

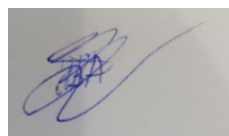
Министерство науки и высшего образования РФ  
Федеральное государственное автономное образовательное учреждение  
высшего образования «Национальный исследовательский технологический  
университет «МИСиС»  
Институт ИТКН  
Кафедра АСУ

## **ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**Выполнил:**

ст. гр. БИВТ-20-1

Янушка А.В.



**Проверил:**

доцент кафедры АСУ

Громов С.В.

Москва 2022

**Цели:**

Целью практики является изучение основ проведения сетевых научных исследований, овладение технологией и инструментами, используемыми в данной области.

**Задачи:**

1. Изучение основ Github и принципов работы с Jupiter.
2. Управление пакетами, окружениями.
3. Основы Continuous Integration
4. Изучение Docker Hub, Mybinder.org
5. Работа с Kaggle
6. Интеграция Kaggle и CI

**Ход работы:**

**Задание 1. Изучение основ Github и принципов работы с Jupiter.**

В процессе выполнения работы был установлен Jupiter Notebook, в котором была реализована задача гребневой регрессии. Результат был сохранен в качестве `ipynb` файла и загружен в репозиторий Github.

```

#импортируем модуль matplotlib.pyplot под именем plt
import matplotlib.pyplot as plt
#аналогично
import numpy as np
#из библиотеки sklearn импортируем модули datasets, linear_model
from sklearn import datasets, linear_model
#аналогично
from sklearn.metrics import mean_squared_error, r2_score

# загружаем набор данных по диабету
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)

# Use only one feature
diabetes_X = diabetes_X[:, np.newaxis, 2]

# разделяем данные на обучающие/тестовые
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

diabetes_y_train = diabetes_y[:-20]
diabetes_y_test = diabetes_y[-20:]

# создаем объект линейной регрессии
regr = linear_model.LinearRegression()

# обучение модели с помощью обучающего набора
regr.fit(diabetes_X_train, diabetes_y_train)

# Делаем прогнозы, используя тестовый набор
diabetes_y_pred = regr.predict(diabetes_X_test)

# строим точечный график
plt.scatter(diabetes_X_test, diabetes_y_test, color="black")
# строим линию среднего значения шириной 3
plt.plot(diabetes_X_test, diabetes_y_pred, color="blue", linewidth=3)

#устанавливаем тики и метки оси x так, чтобы их не было видно
plt.xticks(())
#аналогично для оси y
plt.yticks(())

#отображаем все открытые фигуры
plt.show()

```

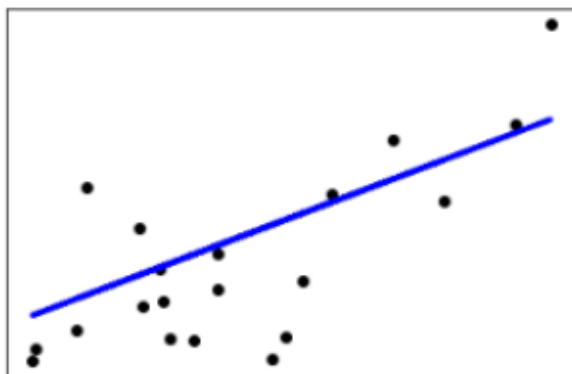


Рисунок 1. Код программы

```
from IPython.display import Image
from IPython.core.display import HTML
Image(url= "https://scikit-learn.org/stable/_images/sphx_glr_plot_ols_001.png")
```

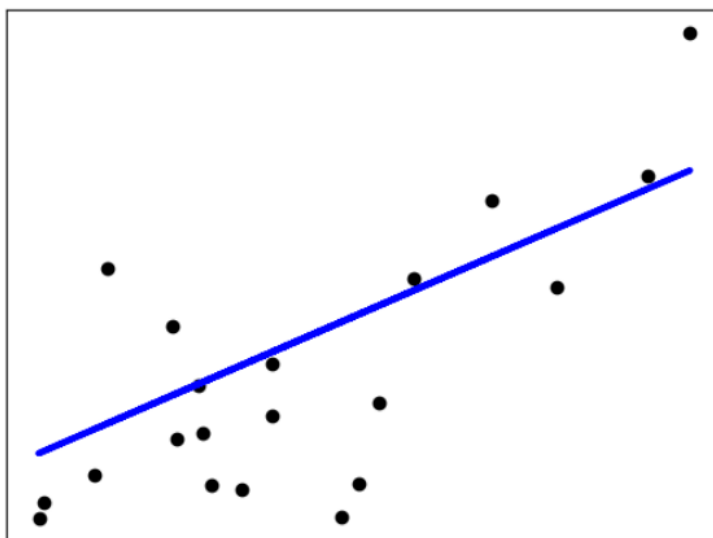


Рисунок 2. Вывод

В процессе выполнения работы был создан текстовый документ, содержащий скрипт и позволяющий конвертировать файл тетрадки в html формат.

Полученный блокнот был сохранен как bat файл. Скрипт был запущен с помощью командной строки, что позволило открывать тетрадку с решением первого домашнего задания в браузере. Командой одноклассников было проверено, что тетрадь автоматически запускается и выполняется в html.

**Рисунок 3. Содержания скрипта для конвертирования файла тетрадки в html**

```
C:\Users\Владелец\Desktop\практика>cn.bat start
```

**Рисунок 4. Запуск скрипта через командную строку**



```
In [136... #импортируем модуль matplotlib.pyplot под именем plt
import matplotlib.pyplot as plt
#аналогично
import numpy as np
#из библиотеки sklearn импортируем модули datasets, linear_model
from sklearn import datasets, linear_model
#аналогично
from sklearn.metrics import mean_squared_error, r2_score

# загружаем набор данных по диабету
diabetes_X, diabetes_y = datasets.load_diabetes(return_X_y=True)

# Use only one feature
diabetes_X = diabetes_X[:, np.newaxis, 2]

# разделяем данные на обучающие/тестовые
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

diabetes_y_train = diabetes_y[:-20]
diabetes_y_test = diabetes_y[-20:]

# создаем объект линейной регрессии
regr = linear_model.LinearRegression()

# обучение модели с помощью обучающего набора
regr.fit(diabetes_X_train, diabetes_y_train)

# Делаем прогнозы, используя тестовый набор
diabetes_y_pred = regr.predict(diabetes_X_test)

# строим точечный график
plt.scatter(diabetes_X_test, diabetes_y_test, color="black")
# строим линию среднего значения шириной 3
plt.plot(diabetes_X_test, diabetes_y_pred, color="blue", linewidth=3)

#устанавливаем титки и метки оси x так, чтобы их не было видно
plt.xticks(())
#аналогично для оси y
plt.yticks(())

#отображаем все открытые фигуры
```

### Задание 3. Основы Continuous Integration

В процессе выполнения работы был создан сценарий и сохранен как yaml файл. С помощью Jupyter Actions был запущен workflow с возможностью скачать вывод в виде html файла.

В папках был создан файл main.yaml, где подробно описан сценарий:

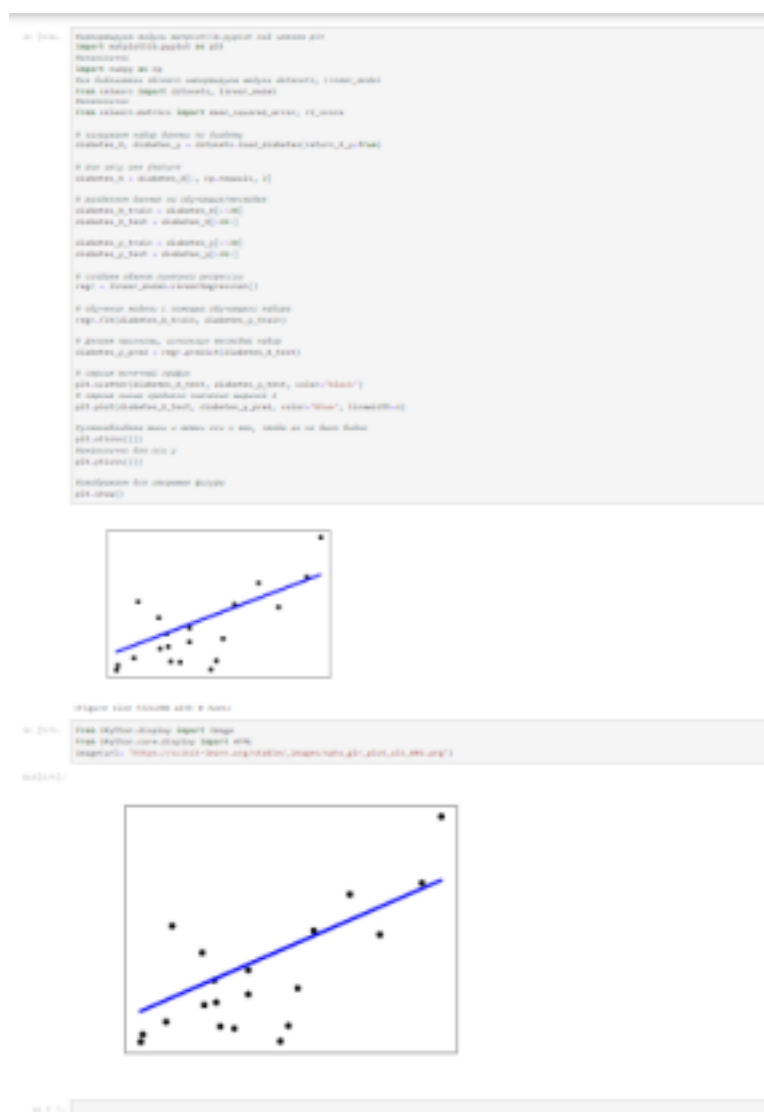
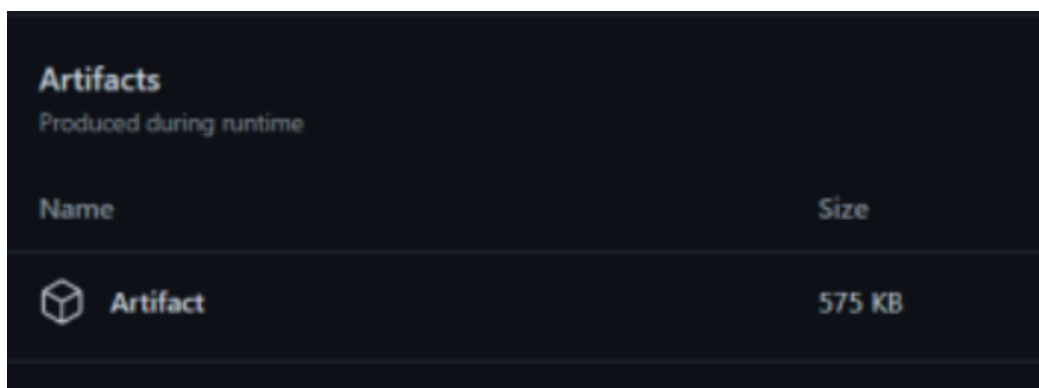


Рисунок 8. Содержание main.yaml файла

После создания файла в раздел actions в Github был запущен workflow. После запуска на компьютер автоматически скачивается архив, в котором содержится тетрадка в html формате:



The screenshot shows the 'Artifacts' section of a GitHub Actions workflow. It indicates that artifacts were 'Produced during runtime'. Below this, there is a table with two columns: 'Name' and 'Size'. A single artifact is listed with the name 'Artifact' (preceded by a cube icon) and a size of '575 KB'.


Artifacts	
Produced during runtime	
Name	Size
 Artifact	575 KB

Рисунок 9. Скачивание вывода в виде html

### **Заключение:**

В ходе выполнения заданий учебной практики были приобретены новые навыки работы с различными сервисами и продуктами, такими как Github Actions, была изучена работа с репозиторием и способы создания скриптов и сценариев.



**Список Литературы:**

1. Документация JupyterDocumentation [Электронный ресурс]. Режим доступа: <https://jupyter.org/documentation> (дата обращения 5.07.2022)
2. Статья Configure GitHub Actions [Электронный ресурс]. Режим доступа: <https://docs.docker.com/ci-cd/github-actions/> (дата обращения 13.07.2022)
- 3.

Документация Docker [Электронный ресурс]. Режим доступа:  
<https://docs.docker.com/> (дата обращения 17.07.2022)