```
import tensorflow
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense , Flatten
```

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

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

```
x_train.shape
```

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

```
len(x_train)
```

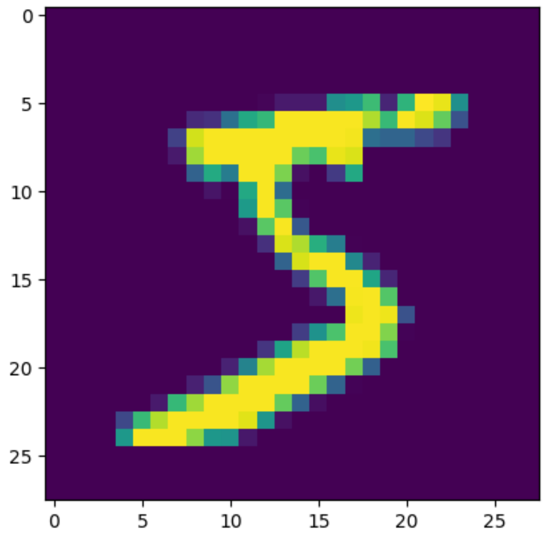
```
60000
```

```
x_train[1]
```

```
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  51, 238, 253,
        253, 190, 114, 253, 228,  47,  79, 255, 168,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,  48, 238, 252, 252,
        179,  12,  75, 121,  21,   0,   0, 253, 243,  50,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,  38, 165, 253, 233, 208,
         84,   0,   0,   0,   0,   0,   0, 253, 252, 165,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   7, 178, 252, 240,  71,  19,
         28,   0,   0,   0,   0,   0,   0, 253, 252, 195,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  57, 252, 252,  63,   0,   0,
          0,   0,   0,   0,   0,   0,   0, 253, 252, 195,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0, 198, 253, 190,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0, 255, 253, 196,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  76, 246, 252, 112,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0, 253, 252, 148,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  85, 252, 230,  25,   0,   0,   0,
          0,   0,   0,   0,   0,   7, 135, 253, 186,  12,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  85, 252, 223,   0,   0,   0,   0,
          0,   0,   0,   0,   7, 131, 252, 225,  71,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  85, 252, 145,   0,   0,   0,   0,
          0,   0,   0,  48, 165, 252, 173,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  86, 253, 225,   0,   0,   0,   0,
          0,   0, 114, 238, 253, 162,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  85, 252, 249, 146,  48,  29,  85,
        178, 225, 253, 223, 167,  56,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  85, 252, 252, 252, 229, 215, 252,
        252, 252, 196, 130,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  28, 199, 252, 252, 253, 252, 252,
        233, 145,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,  25, 128, 252, 253, 252, 141,
         37,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0]], dtype=uint8)
```

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

```
<matplotlib.image.AxesImage at 0x7f20b466edd0>
```



```
x_train = x_train / 255.
x_test = x_test / 255.
```

```
x_train[0]
```

```
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ]])
```

```
model = Sequential()

model.add(Flatten(input_shape = (28 , 28)))
model.add(Dense(128 , activation = "relu"))
model.add(Dense(10 ,activation = 'softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 128)               100480

 dense_1 (Dense)             (None, 10)                1290

=================================================================
Total params: 101,770
Trainable params: 101,770
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(loss = "sparse_categorical_crossentropy" , optimizer = "Adam")
```

```
model.fit(x_train , y_train , epochs = 15 , validation_split = 0.2)
```

```
Epoch 1/15
1500/1500 [==============================] - 8s 5ms/step - loss: 0.2848 - val_loss: 0.1588
Epoch 2/15
1500/1500 [==============================] - 5s 4ms/step - loss: 0.1254 - val_loss: 0.1101
Epoch 3/15
1500/1500 [==============================] - 7s 5ms/step - loss: 0.0864 - val_loss: 0.0999
Epoch 4/15
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0651 - val_loss: 0.0888
Epoch 5/15
1500/1500 [==============================] - 7s 5ms/step - loss: 0.0493 - val_loss: 0.0875
Epoch 6/15
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0402 - val_loss: 0.0867
Epoch 7/15
1500/1500 [==============================] - 7s 4ms/step - loss: 0.0299 - val_loss: 0.0961
Epoch 8/15
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0252 - val_loss: 0.0941
Epoch 9/15
1500/1500 [==============================] - 7s 4ms/step - loss: 0.0189 - val_loss: 0.0938
Epoch 10/15
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0173 - val_loss: 0.1040
Epoch 11/15
1500/1500 [==============================] - 7s 4ms/step - loss: 0.0154 - val_loss: 0.0930
Epoch 12/15
1500/1500 [==============================] - 5s 4ms/step - loss: 0.0104 - val_loss: 0.0983
Epoch 13/15
1500/1500 [==============================] - 12s 8ms/step - loss: 0.0119 - val_loss: 0.1027
Epoch 14/15
1500/1500 [==============================] - 9s 6ms/step - loss: 0.0085 - val_loss: 0.1116
Epoch 15/15
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0075 - val_loss: 0.1158
<keras.callbacks.History at 0x7f20b5ec2cb0>
```

```
y_prob = model.predict(x_test)
y_prob
```

```
313/313 [==============================] - 0s 2ms/step
array([[6.6217028e-12, 4.9329313e-10, 8.7182173e-09, ..., 9.9996823e-01,
             5.2964433e-10, 2.0152461e-06],
            [2.7174196e-16, 2.4983011e-08, 9.9999994e-01, ..., 8.1326343e-25,
             2.1685561e-14, 3.2354656e-27],
            [3.2452921e-10, 9.9996811e-01, 6.6366424e-06, ..., 2.6902881e-06,
             2.1086196e-05, 4.5902857e-10],
            ...,
            [2.7549790e-20, 7.7689950e-14, 2.7030318e-21, ..., 8.5932268e-11,
             3.6087779e-11, 2.4283278e-08],
            [7.4249279e-18, 5.5016094e-21, 1.3112478e-18, ..., 1.6592871e-18,
             5.4723159e-10, 3.1802546e-19],
            [5.2764455e-17, 1.7015963e-22, 3.2704603e-11, ..., 1.3154449e-20,
             8.9802969e-17, 2.3722108e-16]], dtype=float32)
```

```
y_pred = y_prob.argmax(axis = 1)
```

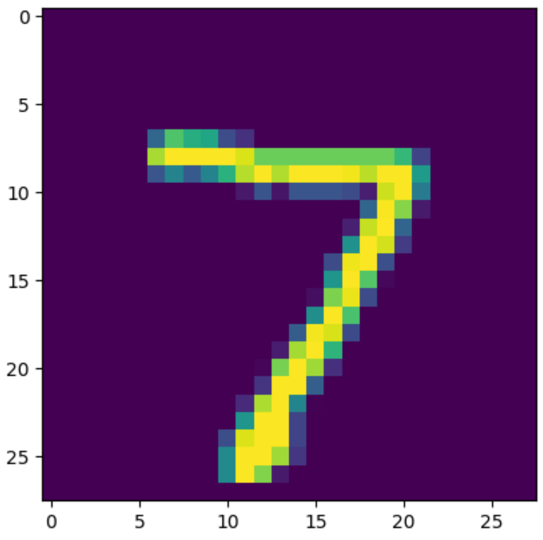
```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(y_test , y_pred)
```

```
0.9767
```

```
# checking the model
plt.imshow(x_test[0])
```

```
<matplotlib.image.AxesImage at 0x7f2083f81b10>
```



```
model.predict(x_test[0].reshape(1 , 28 , 28 )).argmax(axis = 1)
```

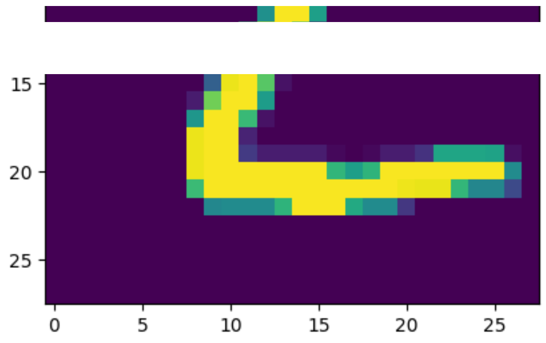
```
1/1 [==============================] - 0s 36ms/step
array([7])
```

```
# checking the model
plt.imshow(x_test[1])
```

```
<matplotlib.image.AxesImage at 0x7f2083fd47f0>
```

```
model.predict(x_test[1].reshape(1 , 28 , 28)).argmax(axis = 1)

    1/1 [==============================] - 0s 41ms/step
    array([2])
```



Colab paid products - Cancel contracts here

✓ 0s completed at 10:24 AM