2274802010480_TranKhanhLinh

LAB2:NUMPY, PANDAS VÀ MATPLOTLIB NÂNG CAO

1.NUMPY

```
import numpy as np
x = np.array([1, 2, 3, 4, 5, 6, 7, 8])
#Lấy số 3
print(x[2])
#lấy giá trị 1 2 3
print(x[:3])
#lấy giá trị 3 đến 8
print(x[2:8])
#lấy từ 3 đến 7
print(x[2:7])
# cách khác
print(x[2:-1])
```

```
#lấy ra số 7 số 8 sử dụng dấu trừ
print(x[-2:])
#lấy index phần tử 2 và 4
print(x[2], x[4])

data = [[1,2],[3,4],[5,6]]
mang = np.array(data)
print(mang)
#lấy số 2
print(mang[0,1])
#lấy số 6
print(mang[-1,1])
#lấy 3 4 5 6
print(mang[1: ,:])
mang[1:]
#cách khác
print(mang[[1,2],:])
#cách khác
print(mang[1:3])
#cách khác ngoài các câu trên dấu trừ
print(mang[-2:])
#lấy 1 3 5
print(mang[:,0])
#lấy 1 và 5
print(mang[[0,2],0])
```

```python
#lấy 1 và 5 theo format start :end: step
print(mang[::2,0])
#cách khác
print(mang[[0,2],0])
print(mang[::2,0])

x = [
    [1,2,3,4],
    [5,6,7,8],
    [9,10,11,12],
    [13,14,15,16]
]
x = np.array(x)
print(x)
#lấy 2 6 10 14
print(x[:,1])
# lấy 7 và 11
print(x[1:3,2])
#lấy 4,7,10
print(x[[0,1,2],[3,2,1]])
#lấy 16 15 14 13
print(x[3,::-1])
#cách khác
print(x[-1,-1::-1])
```

```
3
[1 2 3]
[3 4 5 6 7 8]
[3 4 5 6 7]
[3 4 5 6 7]
[7 8]
3 5
[[1 2]
 [3 4]
 [5 6]]
2
6
[[3 4]
 [5 6]]
[[3 4]
 [5 6]]
[[3 4]
 [5 6]]
[[3 4]
 [5 6]]
[1 3 5]
[1 5]
[1 5]
[1 5]
[1 5]
```

```
[ 7 11]
[ 4  7 10]
[16 15 14 13]
[16 15 14 13]
```

```
#### Bài tập tết về nhà
#### Cho ma trận sau
x = [
    [99,99,99],
    [99,99,99],
    [99,99,99]
]
#### Giả sử 0 là O và 1 là X
#### nhận đầu vào từ phía X và O luân phiên
#### Cho các bạn một cặp chỉ số
#### Nếu phía O nhập ((0,0)) thì ma trận trở thành
x = [
    [X,99,99],
    [99,99,99],
    [99,99,99]
]
#### Nếu phía O nhập ((0,0)) trùng với x thì yêu cầu nhập lại và nếu không thì điền vào ma trận
#### Thử thách của các bạn ở nhà ăn tết: nếu ai đó có 3 ô liên tiếp thì dừng trò chơi
```

```python
import numpy as np

def is_valid_move(board, row, col):
    """Kiểm tra ô có hợp lệ để đi hay không (chưa bị đánh dấu)."""
    return board[row, col] == 99

def make_move(board, row, col, player):
    """Xử lý lượt đi của người chơi."""
    if player == "O" and (row, col) == (0, 0):
        board[:, :] = 99  # Đặt lại toàn bộ bảng về 99
        board[row, col] = "X"  # Đánh dấu (0,0) là "X"
    else:
        board[row, col] = player

def print_board(board):
    """Hiển thị ma trận bàn cờ."""
    for row in board:
        print(" | ".join(str(cell) for cell in row))
    print("-" * 10)  # Dòng ngăn cách

def check_winner(board, player_symbol):
    """Kiểm tra xem người chơi có thắng không."""
    for i in range(3):
```

```
        if all(board[i, j] == player_symbol for j in range(3)) or all(board[j, i] == player_symbol for j in range
            return True
    if all(board[i, i] == player_symbol for i in range(3)) or all(board[i, 2 - i] == player_symbol for i in range(3)):
        return True
    return False

def play_game():
    board = np.full((3, 3), 99, dtype=object)  # Khởi tạo ma trận 3x3
    current_player = "O"
    moves = 0

    while moves < 9:
        print_board(board)
        try:
            move = input(f"Player {current_player}, enter your move (row,col): ")
            row, col = map(int, move.replace("(", "").replace(")", "").split(","))

            if row not in range(3) or col not in range(3):
                print("Invalid input! Row and column must be between 0 and 2.")
                continue

            if is_valid_move(board, row, col):
                make_move(board, row, col, current_player)
                moves += 1
```

```
                # Kiểm tra người thắng
                if check_winner(board, "X"):
                    print_board(board)
                    print("Player X wins!")
                    return
                if check_winner(board, "O"):
                    print_board(board)
                    print("Player O wins!")
                    return

                # Đổi lượt chơi
                current_player = "X" if current_player == "O" else "O"
            else:
                print("Cell already taken, try again!")
        except (ValueError, IndexError):
            print("Invalid input! Please enter move in format (row,col).")

    print_board(board)
    print("It's a draw!")

play_game()
```

```
99 | 99 | 99
99 | 99 | 99
99 | 99 | 99
----------
99 | 99 | 99
O | 99 | 99
99 | 99 | 99
----------
99 | 99 | 99
O | X | 99
99 | 99 | 99
----------
X | 99 | 99
99 | 99 | 99
99 | 99 | 99
----------
```

```
----------
Player O, enter your move (row,col): (2,1)
99 | O | 99
X | O | 99
99 | O | X
----------
Player O wins!
```

```
----------
Player X, enter your move (row,col): (2,2)
99 | O | X
99 | O | X
O | 99 | X
----------
Player X wins!
```

```
----------
Player O, enter your move (row,col): (2,2)
X | O | O
O | X | X
O | X | O
----------
It's a draw!
```

```python
#Cho ma trận sau:
import numpy as np
x = np.array([[1,2,3],
              [4,5,6],
              [7,8,9]])
#lấy 4 5 6
print(x[1,:])
#lấy 2 5
print(x[:2,1])
#lấy 3 4
print(x[[0,1],[2,0]])
#lấy 9,6,3
print(x[::-1,2])
print(x[::-1,2])
```

```
[4 5 6]
[2 5]
[3 4]
[9 6 3]
[9 6 3]
[2, 4, 6, 8, 10]
```

```python
import numpy as np

x = np.array([1,2,3,4,5,6,7,8,9,10])
#Lấy phần tử mảng chẵn
mang_chan = [ num for num in x if num % 2 == 0]
print(mang_chan)
#cách khác
print(x[1::2])
# Tạo ma trận với toàn số 1 với kích thích chỉ định
mang = np.ones((3,3))
print(mang)
np.arange(3)
# tạo ma trận 1 2 3 xen kẽ
mt =np.ones((3,3)) + np.arange(3)
print(mt)
#x =np.array()
#print(x)
#x.shape()
```

```
[2, 4, 6, 8, 10]
[ 2  4  6  8 10]
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
[[1. 2. 3.]
 [1. 2. 3.]
 [1. 2. 3.]]
```

```
    x = np.arange(3)
    x = x.reshape((3,1))
    x
```

```
array([[0],
       [1],
       [2]])
```

```
    #tạo ma trận tăng dần theo chiều dọc
    x = x.reshape((3,1)) + np.arange(3)
    print(x)
```

```
[[0 1 2]
 [1 2 3]
 [2 3 4]]
```

Tạo 1 mảng numpy có kích thước 150x5 hãy tưởng tượng mảng này chứa 150 mẫu về chiều cao, cân nặng, tuổi, [illegible] văn lang.

Chia mảng 4 cột đầu tiên thành 1 biến có tên là x và cột cuối cùng thành y

Chia x thành x_train và x_test chứa 70% dữ liệu và chia y thành y_train và y_test. Trong đó y_train chứa 70% dữ liệu.

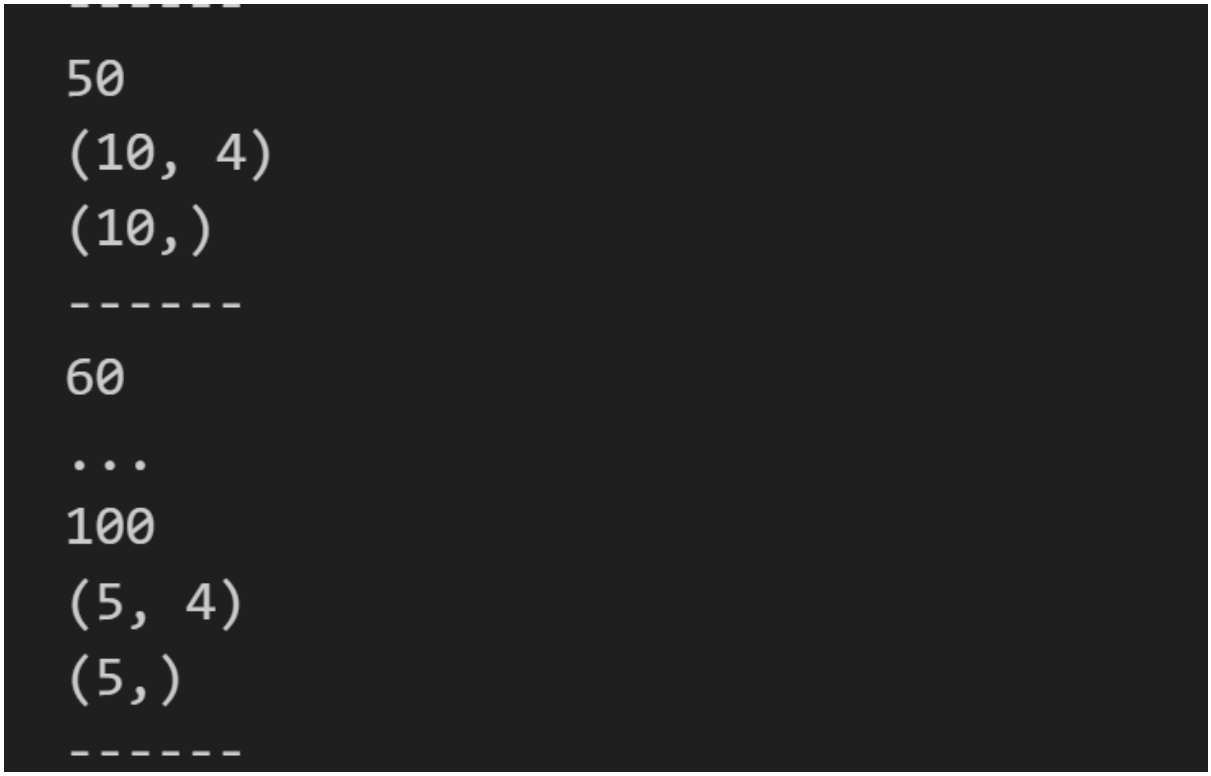
Tạo 10 cặp không chồng chéo của X_train

```
data = np.random.uniform(0,10,(150,5))
data.shape
x = data[:,:-1]
y = data[:,-1]
x.shape, y.shape
#Chia x thành x_train và x_test chứa 70% dữ liệu và chia y thành y_train và y_test. Tro
dataset_size = x.shape[0]
dataset_size
train_size = int(dataset_size * 0.7)
train_size
x_train = x[:train_size]
x_test = x[train_size:]
y_train = y[:train_size]
y_test = y[train_size:]
x_train.shape, x_test.shape, y_train.shape, y_test.shape
#tạo 10 cặp không chồng chéo của x_train
split = x_train.shape[0] // 10
split
```

```
10
```

```
for counter in range(0,x_train.shape[0],split):
    print(counter)
    print(x_train[counter:counter+split].shape)
    print(y_train[counter:counter+split].shape)
    print("------")
```

```
0
(10, 4)
(10,)
------
10
(10, 4)
(10,)
------
20
(10, 4)
(10,)
------
30
(10, 4)
(10,)
------
40
(10, 4)
(10,)
```



Nhận xét:
Input :

Bài toán mô phỏng một tập dữ liệu về 150 với 5 đặc điểm **chiều cao, cân nặng, tuổi, lương, GPA**.

Phân chia dữ liệu:
X biến độc lập: Chứa 4 cột đầu chiều cao, cân nặng, tuổi, lương.Y biến phụ thuộc: Chứa cột GPA để dự đoán.
Chia dữ liệu thành tập huấn luyện và kiểm tra:70-30%
Tạo 10 cặp không chồng chéo từ X_train: Đảm bảo các phần tử không bị trùng lặp.

Output: x_train, x_test, y_train, y_test

10 tập không chồng chéo từ x_Train