

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

DEEP LEARNING TRONG KHOA HỌC DỮ LIỆU
LAB 1

LÀM QUEN VỚI MẠNG NEURAL CƠ BẢN

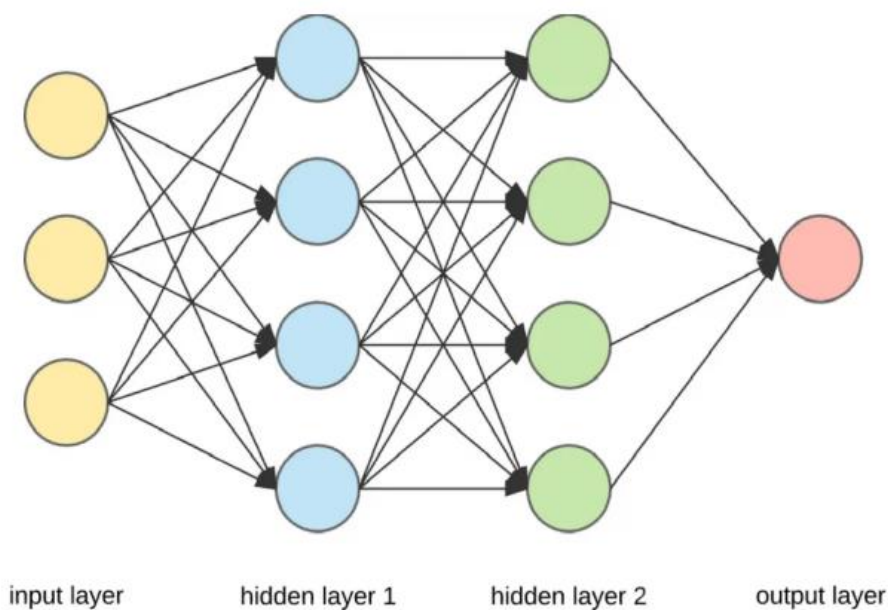
Họ và tên : Lưu Quang Tiến Hoàng

MSSV : 20521342

Lớp : DS201.N11

Mục tiêu

- Ôn tập những kiến thức cơ bản về mạng neural.
- Cài đặt thử nghiệm mô hình mạng neural đơn giản.
- Áp dụng mô hình xây dựng được vào bài toán phân loại ảnh chữ số viết tay.



Minh họa cấu trúc của một mạng neural cơ bản

(?) Mô hình mạng neural trong hình trên có những loại layer nào ?

Mô hình mạng neural ở trong hình trên có ba loại layer:

- Input layer: Là phần dữ liệu mà đã được đưa vào mạng neural để tính toán
- Hidden layer: Là phần layer được tính toán dựa trên dữ liệu được đưa vào
- Output layer: Là phần layer cuối cùng trong mạng neural nơi thu được các dự đoán mong muốn

I, LOAD DỮ LIỆU

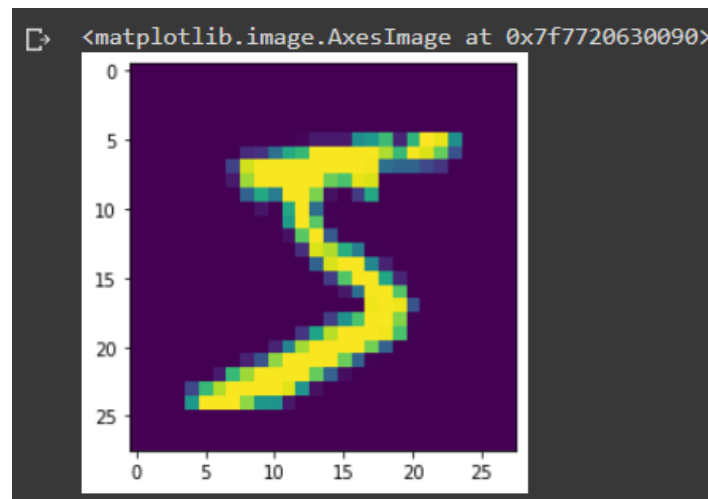
(?) MNIST là viết tắt của cụm từ nào ?

MNIST là viết tắt từ Modified National Institute of Standards and Technology database.

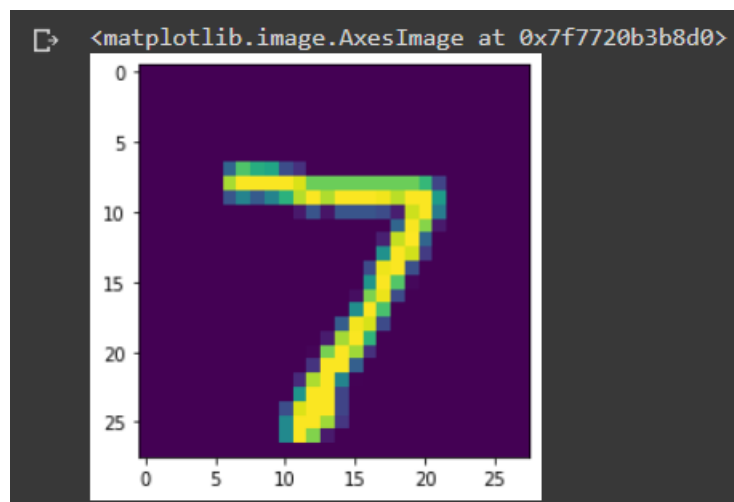
II, CHUẨN BỊ DỮ LIỆU

(?) Sử dụng thư viện Matplotlib để trực quan hóa ảnh train[0] và ảnh test[0] ?

- Train[0] :



- Test[0] :



(?) Hãy khảo sát bộ dữ liệu MNIST do thư viện Keras cung cấp:

- Xác định số lượng ảnh của tập train và tập test ?
- Tính tỉ lệ train : test ?
- Xác định kích thước của mỗi ảnh trong tập train và tập test ?
- Xác định số phần tử ảnh của mỗi ảnh trong tập train và tập test ?

Số lượng ảnh tập train và tập test:

```
[4] X_train.shape  
  
(60000, 28, 28)  
  
[5] X_test.shape  
  
(10000, 28, 28)
```

Tỉ lệ train / test :

```
X_train.shape[0]/X_test.shape[0]  
  
6.0
```

Kích thước của mỗi ảnh trong tập train và tập test là :

Tập train:

```
[66] print(f'{ X_train.shape[1] } * {X_train.shape[2]}')  
  
28 * 28
```

Tập test:

```
[67] print(f'{ X_test.shape[1] } * {X_test.shape[2]}')  
  
28 * 28
```

Số phần tử ảnh của mỗi ảnh trong tập train và tập test là:

Tập train:

```
[68] X_train.shape[1]*X_train.shape[2]
```

784

Tập test:

```
X_test.shape[1]*X_test.shape[2]
```

784

(?) Kiểu dữ liệu set có những tính chất gì ?

- Các phần tử trong tập hợp không có thứ tự.
- Các phần tử này là duy nhất, không cho phép lặp lại.
- Set có thể thay đổi (thêm bớt phần tử) nhưng các phần tử của tập hợp phải ở dạng không thể thay đổi (tức là xác định được dung lượng bộ nhớ ngay khi khai báo).

(?) Xác định số lượng nhãn và liệt kê các nhãn có trong tập train ?

Bao gồm 10 nhãn :

```
[182] labels
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
[183] list_labels=dict()
```

```
for l in labels:  
    list_labels[l] = list(y_train).count(l)
```

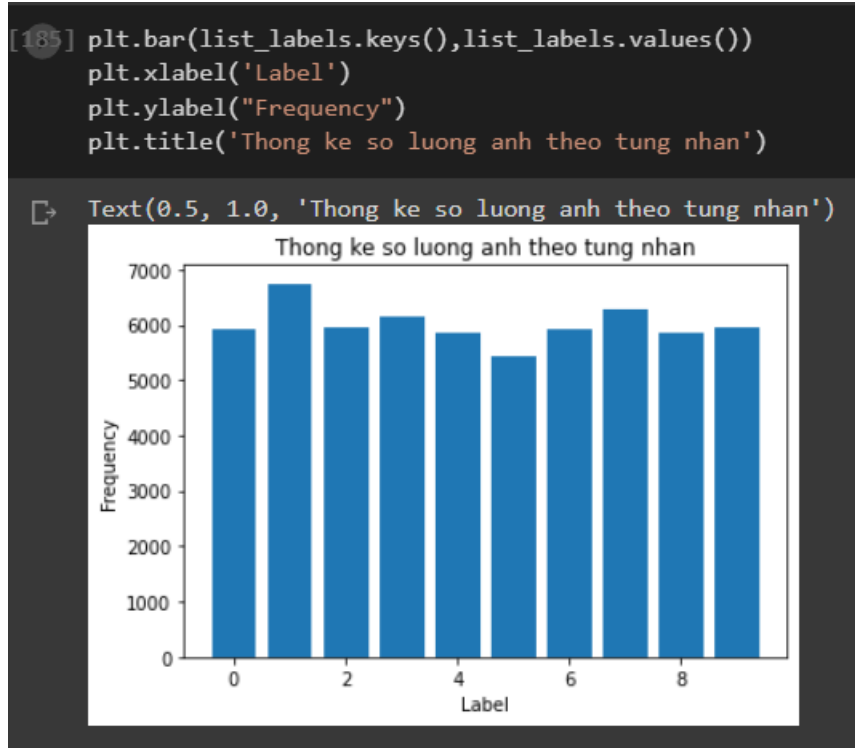
```
[184] list_labels
```

```
{0: 5923,  
1: 6742,  
2: 5958,  
3: 6131,  
4: 5842,  
5: 5421,  
6: 5918,  
7: 6265,  
8: 5851,  
9: 5949}
```

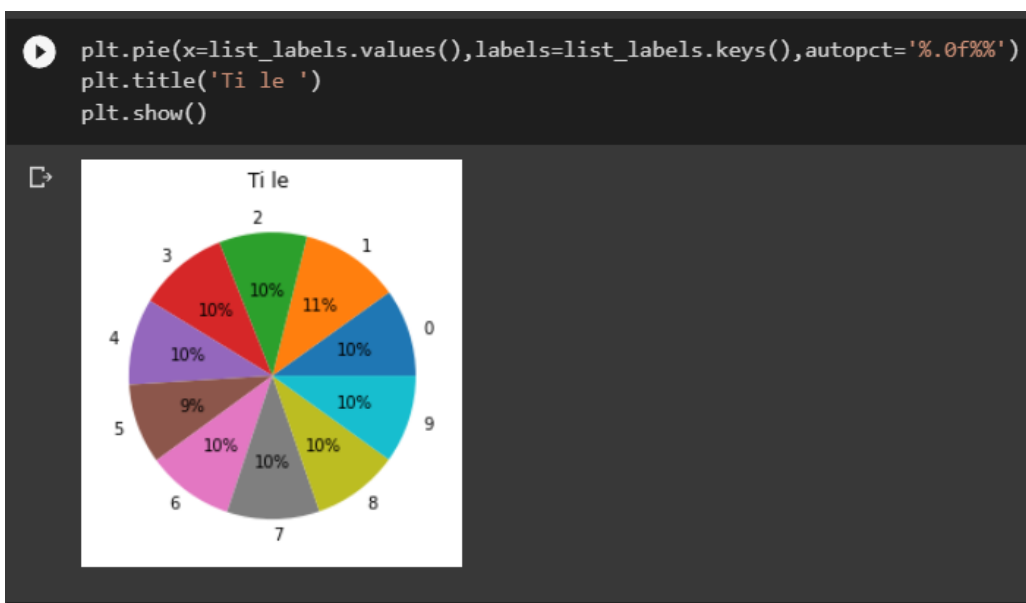
(?) Sử dụng thư viện Matplotlib để vẽ các biểu đồ sau:

- Biểu đồ cột thể hiện sự phân bố các nhãn có trong tập train ?
- Biểu đồ tròn thể hiện tỉ lệ các nhãn có trong tập train ?

Biểu đồ cột:



Biểu đồ tròn:



(?) Sau khi thực hiện thao tác reshape trên tập train:

- Mỗi điểm dữ liệu là một vector có số chiều là bao nhiêu ?
- Các số -1 và 784 có ý nghĩa gì ?
- Viết ra dạng của input $X = \dots$?

- Mỗi điểm dữ liệu là một vector có số chiều là: (784,)

```
[297] X_train_resaped[0].shape
```

```
(784,)
```

- Các số -1 và 784 có ý nghĩa là :
 - + -1 chỉ toàn bộ điểm dữ liệu có trong tập train.
 - + 784 là chiều thay thế cho 28*28 lúc trước.
- Viết ra dạng của input X= :

```
[440] X_train_resaped[0]
```

```
array([[ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        126, 136, 175,  26, 166, 255, 247, 127,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  30, 36, 94, 154, 170, 253,
        253, 253, 253, 253, 225, 172, 253, 242, 195, 64,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  49, 238, 253, 253, 253,
        253, 253, 253, 253, 253, 251, 93, 82, 82, 56, 39,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 18, 219, 253,
        253, 253, 253, 253, 198, 182, 247, 241,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        80, 156, 107, 253, 253, 205, 11,  0, 43, 154,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0, 14,  1, 154, 253, 90,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0, 139, 253, 190,  2,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 11, 190, 253, 70,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, 35,
```

```
[441] print(len(X_train_resaped[0]))
```

784

(?) Điền từ hoặc số thích hợp vào chỗ trống trong những câu dưới đây:

- Đầu ra của dữ liệu sẽ có giá trị trong khoảng $0 \rightarrow 9$ (tương ứng với **(1)** ... chữ số viết tay).
- Mỗi điểm dữ liệu sẽ được phân loại vào một trong **(2)** ... lớp tương ứng.
- Như vậy, đây là một bài toán phân lớp đa lớp (**(3)** ... classification).

Đầu ra của dữ liệu sẽ có giá trị trong khoảng $0 \rightarrow 9$ (tương ứng với 10 chữ số viết tay).

Mỗi điểm dữ liệu sẽ được phân loại vào một trong 10 lớp tương ứng.

Như vậy, đây là một bài toán phân lớp đa lớp (multi-class classification).

(?) Sau khi thực hiện thao tác trên:

- Xác định kích thước của `y_train` ? Các số liệu này tương ứng với những đại lượng nào ?
- Viết ra vector tương ứng `y_train[5]` ?
- Các giá trị "1" và "0" trong mỗi vector có ý nghĩa gì ?

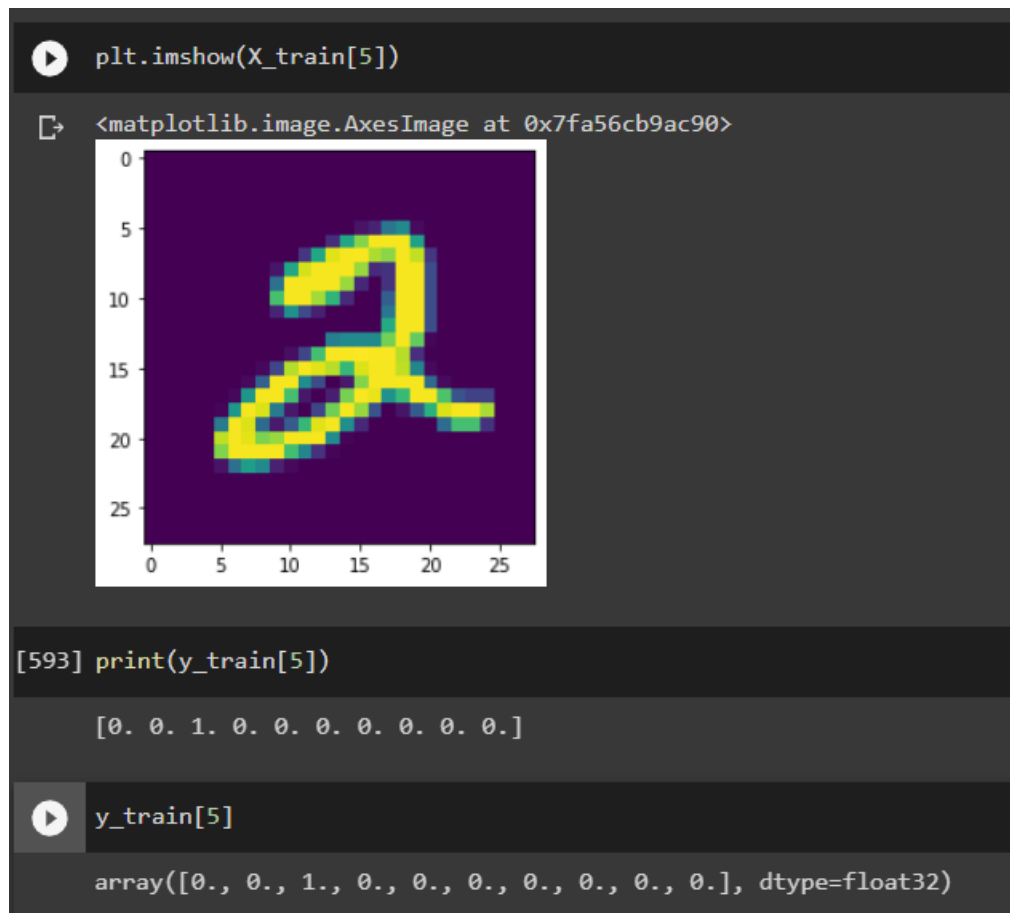
Kích thước của `y_train` : (60000,)

```
[590] y_train.shape
(60000, 10)
```

Viết ra vector tương ứng `y_train[5]` :

```
[591] y_train[5]
array([0., 0., 1., 0., 0., 0., 0., 0., 0., 0.], dtype=float32)
```


Các giá trị “1” và “0” có ý nghĩa: Ma trận như trên là đại diện cho lớp phân loại. Với vị trí của số 1 là đại diện cho chữ số trong ảnh.



III, XÂY DỰNG MẠNG NEURAL BẰNG KERAS

(?) Kể tên và ghi công thức của một số hàm kích hoạt mà bạn biết ?

Sigmoid: $f(x) = \frac{1}{1+\exp^{-x}}$

ReLU: $f(x) = \max(0, x) = \begin{cases} x_i, & \text{if } x_i \geq 0 \\ 0, & \text{if } x_i < 0 \end{cases}$

Tanh: $f(x) = \frac{\exp^x - \exp^{-x}}{\exp^x + \exp^{-x}}$

Softmax: $f(x) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$

(?) Xác định kích thước đầu ra (output_shape) của lớp được thêm vào mô hình ?

Lớp đầu ra của mô hình sẽ là xác suất ứng với mỗi nhãn. Do đó, lớp này sẽ có 10 units.

(?) Tại sao số units ở lớp này lại là 10 ?

Vì số nhãn có trong tập là 10, tức là sẽ có 10 kết quả được suy ra.

(?) Tổng xác suất của các lớp bằng bao nhiêu ?

Tổng xác suất của các lớp bằng 1.

(?) Hai layers trong mô hình trên tương ứng với những loại layer nào trong mô hình mạng neural tổng quát ?

Layer có 10 units là: Output layer.

Layer có size = 784 là: Hidden layer

(?) Quan sát kết quả thực thi câu lệnh, cho biết số lượng tham số của mỗi layer và tổng số lượng tham số (total params) ?

```
Model: "sequential_24"
Layer (type)                Output Shape                Param #
=====
dense_61 (Dense)            (None, 784)                 615440
dense_62 (Dense)            (None, 10)                  7850
=====
Total params: 623,290
Trainable params: 623,290
Non-trainable params: 0
```

Số lượng tham số của hidden layer là 615440

Số lượng tham số của output layer là 7850

Tổng số lượng tham số của hidden layer là 623290

IV, HUẤN LUYỆN MÔ HÌNH

(?) Vì sao cần set giá trị `learning_rate` nhỏ ?

Bởi vì chúng ta đã áp dụng thuật toán tối ưu là Adam, nó sẽ giúp chúng ta thực hiện thuật toán Gradient Descent nhanh hơn và ít trắc trở hơn. Vì vậy ta thiết lập giá trị của `learning_rate` quá lớn dễ bị optimal minimum quá đà.

(?) Viết công thức tính loss Cross Entropy cho bài toán phân lớp nhị phân "cat vs. non-cat" với hai nhãn là $y = 1$ (cat) và $y = 0$ (non-cat) (Chú thích các đại lượng trong công thức) ?

Công thức tính Loss Cross Entropy:

$$H(p) = - \sum_x p(x) \log_2 p(x)$$

- $H(p)$: entropy của xác suất p .
- p : xác suất phân loại mèo.

(?) Tính thời gian huấn luyện mô hình ?

Thời gian huấn luyện mô hình: 38.932519s

```
import time
t1=time.perf_counter()
model.fit(X_train_reshaped, y_train, batch_size=128, epochs=10)
t2=time.perf_counter()
print({t2-t1})
```

```
Epoch 1/10
469/469 [=====] - 4s 7ms/step - loss: 3.3238 - accuracy: 0.0901
Epoch 2/10
469/469 [=====] - 4s 8ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 3/10
469/469 [=====] - 6s 12ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 4/10
469/469 [=====] - 6s 12ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 5/10
469/469 [=====] - 4s 9ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 6/10
469/469 [=====] - 3s 7ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 7/10
469/469 [=====] - 3s 7ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 8/10
469/469 [=====] - 3s 7ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 9/10
469/469 [=====] - 3s 7ms/step - loss: 2.7903 - accuracy: 0.0904
Epoch 10/10
469/469 [=====] - 3s 7ms/step - loss: 2.7903 - accuracy: 0.0904
{38.93251900400003}
```

V, ĐÁNH GIÁ MÔ HÌNH

(?) Kết quả dự đoán của mô hình sẽ là một ma trận có kích thước (a, b).

- Hãy xác định các giá trị a, b ?
- Nêu ý nghĩa của các giá trị a, b ?

Các giá trị: a=10000, b=10.

```
[762] y_pred = model.predict(X_test_reshaped)
```

```
[763] y_pred.shape
```

```
(10000, 10)
```

a=10000 là số lượng dữ liệu được mô hình dự đoán từ tập test.

b=10 là số điểm dữ liệu trong ma trận

(?) Lớp được chọn sẽ có xác suất cao nhất hay thấp nhất ?

Lớp được chọn sẽ có xác suất cao nhất.

(?) Dựa vào kết quả thực thi:

- Viết ra “vector xác suất dự đoán” của ảnh có chỉ số [50] và cho biết kết quả dự đoán ảnh này thuộc lớp nào ? So sánh nhãn dự đoán (`y_pred_label`) và nhãn đúng (`y_test`) của ảnh này ?
- Xác định nhãn phân loại của các ảnh có chỉ số [100], [1001], [4123] trong tập test ? Trực quan hóa các ảnh này để kiểm nghiệm lại kết quả dự đoán ?

Vector ảnh[50]: [-36641.766 , -33266.367 , -36892.457 , -35514.598 ,
-52196.15 , 1860.4623, -65111.723 , -82784.81 , -49448.234 , -10444.562]

```
[104] y_pred[50]
```

```
array([-36641.766 , -33266.367 , -36892.457 , -35514.598 , -52196.15 ,  
       1860.4623, -65111.723 , -82784.81 , -49448.234 , -10444.562 ],  
      dtype=float32)
```

```
y_pred_label[50]
```

5

So sánh nhãn dự đoán và nhãn đúng của ảnh:

```
y_pred_label[50]
```

5

```
[106] y_test[50]
```

6

Xác định nhãn phân loại của các ảnh có chỉ số [100], [1001], [4123] trong tập test ? Trực quan hóa các ảnh này để kiểm nghiệm lại kết quả dự đoán ?

```
y_test[100]
```

6

```
[109] y_test[1001]
```

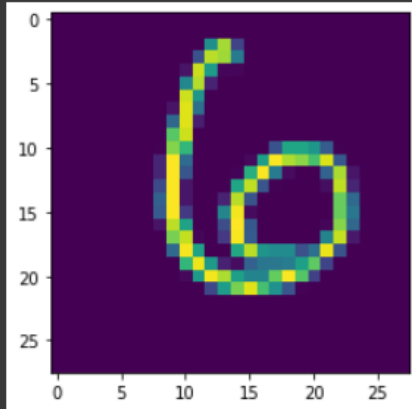
0

```
y_test[4123]
```

8

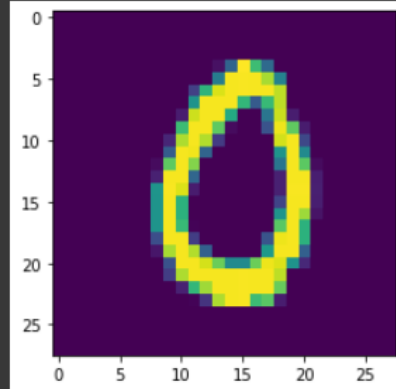
```
plt.imshow(X_test[100])
```

```
<matplotlib.image.AxesImage at 0x7fc328191350>
```



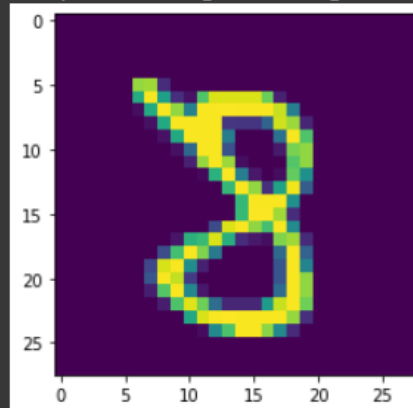
```
plt.imshow(X_test[1001])
```

```
<matplotlib.image.AxesImage at 0x7fc328165ed0>
```



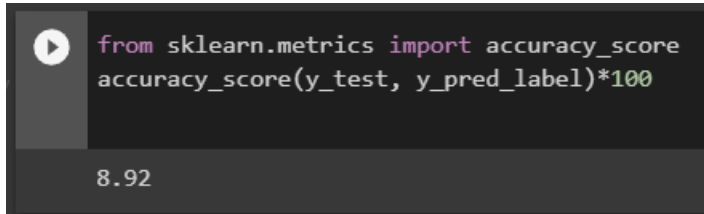
```
plt.imshow(X_test[4123])
```

```
<matplotlib.image.AxesImage at 0x7fc3280cda10>
```



(?) Độ chính xác của mô hình là bao nhiêu ?

Độ chính xác của mô hình là: 8.92



```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred_label)*100
```

8.92

VI, LƯU MÔ HÌNH

(?) Việc lưu lại mô hình có tác dụng gì ?

Việc lưu lại mô hình có tác dụng tiết kiệm rất nhiều thời gian huấn luyện lại mô hình cũ khi ta cần sử dụng lại nó trong tương lai. Thay vì tốn thời gian huấn luyện lại, ta chỉ cần lấy ra mô hình trước đó đã lưu lại rồi chạy là xong. Sau đó, chúng ta có thể load lại mô hình và sử dụng nó để đưa ra dự đoán, hoặc tiếp tục đào tạo nó, hoặc làm bất cứ điều gì chúng ta muốn với nó.