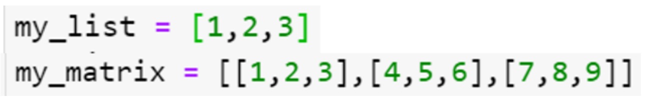
OASIS ML Group TRAINING 00

◆ Numpy

There are lots of useful library when using python. NumPy (or Numpy) is a Linear Algebra Library for Python, the reason it is so important for Data Science with Python is that almost all of the libraries in the PyData Ecosystem rely on NumPy as one of their main building blocks. Numpy has many built-in functions and capabilities. We won't cover them all but instead we will focus on some of the most important aspects of Numpy: vectors, arrays, matrices, and number generation. Please use Numpy to finish.

▫ Practice 01：

Q1：Create arrays directly converting a list or list of lists give below.



```
my_list = [1,2,3]
my_matrix = [[1,2,3],[4,5,6],[7,8,9]]
```

▫ Practice 02：Built-in Methods

There're lots of built-in ways to generate arrays.

Q1：Return evenly spaced values from zero to ten within a given interval 2.

(Hint: use "np.arange")

Q2：Generate matrix of zeros and ones in dimension 3*4.

Q3：Creates a constant number array in dimension 3*4. *Given number：20.*

(Hint: use "np. full")

Q4：Create an evenly spaced array from 0 to 10 over a specified interval 20.

(Hint: use "np. linspace")

Q5：Creates a 5*5 identity matrix

▫ Practice 03：Create random number arrays.

Q1：Create a 5*5 array and populate it with random samples from a uniform distribution over [0, 1]. *(Hint: use "np. random.rand")*

Q2：Create a 5*5 array and samples from the "standard normal" distribution.

(Hint: use "np. random.randn")

Q3：Create a 5*5 array with random integers from 0 to 100. *(Hint: use "np. randint")*

▫ Practice 04：Inspecting array.

Given array：ex_array = np.random.randint(0, 10, (3, 4))

Q1：Show the array dimension (shape).

Q2：Show the length of array.

Q3：Show the Number of array dimensions.

Q4：Show the Number of array elements.

Q5：Show the Data type of array elements.

Q6：Convert array to a different type.

Q7：Returns an array containing the same data with a new shape 2*6.

Copyright @ NCTU OASIS Lab, Shao-Wen, Cheng 2021

- **Practice 05：**

  Given array：ex_array = np.random.randint(0, 20, (3, 5))

  Q1：Find the max, min, median, mean values of the array.

- **Practice 06：**

  Given array：ex_array = np.random.randint(0, 20, (3, 4))

  Q1：Find the index locations using argmin or argmax.

  Q2：Discuss what's difference between axis=0 and axis=1.

- **Practice 07：**

  Given array：ex_array = np.random.randint(0, 20, (3, 4))

  Q1：Sort the array with axis=0 and axis=1, discuss what's difference.

  Q2：Insert a random new value after ex_array. What's happen?

  (Hint：use "append()")

  Given array：ex01_array = np.random.randint(0, 20, 10)

  Q3：Insert new value in the 2nd position of ex01_array.

  Q4：Delete value the 4th position of ex01_array. What's happen?

- **Practice 08：**

  Given array：ex_arra01, ex_arra02 = np.random.randint(0, 20, (2, 4))

  Q1：Merge two array in vertical. *(Hint：use "vstack ()")*

  Q2：Merge two array in horizontal. *(Hint：use "hstack ()")*

  Q3：Merge two array in vertical. *(Hint：use "concatenate (axis = 0) ")*

  Q4：Merge two array in horizontal. *(Hint：use "concatenate (axis = 1)")*

  Given array：arr_q5 = np.random.randint(1, 100,(6, 6))

  Q5：Split array into 3 arrays in vertical or horizontal. *(Hint: use "vsplit/ hsplit")*

- **Practice 09：**

  Given array：ex_array = np.random.randint(0, 20, 10)

  Q1：Select the 3rd elements from the array.

  Q2：Select the elements after 3rd position from the array.

  Q3：Select the 1st ~4th position elements of the array.

  Q4：Create a 2d array with 1 on the border and 0 inside

- **Practice 10：**Broadcasting.

  Given array：ex_array = np.random.randint(0, 20, 10)

  Q1：Change the values of index 0~5 to 100.

  Q2：Change all values to 20.

  Q3：Get an array which composed with ex_array[0:6].

  Q4：Change the values of array in Q3 to 10. Check ex_array, what's happened?

  Q3：Get a copy from the array. *(Hint: use ".copy()")*

Practice 11：2D array (matrices).

General format is **arr_2d[row][col]** or **arr_2d[row,col]**.

Q1：Generate a n*n 2D array .

Q2：Getting the value of (2, 4).

Q3：Show the top right corner with half of array size, which means (n/2*n/2).

Round the number if the number is divisible.

Practice 12：Fancy Indexing

Fancy indexing allows you to select entire rows or columns out of order

Q1：Get the row 6, 4, 2, 7 of given matrix.

```
#Set up matrix
arr2d = np.zeros((10,10))
#Length of array
arr_length = arr2d.shape[1]
#Set up array
for i in range(arr_length):
    arr2d[i] = i
```

Practice 13：Boolean Indexing

```
size_rand = np.random.randint(5, 15)
ex_array = np.random.randint(1, 20, size_rand)
```

Q1：Show the Boolean result in array with condition：value > 6 is True.

For example：array([True, Flase, True……])

Q2：Create array composed with selecting all values satisfied condition above.

Practice 14：Airthemetic

Given array：arr_a, arr_b = np.random.randint(1, 20, (3, 6))

Q1：Show the result of a+b, a-b, a*b, a/b, a>b, a<b

Q2：Show the result of arr_a^3, arr_a inner dot with arr_b

Q3：Show the result of asking question：

- ✓ Total sum values of arr_a.
- ✓ The maximum and minimum value of arr_a.
- ✓ Elementwise addition of arr_a.

Practice 15：

Q1：Taking square roots of arr_a in Practice 14.

Q2：Calculating exponential of arr_a in Practice 14.

Q3：Calculating with sine, cosine, log of arr_a in Practice 14.

Copyright @ NCTU OASIS Lab, Shao-Wen, Cheng 2021

Practice 16：Function Implement.

Define function, then return the value.

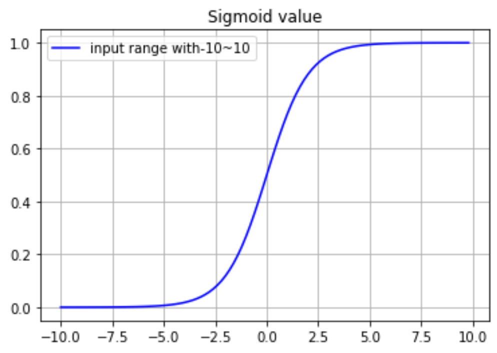
Q1：Build a function that returns the sigmoid of a real number x.

Use math.exp(x) for the exponential function. $\sigma(x) = \frac{1}{1+e^{-x}}$

After finishing define func, use the code below to check the figure is same or not.

```
x = np.arange(-10., 10., 0.2)
out = sigmoid(x)

plt.title("Sigmoid value")
plt.plot(x, out, c='b', label='input range with-10~10')
plt.grid(True)
plt.legend()
plt.show()
```



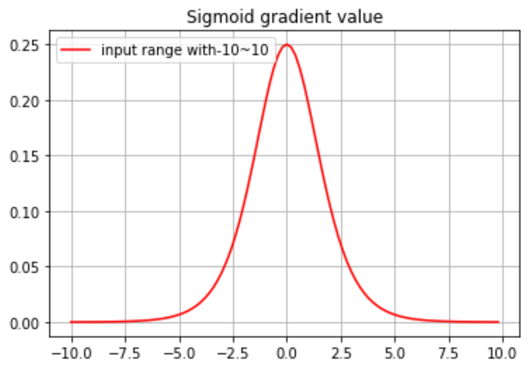
Q2：Implement the function sigmoid_grad() to compute the gradient of the sigmoid function with respect to its input x.

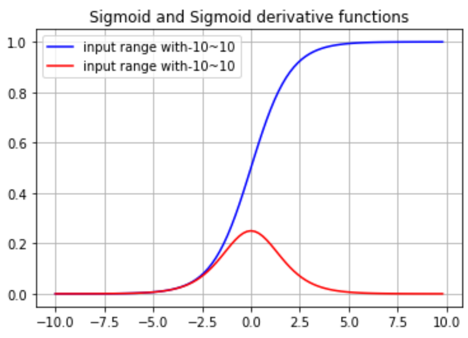
$$\textbf{sigmoid}\quad \textbf{derivate}(\mathrm{x}) = \boldsymbol{\sigma'(x) = \sigma(x)(1-\sigma(x))}$$

After finishing define func, use the code below to check the figure is same or not.

```
print("Q2：Sigmoid gradient")
out_grad = sigmoid_grad(x)

plt.title("Sigmoid gradient value")
plt.plot(x, out_grad, c='r', label='input range with-10~10')
plt.grid(True)
plt.legend()
plt.show()
```





Copyright @ NCTU OASIS Lab, Shao-Wen, Cheng 2021

Q3：Implement the numpy vectorized version of the L1 loss. You may find the function abs(x) (absolute value of x) useful.

- The loss is used to evaluate the performance of your model. The bigger your loss is, the more different your predictions (y^y^) are from the true values (yy). In deep learning, you use optimization algorithms like Gradient Descent to train your model and to minimize the cost.
- **L1 loss** is defined as:

$$L_1(\hat{y}, y) = \sum_{i=0}^{m} |y^{(i)} - \hat{y}^{(i)}|$$

hint: abs(x) = |x

After finishing define func, use the code below to show the result.

```
yhat_ = np.random.rand(10)
y_ = np.random.randint(10)
print("L1 =" , (L1(yhat_, y_)))
```

Q4： Implement the numpy vectorized version of the L2 loss.

There are several way of implementing the L2 loss but you may find the function **np.power()** useful.

- As a reminder, if x=[x1, x2,..., xn], then np.power(x,2) = $\sum_{j=0}^{n} x_j^2$

- **L2 loss** is defined as:

$$L_2(\hat{y}, y) = \sum_{i=0}^{m} (y^{(i)} - \hat{y}^{(i)})^2$$

After finishing define func, use the code below to show the result.

```
yhat_ = np.random.rand(10)
y_ = np.random.randint(10)
print("L2 =" , (L2(yhat_, y_)))
```

Copyright @ NCTU OASIS Lab, Shao-Wen, Cheng 2021

- **Practice 17：I/0**

  Q1：Generate an array then save as text file. *(Hint: use” savetxt()”)*

  Q2：Load from the text file and check correct or not. *(Hint: use” loadtxt”)*

You can also use numpy.save(), np.savez to save file, but using pandas is recommend.

You can find some reference sources form：

- NumPy v1.20 Manual
- numpy-tutorial/rougier
- numpy-tutorial/ eric496
- Python NumPy Tutorial: Learn with Code Examples

Copyright @ NCTU OASIS Lab, Shao-Wen, Cheng 2021