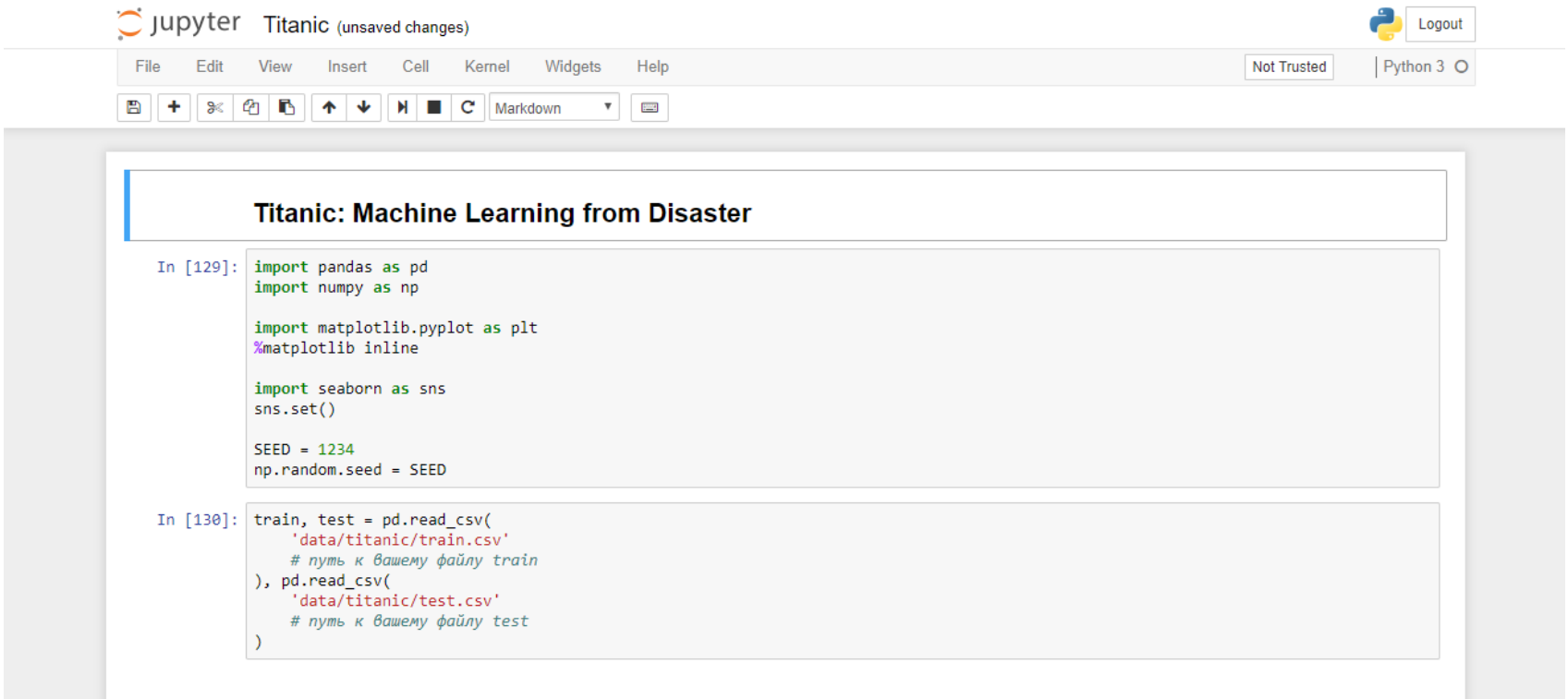
# *Занятие 2*:
# Введение в Numpy и Jupyter notebook

# Jupyter notebook

Ранее рассматривался пакет Anaconda. Напомним, что он включает в себя сотни популярных библиотек для языка Python и работает на Windows, Linux и MacOS. Conda позволяет быстро и легко запускать и модернизировать проекты, использующие scikit-learn, TensorFlow, SciPy и многое другое. Помимо того, в её состав входит Jupyter Notebook, который в дальнейшем будет использоваться для работы. Рассмотрим его интерфейс более подробно:



# Основные особенности использования

- Jupyter представляет информацию в браузере, а выполняет вычисления средствами компьютера (эти действия можно наблюдать в терминале).
- Структура файла *.ipynb* состоит из отдельных блоков нескольких видов. Нас будут интересовать: текст, он же Markdown (включая MD разметку, html и Latex вставки) и Code (блок, в котором можно писать код).
- Ctrl + Enter — выполнить строчку. Shift + Enter — выполнить строчку и перейти на следующую.
- Можно с помощью кнопок сверху создавать новые блоки (insert cell below), перемещать стрелками и удалять (удаление традиционно осуществляется через cut selected cells). Также существуют клавиатурные комбинации.
- Знак звёздочки (*) слева от блока означает, что он выполняется и, соответственно, цифра означает порядок, которым он был выполнен ранее.
- Установив курсор на импортированную функцию, можно нажать Shift+Tab и получить краткую справку по ней. Нажав еще раз, можно получить более полную справку.
- По умолчанию корневая директория сохранения файлов *.ipynb* это *C* : *Users*\%*username*%.
- Дважды кликнув по ячейке, её можно отредактировать.

# Установка дополнительных библиотек

Конечно же, может возникнуть ситуация, когда при решении той или иной задачи на компьютере будет отсутствовать какая-либо необходимая библиотека. Решить данную проблему можно, как правило, одним из двух способов. Для примера, установить через терминал *numpy, scipy, pandas u sklearn* можно следующими вариантами:

- python -m pip install numpy scipy pandas sklearn
- conda install numpy scipy pandas sklearn

Причём возможно установка удастся одним образом и не удастся вторым. Также могут возникнуть какие-либо иные ошибки при установке, специфичные для конкретной библиотеки. Их решение можно найти в документации к библиотеке или на StackOverflow.

# Использование NumPy

**Numpy** позволяет быстро обрабатывать массивы и матрицы. Рассмотрим простейшие примеры использования

***Создание и размер***

```
In [1]: a = np.array([[1,2,3], [4,5,6]])
        a
Out[1]: array([[1, 2, 3],
               [4, 5, 6]])

In [2]: a.shape
Out[2]: (2, 3)
```

***Простые действия***

```
In [3]: a + 2
Out[3]: array([[3, 4, 5],
               [6, 7, 8]])

In [4]: a*2
Out[4]: array([[ 2,  4,  6],
               [ 8, 10, 12]])
```

***Индексация***

```
In [5]: a[0, 0]
Out[5]: 1

In [6]: a[0, 2]
Out[6]: 3

In [7]: a[1, 1]
Out[7]: 5

In [8]: a[:, 0]
Out[8]: array([1, 4])

In [9]: a[:, 1]
Out[9]: array([2, 5])

In [10]: a[:, 2]
Out[10]: array([3, 6])

In [11]: a[0, :]
Out[11]: array([1, 2, 3])

In [12]: a[1, :]
Out[12]: array([4, 5, 6])
```

***Комбинации***

In [13]:
```
b = np.array([[7,8,9], [3,4,5]])
```

In [14]:
```
print(a, '\n\n', b)
```

```
[[1 2 3]
 [4 5 6]]

 [[7 8 9]
 [3 4 5]]
```

In [15]:
```
np.hstack((a,b))
```

Out[15]:
```
array([[1, 2, 3, 7, 8, 9],
       [4, 5, 6, 3, 4, 5]])
```

In [16]:
```
np.vstack((a,b))
```

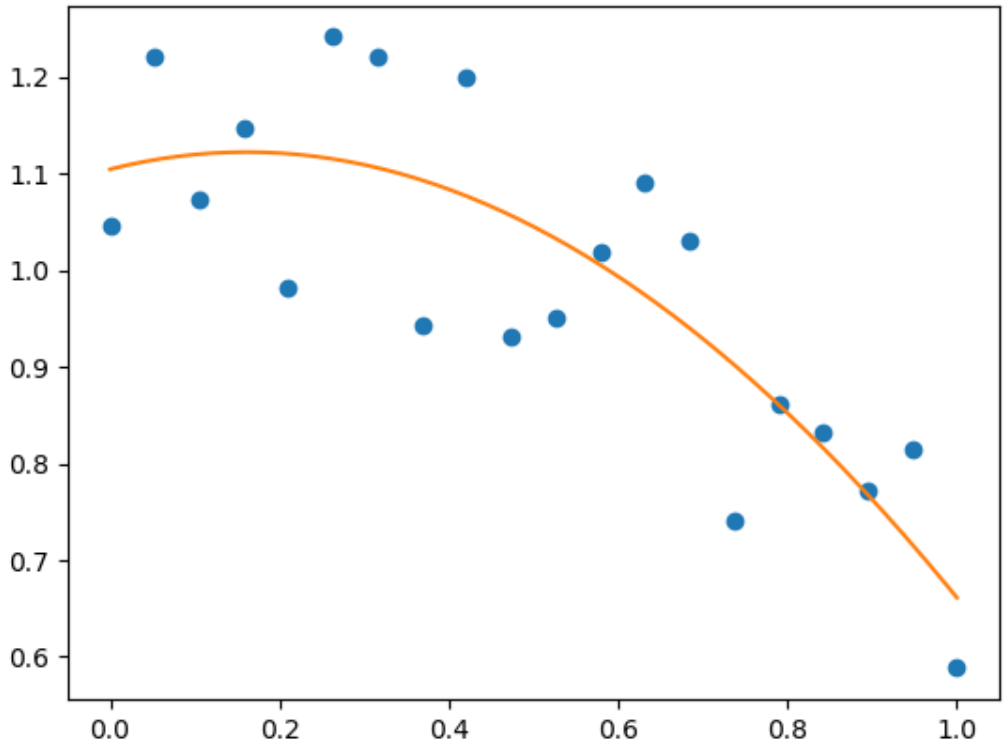
Out[16]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9],
       [3, 4, 5]])
```

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(12)

x = np.linspace(0, 1, 20)
y = np.cos(x) + 0.3*np.random.rand(20)
p = np.poly1d(np.polyfit(x, y, 3))

t = np.linspace(0, 1, 200)
plt.plot(x, y, 'o', t, p(t), '-')
plt.show()
```



# Библиотека SciPy

**SciPy** - библиотека для языка программирования Python с открытым исходным кодом, предназначенная для выполнения научных и инженерных расчётов. Она может использоваться для поиска минимумов и максимумов функций, вычисления интегралов функций, обработки сигналов, обработки изображений, решения обыкновенных дифференциальных уравнений и т.д. Работает он на базе NumPy.

```
>>> from scipy import linalg
>>> arr = np.array([[1, 2],
...                 [3, 4]])
>>> linalg.det(arr)
-2.0
```

```
>>> from scipy import constants
>>> constants.c # speed of light
299792458.0
>>> constants.h # Plank's constant
6.6260693000000002e-034
>>> constants.N_A # Avogadro's number
6.0221415000000003e+023
```

```
>>> from scipy import integrate
>>> integrate.quad(lambda x: math.cos(math.exp(x)), 2, 3)
(-0.063708480528704675, 2.4175070627010321e-014)
```

# Практические задания


1. Запустить задание с прошлого занятия в Jupyter notebook
2. Реализовать в Jupyter notebook алгоритм сортировки пузырьком и протестировать для ряда чисел: $92, 11, 45, 2234, 0, 7, 65$

# Ответы на задания

bubblesort.py

```python
def bubble_sort(some_list):

    is_sorted = False
    last_sorted_item = len(some_list) - 1

    while not is_sorted:

        is_sorted = True

        for i in range(0, last_sorted_item):

            if some_list[i] > some_list[i + 1]:

                some_list[i], some_list[i+1] = some_list[i+1], some_list[i]
                is_sorted = False

        last_sorted_item -= 1

    return some_list


my_numbers = [92,11,45,2234,0,7,65]
print(bubble_sort(my_numbers))
```