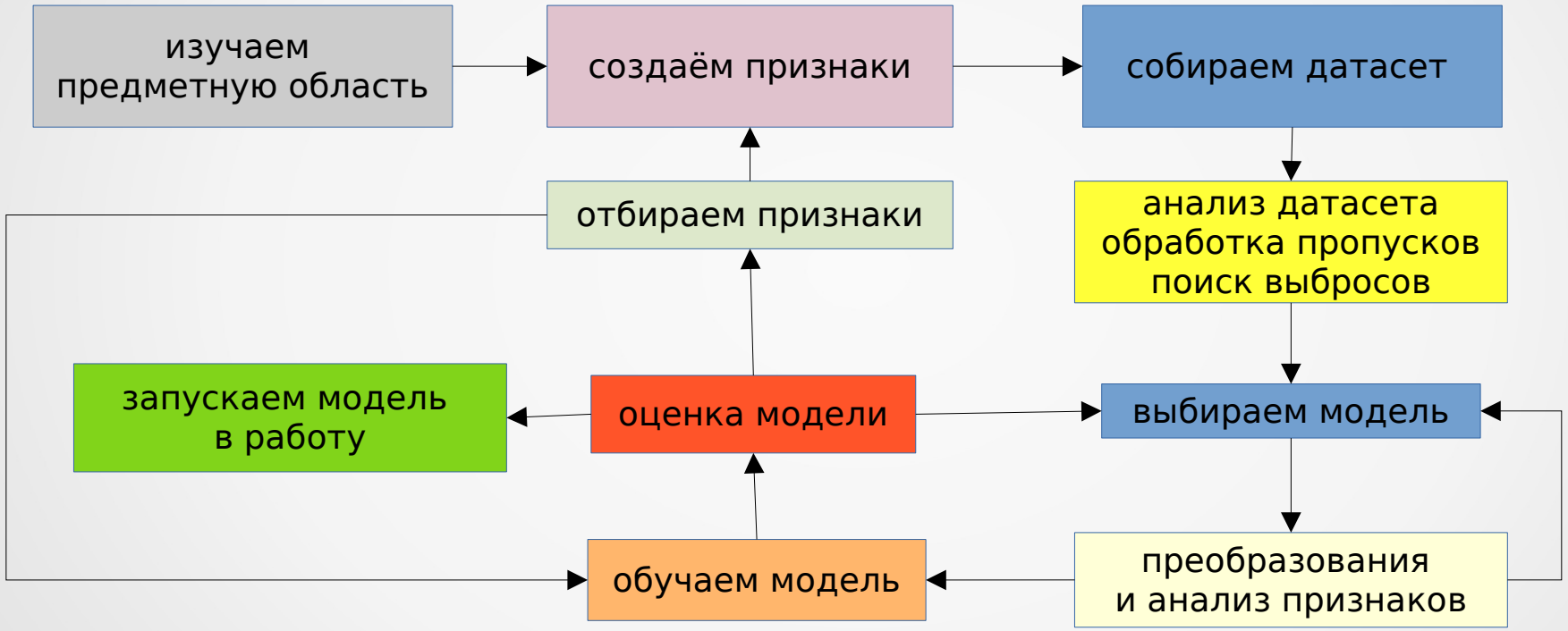
# Язык Python.
# Библиотеки для ML.

Евгений Борисов

# О методах машинного обучения



# схема применения методов ML

**создаём признаки** *( feature extraction / feature engineering )*

отображение данных, специфических для предметной области,
в точки пространства признаков

Типы признаков

- бинарные (да/нет)
- категориальные
- количественные ($\mathbb{R}$)
- порядковые

примеры признаков

для текстов
- TF-IDF
- Word2Vec

для изображений:
- Haar-like features,
- HOG (Histogram of Oriented Gradients)

собираем признаки формируем учебный датасет

# Python: библиотеки для ML

**Инструменты для работы с исходными данных**

Pandas


GeoPandas


Scikit-Image


Librosa


NLTK


**Математические методы и модели ML**

Numpy


Scikit-Learn

**Вспомогательные инструменты**


Matplotlib

SymPy

# Python: Numpy

Numerical Python https://numpy.org

Библиотека для работы с многомерными массивами

```
import numpy as np
```

```
np.array([[0,1,2],[3,4,5]],dtype=np.uint8)
```

```
array([[0, 1, 2],
       [3, 4, 5]], dtype=uint8)
```

```
np.random.rand(2,3).astype(np.float32)
```

```
array([[0.51, 0.36, 0.82],
       [0.68, 0.04, 0.89]], dtype=float32)
```

```
np.eye(i,j) # единичная матрица
```

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

**циклы не используем**

поэлементная обработка в цикле это медленно

**векторизация вычислений**

x * y - поэлементное умножение матриц

x.dot(y ) - умножение матриц

np.vstack( [y,x] ) - объединение матриц

# Python: Matplotlib

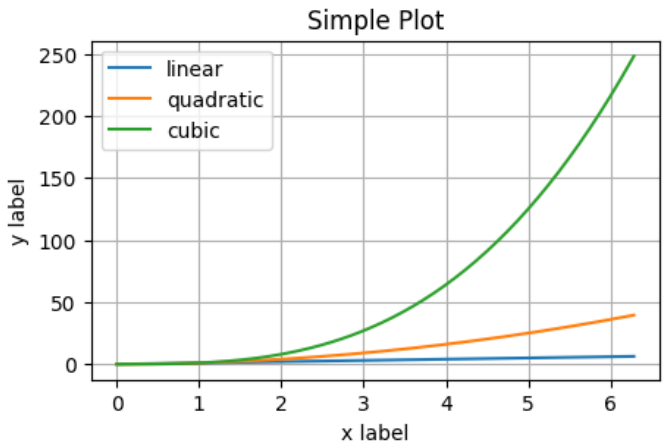
Библиотеки для визуализации данных 2D и 3D графикой

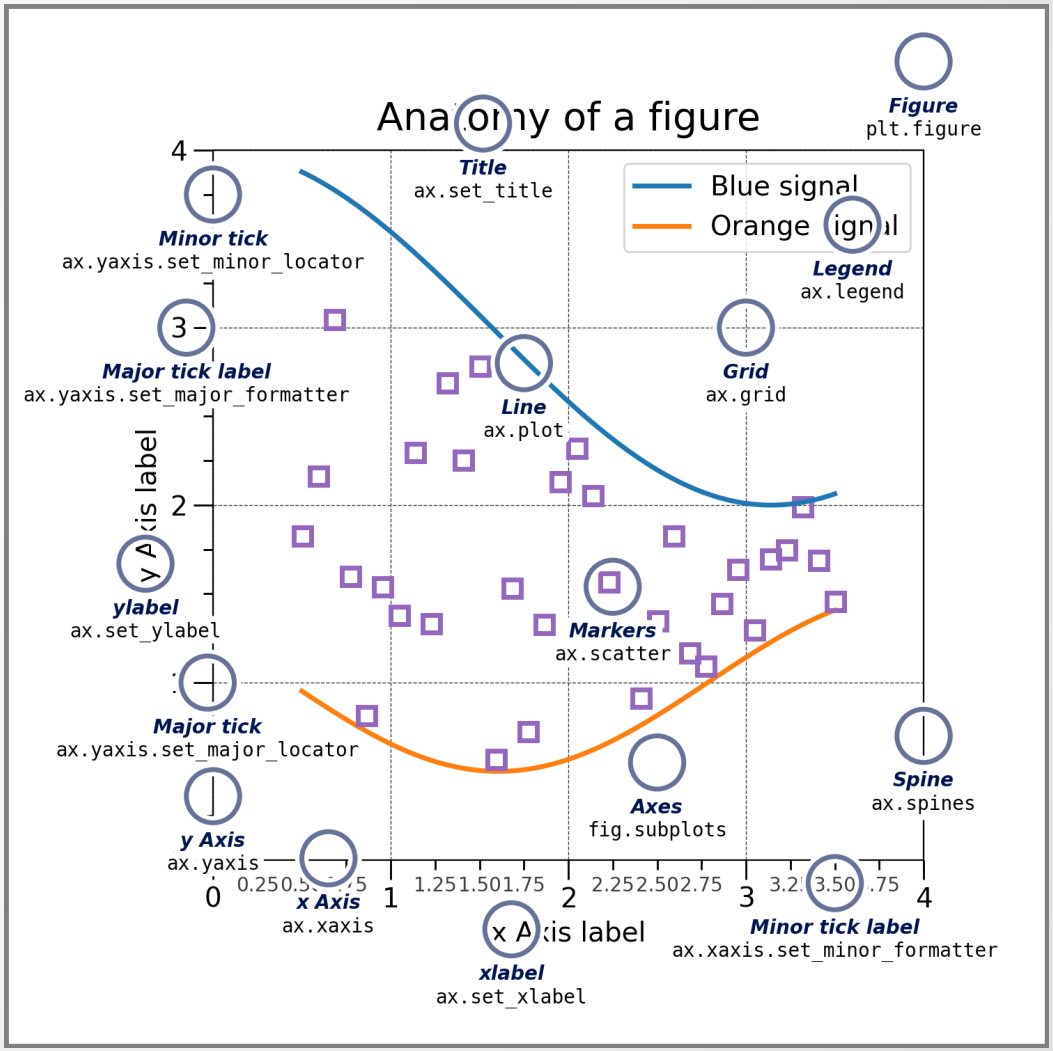
https://matplotlib.org https://seaborn.pydata.org

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

**import** seaborn **as** sns

```
fig, ax = plt.subplots(figsize=(5,3),)
ax.plot(x, x, label='linear')  # Plot some data on the axes.
ax.plot(x, x**2, label='quadratic')  # Plot more data on the axes...
ax.plot(x, x**3, label='cubic')  # ... and some more.
ax.set_xlabel('x label')  # Add an x-label to the axes.
ax.set_ylabel('y label')  # Add a y-label to the axes.
ax.set_title("Simple Plot")  # Add a title to the axes.
ax.legend()  # Add a legend.
ax.grid()
```



# Python: Pandas

Библиотека для работы с табличными данными
https://pandas.pydata.org

**import** pandas **as** pd

pd.Series — столбец

pd.DataFrame — таблица

adf

| x1 | x2 |
|---|---|
| A | 1 |
| B | 2 |
| C | 3 |

+

bdf

| x1 | x3 |
|---|---|
| A | T |
| B | F |
| D | T |

=

| x1 | x2 | x3 |
|---|---|---|
| A | 1 | T |
| B | 2 | F |
| C | 3 | NaN |

```
pd.merge(adf, bdf,
         how='left', on='x1')
```
Join matching rows from bdf to adf.

| | a | b | c |
|---|---|---|---|
| 1 | 4 | 7 | 10 |
| 2 | 5 | 8 | 11 |
| 3 | 6 | 9 | 12 |

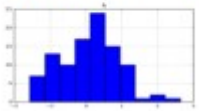
```
df = pd.DataFrame(
        {"a" : [4, 5, 6],
         "b" : [7, 8, 9],
         "c" : [10, 11, 12]},
       index = [1, 2, 3])
```

```
df = pd.DataFrame(
    [[4, 7, 10],
     [5, 8, 11],
     [6, 9, 12]],
    index=[1, 2, 3],
    columns=['a', 'b', 'c'])
```

```
df[df.Length > 7]
```
Extract rows that meet logical criteria.
```
df.drop_duplicates()
```
Remove duplicate rows (only considers columns)
```
df.sample(frac=0.5)
```
Randomly select fraction of rows.
```
df.sample(n=10)
```
Randomly select n rows.
```
df.nlargest(n, 'value')
```
Select and order top n entries.
```
df.nsmallest(n, 'value')
```
Select and order bottom n entries.
```
df.head(n)
```
Select first n rows.
```
df.tail(n)
```
Select last n rows.

```
df.query('Length > 7')
df.query('Length > 7 and Width < 8')
```

```
df.plot.hist()
```
Histogram for each column



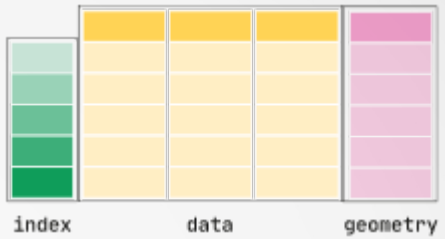
https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

# Python: GeoPandas

GeoPandas

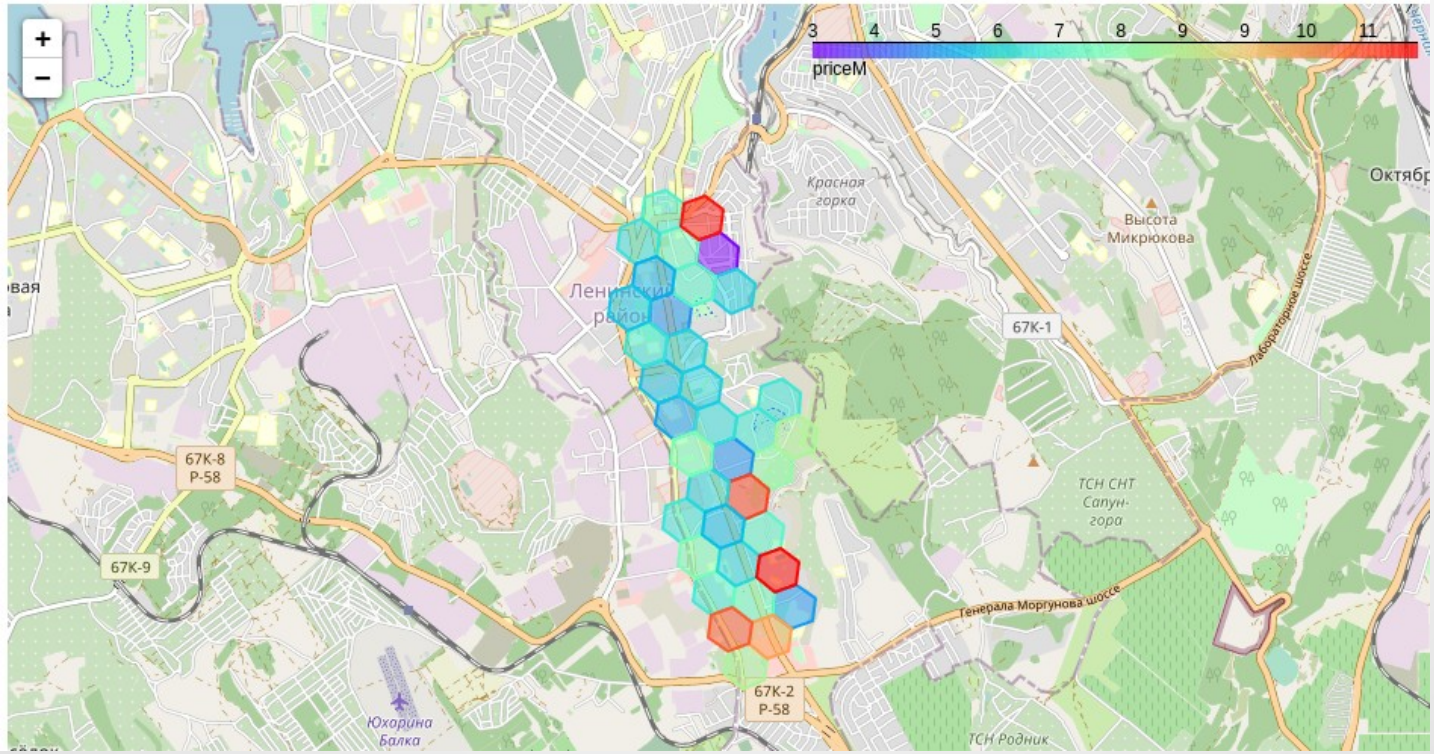
Библиотека для работы с табличными геоданными

https://geopandas.org




```
import geopandas as gpd
```

```
grid.merge(grid_values,on='hex_id').explore('priceM',cmap='rainbow') #'area_name', legend=True)
```



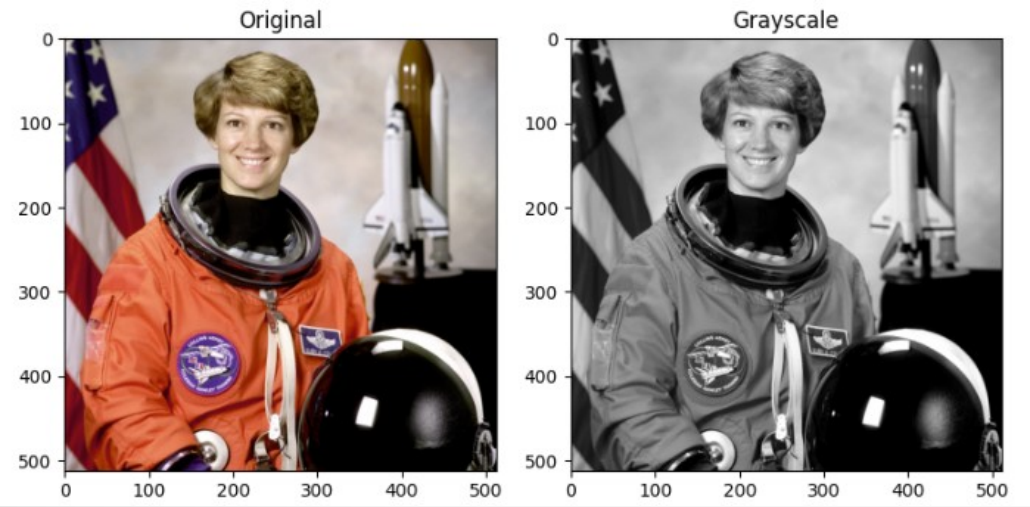
# Python: scikit image

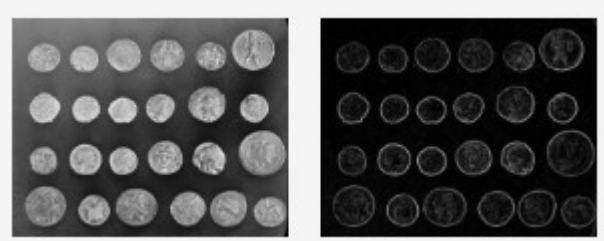
scikit-image
image processing in python

Библиотека для работы с картинками

https://scikit-image.org

**from** skimage **import** data, io, filters



```
from skimage import data, io, filters

image = data.coins()
# ... or any other NumPy array!
edges = filters.sobel(image)
io.imshow(edges)
io.show()
```

# Python: librossa


Библиотека для работы со звуком

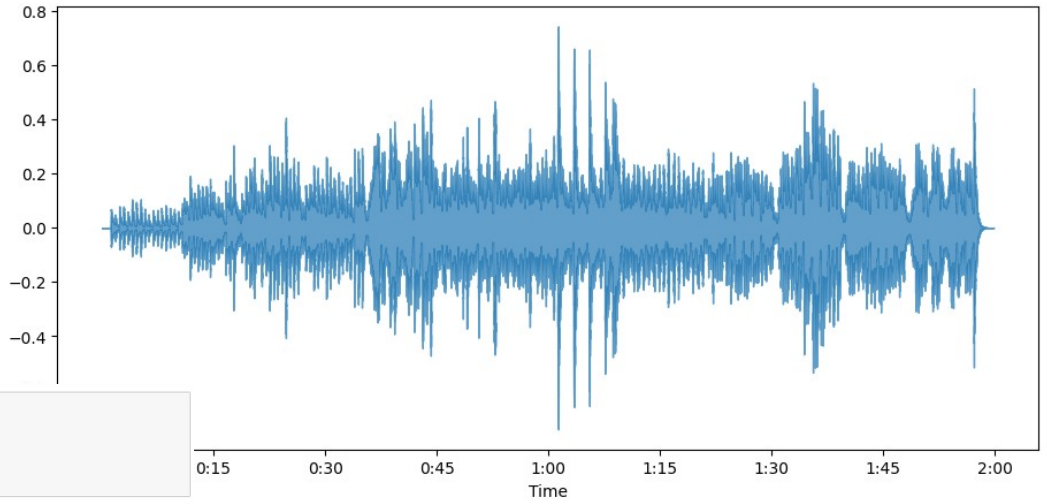
https://librosa.org

```
from IPython.display import  Audio

filename = librosa.example('nutcracker')

Audio(filename=filename,autoplay=True)
```

```
Downloading file 'Kevin_MacLeod_-_P_I_Tcha
a/audio/Kevin_MacLeod_-_P_I_Tchaikovsky_Da
```

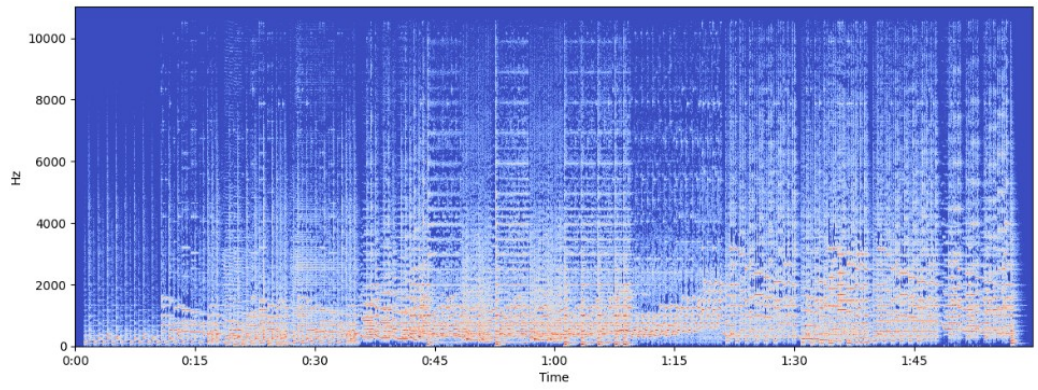

```
# librosa.display.waveshow(y)
plt.figure(figsize=(11, 5))
librosa.display.waveshow(y=y, sr=sr, alpha=.7)
```

```
<librosa.display.AdaptiveWaveplot at 0x7f854028b190>
```




```
# compute the spectrogram
X = librosa.stft(y)
Xdb = librosa.amplitude_to_db(abs(X))
plt.figure(figsize=(14, 5))
librosa.display.specshow(Xdb, sr=sr, x_axis='time', y_axis='hz')
```

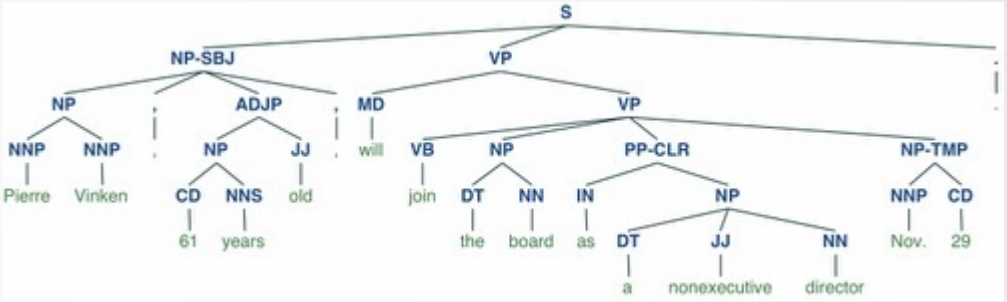
```
<matplotlib.collections.QuadMesh at 0x7f85401f93f0>
```



# Python: NLTK

Библиотека для работы с текстами

https://www.nltk.org




```
>>> import nltk
>>> sentence = """At eight o'clock on Thursday morning
... Arthur didn't feel very good."""
>>> tokens = nltk.word_tokenize(sentence)
>>> tokens
['At', 'eight', "o'clock", 'on', 'Thursday', 'morning',
'Arthur', 'did', "n't", 'feel', 'very', 'good', '.']
>>> tagged = nltk.pos_tag(tokens)
>>> tagged[0:6]
[('At', 'IN'), ('eight', 'CD'), ("o'clock", 'JJ'), ('on', 'IN'),
('Thursday', 'NNP'), ('morning', 'NN')]
```

# Python: SciKit Learn

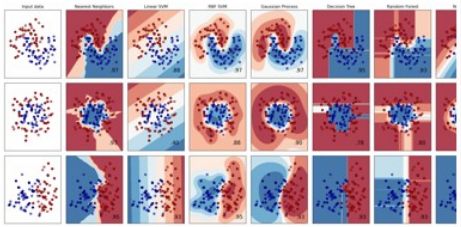
Библиотека методов ML

https://scikit-learn.org

## Classification

Identifying which category an object belongs to.

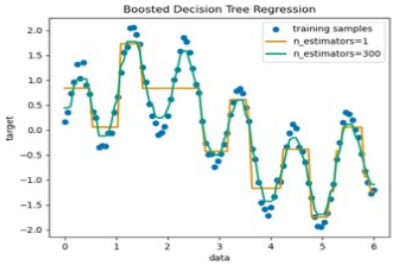
**Applications:** Spam detection, image recognition.
**Algorithms:** SVM, nearest neighbors, random forest, and more...



## Regression

Predicting a continuous-valued attribute associated with an object.

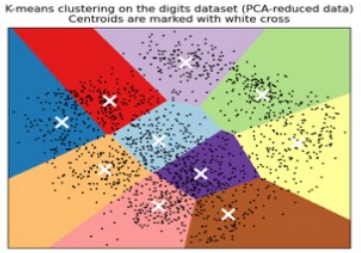
**Applications:** Drug response, Stock prices.
**Algorithms:** SVR, nearest neighbors, random forest, and more...




## Clustering

Automatic grouping of similar objects into sets.

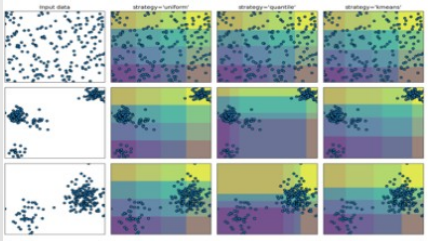
**Applications:** Customer segmentation, Grouping experiment outcomes
**Algorithms:** k-Means, spectral clustering, mean-shift, and more...




## Preprocessing

Feature extraction and normalization.

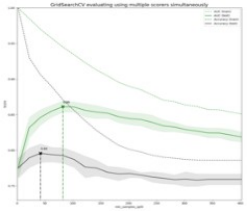
**Applications:** Transforming input data such as text for use with machine learning algorithms.
**Algorithms:** preprocessing, feature extraction, and more...



## Model selection

Comparing, validating and choosing parameters and models.

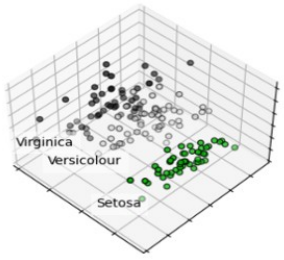
**Applications:** Improved accuracy via parameter tuning
**Algorithms:** grid search, cross validation, metrics, and more...



## Dimensionality reduction

Reducing the number of random variables to consider.

**Applications:** Visualization, Increased efficiency
**Algorithms:** PCA, feature selection, non-negative matrix factorization, and more...



# Python: SciKit Learn

SymPy - Библиотека  символьных вычислений

https://www.sympy.org

```
from sympy.abc import x,y,w,o,u
from sympy import *
```

```
# определим выражение
expr = log(1+exp(-x))
display( expr )
```

$$\log\left(1 + e^{-x}\right)$$

```
# возьмём производную и упростим результат
expr.diff(x).simplify()
```

$$-\frac{1}{e^{x} + 1}$$

# Python: упражнения numpy

https://github.com/rougier/numpy-100

## 100 numpy exercises

This is a collection of exercises that have been collected in the numpy mailing list, on stack overflow and in the numpy documentation. The goal of this collection is to offer a quick reference for both old and new users but also to provide a set of exercises for those who teach.

If you find an error or think you've a better way to solve some of them, feel free to open an issue at https://github.com/rougier/numpy-100.

File automatically generated. See the documentation to update questions/answers/hints programmatically.

Run the `initialize.py` module, then for each question you can query the answer or an hint with `hint(n)` or `answer(n)` for n question number.

In [ ]:
```
%run initialise.py
```

# Python: упражнения pandas

https://github.com/ajcr/100-pandas-puzzles/

## 100 pandas puzzles

Inspired by 100 Numpy exerises, here are 100* short puzzles for testing your knowledge of pandas' power.

Since pandas is a large library with many different specialist features and functions, these excercises focus mainly on the fundamentals of manipulating data (indexing, grouping, aggregating, cleaning), making use of the core DataFrame and Series objects.

Many of the excerises here are stright-forward in that the solutions require no more than a few lines of code (in pandas or NumPy... don't go using pure Python or Cython!). Choosing the right methods and following best practices is the underlying goal.

The exercises are loosely divided in sections. Each section has a difficulty rating; these ratings are subjective, of course, but should be a seen as a rough guide as to how inventive the required solution is.

If you're just starting out with pandas and you are looking for some other resources, the official documentation is very extensive. In particular, some good places get a broader overview of pandas are...

- 10 minutes to pandas
- pandas basics
- tutorials
- cookbook and idioms

# Python: что почитать?

Криволапов С.Я., Хрипунова М.Б. Математика на Python : учебник — Москва: КНОРУС, 2022

Лутц М. Изучаем Python. пер. с англ. — СПб.: ООО “Диалектика”, 2019.

Дейтел П., Дейтел Х. Python: Искусственный интеллект, большие данные и облачные вычисления. — СПб.: Питер, 2020.

100 numpy exercises https://github.com/rougier/numpy-100

100 pandas puzzles https://github.com/ajcr/100-pandas-puzzles/

http://github.com/mechanoid5/ml_lectorium