**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO MÔN: TRÍ TUỆ NHÂN TẠO**

**Đề tài: Nhận dạng chữ số in trên bằng tốt nghiệp**

**Giảng viên : Nguyễn Lan Anh**

**Sinh viên thực hiện : Dương Ngọc Anh**

**Nguyễn Thành Lâm**

**Lớp : 20221IT6043004**

Hà Nội, 2022

# **Lời Nói Đầu**

AI - Artificial Intelligence (Trí Tuệ Nhân Tạo), và cụ thể hơn là Machine Learning (Máy Học) nổi lên như một minh chứng của cuộc cách mạng công nghiệp lần thứ tư. AI hiện diện trong mọi lĩnh vực của đời sống con người, từ kinh tế, giáo dục, y khoa cho đến những công việc nhà, giải trí hay thậm chí là trong quân sự. Những ứng dụng nổi bật trong việc phát triển AI đến từ nhiều lĩnh vực để giải quyết nhiều vấn đề khác nhau. Nhưng những đột phá phần nhiều đến từ Deep Learning (học sâu) - một mảng nhỏ đang mở rộng dần đến từng loại công việc, từ đơn giản đến phức tạp. Deep Learning đã giúp máy tính thực thi những việc tưởng chừng như không thể vào 15 năm trước: phân loại cả ngàn vật thể khác nhau trong các bức ảnh, tự tạo chú thích cho ảnh, bắt chước giọng nói và chữ viết của con người, giao tiếp với con người, hay thậm chí cả sáng tác văn, phim, ảnh, âm nhạc.

Trong quá trình thực hiện đề tài nhóm chúng em xin gửi lời cám ơn chân thành tới cô Nguyễn Lan Anh giảng viên hướng dẫn nhóm chúng em thực hiện đề tài này. Trong quá trình nghiên cứu và thực hiện đề tài được sự chỉ bảo tận tình của các thầy cô, nhóm chúng em đã cố gắn hết sức để hoàn thiện đề tài. Tuy nhiên chúng em rất mong nhận được sự góp ý của thầy và các bạn.

Nhóm em xin chân trọng cảm ơn!

**Mục Lục**

[**Lời Nói Đầu 2**](#_Toc119973560)

[**Chương 1: Tổng quan về mạng Nơ-ron 4**](#_Toc119973561)

[1. Giới thiệu về Neural network 4](#_Toc119973562)

[2. Ảnh trong máy tính 6](#_Toc119973563)

[3. Convolutional neural network (CNN) 13](#_Toc119973564)

[**Chương 2: Xây dựng ứng dụng 16**](#_Toc119973565)

**1. Chuẩn bị dữ liệu**

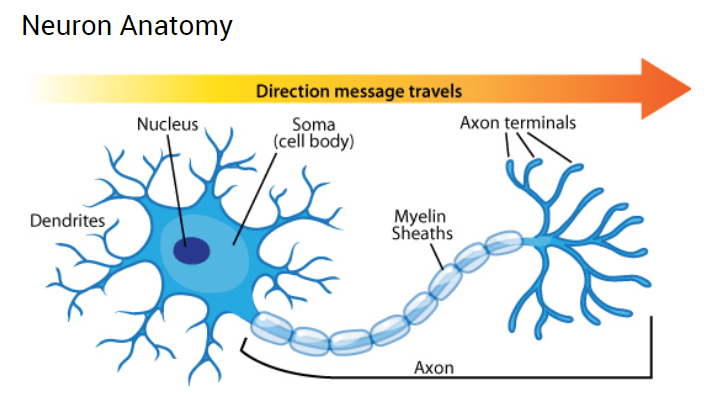
**2. Các bước thực hiện chương trình cụ thể**

**3. Một số lỗi hạn chế**

# **Chương 1: Tổng quan về mạng Nơ-ron**

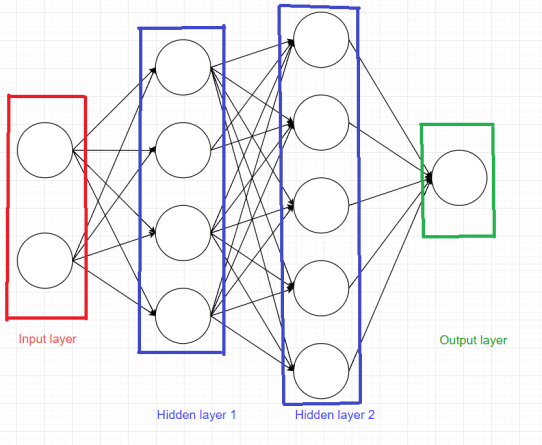
1. **Giới thiệu về Neural network**

* Neural là tính từ của neuron (nơ-ron), network chỉ cấu trúc đồ thị nên neural network (NN) là một hệ thống tính toán lấy cảm hứng từ sự hoạt động của các nơ-ron trong hệ thần kinh.
* Hoạt động của các nơ-ron:



Hình 1.1: Tế bào nơ-ron

* + Nơ-ron là đơn vị cơ bản cấu tạo hệ thống thần kinh và là một phần quan trọng nhất của não. Não chúng ta gồm khoảng 10 triệu nơ-ron và mỗi nơ-ron liên kết với 10.000 nơ-ron khác.
  + Ở mỗi nơ-ron có phần thân (soma) chứa nhân, các tín hiệu đầu vào qua sợi nhánh (dendrites) và các tín hiệu đầu ra qua sợi trục (axon) kết nối với các nơ-ron khác. Hiểu đơn giản mỗi nơ-ron nhận dữ liệu đầu vào qua sợi nhánh và truyền dữ liệu đầu ra qua sợi trục, đến các sợi nhánh của các nơ-ron khác.
  + Mỗi nơ-ron nhận xung điện từ các nơ-ron khác qua sợi nhánh. Nếu các xung điện này đủ lớn để kích hoạt nơ-ron, thì tín hiệu này đi qua sợi trục đến các sợi nhánh của các nơ-ron khác.
* Mô hình neural network lấy cảm hứng từ não bộ và cách nó hoạt động, chứ không phải bắt chước toàn bộ các chức năng của nó

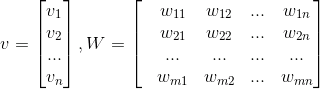


Hình 1.2: Mô hình neural network

* + Layer đầu tiên là input layer, các layer ở giữa được gọi là hidden layer, layer cuối cùng được gọi là output layer. Các hình tròn được gọi là node.
  + Mỗi mô hình luôn có 1 input layer, 1 output layer, có thể có hoặc không các hidden layer. Tổng số layer trong mô hình được quy ước là số layer – 1 (Không tính input layer).
  + Mỗi node trong hidden layer và output layer:
    - Liên kết với tất cả các node ở layer trước đó với các hệ số w riêng.
    - Mỗi node có 1 hệ số bias b riêng.
    - Diễn ra 2 bước: tính tổng linear và áp dụng activation function. Trong đó tổng linear là tổng các giá trị đầu vào, activation function là hàm kích hoạt của mạng.

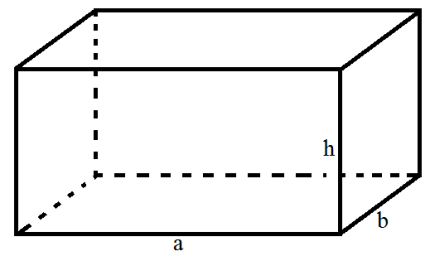
1. **Ảnh trong máy tính**

* Hệ màu RGB:
  + RGB viết tắt của red (đỏ), green (xanh lục), blue (xanh lam), là ba màu chính của ánh sáng khi tách ra từ lăng kính. Khi trộn ba màu trên theo tỉ lệ nhất định có thể tạo thành các màu khác nhau.
  + Với mỗi bộ 3 số r, g, b nguyên trong khoảng [0, 255] sẽ cho ra một màu khác nhau. Do có 256 cách chọn r, 256 cách chọn màu g, 256 cách chọn b => tổng số màu có thể tạo ra bằng hệ màu RGB là: 256 \* 256 \* 256 = 16777216 màu.
* Ảnh màu là một ma trận các pixel mà mỗi pixel biểu diễn một điểm màu. Mỗi điểm màu được biểu diễn bằng bộ 3 số (r,g,b). Để tiện cho việc xử lý ảnh thì sẽ tách ma trận pixel ra 3 channel red, green, blue.
* Tensor:
  + Khi dữ liệu biểu diễn dạng 1 chiều, người ta gọi là vector, mặc định khi viết vector sẽ viết dưới dạng cột.
  + Khi dữ liệu dạng 2 chiều, người ta gọi là ma trận, kích thước là số hàng \* số cột.



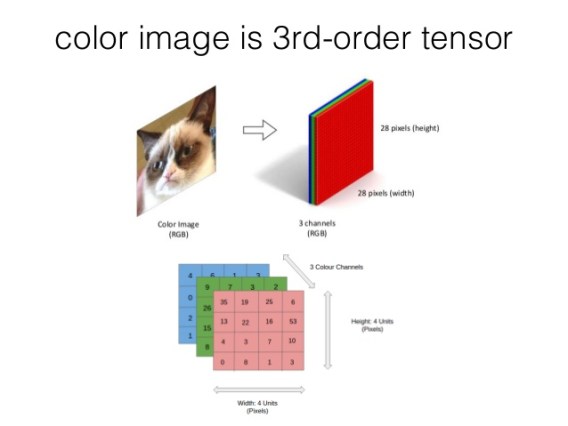
Hình 2.1: Vector v kích thước n, ma trận W kích thước m\*n

* + Khi dữ liệu nhiều hơn 2 chiều thì sẽ được gọi là tensor



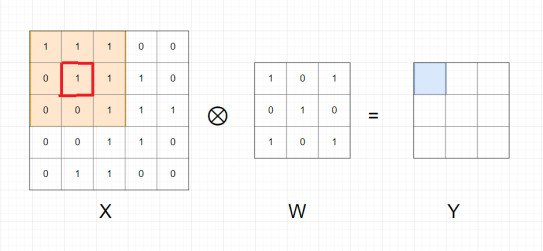
Hình 2.2: Hình hộp chữ nhật kích thước a\*b\*h

* + Tưởng tượng mặt đáy là một ma trận kích thước a \* b, được tạo bởi b vector kích thước a. Cả hình hộp là tensor 3 chiều kích thước a\*b\*h, được tạo bởi xếp h ma trận kích thước a\*b lên nhau.



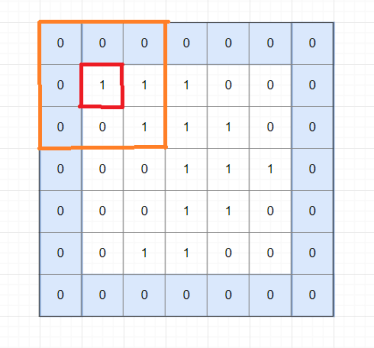
Hình 2.3: ảnh màu kích thước 28\*28, biểu diễn dưới dạng tensor 28\*28\*3

* Phép tính tích chập (convolution)



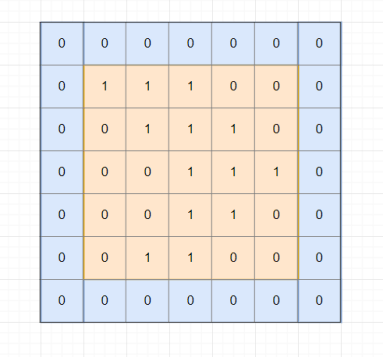
Hình 2.4: Phép tính tích chập

* Padding
  + Mỗi lần thực hiện phép tính convolution xong thì kích thước ma trận Y đều nhỏ hơn X. Nếu muốn ma trận Y thu được có kích thước bằng ma trận X chỉ cần thêm giá trị 0 ở viền ngoài ma trận X.



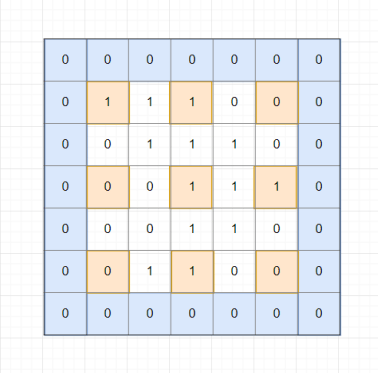
Hình 2.5: Ma trận X khi thêm viền 0 bên ngoài

* + Phép tính này gọi là convolution với padding=1. Padding=k nghĩa là thêm k vector 0 vào mỗi phía của ma trận.
* Stride



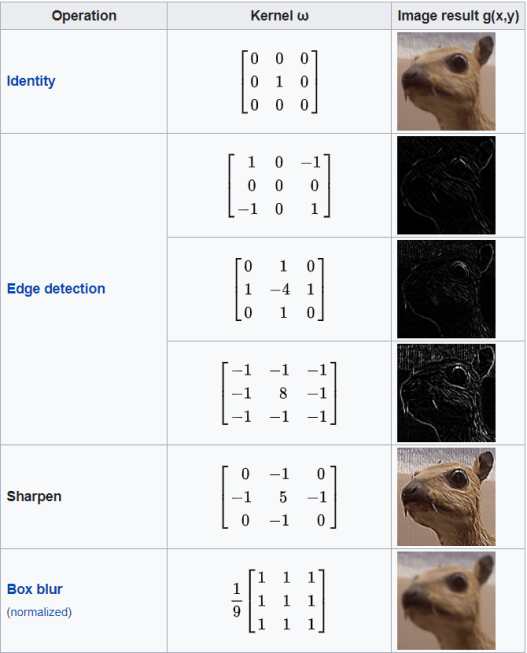
Hình 2.6: stride=1, padding=1

* + Ta thực hiện tuần tự các phần tử trong ma trận X, thu được ma trận Y cùng kích thước ma trận X, ta gọi là stride=1.
  + Nếu stride=k (k > 1) thì ta chỉ thực hiện phép tính convolution trên các phần tử từ vị trí đầu tiên sau đó nhảy k bước theo chiều dọc và ngang cho đến hết ma trận X.



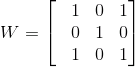
Hình 2.7: padding=1, stride=2

* Ý nghĩa của phép tính convolution
  + Mục đích của phép tính convolution trên ảnh là làm mở, làm nét ảnh; xác định các đường;… Mỗi kernel khác nhau thì sẽ phép tính convolution sẽ có ý nghĩa khác nhau.



Hình 2.8: Ý nghĩa của phép tính tích chập

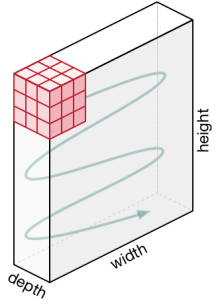
* + Kernel là một ma trận vuông kích thước k\*k sử dụng trong phép tính tích chập



Hình 2.9: kernel kích thước 3\*3

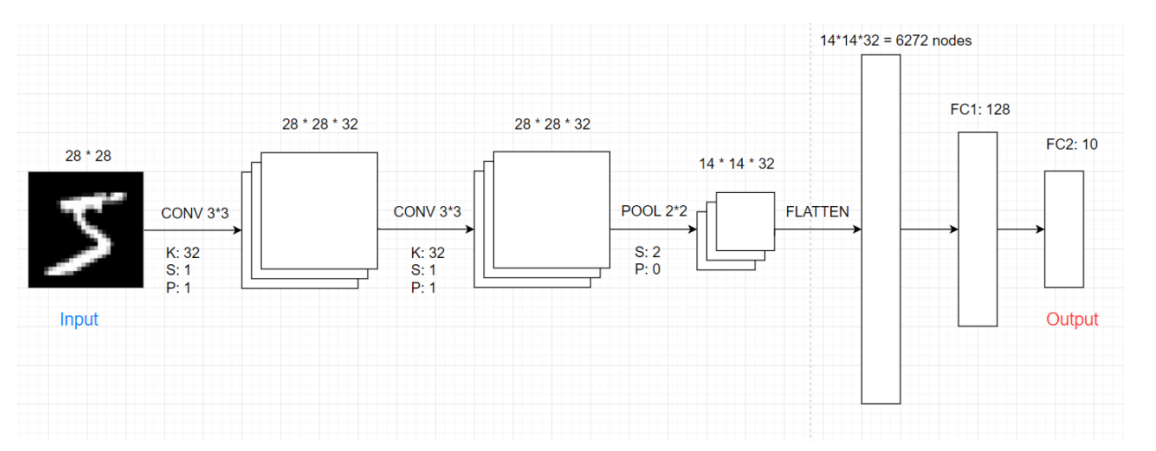
1. **Convolutional neural network (CNN)**

* Mạng CNN còn gọi là mạng nơ-ron tích chập, là một lớp của mạng thần kinh sâu (deep neural network), áp dụng phổ biến nhất để phân tích hình ảnh trực quan.
* CNN có nhiều ứng dụng trong thị giác máy tính, hệ thống gợi ý, phân loại hình ảnh, tính toán hình ảnh y tế (điện toán hình ảnh y tế), xử lý ngôn ngữ tự nhiên, và chuỗi thời gian tài chính.
* Trong bài toán xử lý ảnh, CNN áp dụng các phép tính tích chập để lấy ra các đặc trưng của ảnh.
* Xử lý ảnh với 3 kênh màu (RGB), mạng sẽ sử dụng kernal là 1 tensor 3 chiều kích thước k\*k\*3



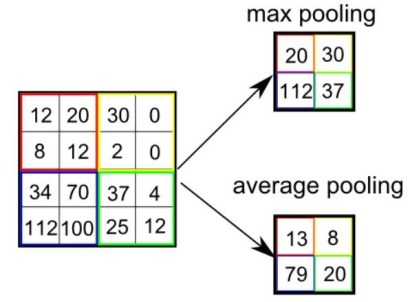
Hình 3.1: Phép tính convolution trên ảnh màu với k=3

* CNN có quy tắc với padding và stride tương tự như xử lý ảnh ở máy tính.



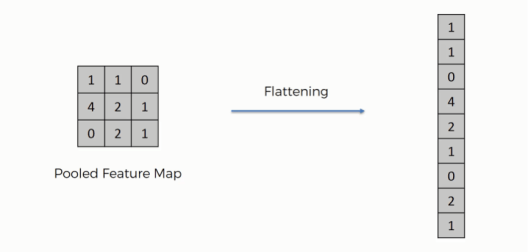
Hình 3.2: Mô hình CNN

* Tổng quát:
  + Input image -> Convolutional layer (Conv) + Pooling layer (Pool) -> Fully connected layer (FC) -> Output.
  + Conv là các phép tính tích chập
  + Pooling layer thường được dùng giữa các convolutional layer, để giảm kích thước dữ liệu nhưng vẫn giữ được các thuộc tính quan trọng. Kích thước dữ liệu giảm giúp giảm việc tính toán trong model.
  + Có 2 loại pooling layer phổ biến là: max pooling và average pooling.



Hình 3.3: Ví dụ về pooling layer

* + Fully connected layer để kết hợp các đặc điểm của ảnh để ra được output của model.
  + Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương đối các đặc điểm của ảnh (ví dụ mắt, mũi, khung mặt, …) thì tensor của output của layer cuối cùng, kích thước H\*W\*D, sẽ được chuyển về 1 vector kích thước (H\*W\*D)



Hình 3.4: Fully connected layer

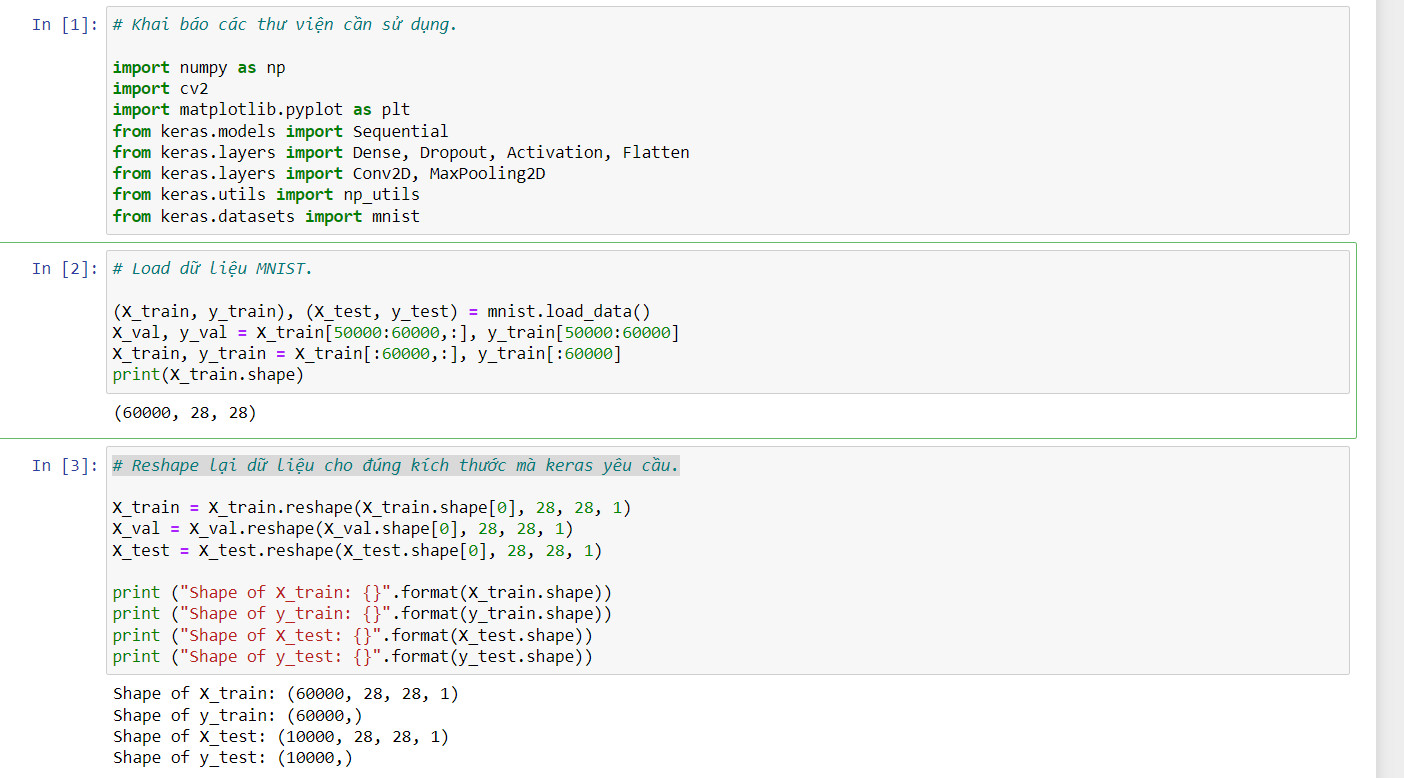
# **Chương 2: Xây dựng ứng dụng**

1. **Chuẩn bị dữ liệu**

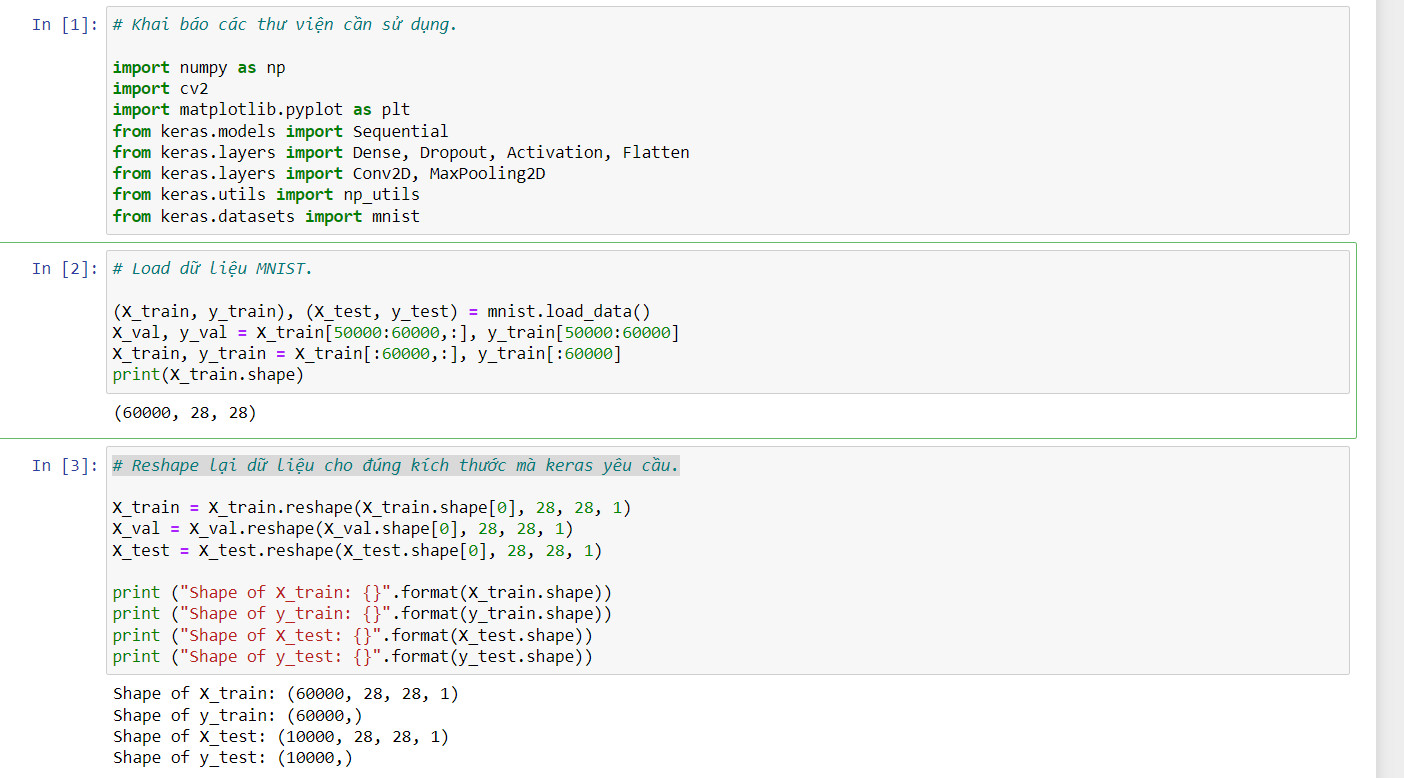
MNIST là bộ cơ sở dữ liệu về chữ số viết tay, bao gồm 2 tập con: training set gồm 60.000 ảnh các chữ số viết tay và test set gồm 10.000 ảnh các chữ số.

1. **Các bước thực hiện chương trình cụ thể**

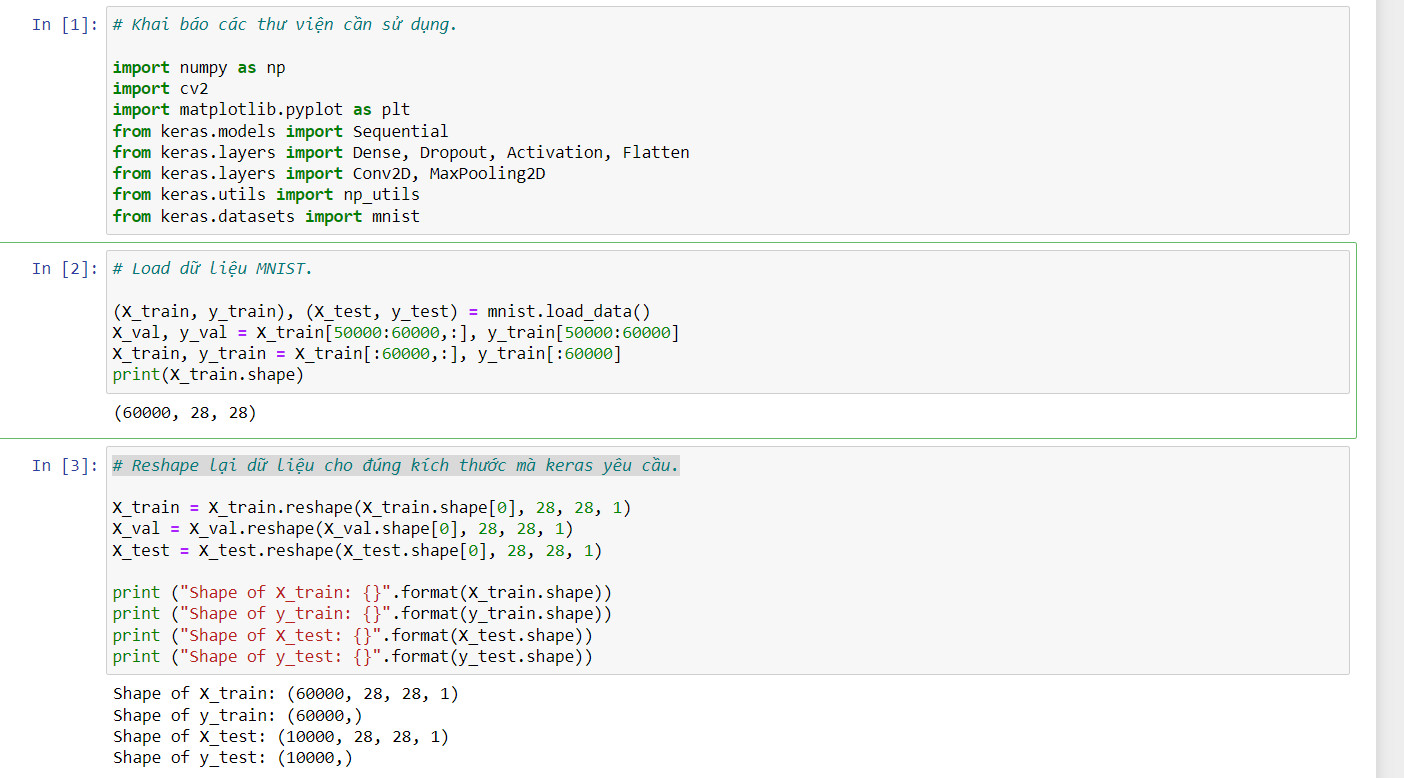
#Bước 1: Khai báo các thư viện cần sử dụng

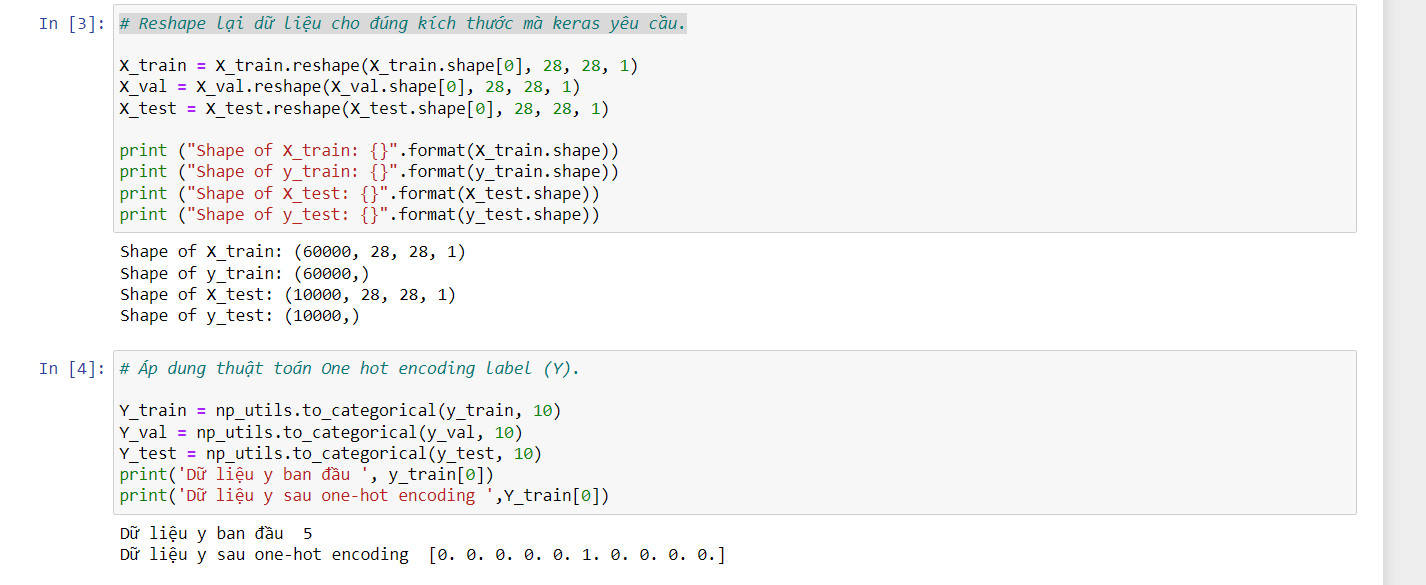


#Bước 2: Load dữ liệu MNIST

****

#Bước 3: Reshape lại dữ liệu cho đúng kích thước mà keras yêu cầu

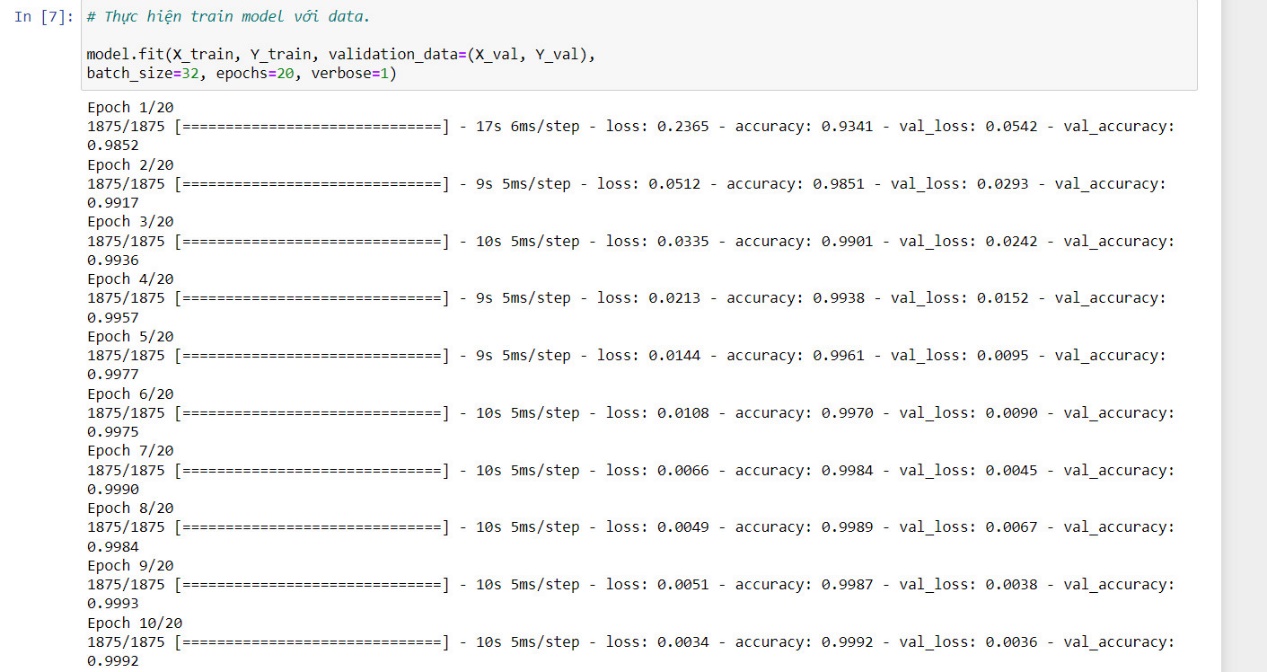


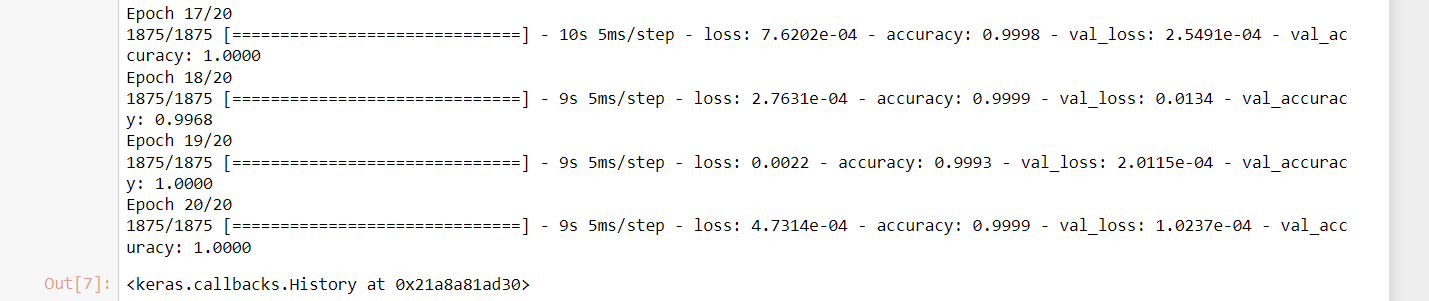
#Bước 4: Áp dụng thuật toán One hot encoding label (Y)

#Bước 5: Định nghĩa model; Compile modelm chỉ ra hàm ***loss function*** nào được sử dụng, phương thức dùng để tối ưu hàm ***loss function***

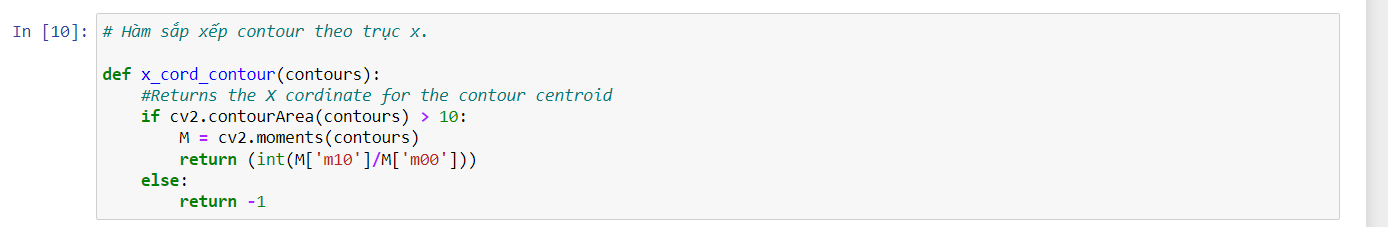


#Bước 6: Thực hiện training model với data

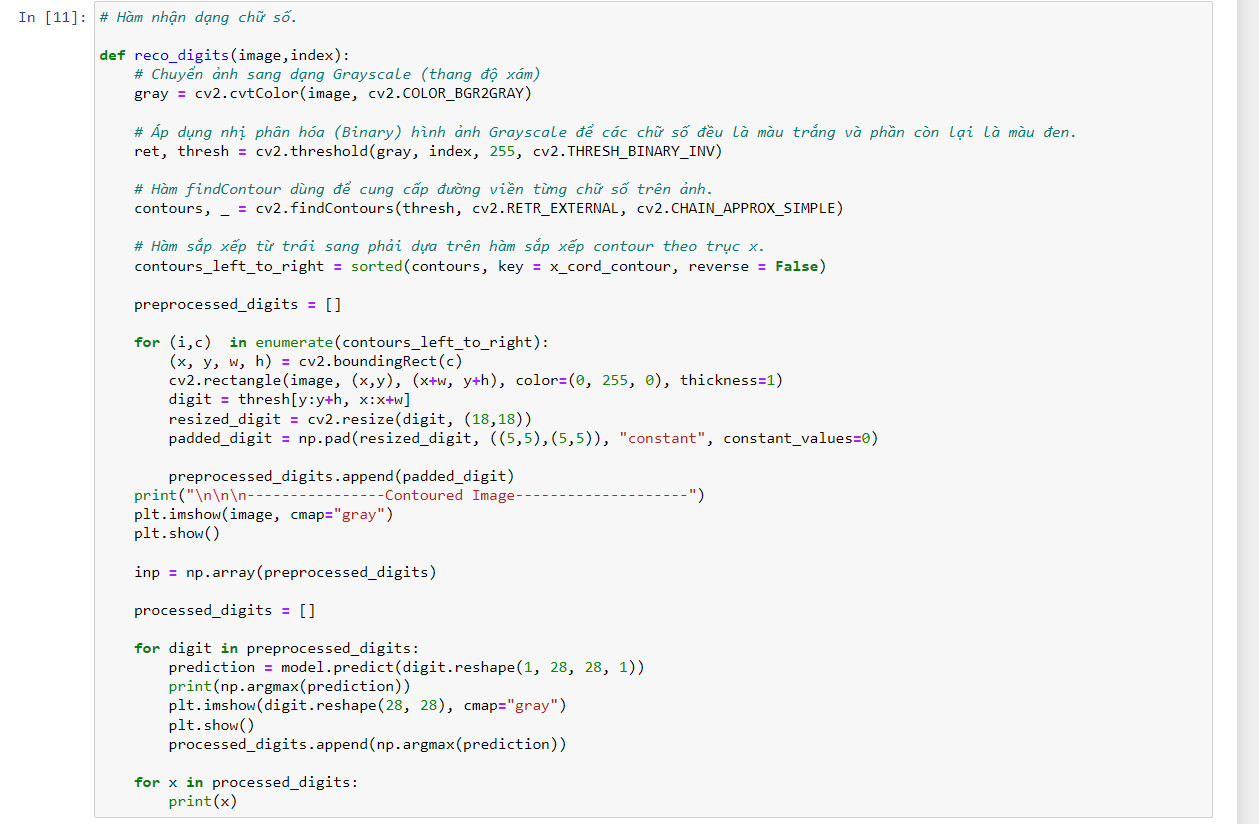




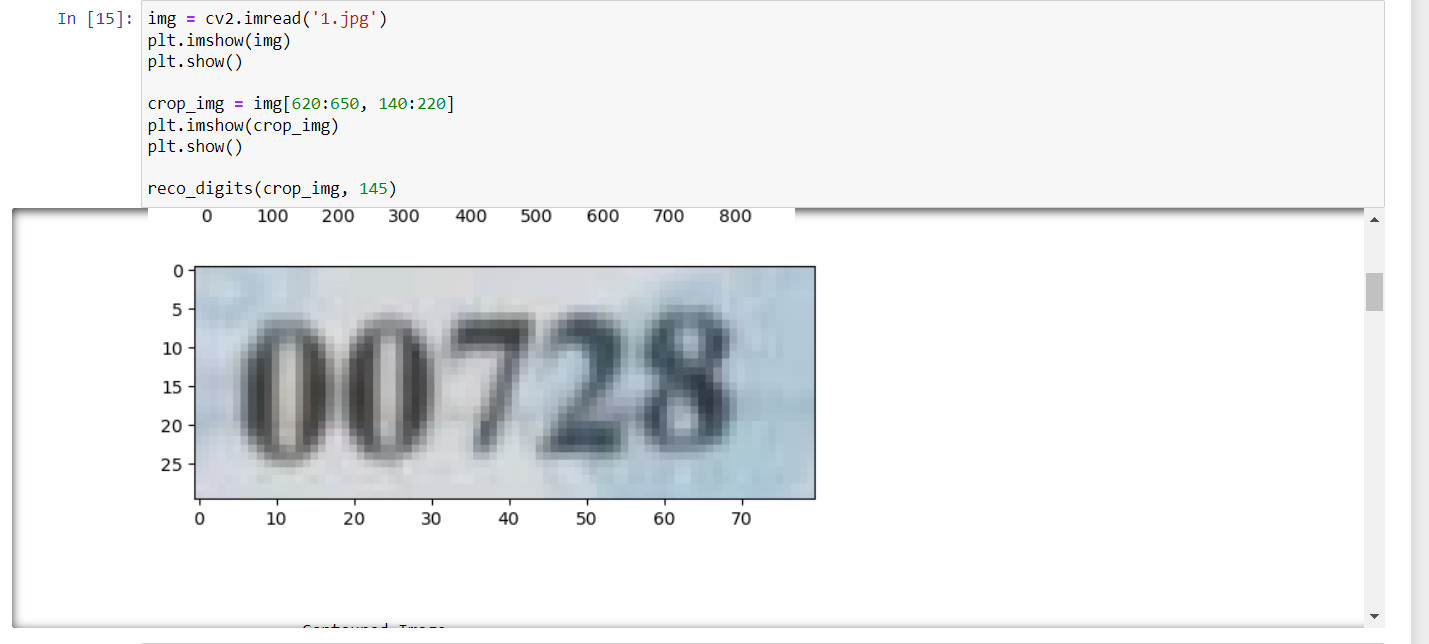
#Bước 7: Hàm sắp xếp contour theo trục x



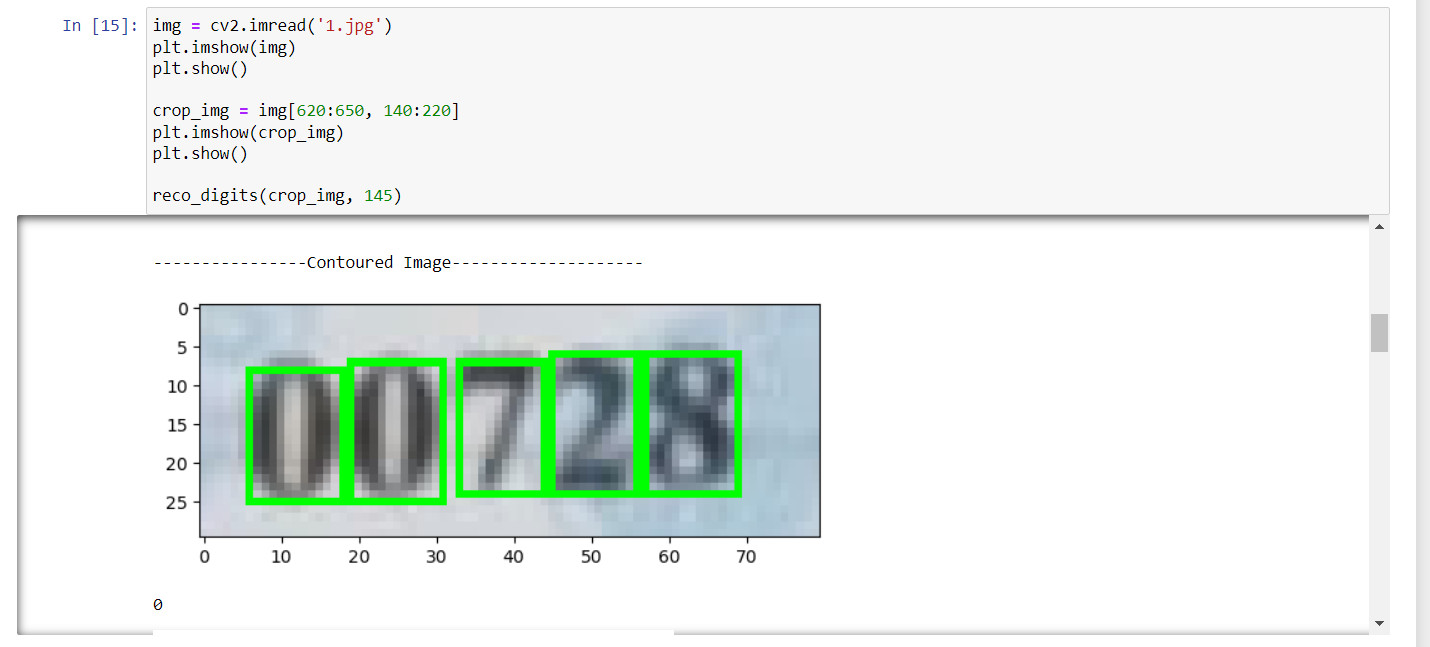
#Bước 8: Hàm nhận dạng chữ số



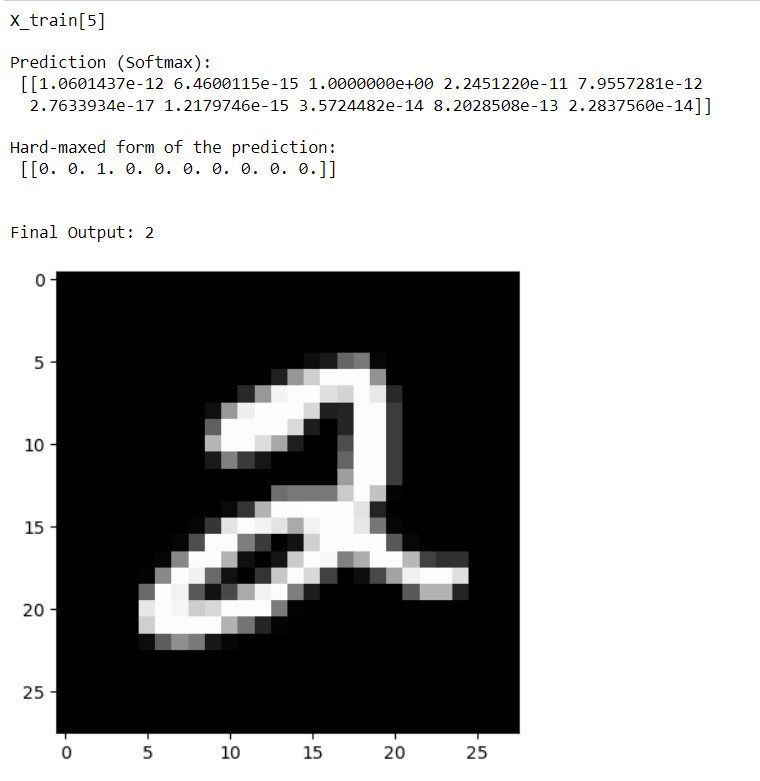
#Bước 9: Crop hình ảnh đưa vào để lấy data phân tích



#Bước 10: Phân tích hình ảnh để đưa ra số liệu



#Bước 11: Nhận dạng chữ số trong data mà máy đã được train



#Bước 12: Đưa ra kết quả phân tích từ dữ liệu đã được training



* Kết quả phân tích: **0 0 7 2 8**

1. **Một số lỗi bị hạn chế**

* Nếu ảnh đưa vào quá mờ, chương trình sẽ biên dịch sai số với những gì chương trình đã được training, dẫn đến kết quả đạt được thấp và ngược lại.
* Bằng tốt nghiệp không sử dụng font chữ số đồng nhất, cho nên 1 số font lạ khiến chương trình nhận diện sai.