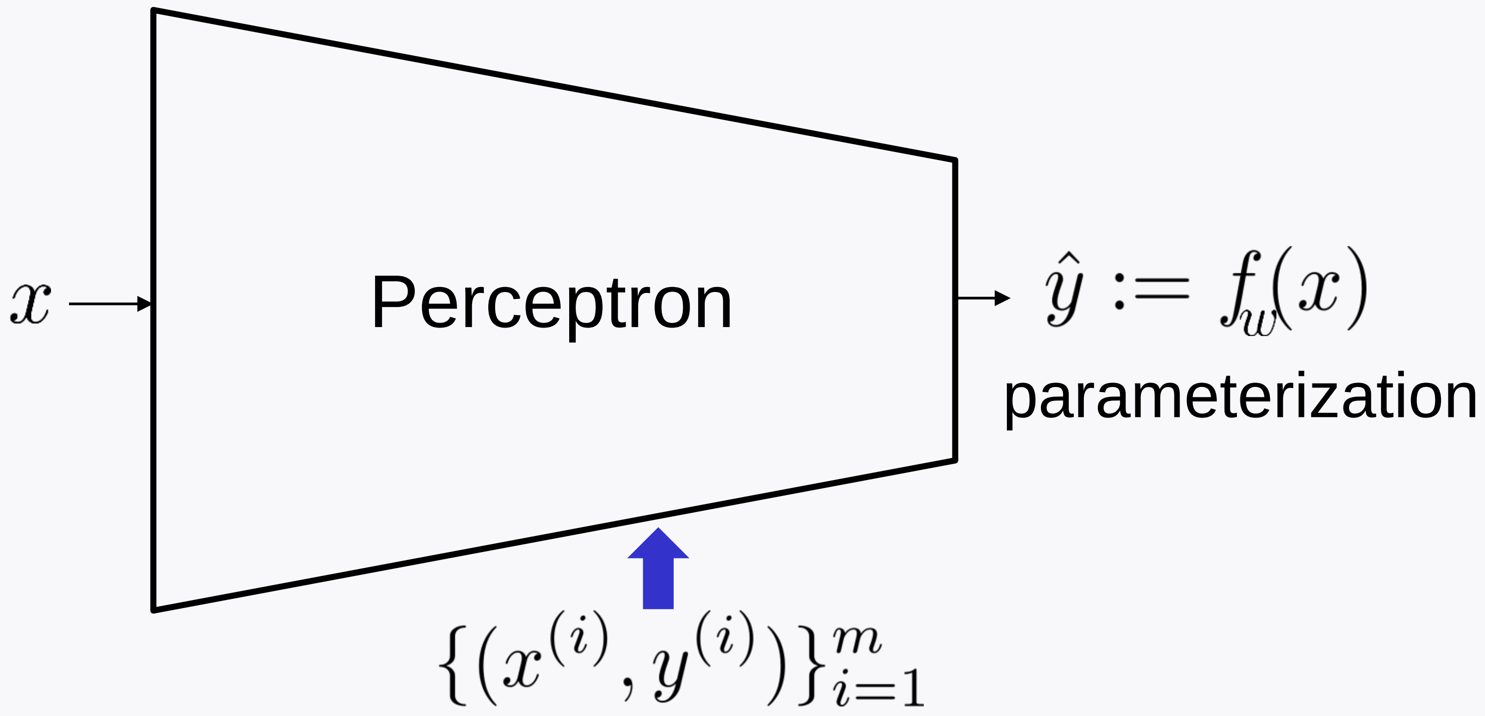
Machine learning & deep learning basics

Practice Session 1

Changho Suh

January 22, 2024

Recap: Machine learning

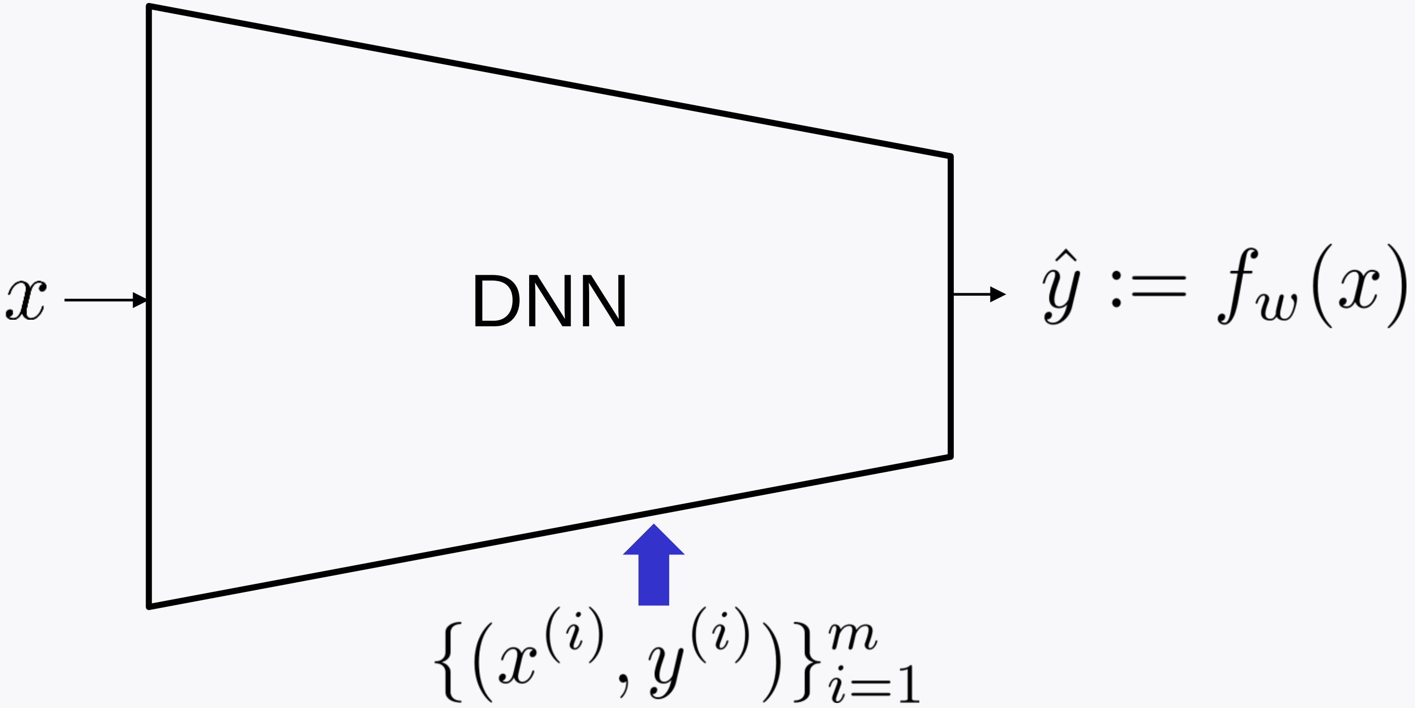


No activation + squared error loss: **Least Squares**

Logistic act. + cross entropy loss: **Logistic regression**

Algorithm: Gradient descent

Recap: Deep neural networks



ReLU (@hidden); Logistic (@output); Cross entropy loss

Algorithm: Gradient descent

Efficient method: backprop

Practical variant: Adam optimizer

Outline

Will study how to implement models via two prominent machine learning platforms:

1. Scikit-learn
2. Tensorflow

Will do this in the context of two simple tasks:

1. Iris plants classification
2. Handwritten digit classification

Focus of PS 1

Will study how to implement models via two prominent machine learning platforms:

1. Scikit-learn
2. Tensorflow

basics

together with Python basics

Will do this in the context of two simple tasks:

1. Iris plants classification
2. Handwritten digit classification

Scikit-learn (Scipy toolkit, since 2007)

A user-friendly **machine learning** library running in Python

Three key features:

1. Easy to use and open sourced
2. Offers many useful public datasets
3. Provide many useful built-in functions for machine learning

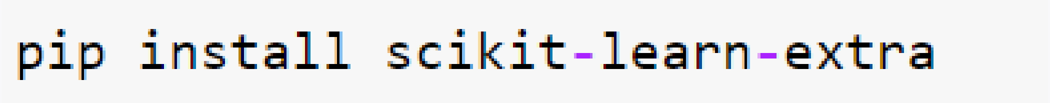
Scikit-learn installation

It comes integrated with Python.

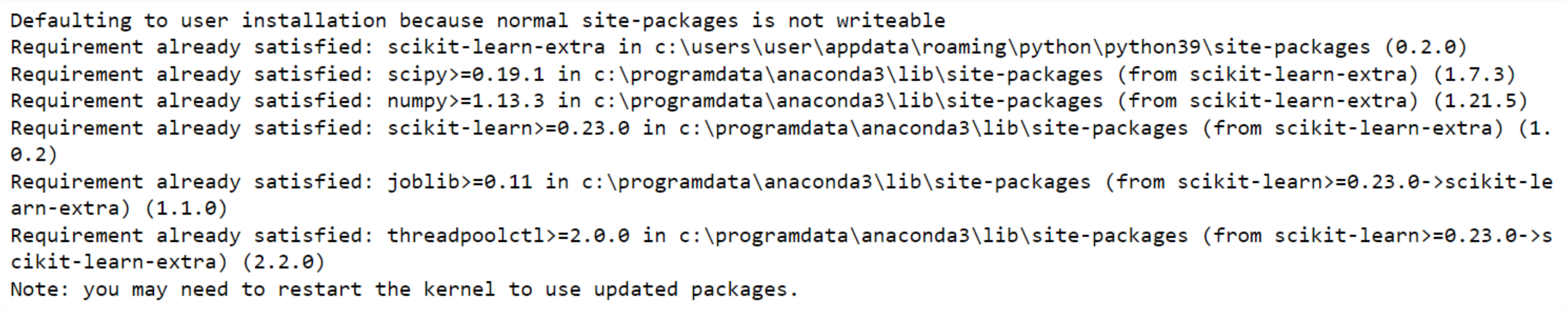
But it might not provide the latest version.

For instance, it does not provide "`sklearn_extra`" package.

Extra package installation



```
pip install scikit-learn-extra
```



```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: scikit-learn-extra in c:\users\user\appdata\roaming\python\python39\site-packages (0.2.0)
Requirement already satisfied: scipy>=0.19.1 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn-extra) (1.7.3)
Requirement already satisfied: numpy>=1.13.3 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn-extra) (1.21.5)
Requirement already satisfied: scikit-learn>=0.23.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn-extra) (1.
0.2)
Requirement already satisfied: joblib>=0.11 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn>=0.23.0->scikit-le
arn-extra) (1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn>=0.23.0->s
cikit-learn-extra) (2.2.0)
Note: you may need to restart the kernel to use updated packages.
```

Scikit-learn: Key packages

1. **`sklearn.datasets`**

Iris plants dataset	**`load_iris`**

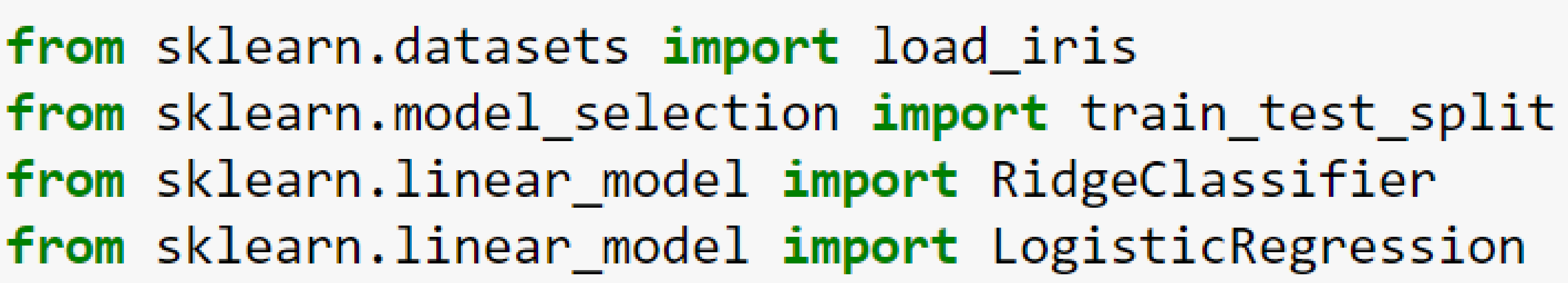
2. **`sklearn.model_selection`**

Dataset split	**`train_test_split`**

3. **`sklearn.linear_model`**

Least Squares	**`RidgeClassifier`**
Logistic regression	**`LogisticRegression`**

Import the key packages



```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import RidgeClassifier
from sklearn.linear_model import LogisticRegression
```

TensorFlow (Google Brain, since 2015)

Another Python-based library for **deep learning**

Three key features:

1. Easy to use and open sourced
2. Efficient for handling tensors and differentiation
3. Supports **Keras** that offers many public datasets & built-in functions for deep learning

TensorFlow installation

```
pip install tensorflow
```

Caution:

Not supported in ipad and iphone yet.

Keras: Key packages

1. **`tensorflow.keras.datasets`**

Handwritten digit dataset	**`mnist`**
Image classification dataset	**`cifar10`**

2. **`tensorflow.keras.models`**

Model framework	**`Sequential`**

3. **`tensorflow.keras.layers`**

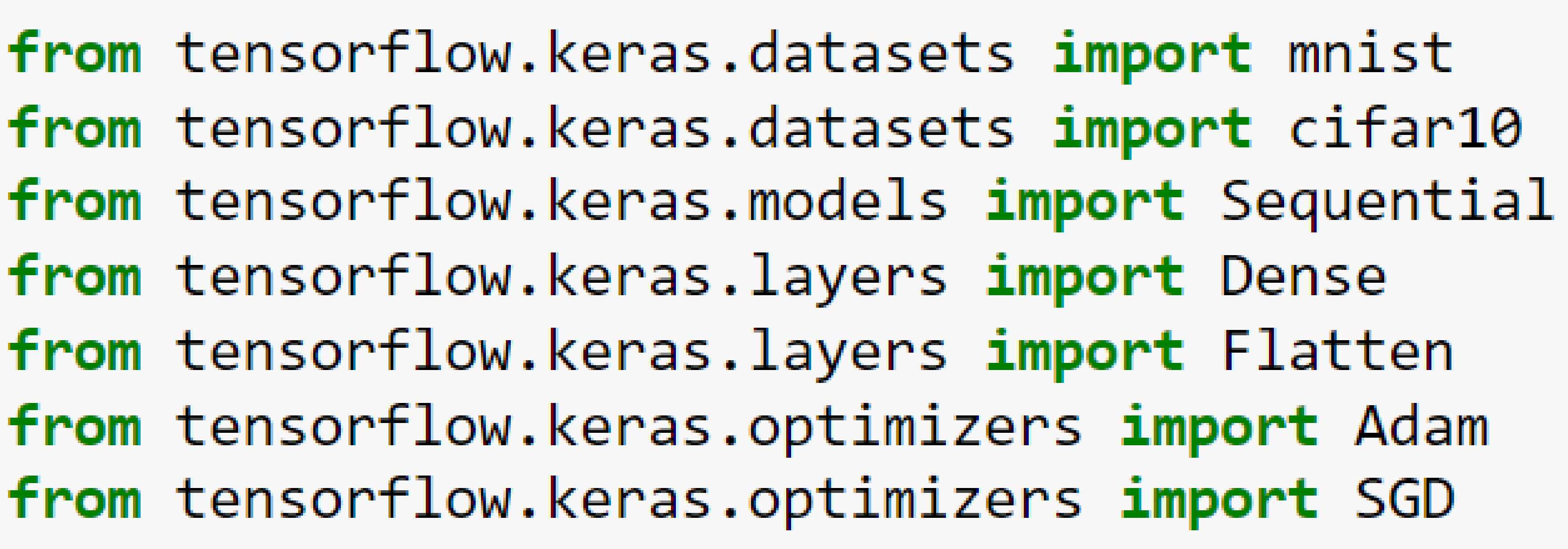
Fully connected layer	**`Dense`**
Vectorization	**`Flatten`**

Keras: Key packages

4. **`tensorflow.keras.optimizers`**

Adam optimizer	**`Adam`**
Stochastic gradient descent	**`SGD`**

Import the key packages



```python
from tensorflow.keras.datasets import mnist
from tensorflow.keras.datasets import cifar10
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.optimizers import SGD
```

Python basics

Python is an easy to use and open sourced programming language.

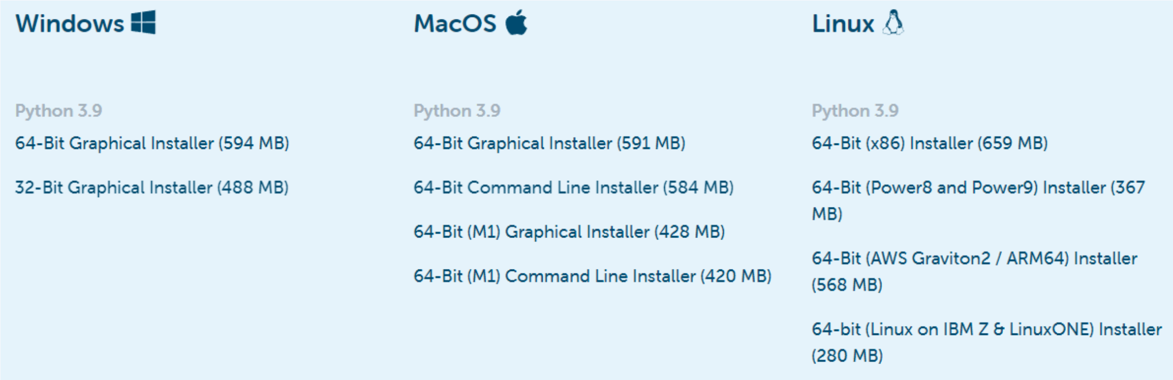
Python requires another software platform:

Jupyter notebook

Jupyter notebook installation

Three ways depending on environments:

1. Anaconda



Windows

Python 3.9

64-Bit Graphical Installer (594 MB)

32-Bit Graphical Installer (488 MB)

MacOS

Python 3.9

64-Bit Graphical Installer (591 MB)

64-Bit Command Line Installer (584 MB)

64-Bit (M1) Graphical Installer (428 MB)

64-Bit (M1) Command Line Installer (420 MB)

Linux

Python 3.9

64-Bit (x86) Installer (659 MB)

64-Bit (Power8 and Power9) Installer (367 MB)

64-Bit (AWS Graviton2 / ARM64) Installer (568 MB)

64-bit (Linux on IBM Z & LinuxONE) Installer (280 MB)

2. Google Colab (cloud-based)

   Packages already installed But limited use

3. Carnets on ipad

   TensorFlow not supported

Python: Key features

1. Easy to use and open sourced

2. Support many useful frameworks:
   Scikit-learn, TensorFlow, Keras

3. Object-oriented programming

Python: Class

Class is a type of object that consists of

1. attributes

2. methods

   Role: modifying the states of attributes

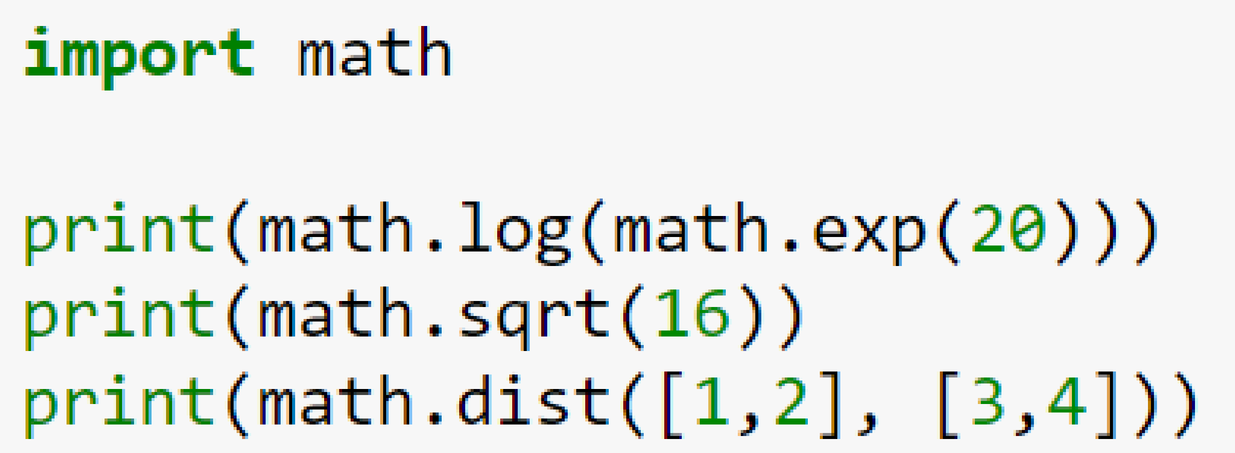
Example: Hubo is a class

attributes:
head, hand, foot

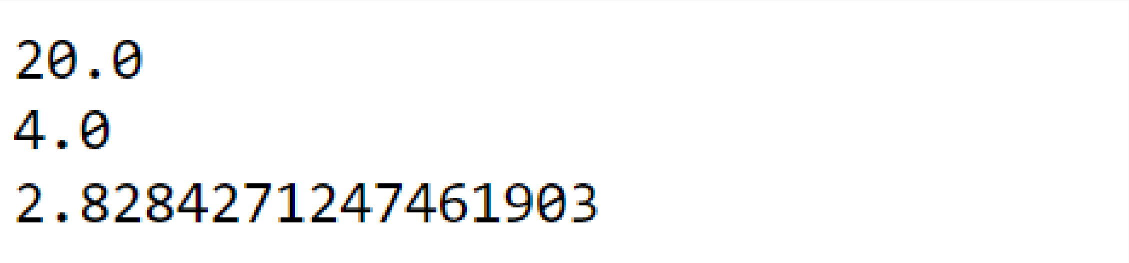
methods:
sit_down(), throw(stuff = ball)

class: Hubo

Key packages: `math`

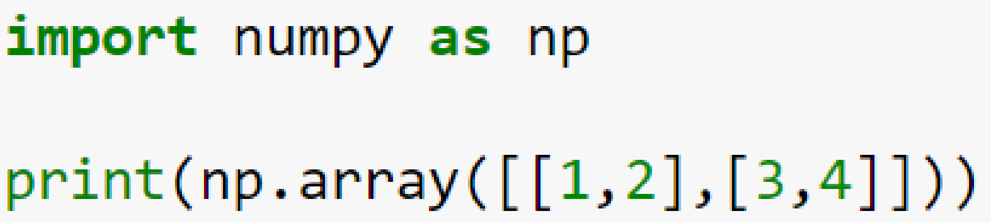


```
import math

print(math.log(math.exp(20)))
print(math.sqrt(16))
print(math.dist([1,2], [3,4]))
```

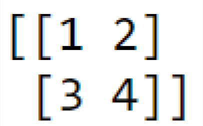


```
20.0
4.0
2.8284271247461903
```

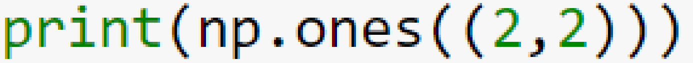
Key packages: `numpy.array`



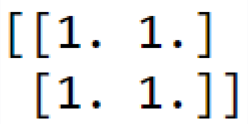
```python
import numpy as np

print(np.array([[1,2],[3,4]]))
```



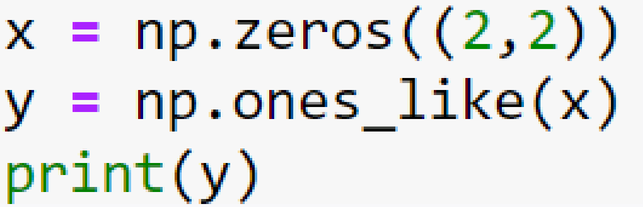
```
[[1 2]
 [3 4]]
```



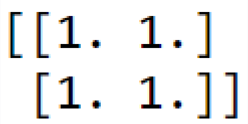
```python
print(np.ones((2,2)))
```



```
[[1. 1.]
 [1. 1.]]
```

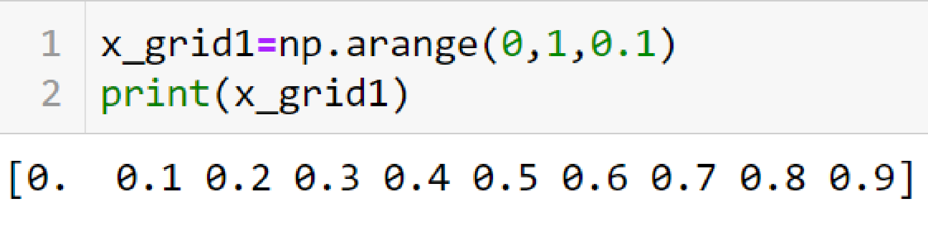


```python
x = np.zeros((2,2))
y = np.ones_like(x)
print(y)
```



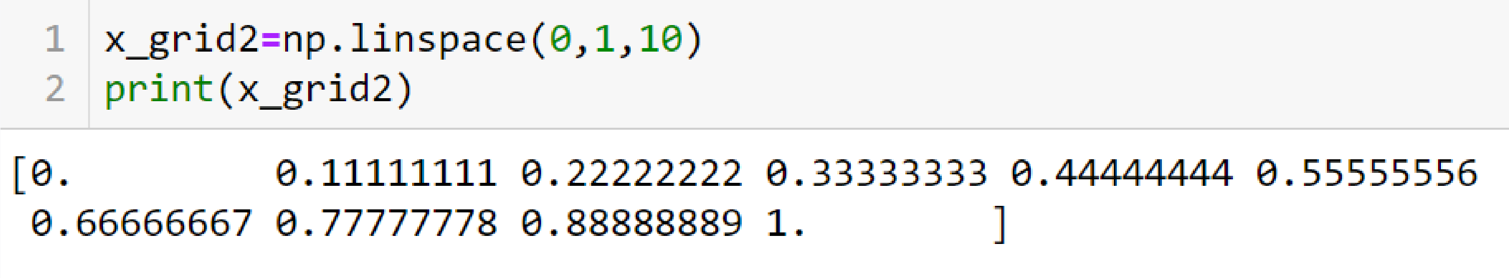
```
[[1. 1.]
 [1. 1.]]
```

Key packages: `numpy.array`



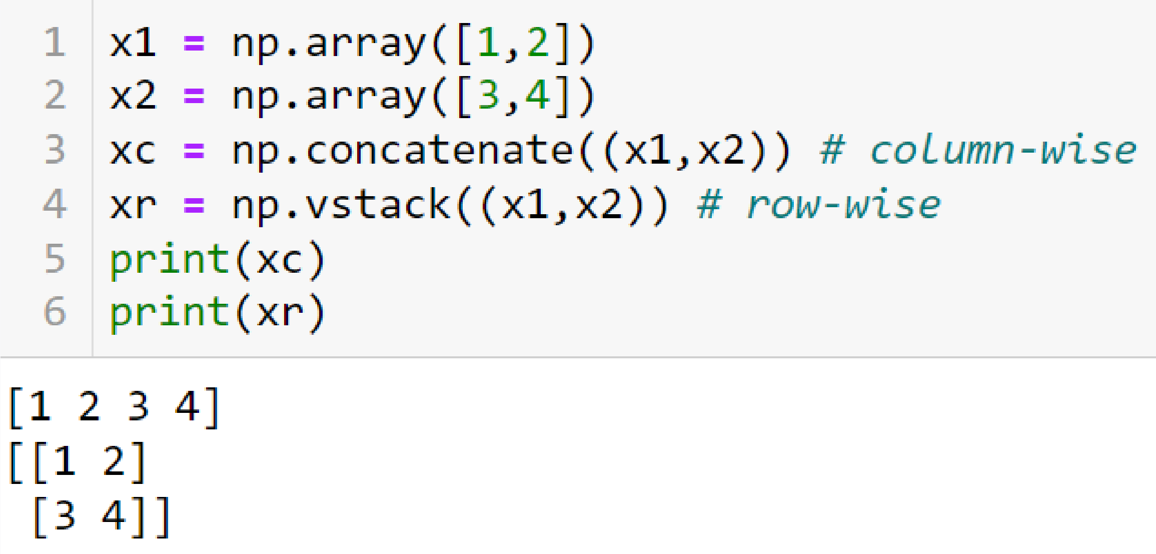
```
x_grid1=np.arange(0,1,0.1)
print(x_grid1)
```

```
[0.  0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9]
```



```
x_grid2=np.linspace(0,1,10)
print(x_grid2)
```

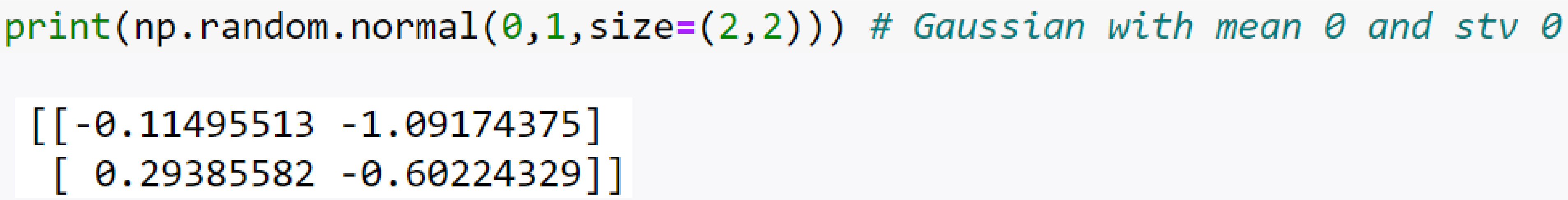
```
[0.         0.11111111 0.22222222 0.33333333 0.44444444 0.55555556
 0.66666667 0.77777778 0.88888889 1.        ]
```



```
x1 = np.array([1,2])
x2 = np.array([3,4])
xc = np.concatenate((x1,x2)) # column-wise
xr = np.vstack((x1,x2)) # row-wise
print(xc)
print(xr)
```

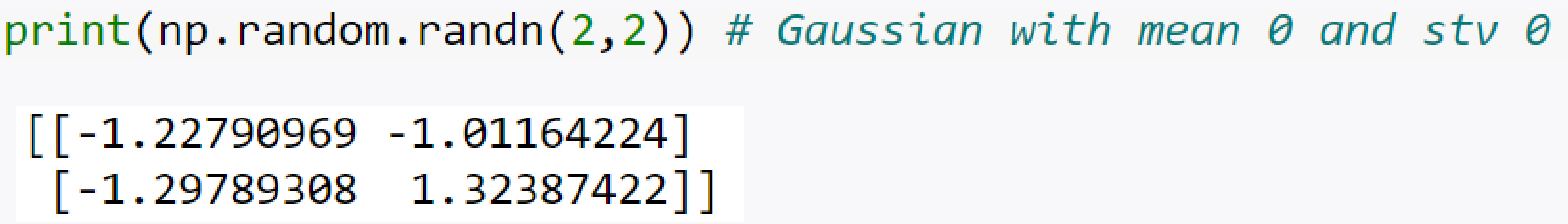
```
[1 2 3 4]
[[1 2]
 [3 4]]
```

Key packages: numpy.random



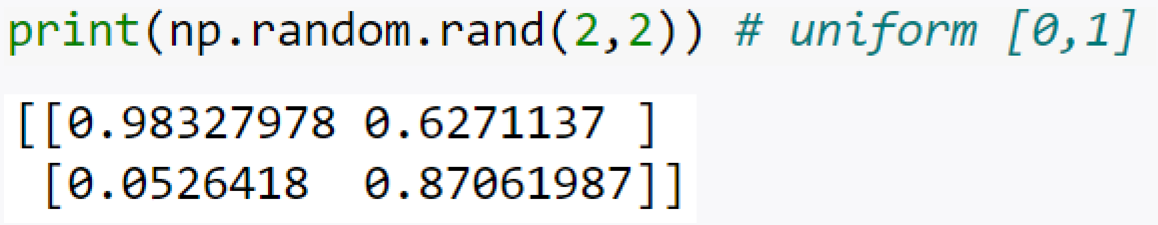
```
print(np.random.normal(0,1,size=(2,2))) # Gaussian with mean 0 and stv 0
```

```
[[-0.11495513 -1.09174375]
 [ 0.29385582 -0.60224329]]
```



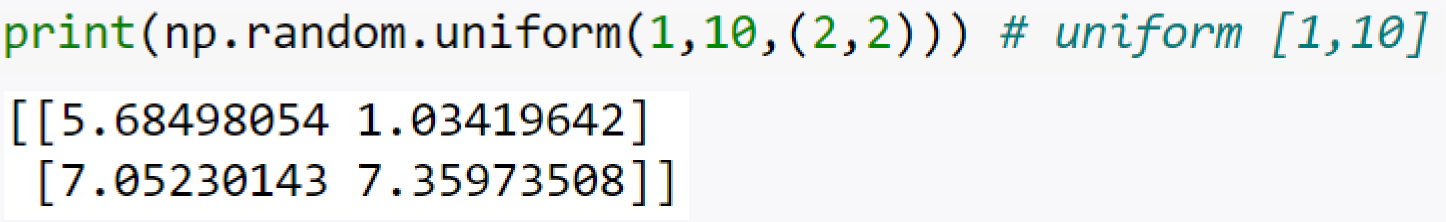
```
print(np.random.randn(2,2)) # Gaussian with mean 0 and stv 0
```

```
[[-1.22790969 -1.01164224]
 [-1.29789308  1.32387422]]
```



```
print(np.random.rand(2,2)) # uniform [0,1]
```

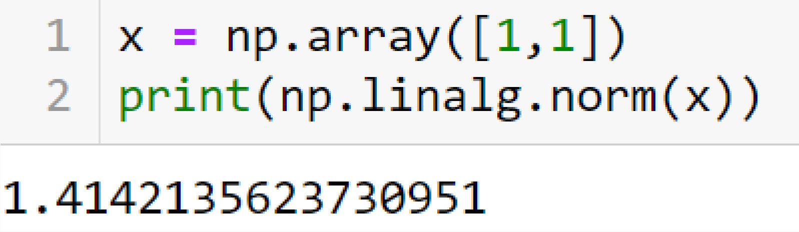
```
[[0.98327978 0.6271137 ]
 [0.0526418  0.87061987]]
```



```
print(np.random.uniform(1,10,(2,2))) # uniform [1,10]
```

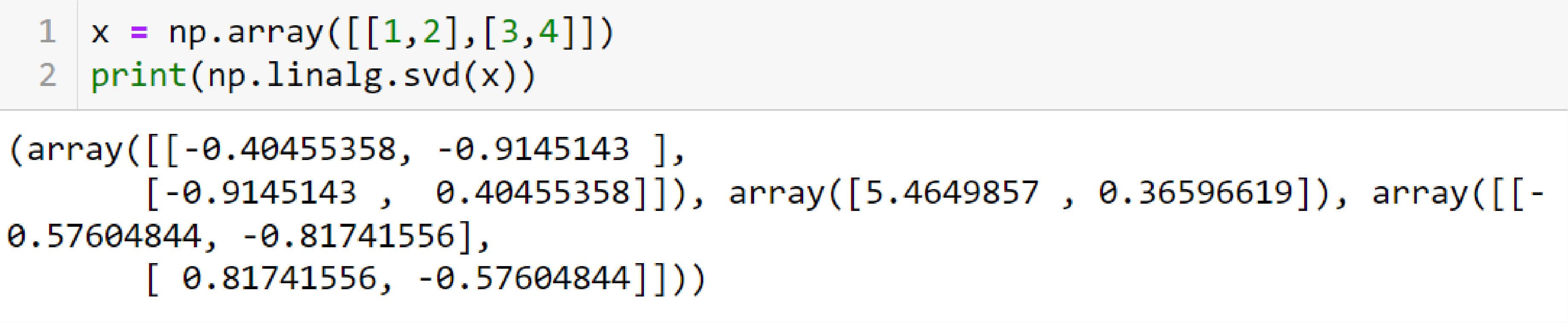
```
[[5.68498054 1.03419642]
 [7.05230143 7.35973508]]
```

Key packages: `numpy.linalg`



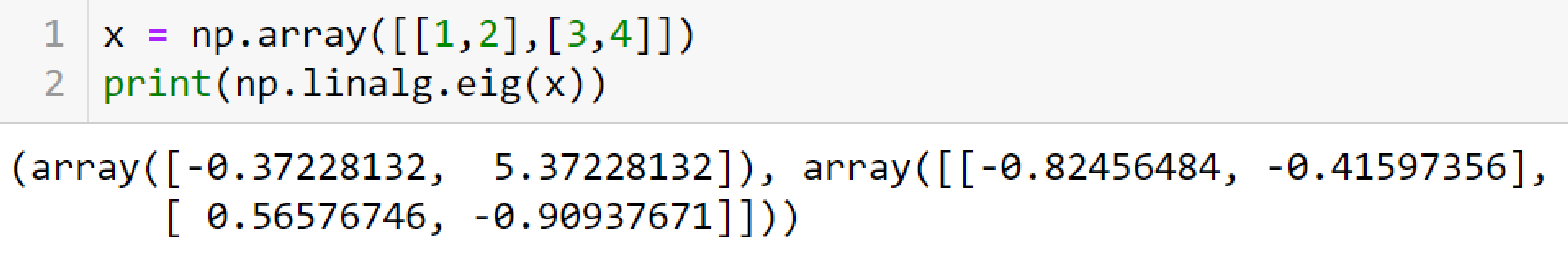
```
x = np.array([1,1])
print(np.linalg.norm(x))
```

```
1.4142135623730951
```



```
x = np.array([[1,2],[3,4]])
print(np.linalg.svd(x))
```

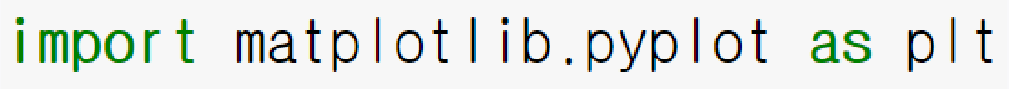
```
(array([[-0.40455358, -0.9145143 ],
       [-0.9145143 ,  0.40455358]]), array([5.4649857 , 0.36596619]), array([[-
0.57604844, -0.81741556],
       [ 0.81741556, -0.57604844]]))
```



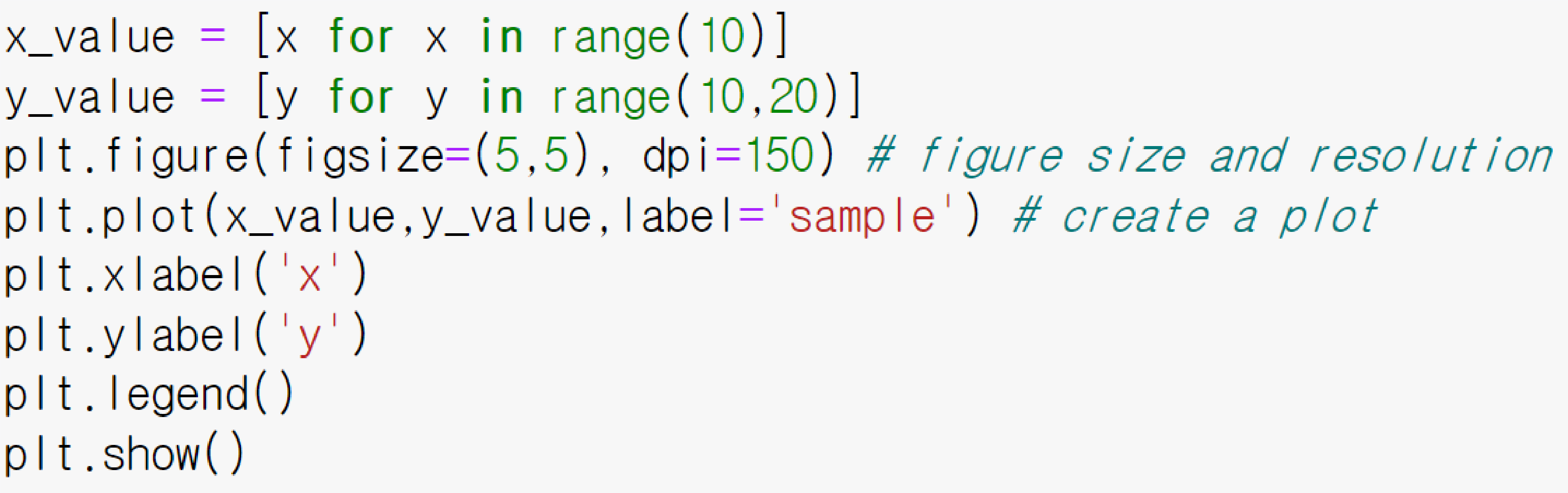
```
x = np.array([[1,2],[3,4]])
print(np.linalg.eig(x))
```

```
(array([-0.37228132,  5.37228132]), array([[-0.82456484, -0.41597356],
       [ 0.56576746, -0.90937671]]))
```

Key packages: `matplotlib`

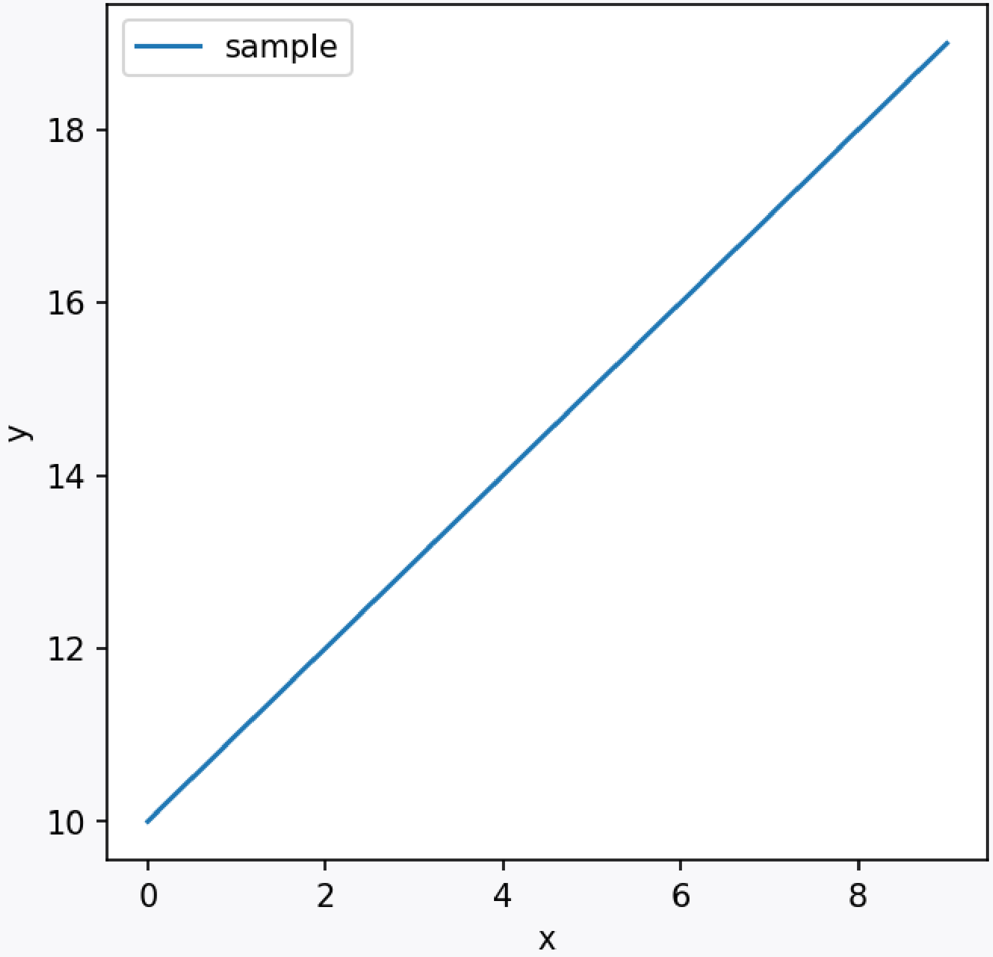


```
import matplotlib.pyplot as plt
```



```
x_value = [x for x in range(10)]
y_value = [y for y in range(10,20)]
plt.figure(figsize=(5,5), dpi=150) # figure size and resolution
plt.plot(x_value,y_value,label='sample') # create a plot
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

Key packages: `matplotlib`



Look ahead

Will learn how to implement **least squares** and **logistic regression** via **sklearn** in the context of Iris plants classification.