```
# data analysis
import tensorflow
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense,Flatten



# import dataset
(x_train,y_train),(x_test,y_test)=keras.datasets.mnist.load_data()


x_train
```

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

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

```
y_train
```

```
array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)
```

```
x_train[0].shape
```

```
(28, 28)
```

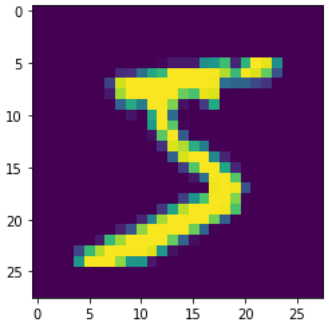
```
x_train[0]
```

```
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,  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,
```

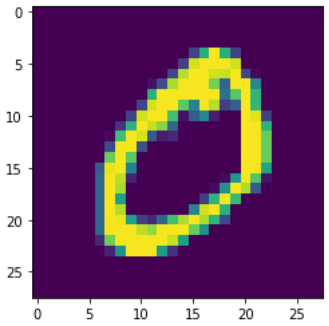
```
import matplotlib.pyplot as plt
plt.imshow(x_train[0])
```

```
<matplotlib.image.AxesImage at 0x7f82a2f5c8b0>
```



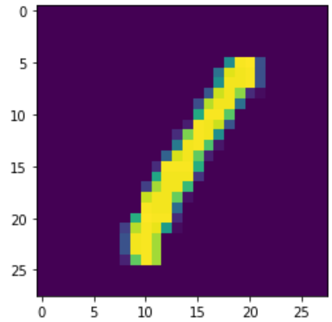

```
plt.imshow(x_train[1])
```

```
<matplotlib.image.AxesImage at 0x7f829d1a9970>
```




```
plt.imshow(x_train[3])
```

```
<matplotlib.image.AxesImage at 0x7f829d17a760>
```



```
x_test[0]
```

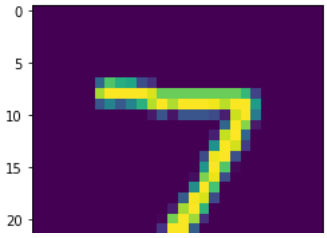
```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   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,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  84, 185, 159, 151,  60,  36,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0, 222, 254, 254, 254, 254, 241, 198,
        198, 198, 198, 198, 198, 198, 198, 170,  52,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,  67, 114,  72, 114, 163, 227, 254,
        225, 254, 254, 254, 250, 229, 254, 254, 140,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,  17,  66,
         14,  67,  67,  67,  59,  21, 236, 254, 106,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,  83, 253, 209,  18,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,  22, 233, 255,  83,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0, 129, 254, 238,  44,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,  59, 249, 254,  62,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0, 133, 254, 187,   5,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   9, 205, 248,  58,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0, 126, 254, 182,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,  75, 251, 240,  57,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
```

```
y_test
```

```
array([7, 2, 1, ..., 4, 5, 6], dtype=uint8)
```

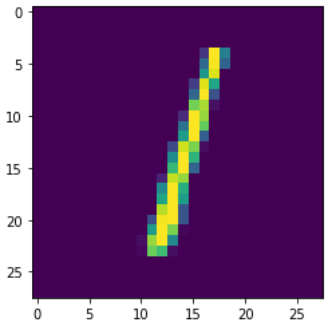
```
plt.imshow(x_test[0])
```

```
<matplotlib.image.AxesImage at 0x7f829d0ae040>
```



```
plt.imshow(x_test[2])
```

```
<matplotlib.image.AxesImage at 0x7f829d098b50>
```



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

```
x_train.shape
```

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

```
y_train
```

```
array([5, 0, 4, ..., 5, 6, 8], dtype=uint8)
```

```
# model building
```

```
model=Sequential()
model.add(Flatten(input_shape=(28,28)))
model.add(Dense(units=125,activation="relu"))
model.add(Dense(units=25,activation="relu"))
```

```
# o/p layer
model.add(Dense(units=10,activation="softmax"))
model.compile(loss="sparse_categorical_crossentropy",optimizer="adam",metrics=["accuracy"])
history=model.fit(x_train,y_train,epochs=20,validation_split=0.20)
```

```
Epoch 1/20
1500/1500 [==============================] - 8s 5ms/step - loss: 2.0035 - accuracy: 0.1997 - val_loss: 1.7984 - val_accuracy: 0.209
Epoch 2/20
1500/1500 [==============================] - 6s 4ms/step - loss: 1.7023 - accuracy: 0.2535 - val_loss: 1.5204 - val_accuracy: 0.304
Epoch 3/20
1500/1500 [==============================] - 7s 5ms/step - loss: 1.4242 - accuracy: 0.3117 - val_loss: 1.3794 - val_accuracy: 0.336
Epoch 4/20
1500/1500 [==============================] - 6s 4ms/step - loss: 1.2605 - accuracy: 0.3973 - val_loss: 1.1725 - val_accuracy: 0.412
Epoch 5/20
1500/1500 [==============================] - 7s 5ms/step - loss: 1.1178 - accuracy: 0.4471 - val_loss: 1.0378 - val_accuracy: 0.490
Epoch 6/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.9416 - accuracy: 0.5357 - val_loss: 0.9041 - val_accuracy: 0.615
Epoch 7/20
1500/1500 [==============================] - 7s 5ms/step - loss: 0.7491 - accuracy: 0.6792 - val_loss: 0.6516 - val_accuracy: 0.695
Epoch 8/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.5811 - accuracy: 0.7031 - val_loss: 0.5936 - val_accuracy: 0.695
Epoch 9/20
1500/1500 [==============================] - 7s 5ms/step - loss: 0.5142 - accuracy: 0.7135 - val_loss: 0.5584 - val_accuracy: 0.696
Epoch 10/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.4632 - accuracy: 0.7744 - val_loss: 0.4568 - val_accuracy: 0.783
Epoch 11/20
1500/1500 [==============================] - 7s 4ms/step - loss: 0.3788 - accuracy: 0.8089 - val_loss: 0.4165 - val_accuracy: 0.802
Epoch 12/20
1500/1500 [==============================] - 5s 4ms/step - loss: 0.3168 - accuracy: 0.8715 - val_loss: 0.3467 - val_accuracy: 0.866
Epoch 13/20
1500/1500 [==============================] - 7s 5ms/step - loss: 0.2631 - accuracy: 0.8852 - val_loss: 0.3182 - val_accuracy: 0.869
Epoch 14/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2360 - accuracy: 0.8877 - val_loss: 0.3501 - val_accuracy: 0.868
Epoch 15/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2256 - accuracy: 0.8886 - val_loss: 0.2972 - val_accuracy: 0.870
```

```
Epoch 16/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2131 - accuracy: 0.8905 - val_loss: 0.2990 - val_accuracy: 0.872
Epoch 17/20
1500/1500 [==============================] - 7s 4ms/step - loss: 0.1803 - accuracy: 0.9486 - val_loss: 0.2325 - val_accuracy: 0.965
Epoch 18/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0878 - accuracy: 0.9893 - val_loss: 0.1617 - val_accuracy: 0.967
Epoch 19/20
1500/1500 [==============================] - 7s 4ms/step - loss: 0.0511 - accuracy: 0.9914 - val_loss: 0.1633 - val_accuracy: 0.969
Epoch 20/20
1500/1500 [==============================] - 6s 4ms/step - loss: 0.0538 - accuracy: 0.9898 - val_loss: 0.1674 - val_accuracy: 0.969
```

```
x_test[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.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.32941176, 0.7254902 , 0.62352941, 0.59215686,
        0.23529412, 0.14117647, 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.87058824, 0.99607843, 0.99607843, 0.99607843,
        0.99607843, 0.94509804, 0.77647059, 0.77647059, 0.77647059,
        0.77647059, 0.77647059, 0.77647059, 0.77647059, 0.77647059,
        0.66666667, 0.20392157, 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.2627451 , 0.44705882, 0.28235294, 0.44705882,
        0.63921569, 0.89019608, 0.99607843, 0.88235294, 0.99607843,
        0.99607843, 0.99607843, 0.98039216, 0.89803922, 0.99607843,
```

```
yprob=model.predict(x_test)
yprob[0]
```

```
313/313 [==============================] - 1s 2ms/step
array([4.2646854e-05, 5.1013085e-05, 6.3467742e-04, 7.7746503e-05,
       9.4124619e-03, 2.9327068e-06, 5.8790439e-09, 9.8967594e-01,
       4.7445305e-06, 9.7875702e-05], dtype=float32)
```

```
ypred=yprob.argmax(axis=1)
ypred
```

```
array([7, 2, 1, ..., 4, 5, 6])
```

```
from sklearn.metrics import classification_report
```

```
print(classification_report(ypred,y_test))
```

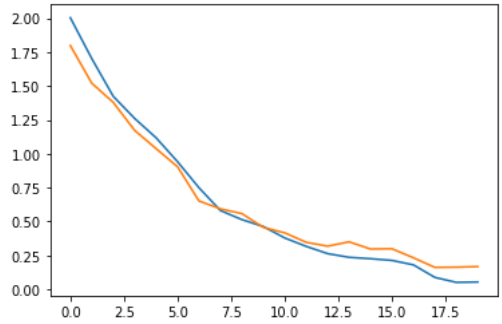
```
              precision    recall  f1-score   support

           0       0.99      0.97      0.98      1007
           1       0.99      0.99      0.99      1142
           2       0.97      0.97      0.97      1034
           3       0.98      0.93      0.96      1060
           4       0.97      0.97      0.97       981
           5       0.95      0.97      0.96       877
           6       0.96      0.99      0.97       933
           7       0.94      0.98      0.96       987
           8       0.96      0.97      0.96       964
           9       0.96      0.96      0.96      1015

    accuracy                           0.97     10000
   macro avg       0.97      0.97      0.97     10000
weighted avg       0.97      0.97      0.97     10000
```

```
plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
```

```
[<matplotlib.lines.Line2D at 0x7f8267ac0cd0>]
```




```
## fasion_mnist datasets
```

```
(xtrain,ytrain),(xtest,ytest)=keras.datasets.fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26421880/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4422102/4422102 [==============================] - 0s 0us/step
```

```
xtrain[0].shape
```

```
(28, 28)
```

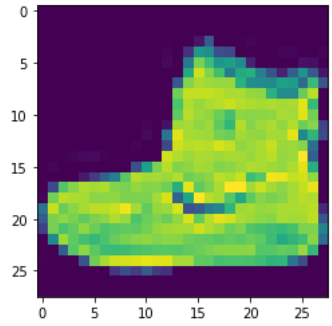
```
xtrain[0]
```

```
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,   1,
          0,   0,  13,  73,   0,   0,   1,   4,   0,   0,   0,   0,   1,
          1,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   3,
          0,  36, 136, 127,  62,  54,   0,   0,   0,   1,   3,   4,   0,
          0,   3],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   6,
          0, 102, 204, 176, 134, 144, 123,  23,   0,   0,   0,   0,  12,
         10,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0, 155, 236, 207, 178, 107, 156, 161, 109,  64,  23,  77, 130,
         72,  15],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   0,
         69, 207, 223, 218, 216, 216, 163, 127, 121, 122, 146, 141,  88,
        172,  66],
```

```
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   1,   1,   0,
 200, 232, 232, 233, 229, 223, 223, 215, 213, 164, 127, 123, 196,
 229,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 183, 225, 216, 223, 228, 235, 227, 224, 222, 224, 221, 223, 245,
 173,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
 193, 228, 218, 213, 198, 180, 212, 210, 211, 213, 223, 220, 243,
 202,   0],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   1,   3,   0,  12,
 219, 220, 212, 218, 192, 169, 227, 208, 218, 224, 212, 226, 197,
 209,  52],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   6,   0,  99,
 244, 222, 220, 218, 203, 198, 221, 215, 213, 222, 220, 245, 119,
 167,  56],
[  0,   0,   0,   0,   0,   0,   0,   0,   0,   4,   0,   0,  55,
 236, 228, 230, 228, 240, 232, 213, 218, 223, 234, 217, 217, 209,
  92,   0],
[  0,   0,   1,   4,   6,   7,   2,   0,   0,   0,   0,   0, 237,
 226, 217, 223, 222, 219, 222, 221, 216, 223, 229, 215, 218, 255,
  77,   0],
[  0,   3,   0,   0,   0,   0,   0,   0,   0,  62, 145, 204, 228,
 207, 213, 221, 218, 208, 211, 218, 224, 223, 219, 215, 224, 244,
 159,   0],
[  0,   0,   0,   0,  18,  44,  82, 107, 189, 228, 220, 222, 217,
 226, 200, 205, 211, 230, 224, 234, 176, 188, 250, 248, 233, 238,
 215,   0],
[  0,  57, 187, 208, 224, 221, 224, 208, 204, 214, 208, 209, 200,
 159, 245, 193, 206, 223, 255, 255, 221, 234, 221, 211, 220, 232,
 246,   0],
[  3, 202, 228, 224, 221, 211, 211, 214, 205, 205, 205, 220, 240,
  80, 150, 255, 229, 221, 188, 154, 191, 210, 204, 209, 222, 228,
 225,   0],
[ 98, 233, 198, 210, 222, 229, 229, 234, 249, 220, 194, 215, 217,
```

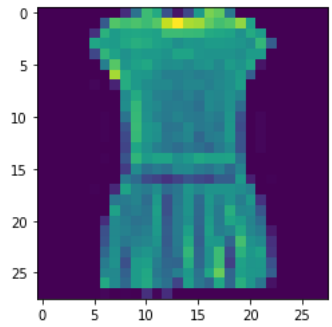
```
import matplotlib.pyplot as plt
plt.imshow(xtrain[0])
```

```
<matplotlib.image.AxesImage at 0x7f826833bee0>
```




```
plt.imshow(xtrain[3])
```

```
<matplotlib.image.AxesImage at 0x7f826b26d670>
```




```
plt.imshow(xtrain[8])
```

```
<matplotlib.image.AxesImage at 0x7f826b3c31f0>
```


```
xtest[0]
```

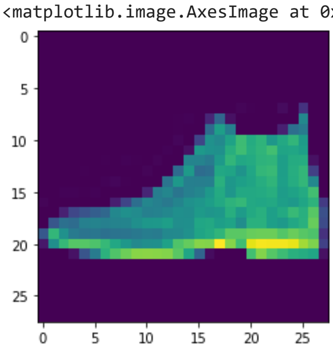
```
array([[  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   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,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   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,   1,   0,   0,   7,   0,  37,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          1,   2,   0,  27,  84,  11,   0,   0,   0,   0,   0,   0, 119,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          1,   0,   0,  88, 143, 110,   0,   0,   0,   0,  22,  93, 106,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          4,   0,  53, 129, 120, 147, 175, 157, 166, 135, 154, 168, 140,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   2,
          0,  11, 137, 130, 128, 160, 176, 159, 167, 178, 149, 151, 144,
          0,   0],
       [  0,   0,   0,   0,   0,   0,   1,   0,   2,   1,   0,   3,   0,
          0, 115, 114, 106, 137, 168, 153, 156, 165, 167, 143, 157, 158,
         11,   0],
       [  0,   0,   0,   0,   1,   0,   0,   0,   0,   0,   3,   0,   0,
         89, 139,  90,  94, 153, 149, 131, 151, 169, 172, 143, 159, 169,
         48,   0],
       [  0,   0,   0,   0,   0,   0,   2,   4,   1,   0,   0,   0,  98,
        136, 110, 109, 110, 162, 135, 144, 149, 159, 167, 144, 158, 169,
        119,   0],
       [  0,   0,   2,   2,   1,   2,   0,   0,   0,   0,  26, 108, 117,
         99, 111, 117, 136, 156, 134, 154, 154, 156, 160, 141, 147, 156,
        178,   0],
       [  3,   0,   0,   0,   0,   0,   0,  21,  53,  92, 117, 111, 103,
        115, 129, 134, 143, 154, 165, 170, 154, 151, 154, 143, 138, 150,
        165,  43],
       [  0,   0,  23,  54,  65,  76,  85, 118, 128, 123, 111, 113, 118,
        127, 125, 139, 133, 136, 160, 140, 155, 161, 144, 155, 172, 161,
        189,  62],
       [  0,  68,  94,  90, 111, 114, 111, 114, 115, 127, 135, 136, 143,
        126, 127, 151, 154, 143, 148, 125, 162, 162, 144, 138, 153, 162,
        196,  58],
       [ 70, 169, 129, 104,  98, 100,  94,  97,  98, 102, 108, 106, 119,
```

```
ytest
```

```
array([9, 2, 1, ..., 8, 1, 5], dtype=uint8)
```

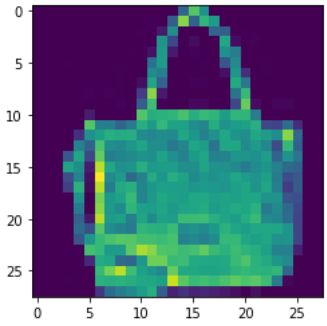
```
plt.imshow(xtest[0])
```

```
<matplotlib.image.AxesImage at 0x7f826bbabb50>
```



```
plt.imshow(xtest[2000])
```

```
<matplotlib.image.AxesImage at 0x7f826b0d03d0>
```




```
xtest.shape
```

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

```
xtrain=xtrain/255
xtest=xtest/255
```

```
xtrain.shape
```

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

```
ytrain
```

```
array([9, 0, 0, ..., 3, 0, 5], dtype=uint8)
```

```
model1=Sequential()
model1.add(Flatten(input_shape=(28,28)))
model1.add(Dense(units=145,activation="relu"))
model1.add(Dense(units=42,activation="relu"))
model1.add(Dense(units=12,activation="softmax"))
model1.compile(loss="sparse_categorical_crossentropy",optimizer="adam",metrics=["accuracy"])
history1=model1.fit(xtrain,ytrain,epochs=22,validation_split=0.20)
```

```
Epoch 1/22
1500/1500 [==============================] - 7s 4ms/step - loss: 0.5229 - accuracy: 0.8149 - val_loss: 0.4000 - val_accuracy: 0.856
Epoch 2/22
1500/1500 [==============================] - 7s 5ms/step - loss: 0.3834 - accuracy: 0.8618 - val_loss: 0.3806 - val_accuracy: 0.866
Epoch 3/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.3453 - accuracy: 0.8726 - val_loss: 0.3748 - val_accuracy: 0.864
Epoch 4/22
1500/1500 [==============================] - 7s 4ms/step - loss: 0.3193 - accuracy: 0.8829 - val_loss: 0.3561 - val_accuracy: 0.873
Epoch 5/22
1500/1500 [==============================] - 7s 5ms/step - loss: 0.2991 - accuracy: 0.8888 - val_loss: 0.3309 - val_accuracy: 0.879
Epoch 6/22
1500/1500 [==============================] - 7s 5ms/step - loss: 0.2862 - accuracy: 0.8945 - val_loss: 0.3233 - val_accuracy: 0.884
Epoch 7/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2698 - accuracy: 0.8992 - val_loss: 0.3383 - val_accuracy: 0.882
Epoch 8/22
1500/1500 [==============================] - 7s 5ms/step - loss: 0.2653 - accuracy: 0.9003 - val_loss: 0.3451 - val_accuracy: 0.878
Epoch 9/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2522 - accuracy: 0.9051 - val_loss: 0.3497 - val_accuracy: 0.877
Epoch 10/22
1500/1500 [==============================] - 7s 4ms/step - loss: 0.2427 - accuracy: 0.9085 - val_loss: 0.3323 - val_accuracy: 0.884
Epoch 11/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2319 - accuracy: 0.9128 - val_loss: 0.3369 - val_accuracy: 0.887
Epoch 12/22
1500/1500 [==============================] - 7s 4ms/step - loss: 0.2233 - accuracy: 0.9156 - val_loss: 0.3331 - val_accuracy: 0.890
Epoch 13/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2164 - accuracy: 0.9198 - val_loss: 0.3272 - val_accuracy: 0.885
Epoch 14/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2104 - accuracy: 0.9206 - val_loss: 0.3261 - val_accuracy: 0.890
Epoch 15/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.2019 - accuracy: 0.9234 - val_loss: 0.3470 - val_accuracy: 0.890
Epoch 16/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1969 - accuracy: 0.9245 - val_loss: 0.3412 - val_accuracy: 0.889
Epoch 17/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1931 - accuracy: 0.9266 - val_loss: 0.3592 - val_accuracy: 0.885
Epoch 18/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1862 - accuracy: 0.9289 - val_loss: 0.3642 - val_accuracy: 0.885
Epoch 19/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1827 - accuracy: 0.9315 - val_loss: 0.3631 - val_accuracy: 0.889
Epoch 20/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1745 - accuracy: 0.9340 - val_loss: 0.3610 - val_accuracy: 0.888
Epoch 21/22
1500/1500 [==============================] - 6s 4ms/step - loss: 0.1766 - accuracy: 0.9338 - val_loss: 0.3621 - val_accuracy: 0.892
```

```
Epoch 22/22
1500/1500 [==============================] - 8s 5ms/step - loss: 0.1672 - accuracy: 0.9367 - val_loss: 0.3807 - val_accuracy: 0.890
```

```
xtest[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.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.01176471,
        0.00392157, 0.        , 0.        , 0.02745098, 0.        ,
        0.14509804, 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.00392157, 0.00784314,
        0.        , 0.10588235, 0.32941176, 0.04313725, 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.46666667, 0.        , 0.        ],
       [0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.00392157, 0.        ,
        0.        , 0.34509804, 0.56078431, 0.43137255, 0.        ,
```

```
yproba=model1.predict(xtest)
yproba[0]
```

```
313/313 [==============================] - 1s 2ms/step
array([8.5230747e-09, 5.8387460e-08, 6.1143937e-11, 1.5296347e-08,
       6.0609816e-11, 2.6613157e-05, 5.7597094e-10, 6.9966274e-03,
       1.2826933e-11, 9.9297661e-01, 1.4302605e-18, 6.6023446e-17],
      dtype=float32)
```

```
y_pred=yproba.argmax(1)
y_pred
```

```
array([9, 2, 1, ..., 8, 1, 5])
```

```
from sklearn.metrics import classification_report
print(classification_report(y_pred,ytest))
```

```
              precision    recall  f1-score   support

           0       0.80      0.87      0.83       921
           1       0.97      0.98      0.98       990
```

```
              2       0.77      0.82      0.80       932
              3       0.89      0.89      0.89      1001
              4       0.87      0.76      0.81      1155
              5       0.96      0.97      0.96       995
              6       0.71      0.71      0.71      1000
              7       0.97      0.91      0.94      1059
              8       0.96      0.97      0.97       996
              9       0.92      0.97      0.95       951

       accuracy                           0.88     10000
      macro avg       0.88      0.88      0.88     10000
   weighted avg       0.88      0.88      0.88     10000
```

```
plt.plot(history.history["loss"])
plt.plot(history.history["val_loss"])
```

```
[<matplotlib.lines.Line2D at 0x7f826e7ccc10>]
```