotlib и numpy для построения графика синусоиды

3. Выполнил примеры из практической работы по Google Colab

```
import torch
print(torch.cuda.is_available())
```

True

Рисунок 5 – Подключил аппаратное ускорение GPU

```
from google.colab import files
uploaded = files.upload()
```

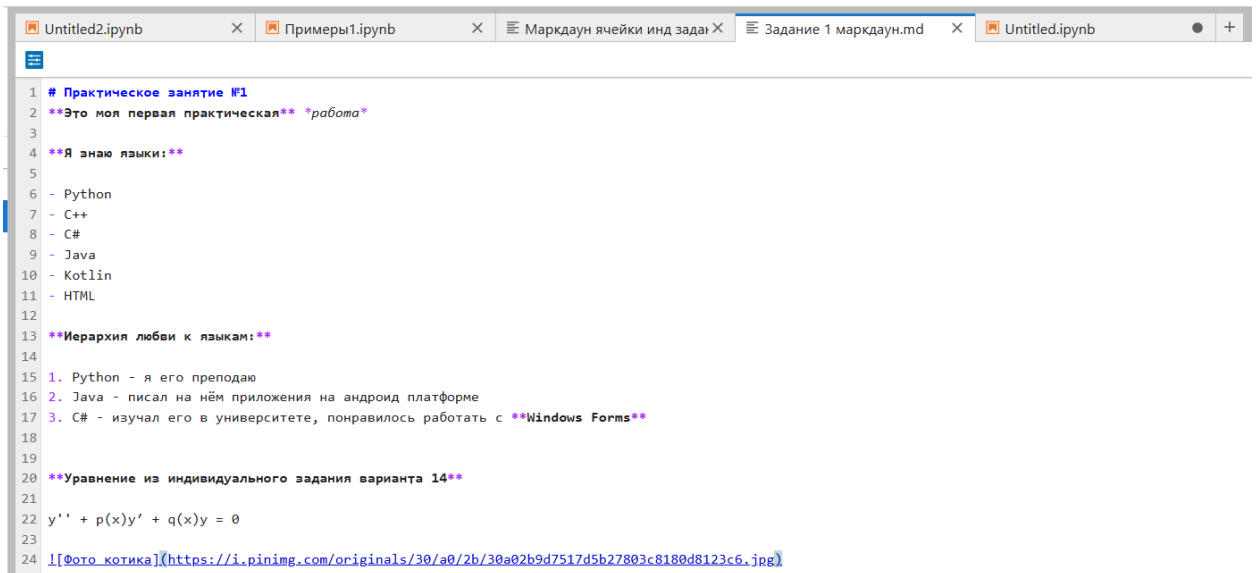
Выбрать файлы Бот телега.rar

- Бот телега.rar(n/a) - 3309060 bytes, last modified: 20.02.2022 - 100% done

Saving Бот телега.rar to Бот телега.rar

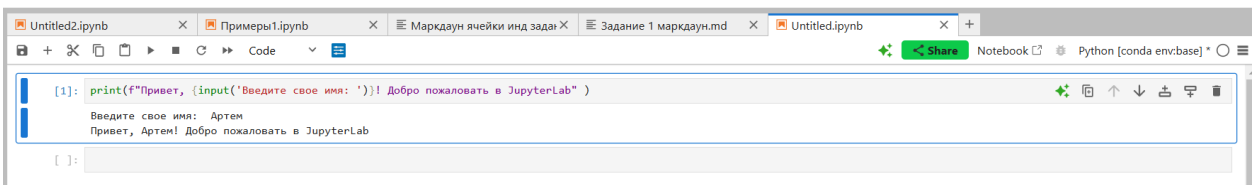
Рисунок 6 – Попробовал использовать работу с файлами и загрузил один из моих архивов

4. Индивидуальное задание



```
1 # Практическое занятие №1
2 **Это моя первая практическая** *работа*
3
4 **Я знаю языки:**
5
6 - Python
7 - C++
8 - C#
9 - Java
10 - Kotlin
11 - HTML
12
13 **Иерархия любви к языкам:**
14
15 1. Python - я его преподаю
16 2. Java - писал на нём приложения на андроид платформе
17 3. C# - изучал его в университете, понравилось работать с **Windows Forms**
18
19
20 **Уравнение из индивидуального задания варианта 14**
21
22  $y'' + p(x)y' + q(x)y = 0$ 
23
24 !@Фото котика(https://i.pinimg.com/originals/30/a0/2b/30a02b9d7517d5b27803c8180d8123c6.jpg)
```

Рисунок 7 – Создал Markdown-ячейку и выполнил индивидуальное задание согласно варианту 14



```
[1]: print(f"Привет, {input('Введите свое имя: ')}! Добро пожаловать в JupyterLab")
Введите свое имя: Артем
Привет, Артем! Добро пожаловать в JupyterLab
```

Рисунок 8 – Разработал консольное приложение, которое запрашивает имя пользователя и выводит приветствие

Ссылка на репозиторий GitHub:

<https://github.com/ArtemStepanovNkey/AI-university/tree/main>

Ответы на контрольные вопросы:

1. Основные отличия JupyterLab от Jupyter Notebook:

- **Интерфейс:** JupyterLab имеет современный интерфейс с поддержкой вкладок и панелей, в то время как Jupyter Notebook использует более простой интерфейс.
- **Множественные документы:** JupyterLab поддерживает одновременную работу с несколькими документами (ноутбуками, текстовыми файлами, терминалами и т.д.) в одном окне, что повышает продуктивность.
- **Расширяемость:** JupyterLab легче расширяется за счет установки плагинов, которые могут добавлять новые функции.
- **Файловая система:** JupyterLab предоставляет улучшенную работу с файловой системой, включая более удобный файловый менеджер.

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

- Откройте JupyterLab и в левой панели выберите "Launcher". Затем нажмите на значок "Notebook" в разделе "Notebook".

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

- JupyterLab поддерживает типы ячеек:
 - Код (Code)
 - Markdown
 - Заголовок (Heading)
- Вы можете переключать тип ячейки с помощью выпадающего меню в панели инструментов или используя горячие клавиши (например, **Y** для ячейки кода и **M** для ячейки markdown).

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

- Чтобы выполнить код в ячейке, можно нажать **Shift + Enter**, что запустит ячейку и перейдет к следующей. Также можно использовать **Ctrl + Enter**, чтобы выполнить ячейку, не переходя к следующей.

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

- Для запуска терминала или текстового редактора перейдите в "Launcher" и выберите соответствующую опцию, например, "Terminal" для терминала или "Text File" для текстового редактора.

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

- JupyterLab имеет встроенный файловый менеджер, который позволяет просматривать, загружать, удалять и организовывать файлы и каталоги. Инструменты для работы с файлами включают "File Browser" и "Text Editor".

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

- Ядра можно управлять через меню "Kernel" на панели инструментов, где можно выбрать следующее: перезапуск ядра, отключение, переключение на другое ядро. Также можно использовать команду "Kernel" в меню для изменения настроек.

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

- JupyterLab позволяет открывать несколько вкладок в одном окне, включая ноутбуки, текстовые файлы, терминалы и даже визуализации. Каждое содержимое может быть перемещено и организовано в панели, что позволяет более эффективно работать с несколькими файлами одновременно.

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

- **%time:** измеряет время выполнения одной команды.

```
%time sum(range(10000))
```

- **%timeit:** несколько запусков для более точного измерения времени.

```
%timeit sum(range(10000))
```

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

- **%bash:** для выполнения команд bash.

- **%python:** для выполнения кода на Python (вызывается по умолчанию).

- **%R:** для выполнения кода на R (если установлен необходимый пакет).

- **%octave:** для выполнения кода на Octave (при наличии соответствующего ядра).

11. Какие основные отличия Google Colab от JupyterLab?

- **Облачное решение:** Google Colab работает в облаке, что позволяет доступ к ноутбукам из любого места без установки ПО.

- **Интеграция с Google Drive:** Colab позволяет легко сохранять и загружать файлы из Google Drive.
- **Бесплатные ресурсы:** Colab предлагает бесплатный доступ к GPU и TPU.
- **Специфические библиотеки:** Colab поставляется с предустановленными библиотеками для машинного обучения, такими как TensorFlow и PyTorch.

12. Как создать новый ноутбук в Google Colab?

- Для создания нового ноутбука в Google Colab, нужно перейти на сайт [Google Colab](https://colab.research.google.com/), затем нажать кнопку "Файл" (File) в верхнем меню и выбрать "Создать новый блокнот" (New notebook).

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

- В Google Colab доступны два основных типа ячеек:
 - Ячейки кода (Code cells) - для выполнения Python-кода.
 - Ячейки текста (Text cells) - для написания поясняющих текстов с использованием разметки Markdown.
- Чтобы переключить тип ячейки, можно воспользоваться выпадающим меню в верхней части ячейки или нажать правую кнопку мыши на ячейке и выбрать соответствующий тип.

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

- Чтобы выполнить код в ячейке, необходимо нажать кнопку "Выполнить" (Run) справа от ячейки или использовать горячие клавиши **Shift + Enter** для выполнения текущей ячейки и перехода к следующей, или **Ctrl + Enter** для выполнения текущей ячейки без перехода.

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

- Google Colab поддерживает несколько способов загрузки и сохранения файлов:

- Загрузка файлов из локального компьютера с использованием встроенных инструментов.
- Сохранение файлов в Google Диск (Google Drive).
- Загрузка файлов из URL.
- Загружая данные из GitHub или других репозиторий.

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

- Для подключения Google Drive к Colab, выполните следующие действия:

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

- После выполнения этого кода появится ссылка для авторизации, после чего доступ к файлам в вашем Google Drive будет доступен по пути `/content/drive/MyDrive`.

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

- Используйте следующий код для загрузки файла из локального компьютера:

```
from google.colab import files uploaded = files.upload()
```

- После выполнения этого кода появится окно выбора файла.

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

- Для просмотра списка файлов можно использовать следующую команду: `!ls`

- Это выведет список файлов и папок в текущем каталоге. Для просмотра файлов в определенной директории можно указать путь, например: `!ls /content`.

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

- Для измерения времени выполнения кода можно использовать магические команды **%time** и **%timeit**:

- **%time** - измеряет время выполнения одной строки:

```
%time sum(range(1000000))
```

- **%timeit** - автоматически выполняет код несколько раз для получения более точных данных:

```
%timeit sum(range(1000000))
```

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

- Для изменения аппаратных ресурсов откройте меню "Среда выполнения" (Runtime) => "Изменить тип среды выполнения" (Change runtime type). Затем в выпадающем списке "Аппаратный ускоритель" (Hardware accelerator) выберите "GPU" или "TPU" по вашему выбору и нажмите "Сохранить".