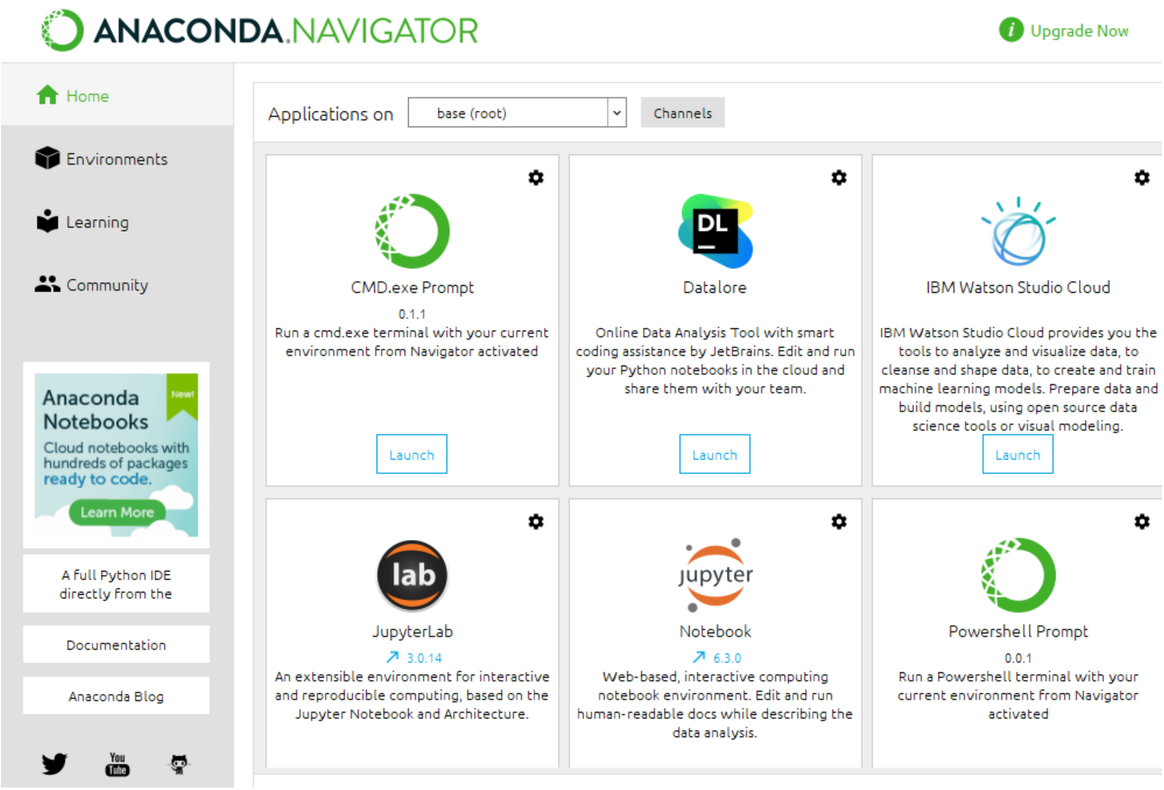
Pre-Requisites and Model Building Screenshots

Date	10 October 2022
Team ID	PNT2022TMID30139
Project Name	A Novel Method for Handwritten Digit Recognition System

Pre-Requisites : Installing Anaconda



CMD.exe Prompt: Run a cmd.exe terminal with your current environment from Navigator activated

Datalore: Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

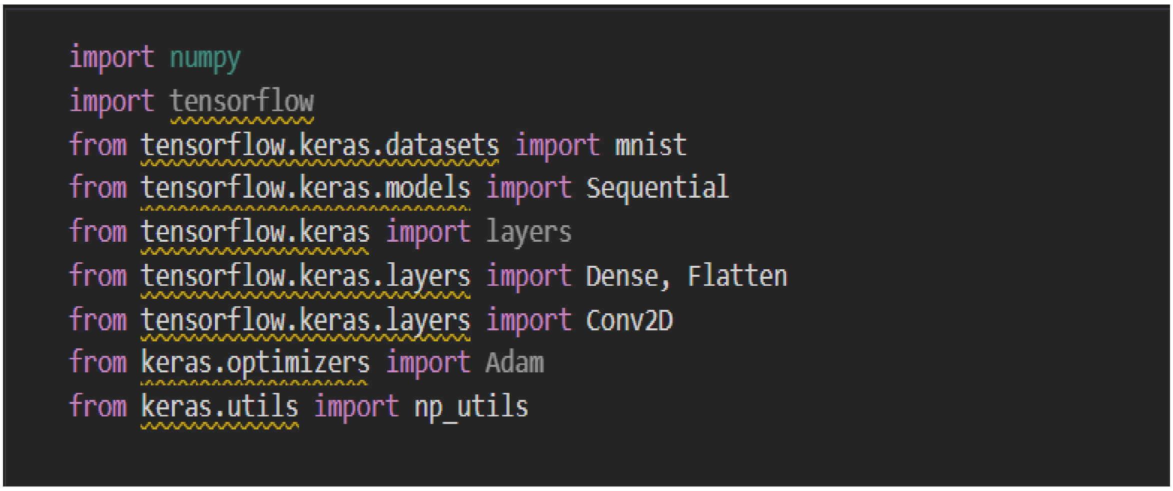
IBM Watson Studio Cloud: IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

JupyterLab: An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook: Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

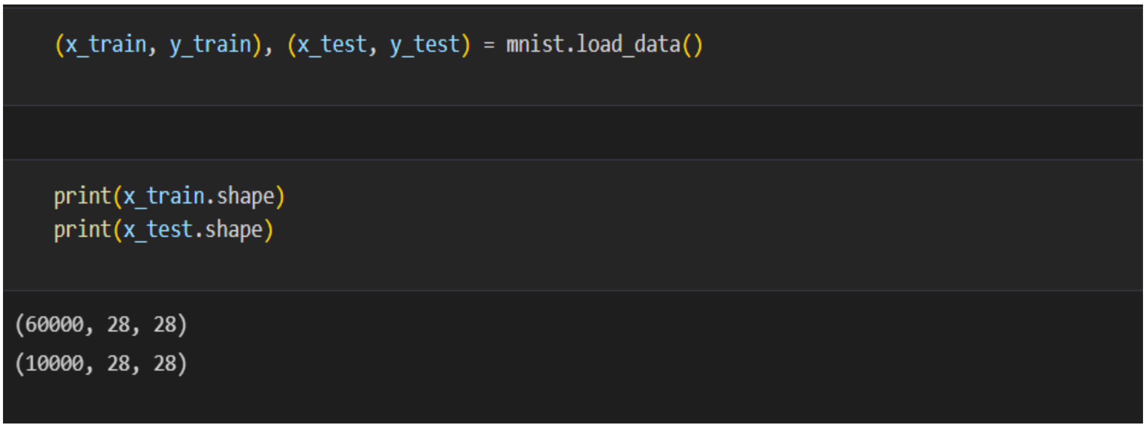
Powershell Prompt: Run a Powershell terminal with your current environment from Navigator activated

IMPORTING THE REQUIRED LIBRARIES:



```
import numpy
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D
from keras.optimizers import Adam
from keras.utils import np_utils
```

LOADING THE DATA:

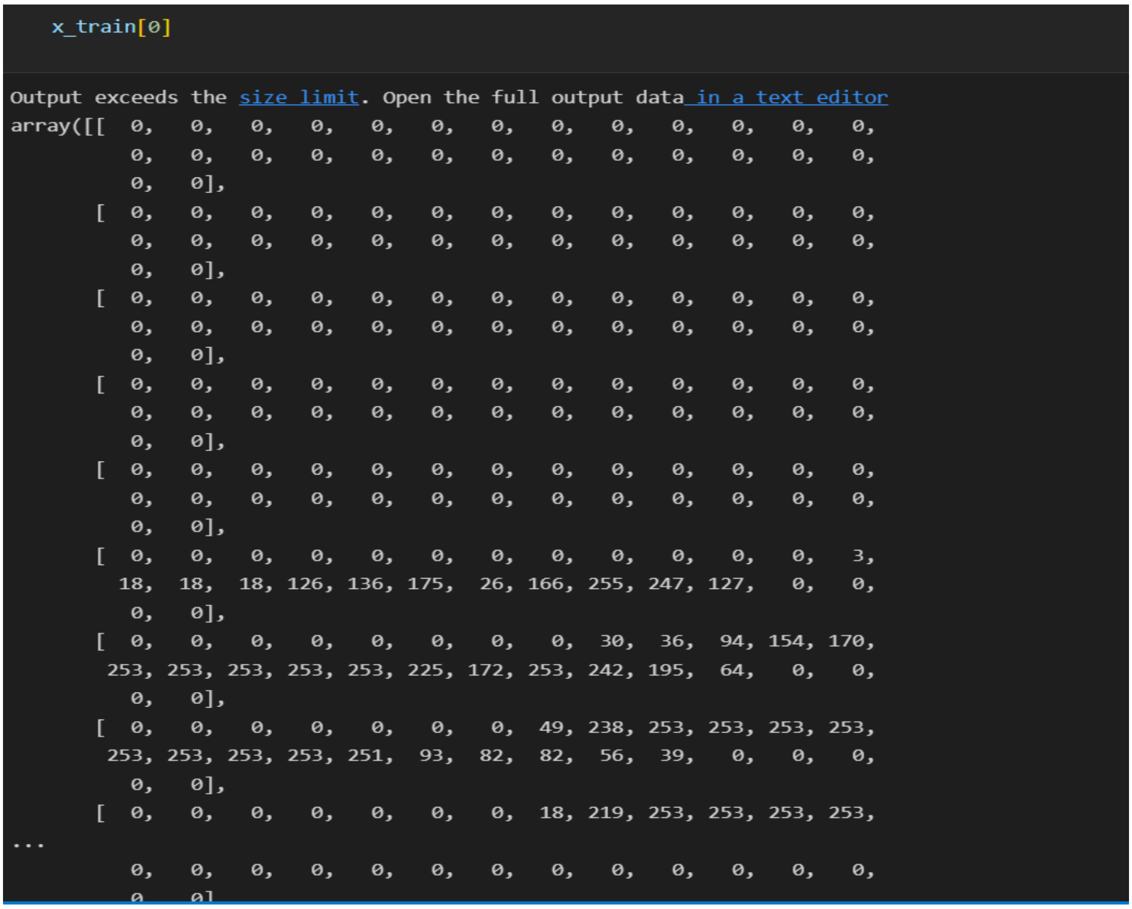


```
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
print(x_train.shape)
print(x_test.shape)
```

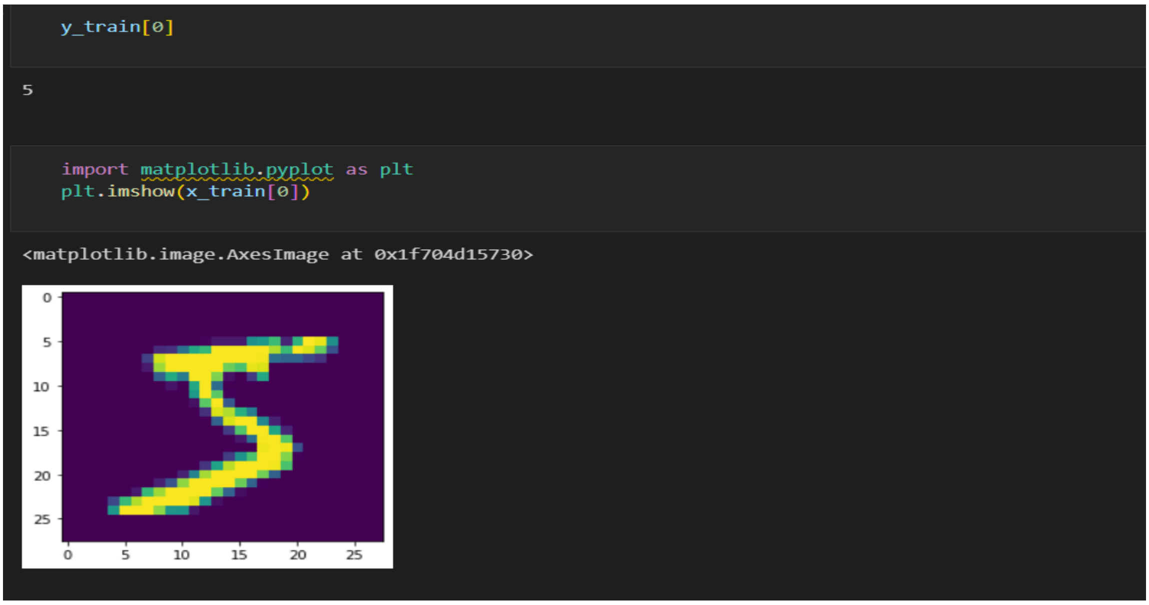
```
(60000, 28, 28)
(10000, 28, 28)
```

ANALYSING THE DATA:



```
x_train[0]
```

```
Output exceeds the size limit. Open the full output data in a text editor
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,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
         18,  18,  18, 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,  18, 219, 253, 253, 253, 253,
...
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0]
```



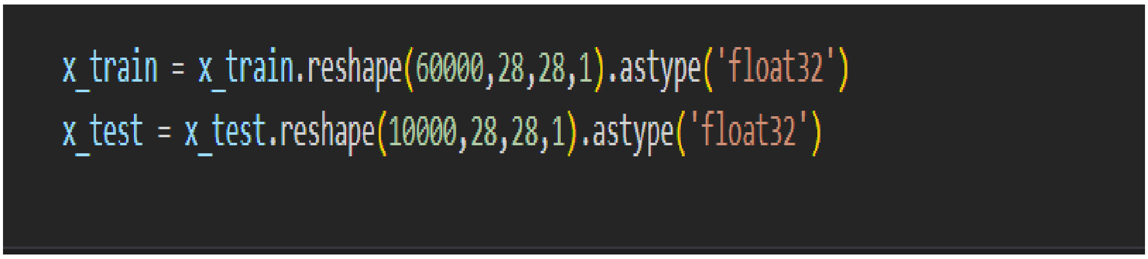
```
y_train[0]
```

```
5
```

```
import matplotlib.pyplot as plt
plt.imshow(x_train[0])
```

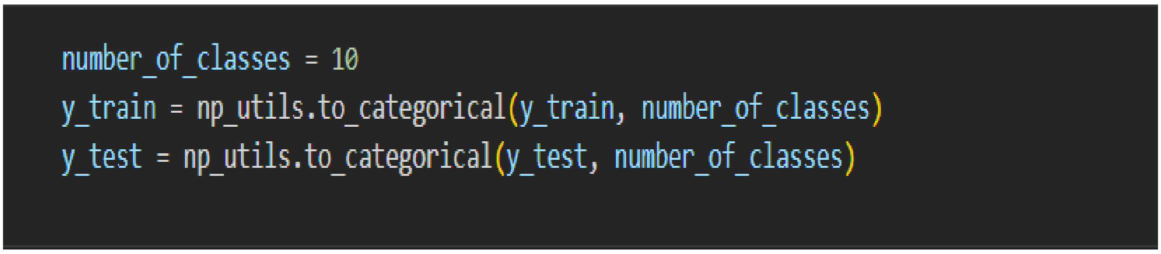
```
<matplotlib.image.AxesImage at 0x1f704d15730>
```

RESHAPING THE DATA:



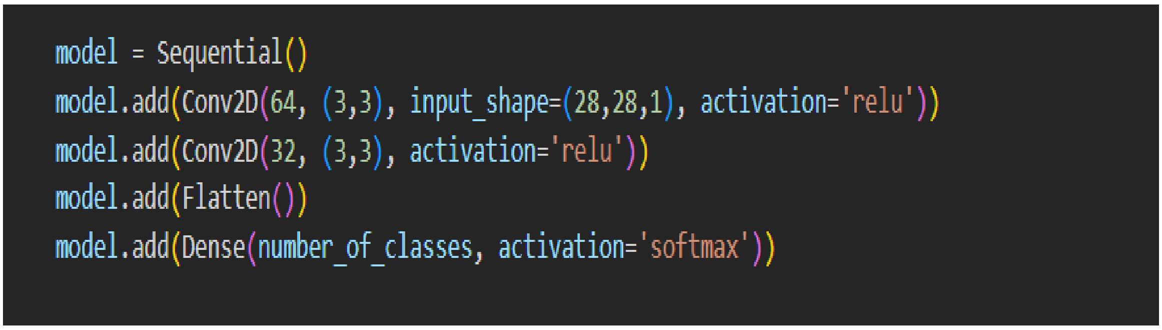
```
x_train = x_train.reshape(60000,28,28,1).astype('float32')
x_test = x_test.reshape(10000,28,28,1).astype('float32')
```

APPLYING ONE HOT ENCODING:



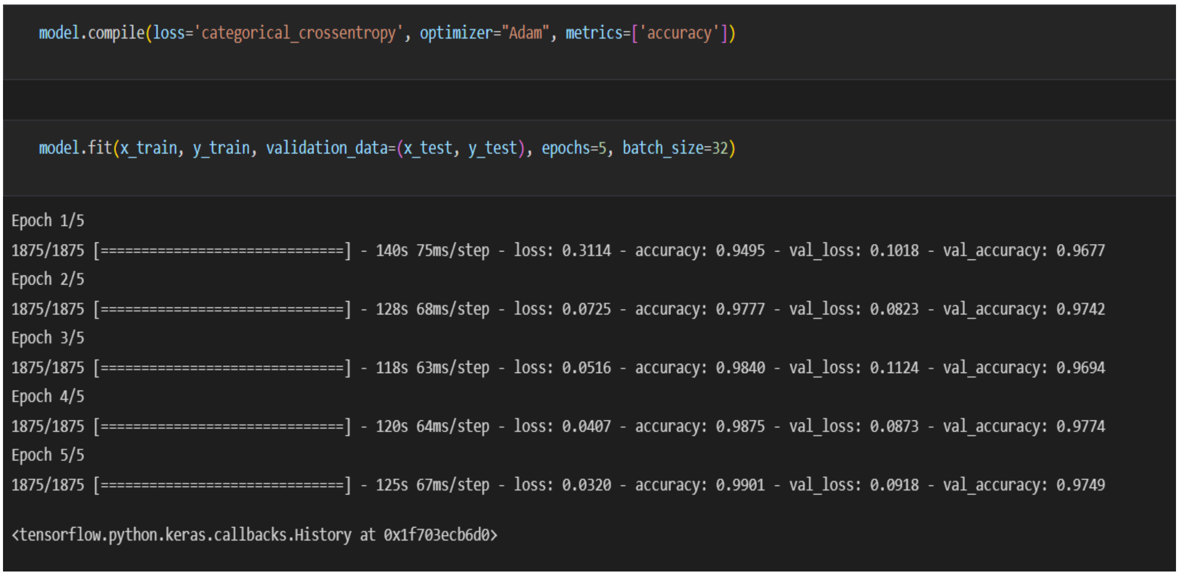
```
number_of_classes = 10
y_train = np_utils.to_categorical(y_train, number_of_classes)
y_test = np_utils.to_categorical(y_test, number_of_classes)
```

ADD CNN LAYERS:



```
model = Sequential()
model.add(Conv2D(64, (3,3), input_shape=(28,28,1), activation='relu'))
model.add(Conv2D(32, (3,3), activation='relu'))
model.add(Flatten())
model.add(Dense(number_of_classes, activation='softmax'))
```

COMPILING AND TRAINING THE MODEL:



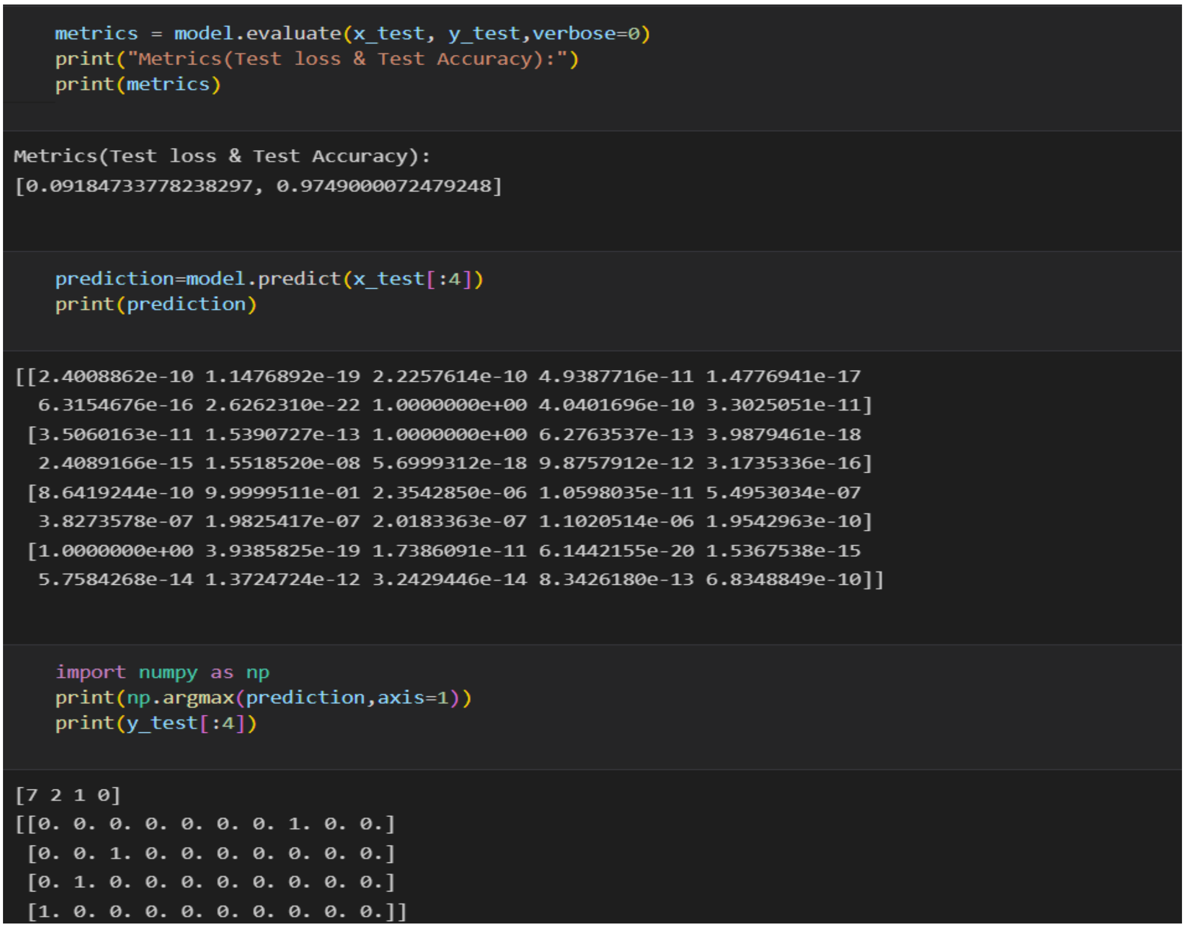
```
model.compile(loss='categorical_crossentropy', optimizer="Adam", metrics=['accuracy'])
```

```
model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=5, batch_size=32)
```

```
Epoch 1/5
1875/1875 [==============================] - 140s 75ms/step - loss: 0.3114 - accuracy: 0.9495 - val_loss: 0.1018 - val_accuracy: 0.9677
Epoch 2/5
1875/1875 [==============================] - 128s 68ms/step - loss: 0.0725 - accuracy: 0.9777 - val_loss: 0.0823 - val_accuracy: 0.9742
Epoch 3/5
1875/1875 [==============================] - 118s 63ms/step - loss: 0.0516 - accuracy: 0.9840 - val_loss: 0.1124 - val_accuracy: 0.9694
Epoch 4/5
1875/1875 [==============================] - 120s 64ms/step - loss: 0.0407 - accuracy: 0.9875 - val_loss: 0.0873 - val_accuracy: 0.9774
Epoch 5/5
1875/1875 [==============================] - 125s 67ms/step - loss: 0.0320 - accuracy: 0.9901 - val_loss: 0.0918 - val_accuracy: 0.9749

<tensorflow.python.keras.callbacks.History at 0x1f703ecb6d0>
```

OBSERVING AND TESTING THE MODEL:



```
metrics = model.evaluate(x_test, y_test,verbose=0)
print("Metrics(Test loss & Test Accuracy):")
print(metrics)
```

```
Metrics(Test loss & Test Accuracy):
[0.09184733778238297, 0.9749000072479248]
```

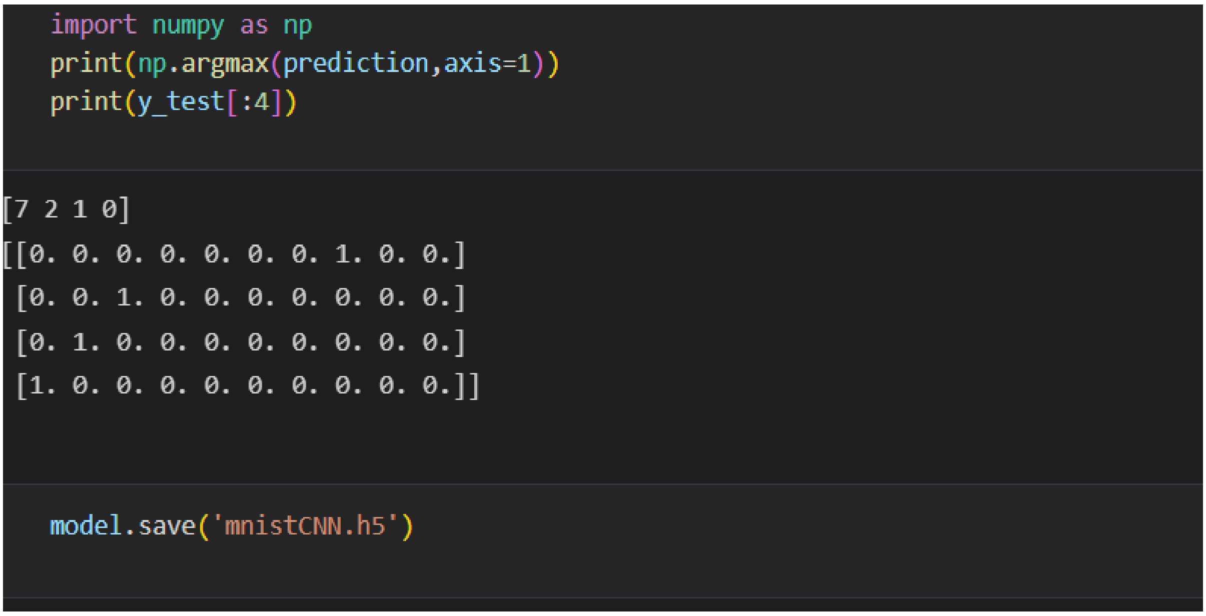
```
prediction=model.predict(x_test[:4])
print(prediction)
```

```
[[2.4008862e-10 1.1476892e-19 2.2257614e-10 4.9387716e-11 1.4776941e-17
  6.3154676e-16 2.6262310e-22 1.0000000e+00 4.0401696e-10 3.3025051e-11]
 [3.5060163e-11 1.5390727e-13 1.0000000e+00 6.2763537e-13 3.9879461e-18
  2.4089166e-15 1.5518520e-08 5.6999312e-18 9.8757912e-12 3.1735336e-16]
 [8.6419244e-10 9.9999511e-01 2.3542850e-06 1.0598035e-11 5.4953034e-07
  3.8273578e-07 1.9825417e-07 2.0183363e-07 1.1020514e-06 1.9542963e-10]
 [1.0000000e+00 3.9385825e-19 1.7386091e-11 6.1442155e-20 1.5367538e-15
  5.7584268e-14 1.3724724e-12 3.2429446e-14 8.3426180e-13 6.8348849e-10]]
```

```
import numpy as np
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

```
[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

SAVE THE MODEL:



```
import numpy as np
print(np.argmax(prediction,axis=1))
print(y_test[:4])
```

```
[7 2 1 0]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

```
model.save('mnistCNN.h5')
```