

# Визуализация данных

---

Для вопросов по курсу:

Иванов Дмитрий Владимирович, [dmitry.ivanov@moevm.info](mailto:dmitry.ivanov@moevm.info)

Префикс в теме письма [CS\_23XX]

# Язык программирования Python. Основы



**Переменная** хранит в себе некоторое значение. Это значение может быть любым: число, строка, список чисел, ....

**Функция** (в программировании) - это программный код, к которому можно обратиться по имени. Мы можем вызвать функцию с параметрами, тем самым изменив ее поведение.

Например, в Python есть функция вывода на экран **print()**. То, что мы передадим в качестве параметра, мы увидим на экране.

```
print(10) # Мы увидим 10  
print(99) # Мы увидим 99
```

# Язык программирования Python. Модули



**Модуль** представляем собой отдельный файл с кодом на Python. Модулями еще называют библиотеки (дополнительные инструменты).

Подключение модуля могут осуществляться несколькими способами:

***import** имя\_модуля*  
***from** имя\_модуля **import** что-то\_конкретное*  
***from** имя\_модуля **import** \**

# Язык программирования Python. Модули (2)



Есть стандартная библиотека Python, в которую входят популярные модули: `math`, `cmath`, `sys` и другие. Чтобы начать их использовать, надо просто импортировать их в код (как на предыдущем слайде).

Но есть и другие модули: *pandas*, *matplotlib*, *folium*, которые придется предварительно установить, чтобы можно было импортировать и использовать:

***pip3 install*** имя\_модуля

# Библиотека pandas



Этапы работы с данными включают:

- обработка данных;
- анализ данных;
- наука о данных.

*Pandas* – библиотека языка Python для обработки данных, подготовки их к анализу.

*Майкл Хейдт, Артем Груздев “Изучаем Pandas”*

Официальный сайт <https://pandas.pydata.org/docs/>



# Библиотека pandas (2)



1. Анализа структурированных данных т.н. «panel data» («панельные данные»).
2. Данные исследований и наблюдений, представленные в виде таблиц.
3. Группировка данные, сводные таблицы и выборка по определенным признакам.
4. Поддерживаются форматы csv, excel, sql, html, hdf и др.

```
import pandas as pd
```



# Столбцы данных в pandas



*Pandas Series* (серия) — одномерные массивы (столбцы)

Индексы помогают обращаться к элементам серии и менять их значения.

# пример создания одного столбца

**import ...**

```
my_series = pd.Series([random.randint(0, 10) for _ in range(5)],
```

```
                        index=['a', 'b', 'c', 'd', 'e'])
```

```
print(my_series.head()) # вывод на экран первых 5 строк
```

```
print(my_series[['a', 'c', 'e']]) # обращение по индексам
```

```
print(my_series.max())
```

```
print(my_series.min())
```

```
print(my_series.mean())
```



# Таблицы данных в pandas



*Pandas DataFrame* — это двумерный массив, структурированный как таблица

Чтение данных: `read_excel(<file path>)`, **`read_csv(<file | buffer | url>)`**

# загружаем готовые данные

**import ...**

`iris = load_iris()`

`df = pd.DataFrame(data=np.c_[iris['data'], iris['target']],` # пример создания вручную:

`columns=iris['feature_names'] + ['target'])`

`print(df.head())`

`df = pd.DataFrame({'first': [1, 2, 3],  
 'second': [7, 7, 8],  
 'third': [6, 6, 5]})`





# Библиотека matplotlib



1. Библиотека для визуализации данных и их свойств.
2. Изначально была создана для ученых, не являющихся программистами.
3. Имеет множество встроенных пакетов (*matplotlib.pyplot* и др.)

***import matplotlib.pyplot as plt***

Официальный сайт: <https://matplotlib.org/>



# Библиотека matplotlib. Основные методы



| Вид графика                          | Методы<br><code>import matplotlib.pyplot as plt</code> |
|--------------------------------------|--|
| Линейные графики                     | <code>plt.plot(x, y)</code>                            |
| Площадная диаграмма (area graph)     | <code>plt.fill_between(x, y)</code>                    |
| Линейчатая и столбчатая диаграммы    | <code>plt.bar(index, values)</code>                    |
| Гистограмма                          | <code>plt.hist(data, bins=20)</code>                   |
| Круговая гистограмма                 | <code>plt.pie(values)</code>                           |
| Диаграмма размаха или «ящик с усами» | <code>plt.boxplot(data)</code>                         |

**mat****lib**



# Линейный график

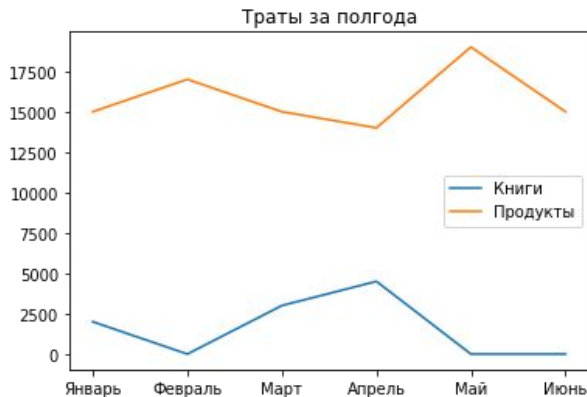
- Линейный график показывает *тенденцию*.
- Допустим, у нас есть данные, которые показывают изменение цены на товар со течением времени.
- Если таких данных много, мы можем воспользоваться линейным графиком: он покажет, возрастает цена или падает за определенный период.
- Если у нас несколько товаров, мы можем сравнить их графики.



# Линейный график. Примеры



`plt.plot(x, y)`



matplotlib



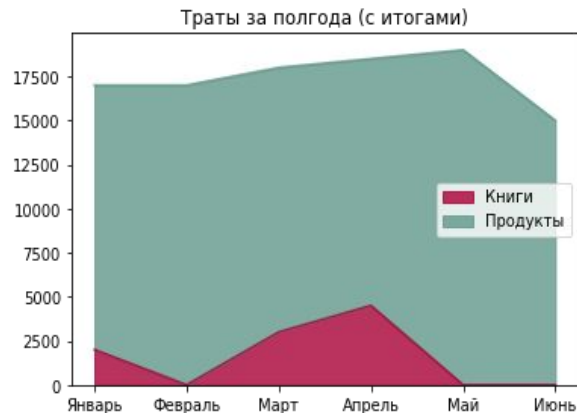
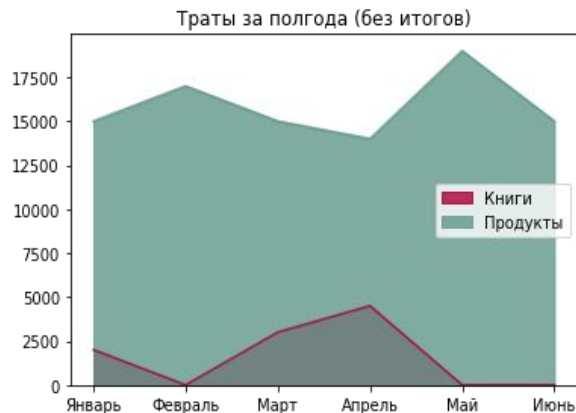
# Площадная диаграмма (area graph)

- Представляет собой линейный график, *но с заполненной цветом областью под линией.*
- Сгруппированные - начинаются с нулевой оси.
- С накопленными областями - каждый следующий график начинается с линии, оставленной предыдущим графиком.



# Площадная диаграмма (area graph).

## Пример



14

`plt.fill_between(x, y)`





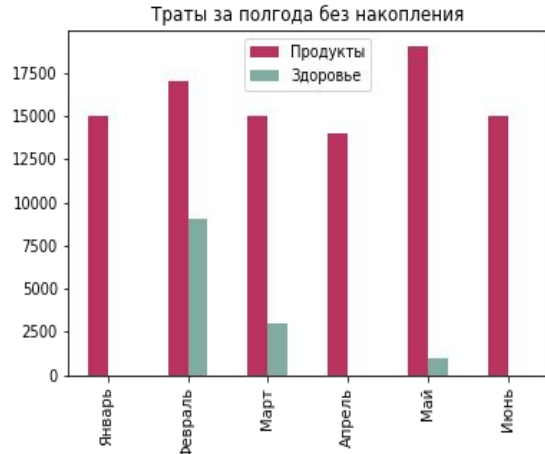
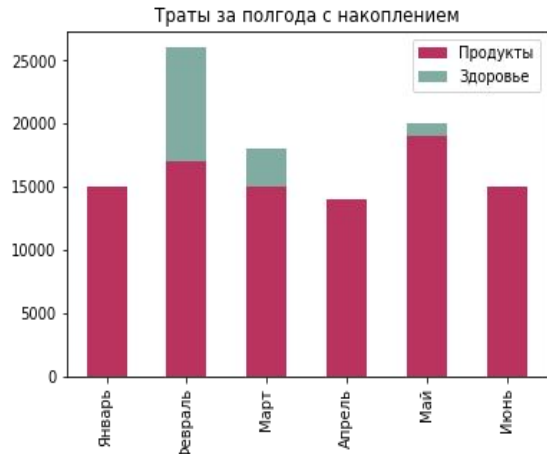
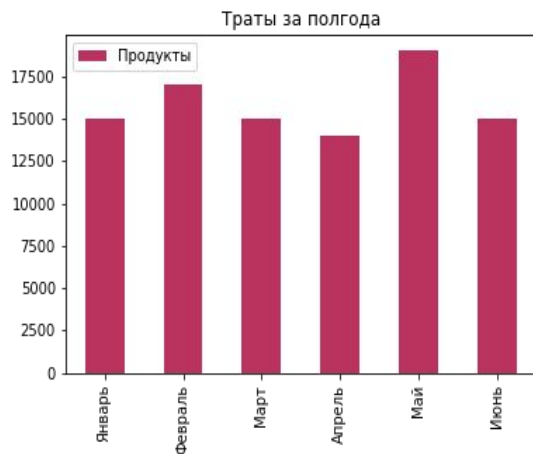
# Линейчатая и столбчатая диаграммы

- В вертикальных столбчатых диаграммах ось X используется для маркировки, а длина столбцов на оси Y соответствует величине измеряемой переменной.
- Здесь величина также может накапливаться



# Линейчатая и столбчатая диаграммы.

## Пример



`plt.bar(index, values)`







# Гистограмма

- Отображает распределение непрерывных данных.
- *Непрерывные данные* - это данные, которые могут принимать любые значения в некотором интервале. Например, средняя продолжительность жизни/температура/вес.
- В противоположность непрерывным данным выделяют дискретные данные - они могут принимать некоторые значения: количество человек в аудитории/количество машин на стоянке.





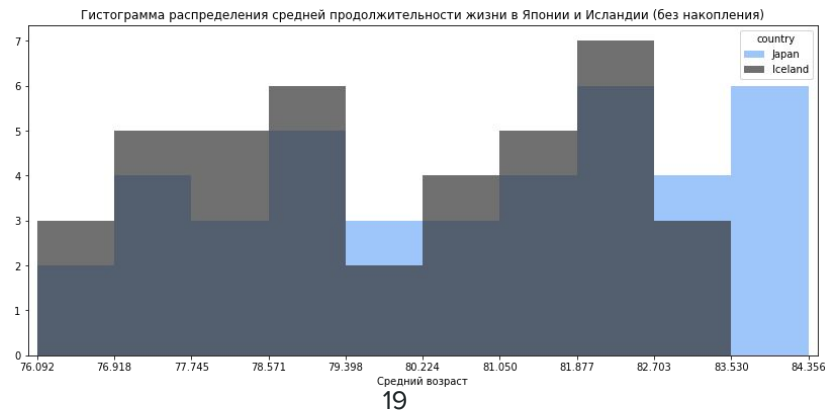
## Гистограмма (2)

- В отличие от линейчатой и столбчатой диаграмм, ось X не делится на взаимоисключающие категории, это сплошная шкала, которая делится на равные интервалы, а по оси Y откладывается количество данных, попадающих в этот интервал.
- Здесь величина по оси Y также может накапливаться

18



# Гистограмма. Пример



`plt.hist(data, bins=20)`





# Круговая диаграмма

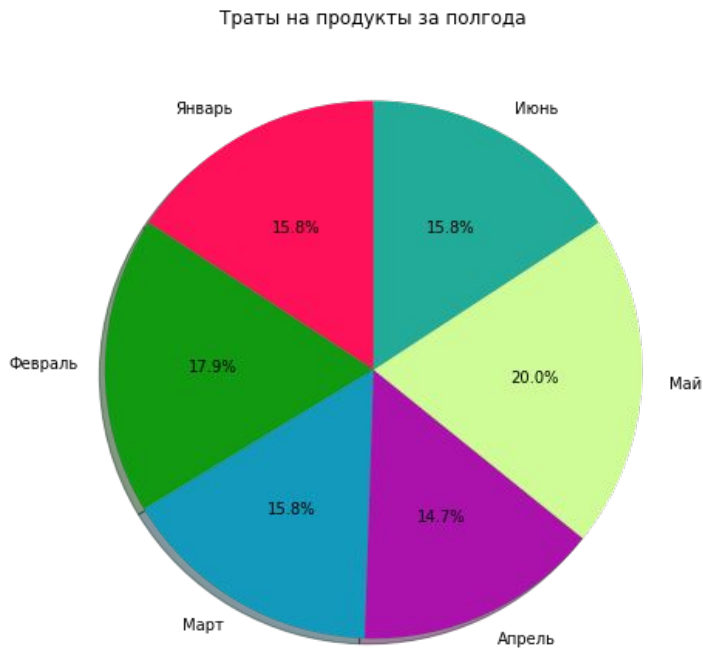
- Представляет собой круговой статистический график , который разделен на части.
- Длина дуги каждой части (и, следовательно, ее центральный угол и площадь) пропорциональна величине, которую эта часть представляет.



# Круговая диаграмма. Пример



`plt.pie(values)`



21





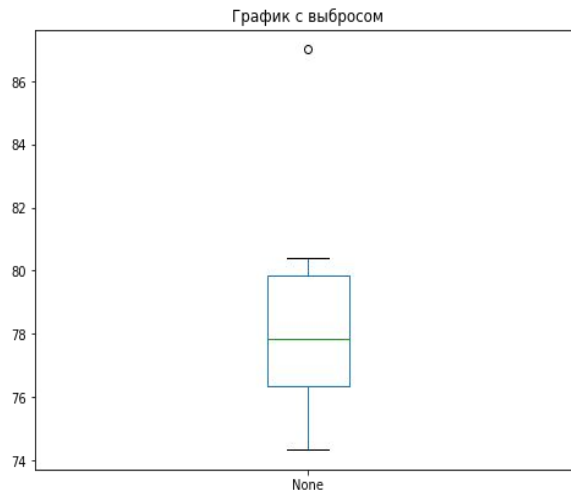
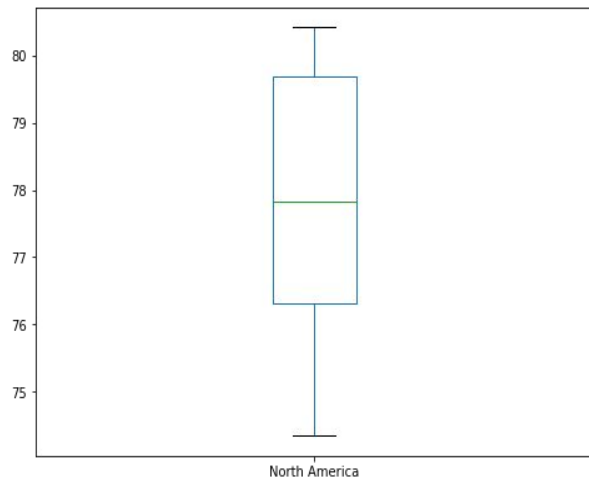
# Диаграмма размаха или ящик с усами

Такой вид диаграммы показывает медиану, нижний и верхний квартили, минимальное и максимальное значение выборки и выбросы.

- Пусть наши данные отсортированы по возрастанию.
- Нижний квартиль показывает значение, перед которым расположены 25% выборки.
- Верхний квартиль показывает значение, после которого расположены 25% выборки.
- Медиана - значение, которое расположено ровно посередине выборки.



# Диаграмма размаха или ящик с усами. Пример



`plt.boxplot(data)`

matplotlib

# Немного практики и полезных ССЫЛОК

---



# Ноутбуки для работы с Python без установки



- альтернатива PyCharm и другим IDE, которые надо устанавливать на ПК
- часто применяется в анализе данных, статистике и пр.

- **Jupyter Notebook**
- **Google Colab**
- **Kaggle**
- ...

The screenshot shows a Jupyter Notebook interface. At the top, it says "jupyter Untitled Last Checkpoint: 9 minutes ago (autosaved)". There are buttons for "Visit repo" and "Copy Binder link". Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar are "Trusted" and "Python 3" buttons. Below the menu bar is a toolbar with icons for file operations, running, and other functions. On the right side of the toolbar, it says "Memory: 201 / 8192 MB". The main area of the notebook contains a code cell with the following code:

```
In [1]: import numpy as np
import matplotlib.pyplot as plt

from tree_node import TreeNode
```

Below the code, there is a traceback error message:

```
ModuleNotFoundError                               Traceback (most recent call last)
<ipython-input-1-984dee29eb66> in <module>
      2 import matplotlib.pyplot as plt
      3
----> 4 from tree_node import TreeNode

ModuleNotFoundError: No module named 'tree_node'
```

At the bottom of the code cell, there is an input prompt "In [ ]:" followed by a text box.

# Практика 1. Основные преобразования и графики

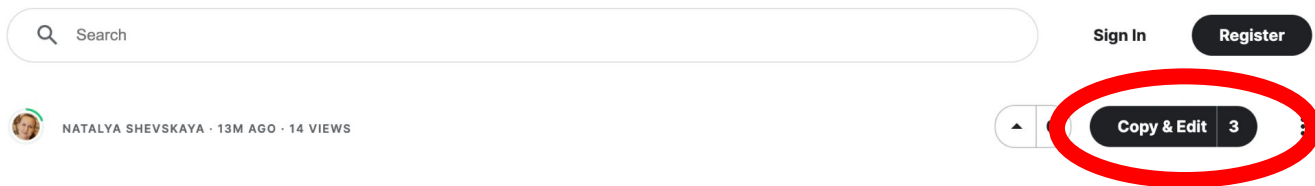
Переходим по ссылке

<http://bit.ly/3UiqDzo>



# Переходим к редактированию ноутбука

кликаем кнопку “Copy & Edit”



## Data-viz: Lesson 1

Python · [World Bank Data \(1960 to 2016\)](#)

Notebook Data Logs Comments (0)

27

Run

26.8s

🕒 Version 4 of 4







matplotlib

# Запуск блока с кодом

Explore and run machine learning code with Kaggle Notebooks. Logged out sessions end after 15 minutes [Sign in](#) or [Register](#)

notebookc1983eb5... Failed to save draft.

File Edit View Run Help

    Run All **Code** Draft Session off (run a cell to start)  

```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import pandas_datareader as pdr
from pandas_datareader import data, wb
import urllib3
```

+ Code + Markdown

```
[ ]:
from warnings import filterwarnings
filterwarnings('ignore')
```

+ Code + Markdown

```
[ ]:
with open('/kaggle/input/lifespan-tablecsv/lifespan_table.csv') as f:
    print(f.readline())
```

```
[ ]:
print(pd.__version__)
print(np.__version__)
print(mpl.__version__)
print(plt)
print(pdr.__version__)
print(wb)
print(urllib3.__version__)
```



28



## Практика 2. Площадная диаграмма

Переходим по ссылке

<http://bit.ly/3AVtS9a>

Кликаем кнопку “Copy & Edit”



29



# Практика 3. Гистограмма диаграмма

Переходим по ссылке

<http://bit.ly/3XJPeQp>

Кликаем кнопку “Copy & Edit”



30



# Работа с геоданными. Библиотека Folium



- Библиотека Folium позволяет создавать интерактивные карты и добавлять различную информацию на эти карты.
- Например, мы можем добавить на карту маркеры, которые точно соответствуют расположению на карте университетов, если у нас есть информация вида:

*Университет\_1 Расположение\_университета\_1*

*Университет\_2 Расположение\_университета\_2*

...

# Folium. Фоновая картограмма

- Вид картограммы, на которой штриховкой различной густоты или краской разной степени насыщенности изображают интенсивность какого-либо показателя в пределах территориальной единицы.
- Удобно использовать, если данные соотносятся с географическими координатами.





# Практика 4. Библиотека с геоданным

Переходим по ссылке

<http://bit.ly/3VV6UHp>

Кликаем кнопку “Copy & Edit”



# Практика 5. Кластеризация

Переходим по ссылке

<http://bit.ly/3VlxLWM>

Кликаем кнопку “Copy & Edit”



Еще немного подсказок по работе с  
данными

---

# Работа с dataframe в pandas

Извлечение данных («срезы»):

```
print(df['target'].unique())  
# [0. 1. 2.], ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']  
print(df[['target', 'sepal width (cm)', 'petal width (cm)']])  
  
print(df.loc[[54, 55, 56], 'target']) # извлечение по  
метке  
print(df.iloc[0:10]) # извлечение по позиции  
setosa = df[df['target'] == 0.0] # извлечение с условием  
versicolor = df[df['target'] == 1.0]  
virginica = df[df['target'] == 2.0]
```

## Работа с dataframe в pandas (2)

**Группировка** по столбцу *"target"*:

```
print(df.groupby('target').size()) # 0.0      50 \n 1.0 50 \n 2.0 50
print(df.groupby("target")["sepal width (cm)"].mean())
print(df.groupby("target")["sepal width (cm)"].min())
```

**Склеивание таблиц** — обязательно использовать те же столбцы:

```
df = pd.DataFrame({'first': [1, 2, 3],
                   'second': [7, 7, 8],
                   'third': [6, 6, 5]})
df_res = df.append(df1, ignore_index=True)
```

```
df1 = pd.DataFrame({'first': [-7, 0, 1],
                    'second': [2, -8, 4],
                    'third': [5, 1, 9]})
```

# Работа с dataframe в pandas (3)

Добавление новых данных (столбцов) в таблицу:

```
df1 = pd.DataFrame()
df1['random'] = pd.Series([random.randint(-100, 100)
                           for _ in range(10)])
df1['ones'] = pd.Series([1] * 10)
df1['new_column'] = (df1['random'] + 3) * df1['ones']
print(df1.head())
```

Добавление новой строки:

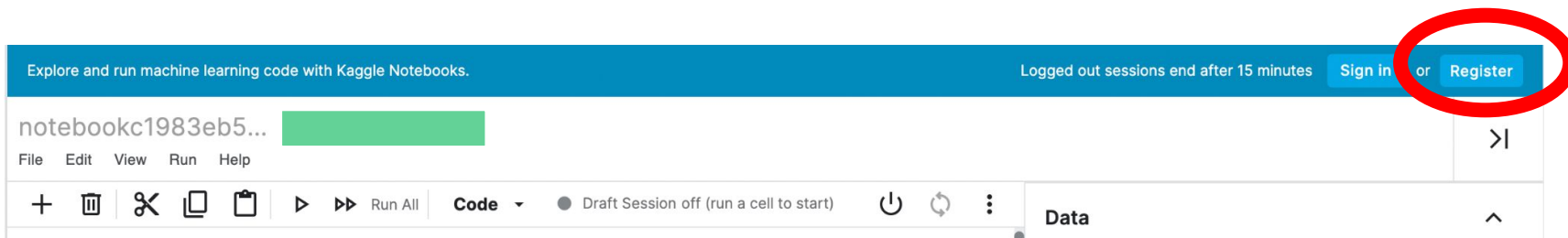
```
df.loc[len(df.index)] = [i for i in range(len(df.columns))]
```

# Использование своих данных на Kaggle

---

# Создание аккаунта со своей почтой

Чтобы загрузить локальные данные потребуется создать аккаунт со своей почтой








# Создание аккаунта со своей почтой (2)


Чтобы загрузить локальные данные потребуется создать аккаунт со своей почтой

[Sign In](#) [Register](#)

 Sign in with Google

 Sign in with your email

 Sign in with Facebook

 Sign in with Yahoo

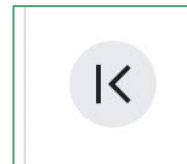
No Account? [Create one.](#)

# Подключение датасетов, которые уже есть на Kaggle

1. На панели инструментов справа жмем кнопку “Add data”

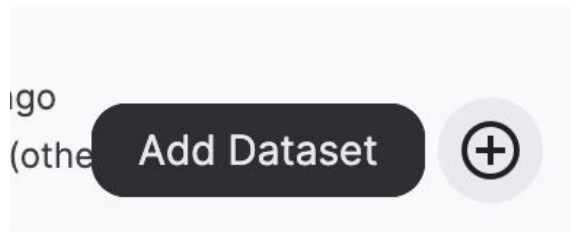


Если панель не видна, найдите справа кнопку: нажмите на нее и панель откроется



# Подключение датасетов, которые уже есть на Kaggle

1. В появившемся поле по очереди вводим названия датасетов и жмем “плюсик”
2. Ждем, когда Kaggle добавит датасет



## Meteorite Landings

NASA · Updated 6y ago  
197 Upvotes · 1 File (CSV) · 701 kB



## 500 Person Gender-Height-Wei...

Yasin Ersever · Updated 4y ago  
213 Upvotes · 1 File (CSV) · 2 kB



## Heights and Weights Dataset

Smit Patel · Updated 3y ago  
78 Upvotes · 1 File (CSV) · 250 kB



## world\_countries.json

Natalya Shevskaya · Updated 1h ago  
0 Upvotes · JSON · 102 kB



# Загрузка локальных датасетов на Kaggle

Файлы доступны по этой ссылке

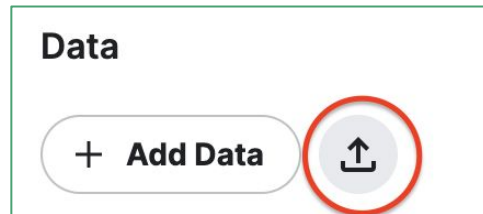
<https://bit.ly/43igh7E>

Файлы надо скачать.



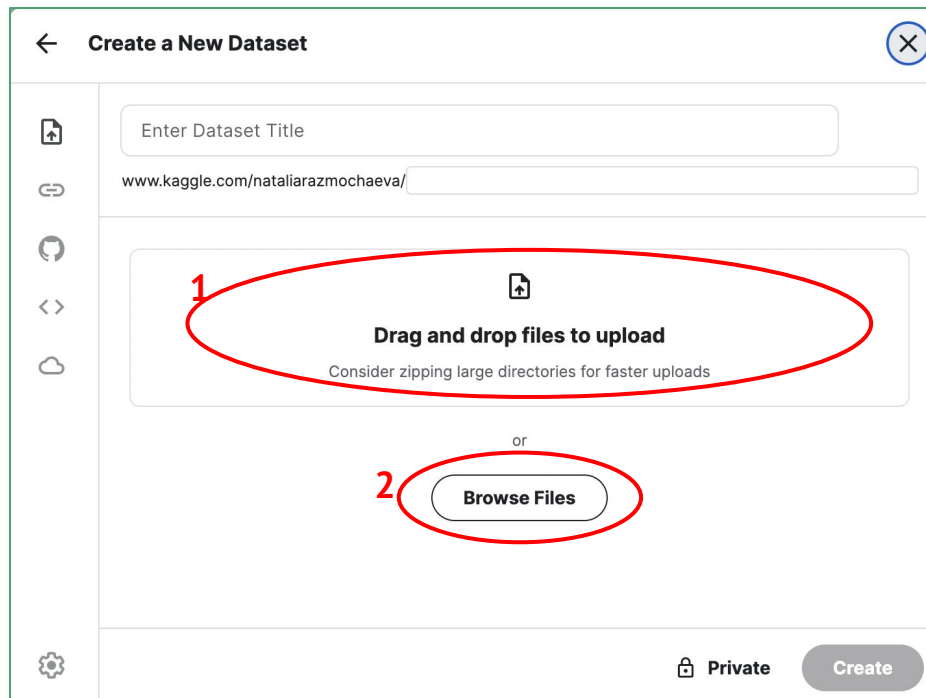
Затем на Kaggle жмем кнопку Upload:

Далее смотрим на открывшееся окошко



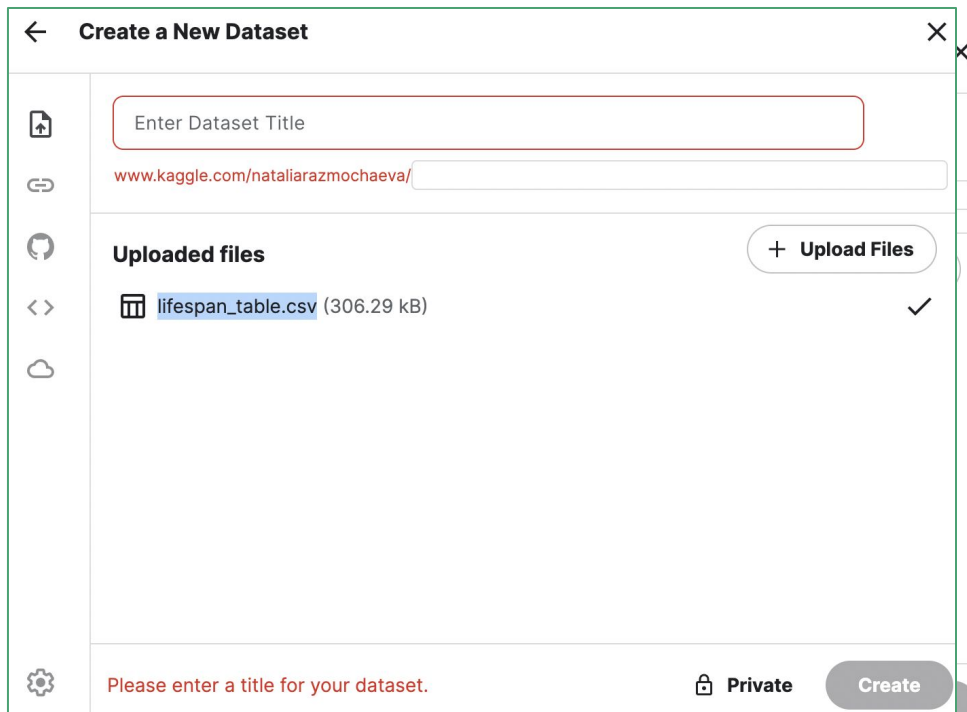
# Загрузка локальных датасетов на Kaggle (2)

1. Локальные файлы можно перетащить левой кнопкой мыши в поле “Drag and drop”
2. Либо файлы можно выбрать через кнопку “Browse File”



# Загрузка локальных датасетов на Kaggle (3)

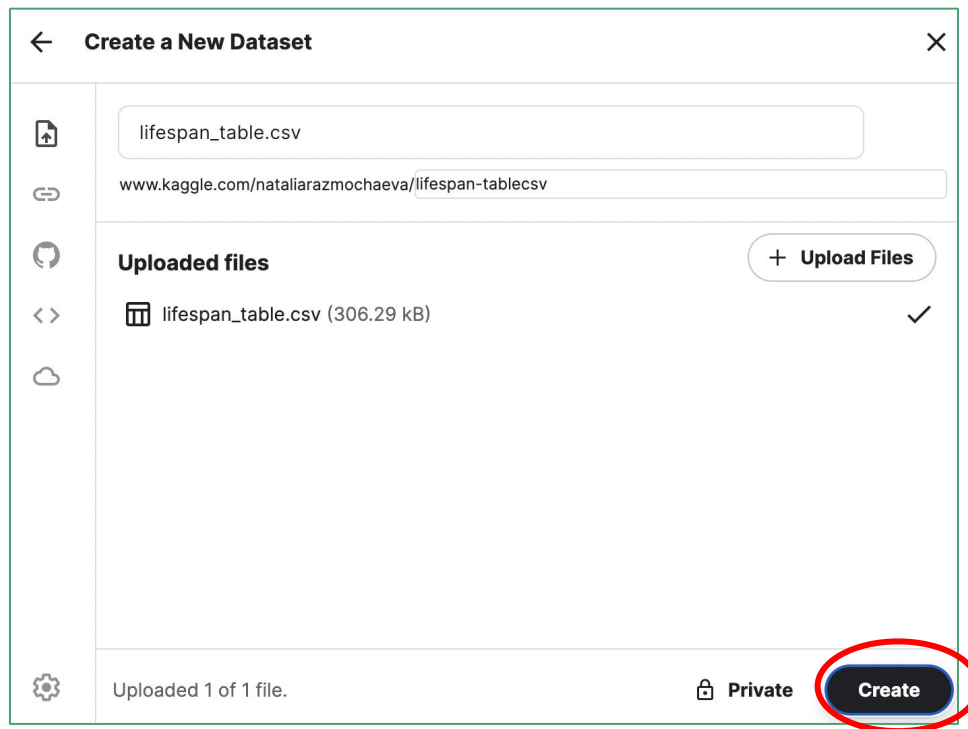
- После выбора файла надо напечатать название датасета (выделено красным)
- Для простоты -- давайте использовать название самого файла
- Печатать название файла следует в поле “Enter Dataset Title”



The screenshot shows the 'Create a New Dataset' window on Kaggle. On the left is a sidebar with icons for file upload, link, refresh, code, and cloud. The main area has a title input field with a red border and placeholder text 'Enter Dataset Title'. Below it is a URL field containing 'www.kaggle.com/nataliarazmochaeva/'. Under the 'Uploaded files' section, a file named 'lifespan\_table.csv' (306.29 kB) is listed with a checkmark. An 'Upload Files' button is in the top right. At the bottom, there is a red error message 'Please enter a title for your dataset.', a 'Private' toggle, and a 'Create' button.

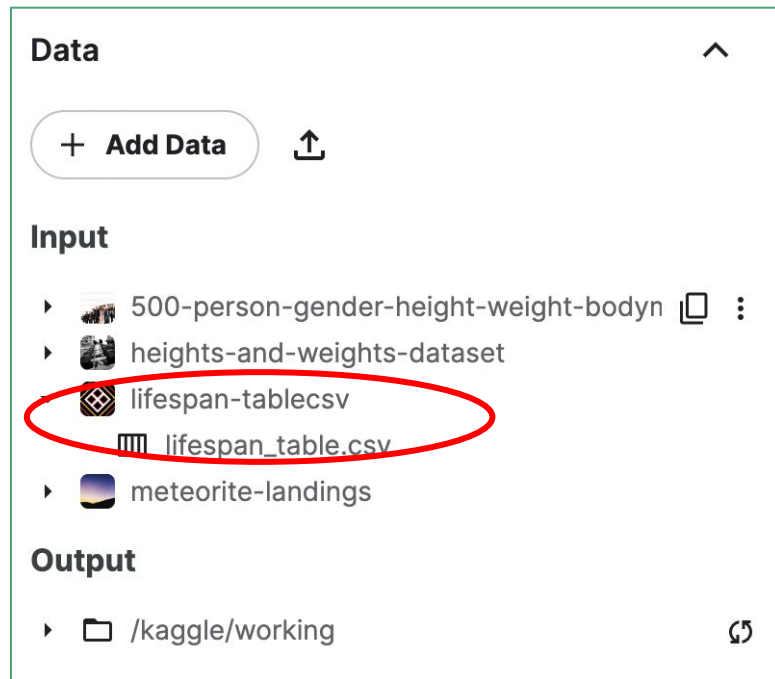
# Загрузка локальных датасетов на Kaggle (4)

- Как только имя файла напечатано, можно жать “Create”



# Загрузка локальных датасетов на Kaggle (5)

- Немного подождав, можно увидеть датасет в списке датасетов (в том же, где и Kaggl'овские)
- Нажав на название датасета, в выпадающем списке можно увидеть сам файл csv, который и загружался в систему





# Вопросы по курсу можно задавать:

---

Иванов Дмитрий Владимирович  
[dmitry.ivanov@moevm.info](mailto:dmitry.ivanov@moevm.info)