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»
Вариант 12

Выполнил:
Ляш Денис Александрович
2 курс, группа ИТС-б-о-23-1,
11.03.02 «Инфокоммуникационные технологии и системы связи»,
направленность (профиль) «Инфокоммуникационные системы и сети», очная форма обучения

_____
(подпись)

Проверил:

Воронкин Р. А.
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2025 г.

Тема: Работа с Jupyter Notebook, JupyterLab и Google Colab

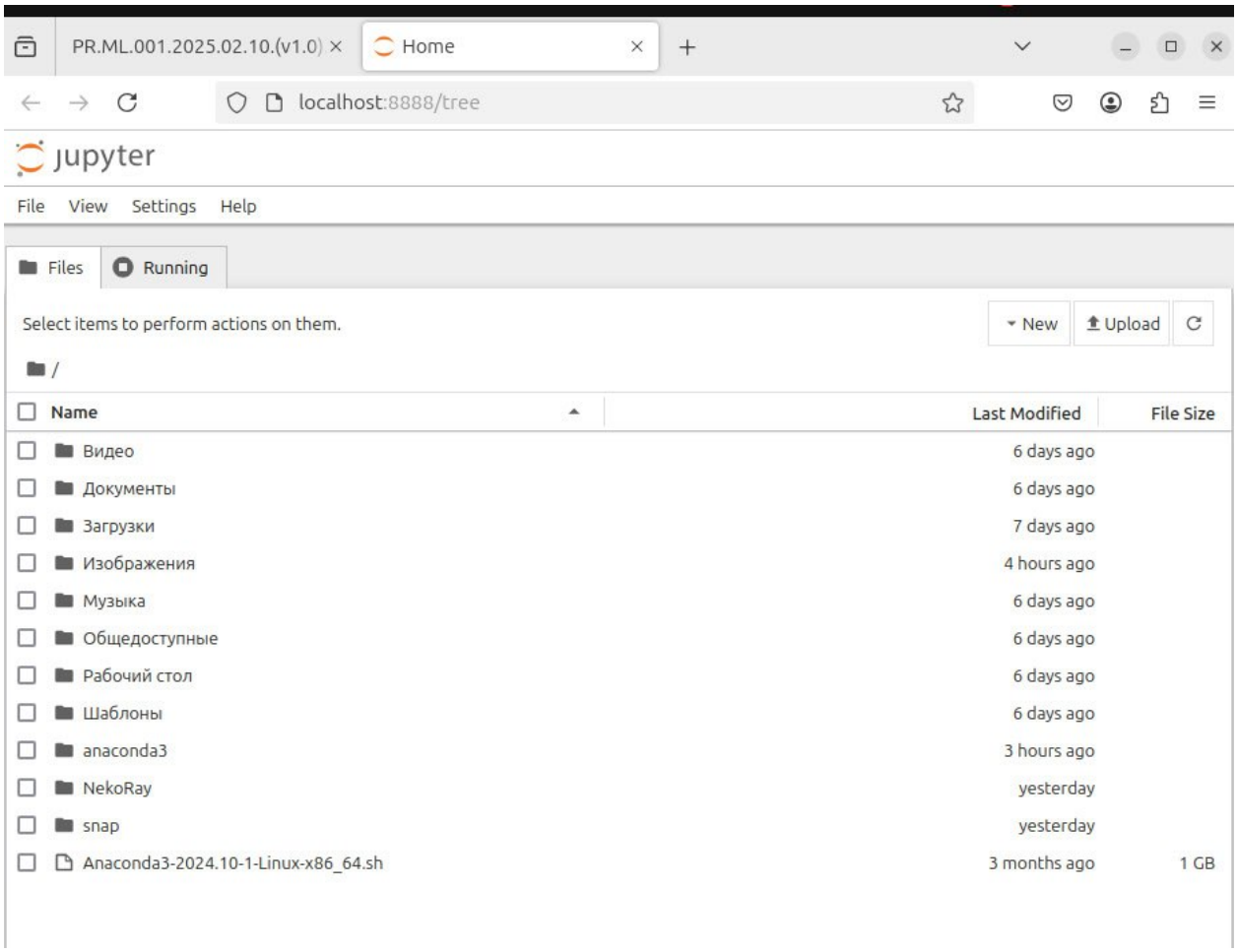
Цель: исследовать базовые возможности интерактивных оболочек Jupyter Notebook, JupyterLab и Google Colab для языка программирования Python.

Ссылка на репозиторий: https://github.com/Zloybanan4ik/lr1.git

Порядок выполнения работы:

1. Запуск Jupiter Notebook

Рисунок 1. Запуск Jupiter Notebook



2. Создание ноутбука

3. Создание ячейки в ноутбуке

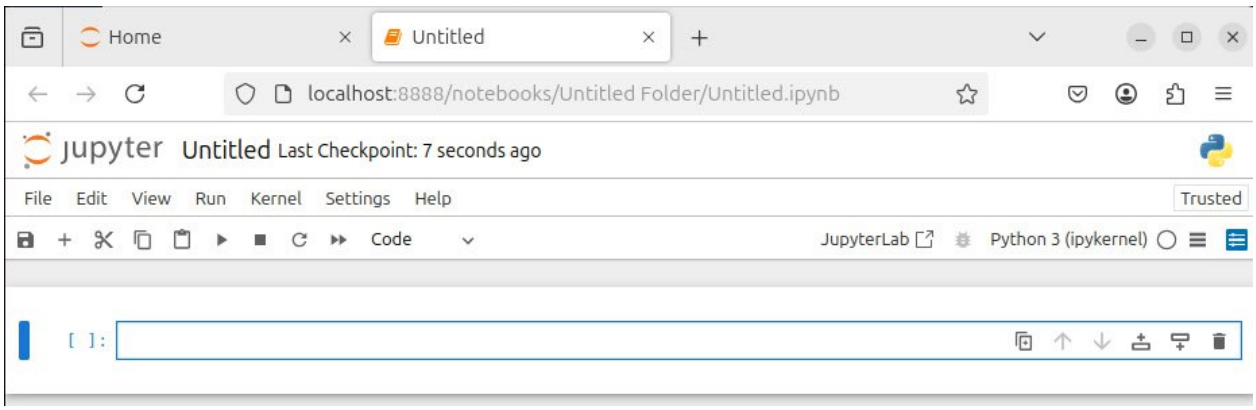


Рисунок 3. Ячейка в ноутбуке

4. Выполнение примеров

```
[1]: 3+2

[1]: 5

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

     12

[3]: n = 7
     for i in range(n):
         print(i * 10)

     0
     10
     20
     30
     40
     50
     60

[8]: 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. Примеры программ

5. Использование вывода изображений

jupyter lr1_Lyash Last Checkpoint: 1 minute ago

ile Edit View Run Kernel Settings Help

Code JupyterLab

[1]:
```
from matplotlib import pylab as plt
%matplotlib inline
```

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

[2]:
```
[<matplotlib.lines.Line2D at 0x2bdf5f907f0>]
```

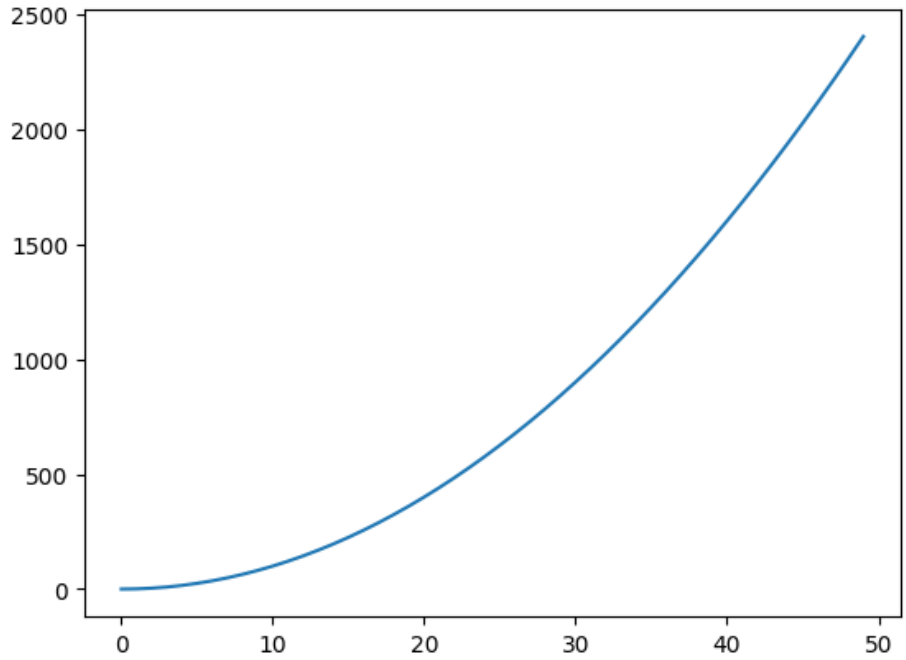


Рисунок 6. Вывод изображений

6. Использование Markdown ячеек

Заголовок первого уровня

Заголовок второго уровня

Полужирный текст, *курсив*, `код в строке`

Список:

- Пункт 1
- Пункт 2
- Пункт 3

Формула: $y = mx + b$

Рисунок 7. Markdown ячейка

7. Выполнение практической работы «Работа с ячейками и Markdown»

Практическое задание №1

Полужирный текст, *курсивный текст*

Нумерованный список:

1. Первый пункт
2. Второй пункт
3. Третий пункт

Маркированный список:

- Элемент A
- Элемент B
- Элемент C

12. Определение обратной матрицы (если детерминант ненулевой):

$$A^{-1} = \frac{1}{\det(A)}\text{adj}(A)$$

Изображение: 

```
[9]: # Ячейка Python-кода
name = input("Введите ваше имя: ")
print(f"Привет, {name}! Добро пожаловать в JupyterLab / Google Colab!")
```

```
Введите ваше имя:  Денис
Привет, Денис! Добро пожаловать в JupyterLab / Google Colab!
```

Рисунок 8. Практическая работа «Работа с ячейками и Markdown»

8. Выполнение практической работы «Работа с файлами»

```
[10]: import os

# 1. Создание и сохранение текстового файла
with open('example.txt', 'w', encoding='utf-8') as file:
    # 2. Запись нескольких строк текста
    file.write("Первая строка текста\n")
    file.write("Вторая строка текста\n")
    file.write("Третья строка текста\n")

# 3. Закрытие файла благодаря with и открытие для чтения
with open('example.txt', 'r', encoding='utf-8') as file:
    content = file.read()
    print("Содержимое файла:")
    print(content)

# 4. Проверка на существование файла
file_exists = os.path.exists('example.txt')
print(f"\nФайл существует: {file_exists}")

# 5. Удаление файла
if file_exists:
    os.remove('example.txt')
    print("Файл успешно удален")
    # Проверка после удаления
    print(f"Файл существует после удаления: {os.path.exists('example.txt')}")
else:
    print("Файл не найден для удаления")
```

```
Содержимое файла:
Первая строка текста
Вторая строка текста
Третья строка текста


Файл существует: True
Файл успешно удален
Файл существует после удаления: False
```

Рисунок 9. Практическая работа «Работа с файлами»

9. Выполнение практической работы «Магические команды Jupyter»

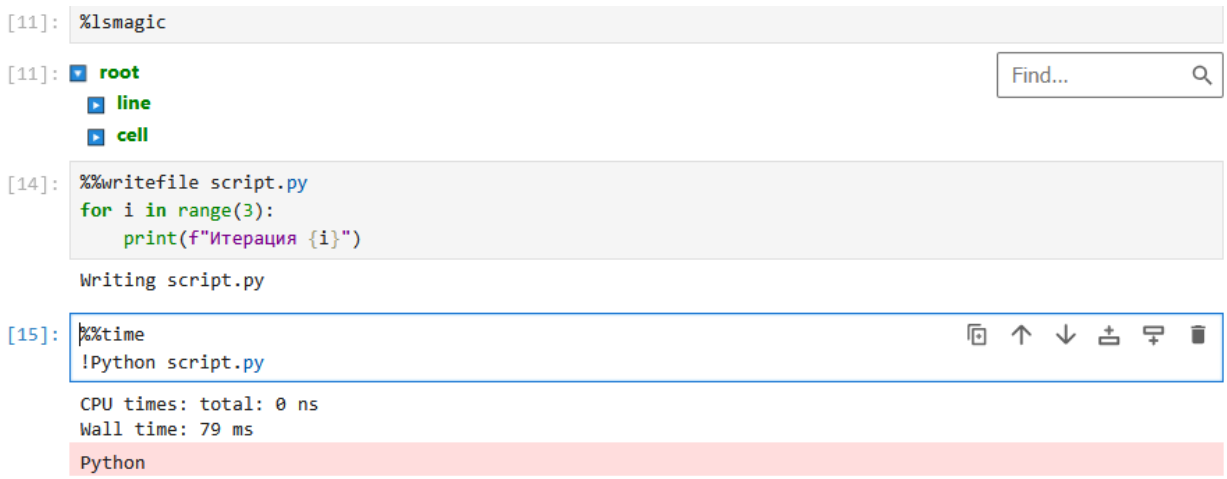


Рисунок 10. Практическая работа «Магические команды Jupyter»

10. Выполнение практической работы «Взаимодействие с оболочкой системы»

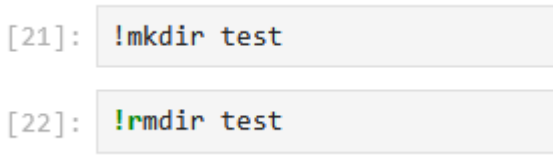


Рисунок 11. Практическая работа «Взаимодействие с оболочкой системы»

Рисунок 12. Созданная папка

11. Выполнение практической работы «Работа с Google Drive и Google Colab»

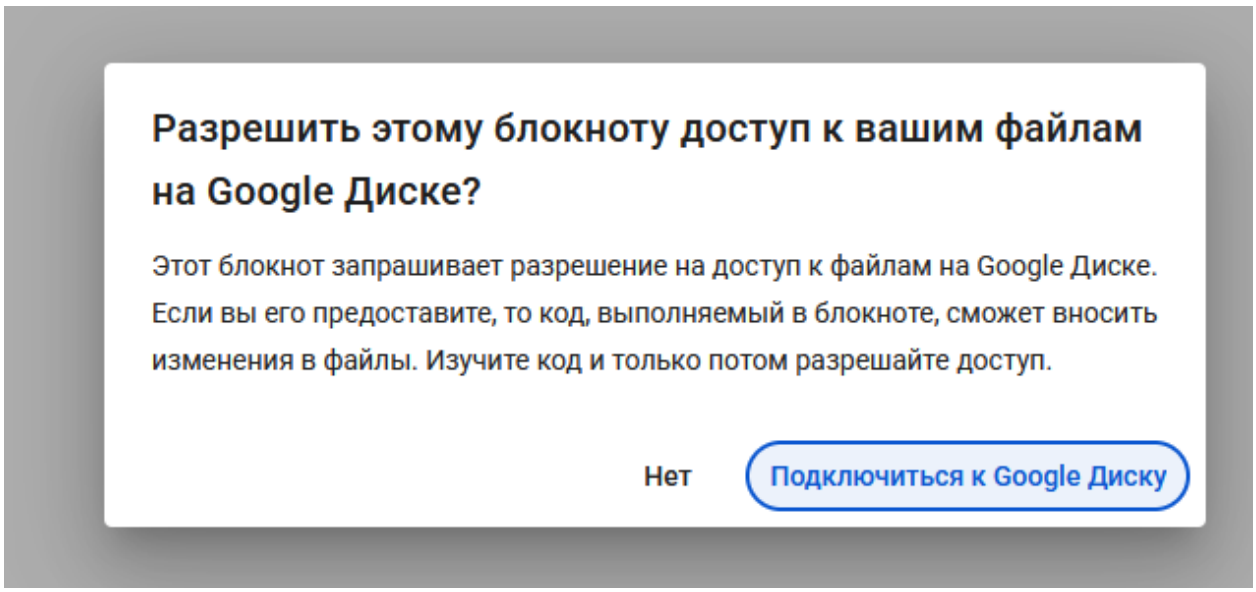


Рисунок 13. Подключение Google Drive к Colab

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

file_path = '/content/drive/MyDrive/my_text_file.txt'

with open(file_path, 'w', encoding='utf-8') as file:
    file.write("Первая строка текста\n")
    file.write("Вторая строка текста\n")
    file.write("Третья строка текста\n")

print("Файл успешно создан в Google Drive")

with open(file_path, 'r', encoding='utf-8') as file:
    content = file.read()
    print("\nСодержимое файла:")
    print(content)

import csv

csv_path = '/content/drive/MyDrive/students.csv'

students = [
    ['ФИО', 'Возраст', 'Группа'],
    ['Ляш Денис Александрович', 20, 'ИТС-б-о-23-1'],
    ['Борцов Богдан Михайлович', 21, 'ИТС-б-о-23-1'],
    ['Бакулин Вадим Романович', 19, 'ИТС-б-о-23-1']
]

with open(csv_path, 'w', encoding='utf-8', newline='') as file:
    writer = csv.writer(file)
    writer.writerows(students)

print("\nCSV-файл успешно создан в Google Drive")
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=
Файл успешно создан в Google Drive

Содержимое файла:
Первая строка текста
Вторая строка текста
Третья строка текста


CSV-файл успешно создан в Google Drive
```

Рисунок 15. Код для работы с файлами

	A	B	C
1	ФИО	Возраст	Группа
2	Ляш Денис Алексан	20	ИТС-б-о-23-1
3	Борцов Богдан Мих	21	ИТС-б-о-23-1
4	Бакулин Вадим Ром	19	ИТС-б-о-23-1

Рисунок 16. Созданная таблица

Ответы на контрольные вопросы:

1. Какие основные отличия JupyterLab от Jupyter Notebook?

В отличие от Jupyter Notebook, JupyterLab позиционируется как комплексная интегрированная среда разработки (IDE). Данная платформа консолидирует в едином пользовательском интерфейсе функциональные возможности Jupyter Notebook, текстового редактора, терминала и файлового менеджера.

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

Процедура создания новой рабочей среды (notebook) в JupyterLab включает в себя последовательный выбор пунктов меню "File", затем "New" и "Notebook". На следующем этапе необходимо определить требуемое ядро исполнения, как правило, Python. В результате выполнения указанных действий будет инициализирован новый notebook.

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

В среде JupyterLab предусмотрена поддержка трех типов ячеек, предназначенных для различных целей: "Code" – для ввода и исполнения программного кода, "Markdown" – для форматирования текстовых пояснений и документации, и "Raw" – для хранения необработанного текста. Для оперативного изменения типа ячейки предусмотрены сочетания клавиш быстрого доступа: клавиша "Y" для переключения в режим "Code" и клавиша "M" для перехода в режим "Markdown".

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

Чтобы выполнить код в ячейке, можно использовать горячие клавиши Shift+Enter (выполнить содержимое ячейки и перейти на ячейку ниже) или Alt+Enter (выполнить содержимое ячейки и вставить новую ячейку ниже).

5. Как запустить терминал или текстовый редактор внутри JupyterLab?

Для инициализации сессии терминала в среде JupyterLab необходимо последовательно выбрать пункты меню "File", затем "New" и "Terminal". Открытие текстового редактора осуществляется аналогичным образом: через пункты меню "File", "New" и "Text File".

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

Функциональность управления файлами и каталогами в JupyterLab реализована посредством файлового менеджера, визуально представляющего иерархическую структуру папок и файлов.

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

Операции, связанные с управлением ядрами исполнения (kernels), в JupyterLab доступны через меню "Kernel". Данный раздел предоставляет возможность перезапуска ядра, принудительного прерывания выполнения кода, а также подключения к другому доступному ядру.

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

Ключевыми особенностями системы вкладок и окон в JupyterLab является возможность одновременного открытия нескольких файлов, гибкая настройка расположения панелей, а также параллельная работа с несколькими notebooks в рамках одного окна, обеспечиваемая системой вкладок и функцией разделения экрана.

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

Для оценки времени выполнения фрагментов кода в JupyterLab предусмотрено использование так называемых "магических команд" %timeit и %%time. В качестве примера, команда %timeit sum(range(1000)) позволяет определить время, необходимое для выполнения функции sum(range(1000)).

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

Интегрированные в JupyterLab "магические команды", а именно %%bash, %%javascript, %%python и другие, обеспечивают возможность исполнения кода, написанного на различных языках программирования, непосредственно в рамках рабочей среды. В частности, команда %%bash предоставляет интерфейс для выполнения команд оболочки Bash.

11. Какие основные отличия Google Colab от JupyterLab?

Ключевое различие между Google Colab и JupyterLab заключается в следующем: Google Colab представляет собой облачную платформу, обеспечивающую бесплатный доступ к вычислительным ресурсам GPU и TPU. Данная среда интегрирована с сервисом Google Диск и предоставляет возможность работы с notebook-ами Jupyter Notebook на удаленных серверах. В свою очередь, JupyterLab является локальной средой разработки, характеризующейся расширенными возможностями конфигурирования и управления файлами.

12. Как создать новый ноутбук в Google Colab?

Для инициализации нового notebook в Google Colab необходимо перейти в меню "Файл" и выбрать пункт "Новый ноутбук". В результате выполнения указанных действий будет открыта рабочая область, содержащая первую ячейку для ввода кода.

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

В Google Colab предусмотрено два основных типа ячеек: "Code", предназначенные для написания и исполнения кода на языке Python, и "Markdown", используемые для оформления документации и текстовых

описаний. Для переключения между типами ячеек реализованы следующие сочетания клавиш: Ctrl+M Y – для активации режима "Code" и Ctrl+M M – для перехода в режим "Markdown".

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

Чтобы выполнить код в ячейке Google Colab, можно использовать горячие клавиши Shift+Enter. Также можно использовать Alt+Enter для выполнения кода и вставки новой ячейки ниже.

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

В Google Colab предусмотрена возможность загрузки файлов двумя способами: посредством графического интерфейса боковой панели (путь "Файлы => Загрузить файлы") или с использованием программной команды files.upload(). Сохранение файлов может быть осуществлено в облачное хранилище Google Диск, либо посредством экспорта в различные форматы, такие как .ipynb, .py и другие.

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

Для интеграции Google Drive с рабочей средой Google Colab необходимо последовательно выполнить следующие действия: импортировать модуль drive из библиотеки google.colab посредством команды from google.colab import drive, а затем осуществить монтирование диска с использованием команды drive.mount('/content/drive'). После выполнения указанных операций появляется возможность доступа и работы с файлами, расположенными на Google Диске.

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

Для загрузки файлов в Google Colab из локального компьютера используется команда from google.colab import files и files.upload().

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

Чтобы посмотреть список файлов в среде Google Colab, можно использовать команду !ls.

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

В среде Google Colab для оценки временных характеристик выполнения фрагментов кода предусмотрено использование "магических команд" %timeit и %%time. В качестве иллюстрации, команда %timeit sum(range(1000)) позволяет определить время, необходимое для выполнения операции суммирования элементов в диапазоне от 0 до 999.

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

Для изменения конфигурации аппаратных ресурсов, выделенных для сессии Google Colab (например, для активации GPU), необходимо выполнить следующие действия: перейти в меню "Среда выполнения", выбрать пункт "Изменить среду выполнения" и в поле "Аппаратный ускоритель" указать требуемый тип ускорителя, например, GPU или TPU.

Вывод: В рамках лабораторного исследования были проанализированы среды разработки: Jupyter Notebook, JupyterLab и Google Colab. В ходе работы изучены возможности использования магических команд для повышения эффективности работы в Anaconda, а также рассмотрена интеграция Google Drive с платформой Google Colab.