Lab 3: NumPy and SciPy

Data structures and Algorithms for CL III

Anna Dick, Lea Grüner

November 23, 2020

NumPy

NumPy is a python library that supports multi-dimensional arrays and matrices and provides many functions to manipulate and work with them

usually imported with `import numpy as np`

Core datastructures in NumPy are its n-dimensional arrays (ndarray)

Difference to built-in list datastructure is that they are homogeneously typed: all elements in the array have to be the same type[1]

[1] They are also faster and use less memory

Creating arrays

```
import numpy as np
>>> np.array([1, 2, 3]) # creates an np.ndarray() object
array([1, 2, 3])
>>> np.zeros(5) # np.ones() also possible
array([ 0.,  0.,  0.,  0.,  0.])
>>> np.arange(5) # np.mgrid[0:5]
array([0, 1, 2, 3, 4])
>>> np.linspace(0, 10, 5)
array([0.0, 2.5, 5.0, 7.5, 10.0])
```

Matrices

```
import numpy as np
>>> np.array([[1, 2], [3, 4]])
array([[1, 2],
       [3, 4]])
>>> np.zeros((3, 2))
array([[0., 0.],
       [0., 0.],
       [0., 0.]]
```

Matrices

```
>>> np.mgrid[0:3:1, 0:2:0.5]
array([[[0. , 0. , 0. , 0. ],
        [1. , 1. , 1. , 1. ],
        [2. , 2. , 2. , 2. ]],
       [[0. , 0.5, 1. , 1.5],
        [0. , 0.5, 1. , 1.5],
        [0. , 0.5, 1. , 1.5]]])
```

Or stack arrays with stack, hstack, vstack or dstack

Numpy.random

Creating random arrays:

```
import numpy as np
>>> rng = np.random.default_rng(0)
>>> rng.random(3)
array([0.63696169, 0.26978671, 0.04097352])
>>> rng.random((3, 2))
array([[0.01652764, 0.81327024],
       [0.91275558, 0.60663578],
       [0.72949656, 0.54362499]])

>>> np.random.uniform(-2,2,3) # default single value
[-0.92400501  1.99754071 -1.44980225]
```

SciPy

Refers to different related entities related to scientific computing in Python, but we will focus on the SciPy library

Builds on NumPy arrays

scipy.stats will probably be the most relevant for you

scipy.stats

Create various distributions that you might know from Statistics class:
Normal, Poisson, Multivariate Normal, Zipf, Bernoulli...

Compute useful statistical values:
Entropy, Mean, Variance, Standard Error

Normal Distribution

Using `norm`:
`norm` has several methods, including mean, median and pdf (probability density function)
`norm.pdf(x, loc=0, scale=1)` creates a probability density function of x, loc specifies the mean and scale the standard deviation

---

```
import numpy as np
from scipy.stats import norm
>>> x = sorted(np.random.uniform(-1,1,5))
>>> norm.pdf(x)
array([0.35541988, 0.39466687, 0.39845481,
       0.3067598, 0.25015239])
```

Multivariate Normal Distribution

Using `multivariate_normal`:
Normally distributed random variable

```
import numpy as np
from scipy.stats import multivariate_normal
>>> x, y = np.mgrid[-1:1:.25, -1:1:.25]
>>> pos = np.dstack((x, y)) # np array of two grids size 8 x 8
>>> rv = multivariate_normal.pdf(pos, mean=[0,0])
```

	0	1	2	3	4	5	6	7
0	0.05855	0.07287	0.08519	0.09356	0.09653	0.09356	0.08519	0.07287
1	0.07287	0.09068	0.10602	0.11644	0.12014	0.11644	0.10602	0.09068
2	0.08519	0.10602	0.12395	0.13613	0.14045	0.13613	0.12395	0.10602
3	0.09356	0.11644	0.13613	0.14951	0.15426	0.14951	0.13613	0.11644
4	0.09653	0.12014	0.14045	0.15426	0.15915	0.15426	0.14045	0.12014
5	0.09356	0.11644	0.13613	0.14951	0.15426	0.14951	0.13613	0.11644
6	0.08519	0.10602	0.12395	0.13613	0.14045	0.13613	0.12395	0.10602
7	0.07287	0.09068	0.10602	0.11644	0.12014	0.11644	0.10602	0.09068

References

Numpy basics:
https://numpy.org/doc/stable/user/absolute_beginners.html
scipy.stats functions:
https://docs.scipy.org/doc/scipy/reference/stats.html
Numpy.random.uniform:
https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.random.uniform.html#numpy.random.uniform scipy.stats.norm:
https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.norm.html