

SCIKIT-LEARN

(самая известная библиотека Python для машинного обучения)

scikit-learn требует наличия пакетов NumPy и SciPy.

Для построения графиков и интерактивной работы необходимо также установить matplotlib, IPython и Jupyter Notebook

Установка свободного дистрибутива Python для научных вычислений, специально предназначенного для Windows, включающего:

NumPy, SciPy, matplotlib, pandas, IPython и scikit-learn

Установка из командной строки Windows **cmd**:

Запуск **cmd**:

Запустится в веб-браузере

```
pip install numpy scipy matplotlib ipython scikit-learn pandas  
ipython3 notebook  
Jupyter notebook
```

[Удобнее работать в [JupyterLab](#):

запуск:

Запустится

инсталл.

```
pip install jupyter lab,  
jupyter lab  
JupyterLab ]
```

Jupyter Notebook, JupyterLab

Интерактивная среда для запуска программного кода в браузере.
Инструмент для анализа данных,
Позволяет легко интегрировать программный код, текст и изображения.

NumPy

Один из основных пакетов для научных вычислений в Python.
Содержит функциональные возможности для работы с многомерными массивами, высокоуровневыми математическими функциями (операции линейной алгебры, преобразование Фурье, генератор псевдослучайных чисел).
Задаёт структуру данных - массив «NumPy»

Класс ndarray, многомерный (n-мерный) массив

```
import numpy as np
x = np.array([[1, 2, 3], [4, 5, 6]])
print("x:\n{}".format(x))
x:
[[1 2 3]
 [4 5 6]]
```

SciPy

Библиотека для научных вычислений: матричные вычисления, процедуры линейной алгебры, оптимизация, обработка сигналов, статистика.

SCIKIT-LEARN использует набор функций SciPy для реализации своих алгоритмов.

Пакет `scipy.sparse` создает разреженные матрицы (sparse matrices), которые представляют собой еще один формат данных для SCIKIT-LEARN.

Разреженная матрица - это матрица с преимущественно нулевыми элементами.

Подробную информацию о разреженных матрицах SciPy можно найти в [SciPy Lecture Notes](#)

```
# (Создаем 2D массив NumPy с единицами по главной диагонали и нулями в остальных ячейках)
from scipy import sparse
eye = np.eye(4)
#numpy.eye(R, C = None, k = 0, dtype = type <'float'>) : Return a matrix having 1's on the diagonal and 0's elsewhere w.r.t. k
print("массив NumPy:\n{}".format(eye))
массив NumPy:
[[1.  0.  0.  0.]
 [0.  1.  0.  0.]
 [0.  0.  1.  0.]
 [0.  0.  0.  1.]]
```

SciPy

```
# Массив NumPy преобразуем в разреженную матрицу SciPy в формате CSR
# Compressed Sparse Row Format (CSR), Compressed Sparse Column Format (CSC)
sparse_matrix = sparse.csr_matrix(eye) # единичная - по диагонали 1, ост.0
print("\nразреженная матрица SciPy в формате CSR:\n{}".format(sparse_matrix))
разреженная матрица SciPy в формате CSR:
(0, 0) 1.0
(1, 1) 1.0
(2, 2) 1.0
(3, 3) 1.0
```

```
# Создание разреженной матрицы с использованием формата
# COO (coordinate format) – координатный формат, задаем только координаты ненулевые элементов матрицы
# (номера строк и столбцов)
data = np.ones(4)
row_indices = np.arange(4)
col_indices = np.arange(4)
eye_coo = sparse.coo_matrix((data, (row_indices, col_indices)))
print("формат COO:\n{}".format(eye_coo))
формат COO:
(0, 0) 1.0
(1, 1) 1.0
(2, 2) 1.0
(3, 3) 1.0
```

Задание: создать разреженную матрицу M , $\dim(M)=10 \times 6$, где $M_{2,4}=M_{6,4}=M_{2,5}=M_{6,6}=1$ с использованием обоих форматов. Вывести на печать, сравнить.

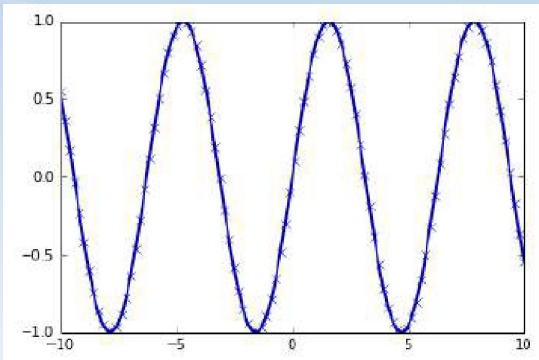
Matplotlib

Основная библиотека для построения графиков.

Включает функции для создания высококачественных визуализаций типа линейных диаграмм, гистограмм, диаграмм разброса и т.д.

При работе в Jupyter Notebook можно вывести рисунок прямо в браузере с помощью встроенных команд `%matplotlib notebook` и `%matplotlib inline`.

```
# Построение графика с использованием библиотек Matplotlib
%matplotlib inline
import matplotlib.pyplot as plt
x = np.linspace(-10, 10, 100)           # переменная X из 100 чисел от -10 до 10 (ось абсцисс)
y = np.sin(x)                          # функция от X
plt.plot(x, y, marker="x")             # построение графика
```



Pandas

Библиотека для обработки и анализа данных.

Построена на основе структуры данных DataFrame (таблицы, похожие на таблицы Excel). Имеет широкие возможности по работе с таблицами, в частности, позволяет выполнять SQL-подобные запросы.

В отличие от NumPy, который требует, чтобы все записи в массиве были одного и того же типа, в pandas каждый столбец может иметь отдельный тип (например, целые числа, даты, числа с плавающей точкой и строки).

Способна работать с различными форматами файлов и баз данных, например, с файлами SQL, Excel и CSV.

Подробная информация – в книге

McKinney W. Python for Data Analysis. Data Wrangling with Pandas, NumPy, and Ipython. O'Reilly, 2012

```
# Пример создания DataFrame таблицы
```

```
inlineimport pandas as pd
```

```
# набор данных с характеристиками пользователей
```

```
data = {'Name': ["John", "Anna", "Peter", "Linda"], 'Location' : ["New York", "Paris", "Berlin", "London"], 'Age' : [24, 13, 53, 33]}
```

```
data_pandas = pd.DataFrame(data)
```

```
display(data_pandas)
```

```
# IPython.display позволяет "красиво напечатать" таблицу
```

| | Age | Location | Name |
|---|-----|----------|-------|
| 0 | 24 | New York | John |
| 1 | 13 | Paris | Anna |
| 2 | 53 | Berlin | Peter |
| 3 | 33 | London | Linda |

в Jupyter notebook

| | Name | Location | Age |
|---|-------|----------|-----|
| 0 | John | New York | 24 |
| 1 | Anna | Paris | 13 |
| 2 | Peter | Berlin | 53 |
| 3 | Linda | London | 33 |

в JupyterLab

2. Задача классификации. OneR (one rule) алгоритм

Вспомнить:

class (target, цель)

Есть ли на фото тигр?

Болен ли пациент таким-то заболеванием?

Продается ли этот товар нужными объемами?

классификация

Обучить классификатор на известных классах так, чтобы при предъявлении ему неизвестного класса, он отнес бы его к одному из известных.

Задача: классифицировать сорта цветков ириса

Исходные данные:

features

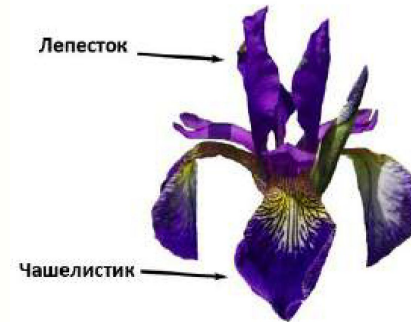
- длина и ширина лепестков (см),
- длина и ширина чашелистиков (см).

Возможные сорта *classes*

- Setosa,
- Versicolor,
- Virginica

различаются на основе перечисленных характеристик (признаков, features)

Цель: построить классификатор (модель машинного обучения), который сможет обучиться на основе перечисленных характеристик цветков ириса, классифицированных по сортам, и затем предскажет сорт для любого далее предъявляемого ему цветка ириса. *labels*



? Это обучение с учителем или без?

Поскольку есть примеры классов, то решаемая задача является задачей *обучения с учителем*

2. Задача классификации. OneR (one rule) алгоритм

Загрузить файл данных из модуля `datasets` библиотеки `scikit-learn`, вызвав функцию `load_iris`:

```
# загрузка файла данных
import numpy as np
from sklearn.datasets import load_iris
iris_dataset = load_iris()
X = iris_dataset.data
y = iris_dataset.target
```

Объект `iris` содержит ключи и значения. Просмотр структуры

```
# Структура - ключи и значения
print("Ключи iris_dataset: \n{}".format(iris_dataset.keys()))
Ключи iris_dataset:
dict_keys(['target_names', 'feature_names', 'DESCR', 'data', 'target'])

# ключ DESCR – краткое описание набора данных/ Просмотр DESCR одним из способов:
print(iris_dataset.DESCR)
# print("Ключи iris_dataset: \n{}".format(iris_dataset.keys()))
# print("Ключи iris_dataset: {}".format(iris_dataset.DESCR))

# Сами данные записаны в массивах target и data. data – массив NumPy, который содержит количественные измерения длины
# чашелистиков, ширины чашелистиков, длины лепестков и ширины лепестков:
print("Тип массива data: {}".format(type(iris_dataset['data'])))
Тип массива data: <class 'numpy.ndarray'>

# Строки в data соответствуют цветам ириса = примерам (samples), а столбцы - 4 характеристики (признака, features)
print("Форма (shape) массива data: {}".format(iris_dataset['data'].shape))
Форма (shape) массива data: (150, 4)
```

Задание 1: вывести на печать первые 5 примеров (*samples*) массива `data`

2. Задача классификации. OneR (one rule) алгоритм

Для решения задачи классификации с учителем надо иметь 2 набора данных:

-обучающие данные (training data, training set).

-тестовые данные (test data, test set, hold-out set).

Функция `train_test_split` (библиотека `scikit-learn`) перемешивает исходный набор данных случайным образом и разбивает его на две части: обучающий набор = 75% samples, тестовый набор = 25% samples

Чтобы в точности для отладки повторно воспроизвести случайное перемешивание, в генераторе псевдослучайных чисел зададим

фиксированное стартовое значение `random_state=0`

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(iris_dataset['data'], iris_dataset['target'], random_state=0)
```

форма массива X_train: (112, 4)

форма массива y_train: (112,)

форма массива X_test: (38, 4)

форма массива y_test: (38,)

Качественный анализ данных: матрица диаграмм рассеяния

Для пары признаков – на плоскости (`scatter plot`). Если признаков больше, то строятся матрицы диаграммы (`scatterplot matrix`, `pair plots`) для всех возможных пар (в `pandas` функция `scatter_matrix`)

матрица диаграмм рассеяния

создаем dataframe из данных в массиве X_train

маркируем столбцы, используя строки в `iris_dataset.feature_names`

создаем матрицу рассеяния из dataframe, цвет точек атоматом, По диагонали - гистограммы каждого признака

```
import pandas as pd
```

```
from pandas import plotting
```

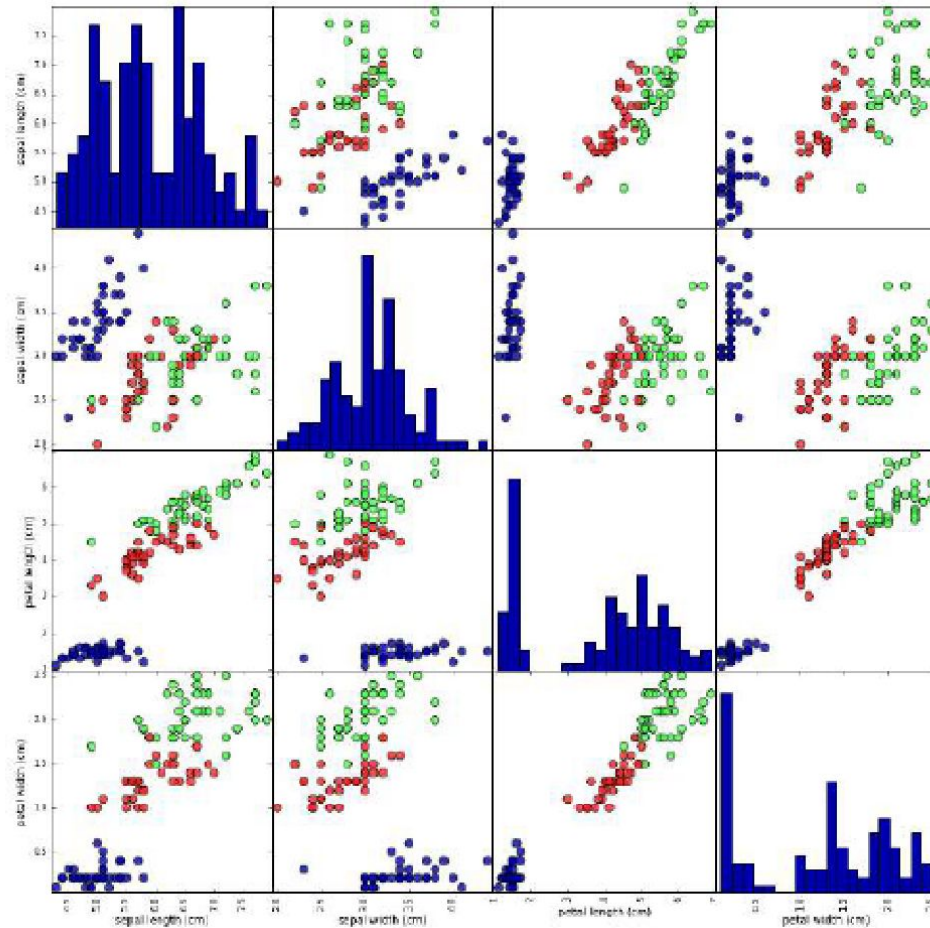
```
%matplotlib inline
```

```
iris_dataframe = pd.DataFrame(X_train, columns=iris_dataset.feature_names)
```

```
grr = plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o',
```

```
hist_kwds={'bins': 20}, s=60, alpha=.8)
```

2. Задача классификации. OneR (one rule) алгоритм



Задание 3: сделать вывод по матрицам рассеяния

**Признаки позволяют относительно хорошо разделить три класса
Модель машинного обучения, вероятно,
сможет научиться разделять их.**

2. Задача классификации. OneR (one rule) алгоритм

Для решения задачи классификации с учителем (построения классификатора) используем метод k-means (ближайших соседей)

Библиотека [scikit-learn](#), где модели машинного обучения реализованы в собственных классах, называемых [Estimator](#).

Алгоритм классификации на основе метода k ближайших соседей реализован в классификаторе [KNeighborsClassifier](#) модуля [neighbors](#).

Задание4: формализовать использование метода k-means для решения рассматриваемой задачи классификации, пояснить особенности его использования

- Тренировка выполняется на обучающем наборе данных;
- В ходе классификации вводимой точки данных алгоритм находит точку в обучающем наборе, которая ближе всего находится к вводу;
- присвоение метки (классификация, отнесение к классу) введенной новой точки.

k означает: вместо того, чтобы использовать лишь «ближайшего соседа» к вводу, рассматривается любое фиксированное число $k > 1$ соседей (например, три, или пять, или более соседей).

Т.о. классификация (прогноз, predict) для вводимой точки данных выполняется для того класса, которому принадлежит большинство из k соседей.

2. Задача классификации. OneR (one rule) алгоритм

Создать объект-экземпляр класса, задав параметры модели: количество соседей k (установим $k = 1$)

```
# Создать объект-экземпляр класса, задав параметры модели: количество соседей k= 1
```

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = KNeighborsClassifier(n_neighbors=1)
```

```
# Объект knn включает алгоритм, который будет использоваться для  
# построения модели на обучающих данных, а также алгоритм,  
# который сгенерирует прогнозы для новых точек данных
```

Для обучения вызывать метод `fit` объекта `knn`, который принимает в качестве аргументов массивы `NumPy`: `X_train` и `y_train`, содержащие обучающие и тестовые данные

```
knn.fit(X_train, y_train)
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=1, p=2,  
weights='uniform')
```

Почти все параметры классификатора `KNeighborsClassifier` имеют значения по умолчанию (параметр `n_neighbors=1` задавали).

Большинство классификаторов в `scikit-learn` имеют массу параметров, но большая часть из них связана с оптимизацией скорости вычислений или предназначена для особых случаев использования.

Не стоит подробно останавливаться на всех параметрах, выводимых классификатором