```
import numpy as np
import pandas as pd
from numpy import unique, argmax
from tensorflow.keras.datasets.mnist import load_data
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dropout
from tensorflow.keras.utils import plot_model
import matplotlib.pyplot as plt
from tensorflow.keras.datasets import mnist


(train_x, train_y), (test_x, test_y) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

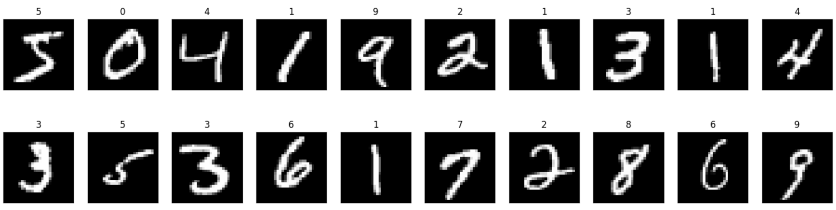
```
#printing the shapes
print(train_x.shape, train_y.shape)
print(test_x.shape , test_y.shape)
```

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

```
#normalizing the pixel values of images
train_x = train_x.astype('float32')/255.0
```



```
#plotting images of dataset
fig = plt.figure(figsize = (20,5))
for i in range(20):
 ax= fig.add_subplot(2, 10, i+1, xticks=[], yticks=[])
 ax.imshow(np.squeeze(train_x[i]), cmap='gray')
 ax.set_title(train_y[i])
```




```
shape = train_x.shape[1:]
shape
```

```
(28, 28)
```

```
#CNN Model
from tensorflow.keras.layers import MaxPooling2D as MaxPool2D
model = Sequential()
#adding convolutional layer
model.add(Conv2D(32, (3,3), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPool2D((2,2)))
model.add(Conv2D(48, (3,3), activation='relu'))
model.add(MaxPool2D((2,2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(500, activation='relu'))
model.add(Dense(10, activation='softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2D  (None, 13, 13, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 11, 11, 48)        13872

 max_pooling2d_1 (MaxPooling  (None, 5, 5, 48)         0
 2D)

 dropout (Dropout)           (None, 5, 5, 48)          0

 flatten (Flatten)           (None, 1200)              0

 dense (Dense)               (None, 500)               600500

 dense_1 (Dense)             (None, 10)                5010

=================================================================
Total params: 619,702
Trainable params: 619,702
Non-trainable params: 0
_________________________________________________________________
```

```
#compiling model
model.compile(optimizer='adam', loss = 'sparse_categorical_crossentropy',metrics= ['accuracy'] )
                                                                    dation_split = 0.1)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

```
Epoch 1/10
422/422 - 44s - loss: 0.2454 - accuracy: 0.9234 - val_loss: 0.0561 - val_accuracy: 0.9840 - 44s/epoch - 104ms/step
Epoch 2/10
422/422 - 42s - loss: 0.0822 - accuracy: 0.9735 - val_loss: 0.0371 - val_accuracy: 0.9888 - 42s/epoch - 99ms/step
Epoch 3/10
422/422 - 43s - loss: 0.0598 - accuracy: 0.9814 - val_loss: 0.0356 - val_accuracy: 0.9892 - 43s/epoch - 103ms/step
Epoch 4/10
422/422 - 42s - loss: 0.0497 - accuracy: 0.9842 - val_loss: 0.0281 - val_accuracy: 0.9915 - 42s/epoch - 99ms/step
Epoch 5/10
422/422 - 42s - loss: 0.0404 - accuracy: 0.9871 - val_loss: 0.0304 - val_accuracy: 0.9905 - 42s/epoch - 99ms/step
Epoch 6/10
422/422 - 43s - loss: 0.0354 - accuracy: 0.9885 - val_loss: 0.0288 - val_accuracy: 0.9908 - 43s/epoch - 102ms/step
Epoch 7/10
422/422 - 41s - loss: 0.0320 - accuracy: 0.9898 - val_loss: 0.0279 - val_accuracy: 0.9920 - 41s/epoch - 98ms/step
Epoch 8/10
422/422 - 42s - loss: 0.0284 - accuracy: 0.9905 - val_loss: 0.0275 - val_accuracy: 0.9937 - 42s/epoch - 98ms/step
Epoch 9/10
422/422 - 45s - loss: 0.0266 - accuracy: 0.9913 - val_loss: 0.0268 - val_accuracy: 0.9933 - 45s/epoch - 106ms/step
Epoch 10/10
422/422 - 42s - loss: 0.0246 - accuracy: 0.9914 - val_loss: 0.0270 - val_accuracy: 0.9920 - 42s/epoch - 100ms/step
```

+ Code + Text

```
loss, accuracy= model.evaluate(test_x, test_y, verbose = 0)
print(f'Accuracy: {accuracy*100}')
```

```
Accuracy: 99.27999973297119
```

✓ 4s completed at 10:39 PM

Automatic saving failed. This file was updated remotely or in another tab. Show diff