SPRINT 1

TEAM ID : PNT2022TMID39460

```python
import numpy as np
import tensorflow #open source used for both ML and DL for computation
from tensorflow.keras.datasets import mnist #mnist dataset
from tensorflow.keras.models import Sequential #it is a plain stack of layers
from tensorflow.keras import layers #A Layer consists of a tensor- in tensor-out computat
from tensorflow.keras.layers import Dense, Flatten #Dense-Dense Layer is the regular deepl
#faltten -used fot flattening the input or change the dimension
from tensorflow.keras.layers import Conv2D #onvoLutiona l Layer
from keras.optimizers import Adam #opt imizer
from keras. utils import np_utils #used for one-hot encoding
import matplotlib.pyplot as plt   #used for data visualization
```

LOAD DATA

```python
(x_train, y_train), (x_test, y_test)=mnist.load_data () #splitting the mnist data into tra
```

```python
print (x_train.shape)  #shape is used for give the dimens ion values #60000-rows 28x28-pix
print (x_test.shape)
```

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

```python
x_train[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,   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,    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,   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,  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,   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,   35,
         241,  225,  160,  108,    1,    0,    0,    0,    0,    0,    0,    0,    0,
           0,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          81,  240,  253,  253,  119,   25,    0,    0,    0,    0,    0,    0,    0,
           0,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
           0,   45,  186,  253,  253,  150,   27,    0,    0,    0,    0,    0,    0,
           0,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
           0,    0,   16,   93,  252,  253,  187,    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,  249,  253,  249,   64,    0,    0,    0,    0,    0,
           0,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
           0,   46,  130,  183,  253,  253,  207,    2,    0,    0,    0,    0,    0,
           0,    0],
       [   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   39,
         148,  229,  253,  253,  253,  250,  182,    0,    0,    0,    0,    0,    0,
```
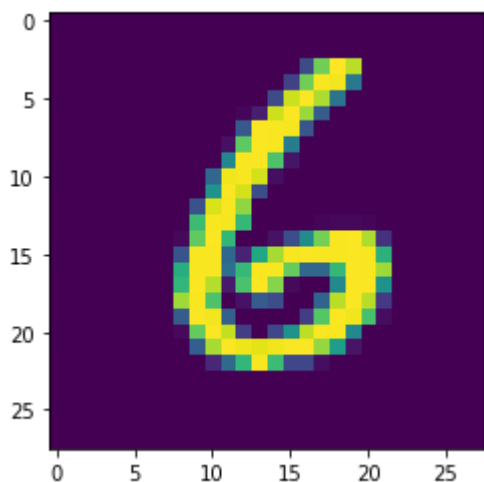
```
plt.imshow(x_train[6000])     #ploting the index=image
```

```
<matplotlib.image.AxesImage at 0x7fd4a9077a90>
```



```
np.argmax(y_train[6000])
```

```
0
```

Reshaping Dataset

```
#Reshaping to format which CNN expects (batch, height, width, channels)
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   #storing the no of classes in a variable


y_train = np_utils.to_categorical (y_train, number_of_classes) #converts the output in bin
y_test = np_utils.to_categorical (y_test, number_of_classes)
```

Colab paid products - Cancel contracts here

✓ 0s   completed at 8:02 PM   ● ✕