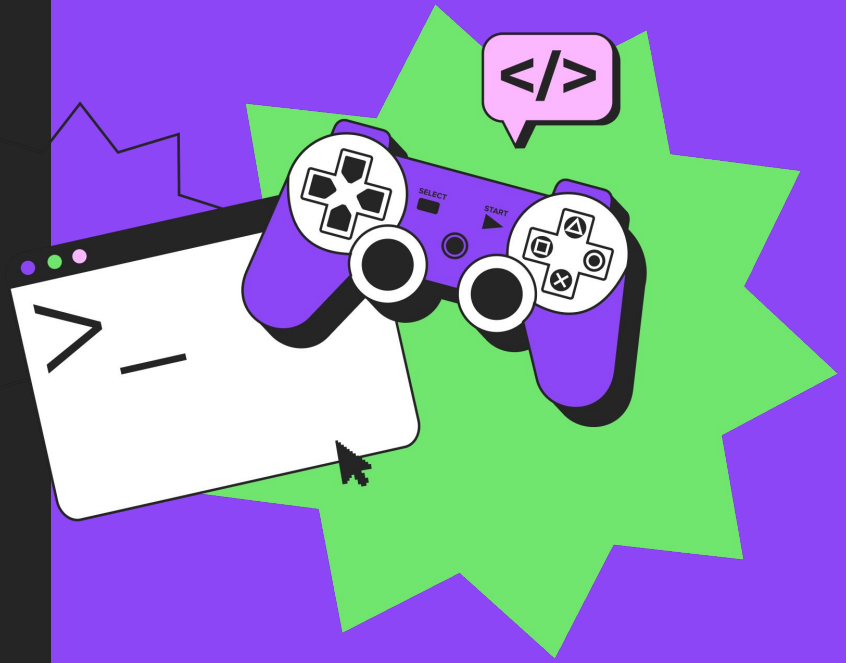
GeekBrains

Знакомство с библиотеками

А что вообще есть? И где искать?

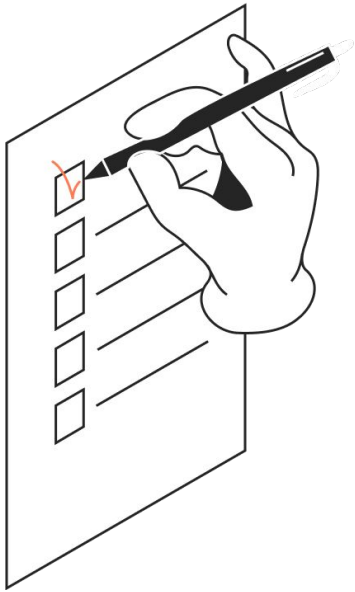


Александра Корнеева

- Computer Vision Engineer в М.Видео и Эльдорадо;
- Аналитик-разработчик в Авито;
- Выращиваю салат с помощью видеокамер.

Что будет на уроке сегодня

- 📌 Знакомство с библиотеками анализа данных на Python;
- 📌 Pandas. Работа с датафреймами;
- 📌 NumPy. Матрицы и линейная алгебра;
- 📌 Scipy. Математические операции;
- 📌 Matplotlib. Визуализация;
- 📌 Scikit-Learn. Машинное обучение;
- 📌 А что ещё?



*** Весь код со слайдов вы найдёте в Jupyter Notebook. Не переживайте** 😌

Pandas.
Вы подумали не о тех пандах

Основные структуры данных и индексация по ним

Pandas Series (серия) — одномерный массив.

- Индексы можно указать самостоятельно;
- Индекс может быть числом (1, 2...), буквой (‘a’, ‘b’..), любым неизменяемым объектом;
- Если не задать индексацию, она проставляется автоматически (0, 1, 2...).

```
1 s = pd.Series(['Петя', 'Вася', 'Саша'])
2 s
```

```
0    Петя
1    Вася
2    Саша
dtype: object
```

```
1 s[1]
```

```
'Вася'
```

```
1 s = pd.Series(['Петя', 'Вася', 'Саша'], index=['a', 'b', 'c'])
2 s
```

```
a    Петя
b    Вася
c    Саша
dtype: object
```

```
1 s[1], s['c']
```

```
('Вася', 'Саша')
```

Основные структуры данных и индексация по ним

Pandas DataFrame — двумерный массив, похожий на таблицу.

- Столбец DataFrame — это Series;
- Индексация по столбцам;
- Для доступа к строке используется метод **.iloc**

```
1 data = {'name':['Ира','Петя','Валя','Коля'],'age':[23, 22, 21, 24]}
2 data = pd.DataFrame(data)
3 data
```

	name	age
0	Ира	23
1	Петя	22
2	Валя	21
3	Коля	24

```
1 data['name'], type(data['name']) # data.name
```

```
(0     Ира
 1    Петя
 2    Валя
 3    Коля
 Name: name, dtype: object,
 pandas.core.series.Series)
```

```
1 data.iloc[1]
```

```
name    Петя
age       22
Name: 1, dtype: object
```

```
1 data.iloc[1].age
```

```
22
```

Загрузка и сохранение данных

```
1 data = pd.read_csv('gb/winemag-data_first150k.csv', index_col=0)
2 data.head() # data.tail()
```

	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	NaN	Tinta de Toro	Bodega Carmen Rodríguez
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi
4	France	This is the top wine from La Bégude, named aft...	La Brûlade	95	66.0	Provence	Bandol	NaN	Provence red blend	Domaine de la Bégude

```
1 data.to_csv('gb/winemag-data_first150k_copy.csv')
```

Описание загруженного датасета

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 150930 entries, 0 to 150929
Data columns (total 10 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   country      150925 non-null  object 
 1   description  150930 non-null  object 
 2   designation  105195 non-null  object 
 3   points       150930 non-null  int64  
 4   price        137235 non-null  float64
 5   province     150925 non-null  object 
 6   region_1     125870 non-null  object 
 7   region_2     60953 non-null   object 
 8   variety      150930 non-null  object 
 9   winery       150930 non-null  object 
dtypes: float64(1), int64(1), object(8)
memory usage: 12.7+ MB
```

```
1 data.describe()
```

	points	price
count	150930.000000	137235.000000
mean	87.888418	33.131482
std	3.222392	36.322536
min	80.000000	4.000000
25%	86.000000	16.000000
50%	88.000000	24.000000
75%	90.000000	40.000000
max	100.000000	2300.000000

Индексация

```python
# колонка
data['price'] # data.price
```

```
0         235.0
1         110.0
2          90.0
3          65.0
4          66.0
          ...
150925     20.0
150926     27.0
150927     20.0
150928     52.0
150929     15.0
Name: price, Length: 150930, dtype: float64
```

```python
# строка
data.iloc[3]
```

```
country                                                    US
description    This spent 20 months in 30% new French oak, an...
designation                                          Reserve
points                                                    96
price                                                   65.0
province                                              Oregon
region_1                                  Willamette Valley
region_2                                  Willamette Valley
variety                                          Pinot Noir
winery                                               Ponzi
Name: 3, dtype: object
```

Срез датасета

```
1 df1 = data[0:2]
2 df2 = data[2:5]
3 print(df1.shape, df2.shape)
```

```
(2, 10) (3, 10)
```

```
1 df1
```

	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	NaN	Tinta de Toro	Bodega Carmen Rodríguez

```
1 df2
```

	country	description	designation	points	price	province	region_1	region_2	variety	winery
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi
4	France	This is the top wine from La Bégude, named aft...	La Brûlade	95	66.0	Provence	Bandol	NaN	Provence red blend	Domaine de la Bégude

Конкатенация («склеивание») датасетов

```
1 pd.concat([df1, df2], ignore_index=True)
```

	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	NaN	Tinta de Toro	Bodega Carmen Rodríguez
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	Sonoma	Sauvignon Blanc	Macauley
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	Willamette Valley	Pinot Noir	Ponzi
4	France	This is the top wine from La Bégude, named aft...	La Brûlade	95	66.0	Provence	Bandol	NaN	Provence red blend	Domaine de la Bégude

NumPy.
Матрицы и линейная алгебра

Массивы NumPy

```
# одномерный
arr_1 = np.array([1, 2, 3, 4, 5, 6])
arr_1
```

```
array([1, 2, 3, 4, 5, 6])
```

```
# многомерный (двумерный)
arr_2 = np.array([[1, 2, 3], [5, 6, 7]])
arr_2
```

```
array([[1, 2, 3],
       [5, 6, 7]])
```

```
# растягиваем в одномерный
arr_2.ravel()
```

```
array([1, 2, 3, 5, 6, 7])
```

```
arr_1.shape, arr_2.shape
```

```
((6,), (2, 3))
```

```
arr_3 = arr_1.reshape(2, 3)
arr_3
```

```
array([[1, 2, 3],
       [4, 5, 6]])
```

- **.ravel()** — «растягивание» в одномерный массив;
- **.shape** — размерность массива;
- **.reshape()** — изменение размерности массива.

Создание NumPy-массивов определённого типа

```
np.zeros((3,3))
```

```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

```
np.ones((3, 3))
```

```
array([[1., 1., 1.],
       [1., 1., 1.],
       [1., 1., 1.]])
```

```
np.eye(4)
```

```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

```
a = np.array([[0, 1], [2, 3], [4, 5]])
np.ones_like(a)
```

```
array([[1, 1],
       [1, 1],
       [1, 1]])
```

- **zeroes((n, m))** — массив из нулей;
- **ones((n, m))** — массив из единиц;
- **eye(n)** — единичная матрица;
- **ones_like(arr)** — матрица такой же размерности, как и arr, но единичная;
- **zeros_like(arr)** — матрица такой же размерности, как и arr, но нулевая.

Работа с измерениями

Работа с измерениями заложена практически во все функции NumPy.

```
a = np.array([[0, 1, 2], [3, 4, 5], [6, 8, 7]])
a
```

```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 8, 7]])
```

```
a.max()
```

```
8
```

```
# минимальные элементы по столбцам
a.min(axis = 0)
```

```
array([0, 1, 2])
```

```
# минимальные элементы по строкам
a.min(axis = 1)
```

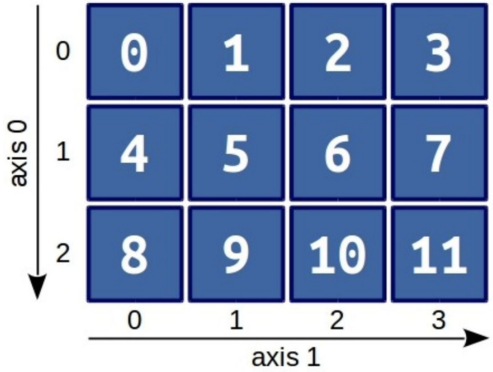
```
array([0, 3, 6])
```

```
# среднее по столбцам
a.mean(axis = 0)
```

```
array([3.        , 4.33333333, 4.66666667])
```

```
# стандартное октлонение по строкам
np.std(a, axis = 1)
```

```
array([0.81649658, 0.81649658, 0.81649658])
```



SciPy.
Математические операции

Основные модули, которые нам понадобятся

constants

integrate

linalg

Constants

constants

integrate

linalg

Физические и математические константы

```
1 from scipy import constants
```

```
1 constants.pi, constants.Planck, constants.g, constants.Avogadro
```

```
(3.141592653589793, 6.62607015e-34, 9.80665, 6.02214076e+23)
```

Integrate. Нам больше не нужен wolframalpha

constants

integrate

linalg

Решения интегральных и обычных дифференциальных уравнений

- Берем интеграл $\int_0^4 x^2\,dx$

```
1  f1 = lambda x: x**2
2  integrate.quad(f1, 0, 4)
```

```
(21.333333333333336, 2.368475785867001e-13)
```

$\int_0^{\inf} x^2\,dx$

```
1  invexp = lambda x: np.exp(-x)
2  integrate.quad(invexp, 0, np.inf)
```

```
(1.0000000000000002, 5.842606742906004e-11)
```

Linalg. Линейная алгебра

constants

integrate

linalg

```
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 10]])
arr
```

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8, 10]])
```

```
# определитель матрицы
linalg.det(arr)
```

```
-3.0000000000000013
```

```
# собственные значения и собственные векторы
linalg.eig(arr)
```

```
(array([16.70749332+0.j, -0.90574018+0.j,  0.19824686+0.j]),
 array([[-0.22351336, -0.86584578,  0.27829649],
        [-0.50394563,  0.0856512 , -0.8318468 ],
        [-0.83431444,  0.4929249 ,  0.48018951]]))
```

```
# обратная матрица
linalg.inv(arr)
```

```
array([[-0.66666667, -1.33333333,  1.        ],
       [-0.66666667,  3.66666667, -2.        ],
       [ 1.        , -2.        ,  1.        ]])
```

```
# решение СЛАУ Ax=b
b = np.array([1, 2, 3])
linalg.solve(arr, b)
```

```
array([-3.33333333e-01,  6.66666667e-01,  3.17206578e-17])
```

Matplotlib. Визуализация

Объект Figure

```python
fig = plt.figure()   # Создание объекта Figure
plt.scatter(1.0, 1.0)   # scatter - метод для нанесения маркера в точке (1.0, 1.0)

plt.show()
```

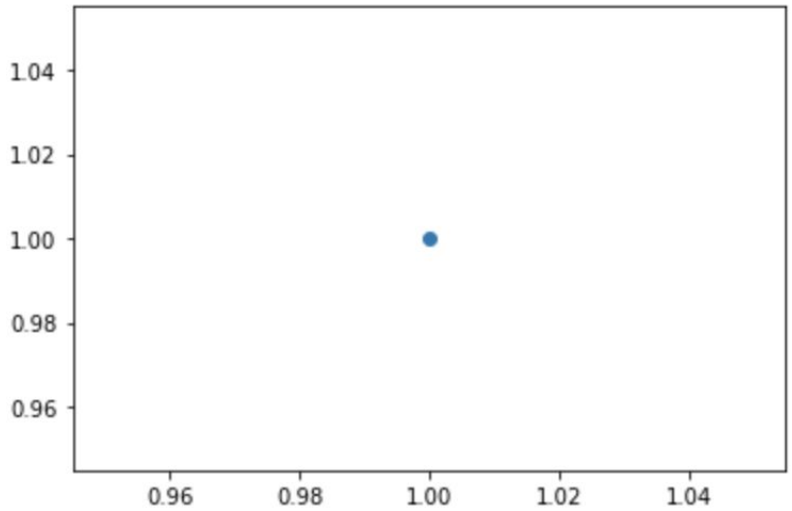


График линии (.plot)

```
1  # график линии
2  plt.plot((0, 1, 2, 3, 4, 5, 6, 7), (-1, 3, 5, -3, 5, 6, -9, 2))
3  plt.grid()
4  plt.show()
```

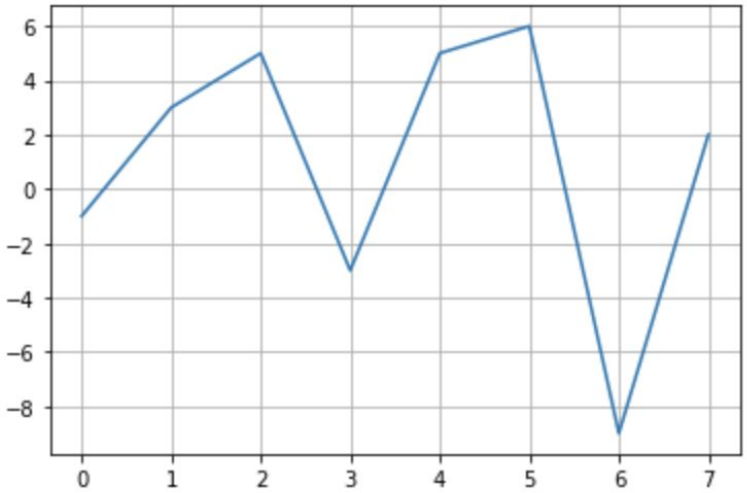


График замкнутой линии (.plot)

```
1 # замкнутые фигуры
2 plt.plot((0, 0, 1, 1, 0), (0, 1, 1, 0, 0))
3 plt.plot((0.1, 0.5, 0.9, 0.1), (0.1, 0.9, 0.1, 0.1))
4 plt.show()
```

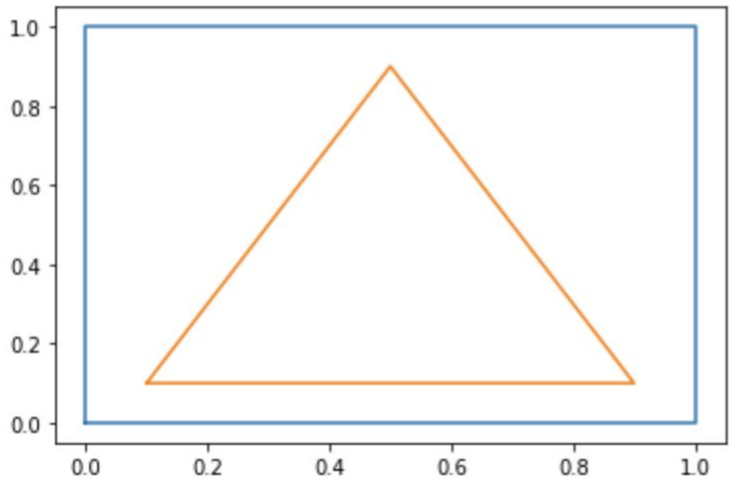
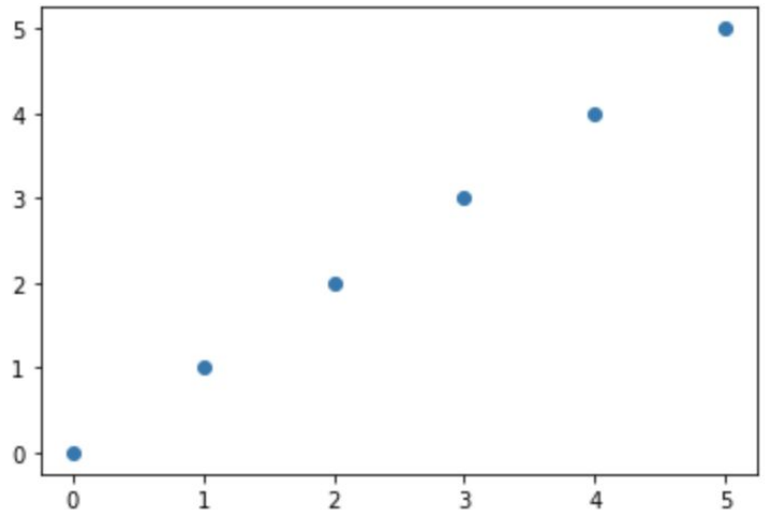


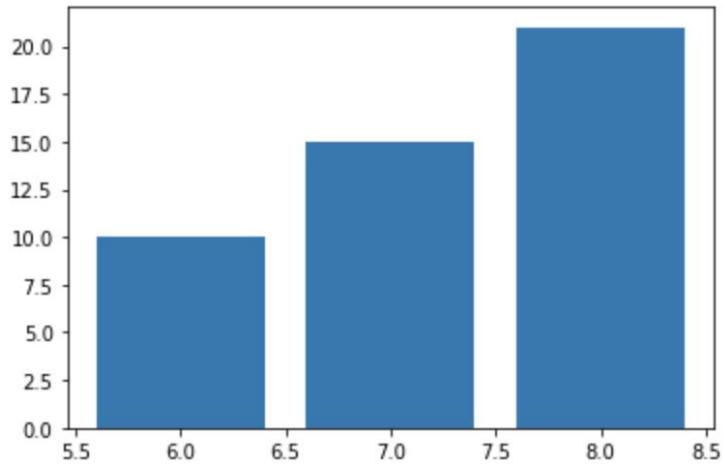
График множества точек (.scatter)

```
1 # график множества точек
2 plt.scatter([0, 1, 2, 3, 4 , 5], [0, 1, 2, 3, 4 , 5])
3 plt.show()
```



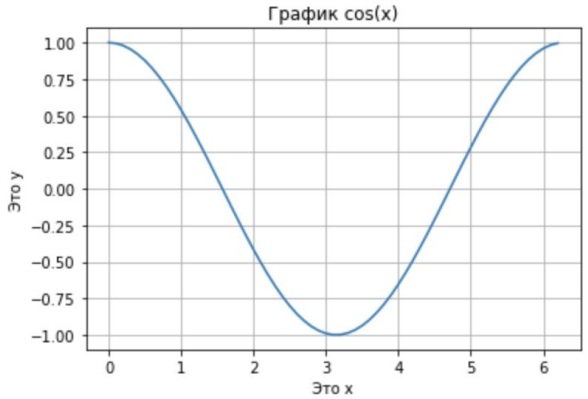
Гистограмма

```
1 #гистограмма
2 plt.bar([6, 7, 8], [10, 15, 21])
3 plt.show()
```



Дополнительные элементы на графике

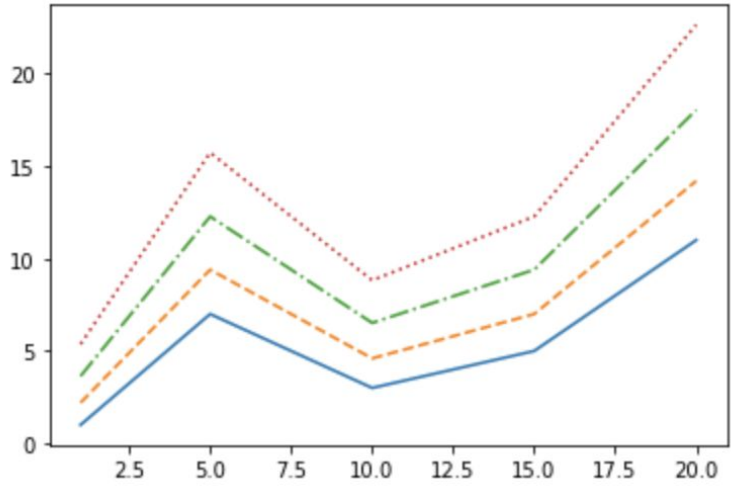
```
lag = 0.1
x = np.arange(0.0, 2 * np.pi, lag)
y = np.cos(x)

fig = plt.figure()
plt.plot(x, y)
plt.grid()

plt.title('График cos(x)')
plt.ylabel('Это y')
plt.xlabel('Это x');
```



- **plt.plot** — график по точкам;
- **plt.grid** — сетка;
- **plt.title** — название графика;
- **plt.ylabel/plt.xlabel** — наименование осей.

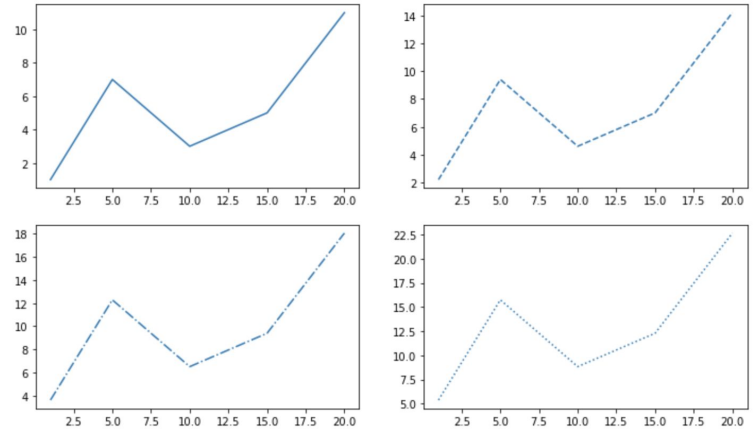
Несколько графиков в одной области

```
x = [1, 5, 10, 15, 20]
y1 = [1, 7, 3, 5, 11]
y2 = [i*1.2 + 1 for i in y1]
y3 = [i*1.2 + 1 for i in y2]
y4 = [i*1.2 + 1 for i in y3]
plt.plot(x, y1, '-', x, y2, '--', x, y3, '-.', x, y4, ':');
```



Несколько графиков в разных областях

```
# Настройка размеров подложки
plt.figure(figsize=(12, 7))

# Вывод графиков
plt.subplot(2, 2, 1)
plt.plot(x, y1, '-')

plt.subplot(2, 2, 2)
plt.plot(x, y2, '--')

plt.subplot(2, 2, 3)
plt.plot(x, y3, '-.')

plt.subplot(2, 2, 4)
plt.plot(x, y4, ':');
```



Scikit-Learn. Машинное обучение

Что можно найти в Scikit-Learn?

sklearn.preprocessing.StandardScaler — нормализация данных (приведение дисперсии к единице, математического ожидания — к нулю).

предварительная обработка данных

выбор модели

```
x = np.array([[0.1, 1.0, 22.8], [0.5, 5.0, 41.2], [1.2, 12.0, 2.8], [0.8, 8.0, 14.0]])
x
```

```
array([[ 0.1,  1. , 22.8],
       [ 0.5,  5. , 41.2],
       [ 1.2, 12. ,  2.8],
       [ 0.8,  8. , 14. ]])
```

```
scaler = StandardScaler()
scaled_x = scaler.fit_transform(x)
scaler.scale_
```

```
array([ 0.40311289,  4.03112887, 14.04421589])
```

```
scaler.mean_
```

```
array([ 0.65,  6.5 , 20.2 ])
```

```
scaler.var_
```

```
array([1.6250e-01, 1.6250e+01, 1.9724e+02])
```

```
scaled_x.mean().round(decimals=4)
```

```
0.0
```

```
scaled_x.std(axis=0)
```

```
array([1., 1., 1.])
```

Что можно найти в Scikit-Learn?

sklearn.preprocessing.OneHotEncoder — преобразование категориальных данных в числа.

предварительная обработка данных

выбор модели

```
roles = np.array([('Tom', 'manager'), ('Mary', 'developer'), ('Ann', 'recruiter'), ('Jim', 'developer')])
```

```
encoder = OneHotEncoder()
encoded_roles = encoder.fit_transform(roles[:, [1]])
encoded_roles.toarray()
```

```
array([[0., 1., 0.],
       [1., 0., 0.],
       [0., 0., 1.],
       [1., 0., 0.]])
```

Что можно найти в Scikit-Learn?

sklearn.model_selection.train_test_split() — разбиение данных на выборки трейн и тест.

предварительная обработка данных

выбор модели

```
x, y = np.arange(1, 21).reshape(-1, 2), np.arange(3, 40, 4)
x, y
```

```
(array([[ 1,  2],
        [ 3,  4],
        [ 5,  6],
        [ 7,  8],
        [ 9, 10],
        [11, 12],
        [13, 14],
        [15, 16],
        [17, 18],
        [19, 20]]),
 array([ 3,  7, 11, 15, 19, 23, 27, 31, 35, 39]))
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.4)
```

```
x_train, y_train
```

```
(array([[17, 18],
        [ 7,  8],
        [ 9, 10],
        [13, 14],
        [15, 16],
        [11, 12]]),
 array([35, 15, 19, 27, 31, 23]))
```

```
x_test, y_test
```

```
(array([[ 5,  6],
        [19, 20],
        [ 3,  4],
        [ 1,  2]]),
 array([11, 39,  7,  3]))
```

Модели машинного обучения

регрессия

- LinearRegression — линейная регрессия
- DecisionTreeRegressor — дерево решений

кластеризация

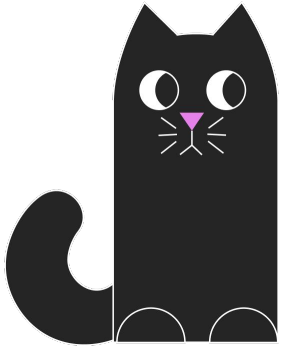
- KMeans — логистическая регрессия
- DecisionTreeClassifier — дерево решений

классификация

- LogisticRegression — логистическая регрессия
- DecisionTreeClassifier — дерево решений
- KNeighborsClassifier — k-ближайших соседей

Практическое задание

Ищите практическое задание в notebook с уроком.



Что мы узнали сегодня на уроке

- 📌 Познакомились с библиотеками анализа данных на Python;
- 📌 Pandas;
- 📌 NumPy;
- 📌 Scipy;
- 📌 Matplotlib;
- 📌 Scikit-Learn;

Вопросы?

Вопросы?

Вопросы?