NumPy-массивы и их возможности

Владислав Порицкий

itstep.by

20 октября 2017 г.

Общие сведения

- NumPy – вычислительная библиотека, которая лежит в основе экосистемы анализа данных в Python.
- Реализована главным образом на C и Fortran.
- Импортировать принято так:

```
import numpy as np
```

- Основная структура данных – быстрый многомерный массив `np.array`.
- В отличие от списков, массивы могут содержать элементы только какого-то одного типа.
- NumPy предоставляет много встроенных типов данных и механизм для конструирования новых типов.

Инициализация массива

- Из существующих данных, например, из списка:

```
arr = np.array([[3, 15], [10, 20]])
```

- Тип можно указать явно (аргумент `dtype`).
- Пустой массив без инициализации: `np.empty`.
- Нулями: `np.zeros`.
- Единицами: `np.ones`.
- Случайными числами: `np.random.rand` и др.
- 1D массив диапазоном чисел: `np.arange` (сигнатура аналогична встроенной функции `range`).
- 1D массив числами через определённые промежутки: `np.linspace` и `np.logspace` (осторожно с округлением).
- 2D массив – единичная матрица: `np.eye`.

Типы данных

- Булевские значения: `np.bool_`
  - На хранение затрачивается 1 байт.
- Целые числа: `np.int8` ... `np.int64` (знаковые) и `np.uint8` ... `np.uint64` (беззнаковые)
- Вещественные числа: `np.float16`, `np.float32`, `np.float64`
- Поддерживаются комплексные числа.
- Можно создавать собственные типы с помощью функции `np.dtype`.

Размерность и индексация массивов

- Размерность массива: атрибут `shape`.
  - ▷ У 1D массива – 1-элементный кортеж.
  - ▷ У 2D массива – пара: количество строк, количество столбцов.
- Изменить размерность на указанную, если это возможно: метод или функция `reshape`.
- Развернуть в 1D: метод или функция `ravel`.

Размерность и индексация массивов

- Как и в последовательностях Python, поддерживаются индексы и срезы.
- У многомерного массива можно брать индексы / срезы по нескольким осям:

```
arr = np.eye(3)
arr[1:3, 2]                    # array([0, 1])
```

- Операторы сравнения, применённые к массиву и числу, дают булевозначную маску.
- По этой маске можно выбирать элементы:

```
arr = np.arange(5)
arr[arr > 2]                   # array([3, 4])
```

- Инверсия маски: оператор ~.

Операции над массивами

- Прибавление числа, умножение на число: перегруженные операторы `+` и `*`.
- Оператором `+` также можно построчно прибавлять 1D массив к 2D массиву подходящего размера.
- Транспонирование 2D массива: атрибут `T`.
- Сцепление двух массивов: `np.concatenate`.
- В частности, сцепление двух 2D массивов вдоль вертикальной или горизонтальной оси: `np.vstack`, `np.hstack`.
- Скалярное произведение векторов (1D) или матричное произведение матриц (2D): np.dot.