DATASET PREPROCESSING

```
In [1]: import matplotlib.pyplot as plt
        import tensorflow

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

```
C:\ProgramData\Anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarn
ing: A NumPy version >=1.16.5 and <1.23.0 is required for this version of
SciPy (detected version 1.24.3
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

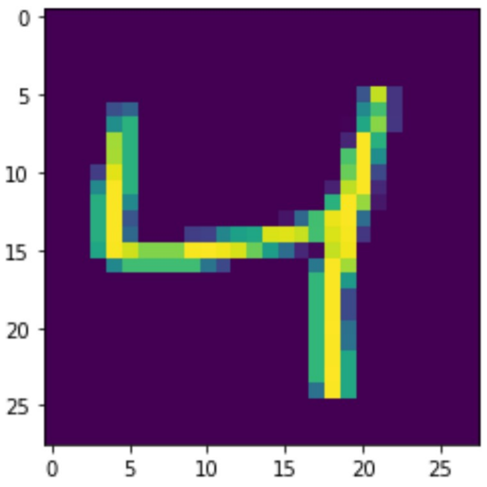
```
In [2]: (X_train,y_train),(X_test,y_test) = keras.datasets.mnist.load_data()
```

```
In [3]: X_test.shape
        X_train.shape
        y_train.shape
        y_test.shape

Out[3]: (10000,)
```

```
In [4]: plt.imshow(X_train[2])

Out[4]: <matplotlib.image.AxesImage at 0x1b98a014d00>
```



```
In [5]: X_train = X_train/255
        X_test = X_test/255
```

SCENARIO - 1

```
In [6]: model1 = Sequential()
        model1.add(Flatten(input_shape=(28,28)))
        model1.add(Dense(128,activation='sigmoid')) #2^n < 28*28
        model1.add(Dense(32,activation='sigmoid'))
        model1.add(Dense(10,activation='softmax')) #no of classes
```

In [7]:
```
model1.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 flatten (Flatten)           (None, 784)               0         
                                                                 
 dense (Dense)               (None, 128)               100480    
                                                                 
 dense_1 (Dense)             (None, 32)                4128      
                                                                 
 dense_2 (Dense)             (None, 10)                330       
                                                                 
=================================================================
Total params: 104938 (409.91 KB)
Trainable params: 104938 (409.91 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [8]:
```
optimizer = keras.optimizers.Adagrad(learning_rate=0.001)
model1.compile(loss='sparse_categorical_crossentropy',optimizer=optimizer,m
In [9]: history1 = model1.fit(X_train,y_train,epochs=24,validation_split=0.3)
```

```
Epoch 1/24
1313/1313 [==============================] - 2s 2ms/step - loss: 2.3191 -
accuracy: 0.1362 - val_loss: 2.2736 - val_accuracy: 0.2373
Epoch 2/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.2504 -
accuracy: 0.2837 - val_loss: 2.2285 - val_accuracy: 0.3399
Epoch 3/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.2067 -
accuracy: 0.4120 - val_loss: 2.1835 - val_accuracy: 0.4909
Epoch 4/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.1605 -
accuracy: 0.5235 - val_loss: 2.1351 - val_accuracy: 0.5516
Epoch 5/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.1104 -
accuracy: 0.5661 - val_loss: 2.0828 - val_accuracy: 0.5968
Epoch 6/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.0569 -
accuracy: 0.5964 - val_loss: 2.0277 - val_accuracy: 0.6204
Epoch 7/24
1313/1313 [==============================] - 2s 1ms/step - loss: 2.0011 -
accuracy: 0.6168 - val_loss: 1.9709 - val_accuracy: 0.6324
Epoch 8/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.9443 -
accuracy: 0.6233 - val_loss: 1.9134 - val_accuracy: 0.6456
Epoch 9/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.8877 -
accuracy: 0.6357 - val_loss: 1.8567 - val_accuracy: 0.6519
Epoch 10/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.8323 -
accuracy: 0.6417 - val_loss: 1.8018 - val_accuracy: 0.6597
Epoch 11/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.7789 -
accuracy: 0.6486 - val_loss: 1.7490 - val_accuracy: 0.6679
Epoch 12/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.7279 -
accuracy: 0.6565 - val_loss: 1.6988 - val_accuracy: 0.6734
Epoch 13/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.6795 -
accuracy: 0.6634 - val_loss: 1.6513 - val_accuracy: 0.6782
Epoch 14/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.6338 -
accuracy: 0.6684 - val_loss: 1.6064 - val_accuracy: 0.6853
Epoch 15/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.5907 -
accuracy: 0.6762 - val_loss: 1.5640 - val_accuracy: 0.6927
Epoch 16/24
1313/1313 [==============================] - 2s 2ms/step - loss: 1.5501 -
accuracy: 0.6822 - val_loss: 1.5242 - val_accuracy: 0.6988
Epoch 17/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.5119 -
accuracy: 0.6890 - val_loss: 1.4867 - val_accuracy: 0.7036
Epoch 18/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.4759 -
accuracy: 0.6946 - val_loss: 1.4513 - val_accuracy: 0.7082
Epoch 19/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.4419 -
accuracy: 0.6999 - val_loss: 1.4179 - val_accuracy: 0.7142
```

```
Epoch 20/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.4098 -
accuracy: 0.7056 - val_loss: 1.3864 - val_accuracy: 0.7187
Epoch 21/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.3794 -
accuracy: 0.7110 - val_loss: 1.3565 - val_accuracy: 0.7241
Epoch 22/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.3506 -
accuracy: 0.7163 - val_loss: 1.3282 - val_accuracy: 0.7281
Epoch 23/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.3232 -
accuracy: 0.7215 - val_loss: 1.3012 - val_accuracy: 0.7329
Epoch 24/24
1313/1313 [==============================] - 2s 1ms/step - loss: 1.2972 -
accuracy: 0.7263 - val_loss: 1.2756 - val_accuracy: 0.7373
```

In [10]:
```python
y_prob = model1.predict(X_test)
y_pred = y_prob.argmax(axis=1)
```

```
313/313 [==============================] - 0s 731us/step
```

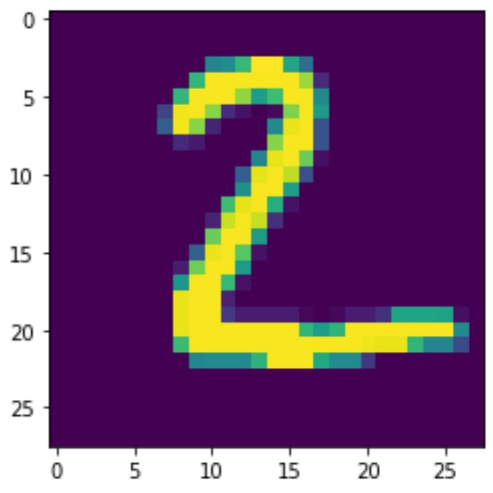
In [11]:
```python
from sklearn.metrics import accuracy_score
a1 = accuracy_score(y_test,y_pred)
a1
```

Out[11]:
```
0.7381
```

In [12]:
```python
plt.imshow(X_test[1])
model1.predict(X_test[1].reshape(1,28,28)).argmax(axis=1)
```

```
1/1 [==============================] - 0s 13ms/step
```

Out[12]:
```
array([2], dtype=int64)
```



SCENARIO - 2

```
In [13]: model2 = Sequential()  ###
         model2.add(Flatten(input_shape=(28,28)))
         model2.add(Dense(128,activation='relu')) #change - 1
         model2.add(Dense(32,activation='relu'))
         model2.add(Dense(10,activation='softmax'))
         model2.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_1 (Flatten)         (None, 784)               0

 dense_3 (Dense)             (None, 128)               100480

 dense_4 (Dense)             (None, 32)                4128

 dense_5 (Dense)             (None, 10)                330

=================================================================
Total params: 104938 (409.91 KB)
Trainable params: 104938 (409.91 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [14]:
```
optimizer = keras.optimizers.Adam(learning_rate=0.001) #change - 2
model2.compile(loss='sparse_categorical_crossentropy',optimizer=optimizer,m
history2 = model2.fit(X_train,y_train,epochs=25,validation_split=0.2) #chan
```

```
Epoch 1/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.2886 -
accuracy: 0.9160 - val_loss: 0.1444 - val_accuracy: 0.9571
Epoch 2/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.1160 -
accuracy: 0.9660 - val_loss: 0.1053 - val_accuracy: 0.9686
Epoch 3/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0781 -
accuracy: 0.9765 - val_loss: 0.1139 - val_accuracy: 0.9665
Epoch 4/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0586 -
accuracy: 0.9823 - val_loss: 0.1057 - val_accuracy: 0.9686
Epoch 5/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0462 -
accuracy: 0.9854 - val_loss: 0.0981 - val_accuracy: 0.9712
Epoch 6/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0359 -
accuracy: 0.9884 - val_loss: 0.1083 - val_accuracy: 0.9702
Epoch 7/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0312 -
accuracy: 0.9896 - val_loss: 0.1004 - val_accuracy: 0.9748
Epoch 8/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0255 -
accuracy: 0.9920 - val_loss: 0.1112 - val_accuracy: 0.9727
Epoch 9/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0213 -
accuracy: 0.9928 - val_loss: 0.1220 - val_accuracy: 0.9710
Epoch 10/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0197 -
accuracy: 0.9937 - val_loss: 0.1225 - val_accuracy: 0.9719
Epoch 11/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0150 -
accuracy: 0.9954 - val_loss: 0.1243 - val_accuracy: 0.9732
Epoch 12/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0150 -
accuracy: 0.9951 - val_loss: 0.1348 - val_accuracy: 0.9745
Epoch 13/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0141 -
accuracy: 0.9955 - val_loss: 0.1354 - val_accuracy: 0.9723
Epoch 14/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0127 -
accuracy: 0.9959 - val_loss: 0.1363 - val_accuracy: 0.9744
Epoch 15/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0110 -
accuracy: 0.9963 - val_loss: 0.1335 - val_accuracy: 0.9748
Epoch 16/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0117 -
accuracy: 0.9962 - val_loss: 0.1500 - val_accuracy: 0.9742
Epoch 17/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0111 -
accuracy: 0.9964 - val_loss: 0.1772 - val_accuracy: 0.9705
Epoch 18/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0101 -
accuracy: 0.9965 - val_loss: 0.1524 - val_accuracy: 0.9756
Epoch 19/25
```

```
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0095 -
accuracy: 0.9968 - val_loss: 0.1913 - val_accuracy: 0.9697
Epoch 20/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0086 -
accuracy: 0.9969 - val_loss: 0.1479 - val_accuracy: 0.9739
Epoch 21/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0097 -
accuracy: 0.9969 - val_loss: 0.1573 - val_accuracy: 0.9732
Epoch 22/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0085 -
accuracy: 0.9970 - val_loss: 0.1567 - val_accuracy: 0.9741
Epoch 23/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0071 -
accuracy: 0.9974 - val_loss: 0.1653 - val_accuracy: 0.9757
Epoch 24/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0082 -
accuracy: 0.9970 - val_loss: 0.1696 - val_accuracy: 0.9751
Epoch 25/25
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0085 -
accuracy: 0.9972 - val_loss: 0.1741 - val_accuracy: 0.9752
```

In [15]:
```python
from sklearn.metrics import accuracy_score
y_prob = model2.predict(X_test)  ###
y_pred = y_prob.argmax(axis=1)
a2 = accuracy_score(y_test,y_pred)
a2
```

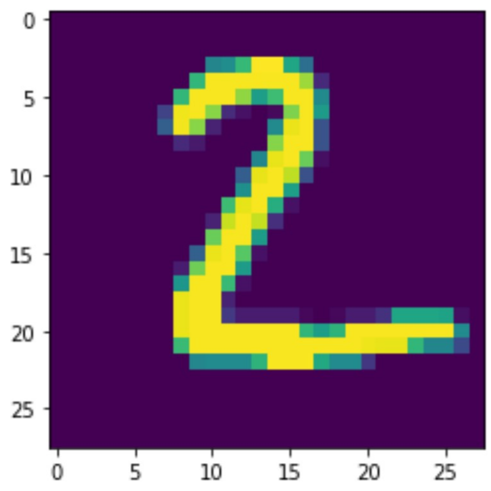
```
313/313 [==============================] - 0s 716us/step
```

Out[15]:
```
0.9759
```

In [16]:
```python
plt.imshow(X_test[1])
model2.predict(X_test[1].reshape(1,28,28)).argmax(axis=1) ###
```

```
1/1 [==============================] - 0s 11ms/step
```

Out[16]:
```
array([2], dtype=int64)
```



SCENARIO - 3

```
In [17]: model3 = Sequential()  ###
         model3.add(Flatten(input_shape=(28,28)))
         model3.add(Dense(128,activation='tanh')) #change - 1
         model3.add(Dense(32,activation='tanh'))
         model3.add(Dense(10,activation='softmax'))
         model3.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten_2 (Flatten)         (None, 784)               0

 dense_6 (Dense)             (None, 128)               100480

 dense_7 (Dense)             (None, 32)                4128

 dense_8 (Dense)             (None, 10)                330

=================================================================
Total params: 104938 (409.91 KB)
Trainable params: 104938 (409.91 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
In [18]: optimizer = keras.optimizers.Adadelta(learning_rate=0.001) #change - 2
         model3.compile(loss='sparse_categorical_crossentropy',optimizer=optimizer,m
         history3 = model3.fit(X_train,y_train,epochs=21,validation_split=0.1) #chan
```

```
Epoch 1/21
1688/1688 [==============================] - 3s 1ms/step - loss: 2.2132 -
accuracy: 0.1669 - val_loss: 2.0641 - val_accuracy: 0.2740
Epoch 2/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.9646 -
accuracy: 0.3836 - val_loss: 1.8337 - val_accuracy: 0.4960
Epoch 3/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.7634 -
accuracy: 0.5368 - val_loss: 1.6451 - val_accuracy: 0.5973
Epoch 4/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.5989 -
accuracy: 0.6063 - val_loss: 1.4906 - val_accuracy: 0.6543
Epoch 5/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.4651 -
accuracy: 0.6480 - val_loss: 1.3643 - val_accuracy: 0.6937
Epoch 6/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.3558 -
accuracy: 0.6786 - val_loss: 1.2607 - val_accuracy: 0.7212
Epoch 7/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.2660 -
accuracy: 0.7022 - val_loss: 1.1752 - val_accuracy: 0.7457
Epoch 8/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.1915 -
accuracy: 0.7195 - val_loss: 1.1035 - val_accuracy: 0.7663
Epoch 9/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.1284 -
accuracy: 0.7357 - val_loss: 1.0421 - val_accuracy: 0.7827
Epoch 10/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.0740 -
accuracy: 0.7497 - val_loss: 0.9891 - val_accuracy: 0.7972
Epoch 11/21
1688/1688 [==============================] - 2s 1ms/step - loss: 1.0268 -
accuracy: 0.7617 - val_loss: 0.9429 - val_accuracy: 0.8078
Epoch 12/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.9855 -
accuracy: 0.7722 - val_loss: 0.9022 - val_accuracy: 0.8173
Epoch 13/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.9487 -
accuracy: 0.7814 - val_loss: 0.8659 - val_accuracy: 0.8260
Epoch 14/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.9158 -
accuracy: 0.7891 - val_loss: 0.8332 - val_accuracy: 0.8345
Epoch 15/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.8860 -
accuracy: 0.7967 - val_loss: 0.8039 - val_accuracy: 0.8423
Epoch 16/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.8591 -
accuracy: 0.8036 - val_loss: 0.7771 - val_accuracy: 0.8500
Epoch 17/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.8344 -
accuracy: 0.8106 - val_loss: 0.7527 - val_accuracy: 0.8552
Epoch 18/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.8118 -
accuracy: 0.8162 - val_loss: 0.7303 - val_accuracy: 0.8600
Epoch 19/21
```

```
1688/1688 [==============================] - 2s 1ms/step - loss: 0.7909 -
accuracy: 0.8209 - val_loss: 0.7095 - val_accuracy: 0.8657
Epoch 20/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.7715 -
accuracy: 0.8259 - val_loss: 0.6904 - val_accuracy: 0.8700
Epoch 21/21
1688/1688 [==============================] - 2s 1ms/step - loss: 0.7536 -
accuracy: 0.8304 - val_loss: 0.6726 - val_accuracy: 0.8742
```

In [19]:
```
from sklearn.metrics import accuracy_score
y_prob = model3.predict(X_test)  ###
y_pred = y_prob.argmax(axis=1)
a3 = accuracy_score(y_test,y_pred)
a3
```

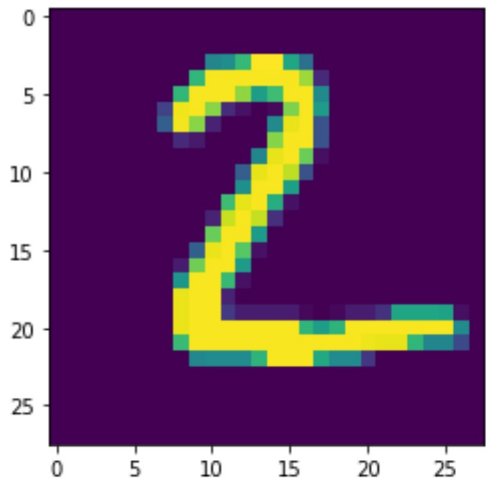
```
313/313 [==============================] - 0s 816us/step
```

Out[19]:
```
0.8442
```

In [20]:
```
plt.imshow(X_test[1])
model3.predict(X_test[1].reshape(1,28,28)).argmax(axis=1) ###
```

```
1/1 [==============================] - 0s 12ms/step
```

Out[20]:
```
array([2], dtype=int64)
```



SCENARIO - 4

In [21]:
```
model4 = Sequential()  ###
model4.add(Flatten(input_shape=(28,28)))
model4.add(Dense(128,activation='sigmoid')) #change - 1
model4.add(Dense(32,activation='sigmoid'))
model4.add(Dense(10,activation='softmax'))
model4.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 flatten_3 (Flatten)         (None, 784)               0         
                                                                 
 dense_9 (Dense)             (None, 128)               100480    
                                                                 
 dense_10 (Dense)            (None, 32)                4128      
                                                                 
 dense_11 (Dense)            (None, 10)                330       
                                                                 
=================================================================
Total params: 104938 (409.91 KB)
Trainable params: 104938 (409.91 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

In [22]:
```python
optimizer = keras.optimizers.Adam(learning_rate=0.001) #change - 2
model4.compile(loss='sparse_categorical_crossentropy',optimizer=optimizer,m
history4 = model4.fit(X_train,y_train,epochs=23,validation_split=0.2) #chan
```

```
Epoch 1/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.6130 -
accuracy: 0.8598 - val_loss: 0.2550 - val_accuracy: 0.9304
Epoch 2/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.2186 -
accuracy: 0.9379 - val_loss: 0.1781 - val_accuracy: 0.9497
Epoch 3/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.1538 -
accuracy: 0.9563 - val_loss: 0.1418 - val_accuracy: 0.9609
Epoch 4/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.1155 -
accuracy: 0.9669 - val_loss: 0.1182 - val_accuracy: 0.9658
Epoch 5/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0896 -
accuracy: 0.9747 - val_loss: 0.1065 - val_accuracy: 0.9694
Epoch 6/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0716 -
accuracy: 0.9799 - val_loss: 0.0970 - val_accuracy: 0.9716
Epoch 7/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0585 -
accuracy: 0.9837 - val_loss: 0.0910 - val_accuracy: 0.9723
Epoch 8/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0470 -
accuracy: 0.9874 - val_loss: 0.0930 - val_accuracy: 0.9727
Epoch 9/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0386 -
accuracy: 0.9898 - val_loss: 0.0891 - val_accuracy: 0.9732
Epoch 10/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0304 -
accuracy: 0.9920 - val_loss: 0.0896 - val_accuracy: 0.9742
Epoch 11/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0249 -
accuracy: 0.9940 - val_loss: 0.0954 - val_accuracy: 0.9728
Epoch 12/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0199 -
accuracy: 0.9953 - val_loss: 0.0884 - val_accuracy: 0.9749
Epoch 13/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0159 -
accuracy: 0.9965 - val_loss: 0.0919 - val_accuracy: 0.9737
Epoch 14/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0127 -
accuracy: 0.9974 - val_loss: 0.0882 - val_accuracy: 0.9759
Epoch 15/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0102 -
accuracy: 0.9982 - val_loss: 0.0920 - val_accuracy: 0.9743
Epoch 16/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0078 -
accuracy: 0.9987 - val_loss: 0.0949 - val_accuracy: 0.9740
Epoch 17/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0071 -
accuracy: 0.9987 - val_loss: 0.0989 - val_accuracy: 0.9747
Epoch 18/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0055 -
accuracy: 0.9990 - val_loss: 0.0943 - val_accuracy: 0.9766
Epoch 19/23
```

```
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0045 -
accuracy: 0.9993 - val_loss: 0.0979 - val_accuracy: 0.9757
Epoch 20/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0032 -
accuracy: 0.9995 - val_loss: 0.0983 - val_accuracy: 0.9759
Epoch 21/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0041 -
accuracy: 0.9993 - val_loss: 0.1012 - val_accuracy: 0.9753
Epoch 22/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0016 -
accuracy: 0.9999 - val_loss: 0.1145 - val_accuracy: 0.9740
Epoch 23/23
1500/1500 [==============================] - 2s 1ms/step - loss: 0.0039 -
```

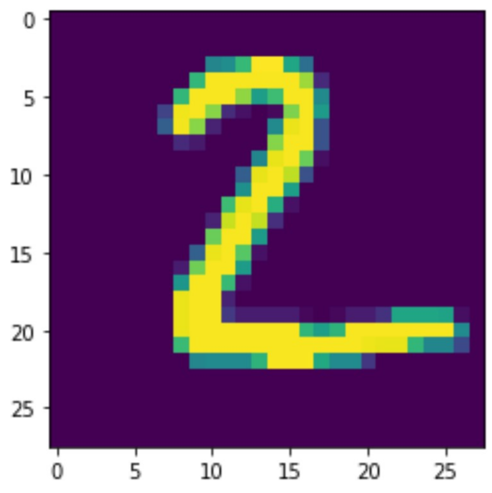
In [23]:
```python
from sklearn.metrics import accuracy_score
y_prob = model4.predict(X_test)  ###
y_pred = y_prob.argmax(axis=1)
a4 = accuracy_score(y_test,y_pred)
```

```
313/313 [==============================] - 0s 716us/step
```

In [24]:
```python
plt.imshow(X_test[1])
model4.predict(X_test[1].reshape(1,28,28)).argmax(axis=1) ###
```

```
1/1 [==============================] - 0s 11ms/step
```

Out[24]:
```
array([2], dtype=int64)
```



VISUALIZATION

In [25]:
```python
import numpy as np
```

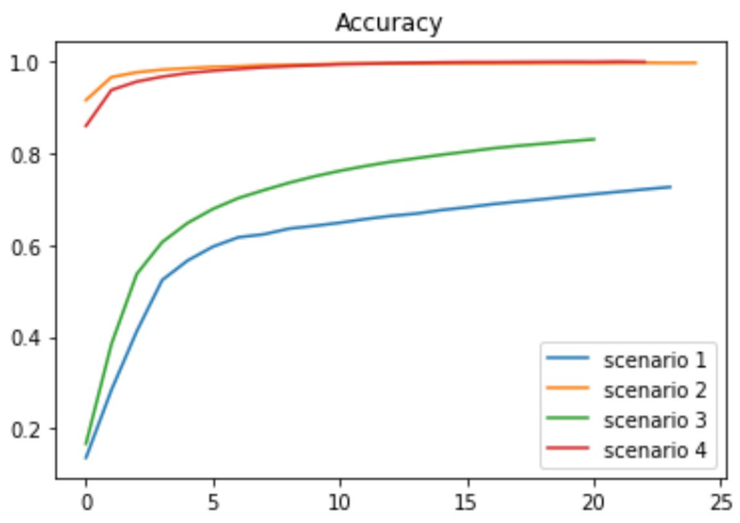
In [26]:
```python
a = [a1,a2,a3,a4]
s = ['scenario 1','scenario 2','scenario 3','scenario 4']
```

In [27]:
```python
a
```

Out[27]:
```
[0.7381, 0.9759, 0.8442, 0.9764]
```

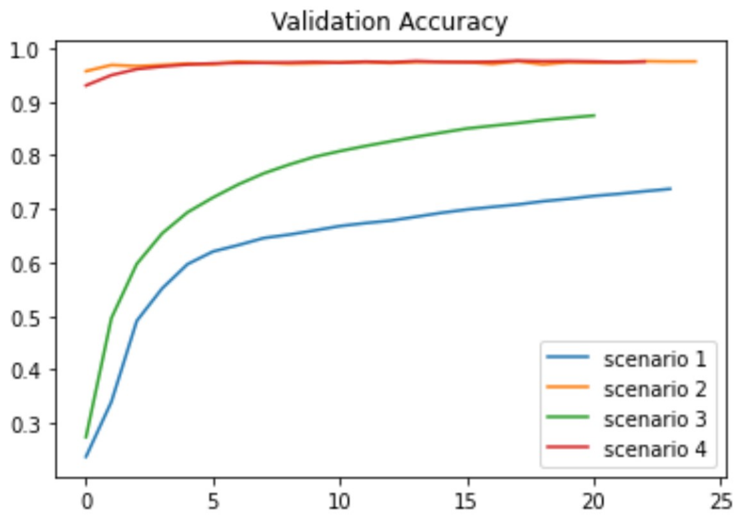
```
In [28]: plt.title("Accuracy")
         plt.plot(history1.history['accuracy'])
         plt.plot(history2.history['accuracy'])
         plt.plot(history3.history['accuracy'])
         plt.plot(history4.history['accuracy'])
         plt.legend(['scenario 1','scenario 2','scenario 3','scenario 4'])

Out[28]: <matplotlib.legend.Legend at 0x1b9b3b4bf40>
```



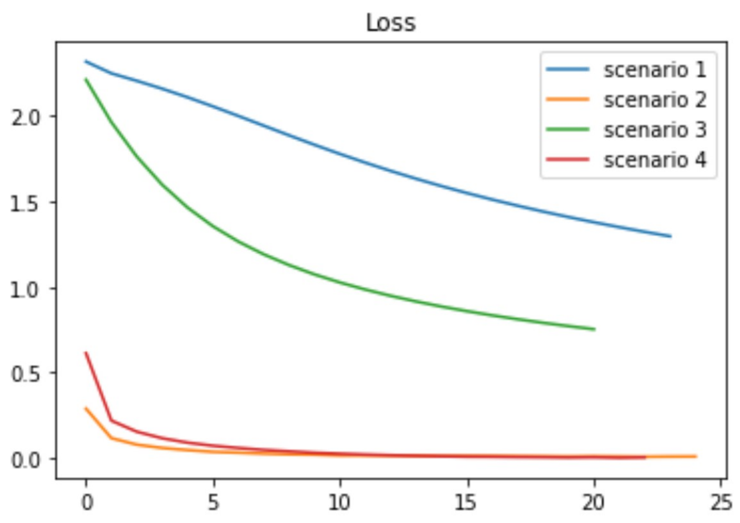
```
In [29]: plt.title("Validation Accuracy")
         plt.plot(history1.history['val_accuracy'])
         plt.plot(history2.history['val_accuracy'])
         plt.plot(history3.history['val_accuracy'])
         plt.plot(history4.history['val_accuracy'])
         plt.legend(['scenario 1','scenario 2','scenario 3','scenario 4'])

Out[29]: <matplotlib.legend.Legend at 0x1b9a9b59f10>
```



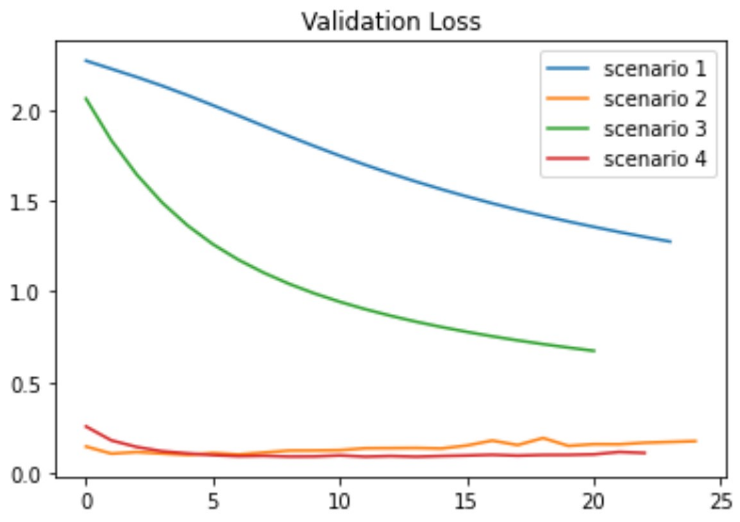
```
In [30]: plt.title("Loss")
         plt.plot(history1.history['loss'])
         plt.plot(history2.history['loss'])
         plt.plot(history3.history['loss'])
         plt.plot(history4.history['loss'])
         plt.legend(['scenario 1','scenario 2','scenario 3','scenario 4'])

Out[30]: <matplotlib.legend.Legend at 0x1b9a7d0b160>
```

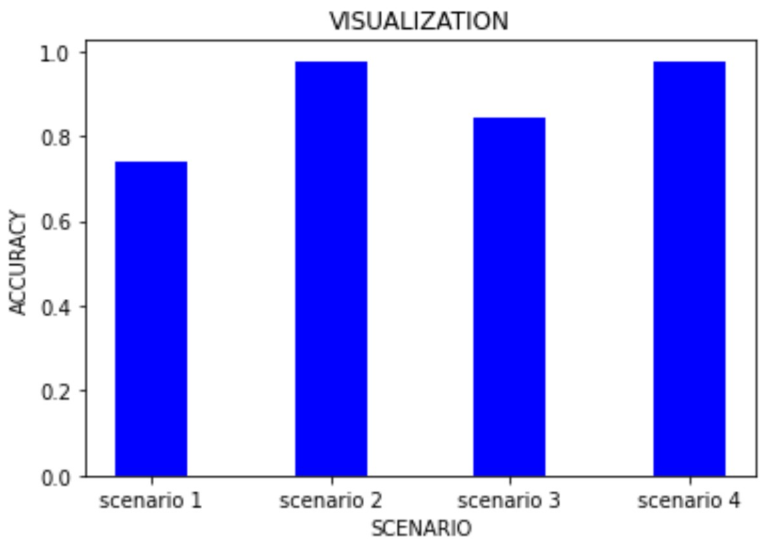


```
In [31]: plt.title("Validation Loss")
         plt.plot(history1.history['val_loss'])
         plt.plot(history2.history['val_loss'])
         plt.plot(history3.history['val_loss'])
         plt.plot(history4.history['val_loss'])
         plt.legend(['scenario 1','scenario 2','scenario 3','scenario 4'])

Out[31]: <matplotlib.legend.Legend at 0x1b9b1bf4040>
```



In [32]:
```
plt.bar(s,a,color='blue',width = 0.4)
plt.xlabel("SCENARIO")
plt.ylabel("ACCURACY")
plt.title("VISUALIZATION")
plt.show()
```


VISUALIZATION
ACCURACY
scenario 1
scenario 2
scenario 3
scenario 4
SCENARIO