```
import numpy as np
import pandas as pd
import random
import tensorflow as tf
import matplotlib.pyplot as plt


from tensorflow.keras.models import Sequential


from tensorflow.keras.layers import Flatten,Conv2D,Dense,MaxPooling2D


from tensorflow.keras.optimizers import SGD
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.datasets import mnist


(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

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

```
print(X_train.shape)
(60000, 28, 28)
```

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

```
X_train[0].min(), X_train[0].max()
(0, 255)
```

```
     (0, 255)
```

```
X_train = (X_train - 0.0) / (255.0 - 0.0)
X_test = (X_test - 0.0) / (255.0 - 0.0)
X_train[0].min(), X_train[0].max()
(0.0, 1.0)
```

```
     (0.0, 1.0)
```

```
def plot_digit(image, digit, plt, i):
  plt.subplot(4, 5, i + 1)
  plt.imshow(image, cmap=plt.get_cmap('gray'))
  plt.title(f"Digit: {digit}")
  plt.xticks([])
  plt.yticks([])
  plt.figure(figsize=(16, 10))


for i in range(20):
  plot_digit(X_train[i], y_train[i], plt, i)
  plt.show()
```

Digit: 5

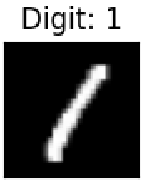
```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 0

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 4

```
<Figure size 1600x1000 with 0 Axes>
```


Digit: 1

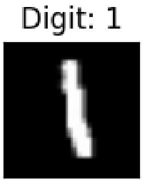
```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 9

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 2

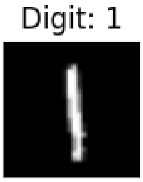
```
<Figure size 1600x1000 with 0 Axes>
```


Digit: 1

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 3

```
<Figure size 1600x1000 with 0 Axes>
```


Digit: 1

```
<Figure size 1600x1000 with 0 Axes>
```

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 4

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 3

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 5

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 3

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 6

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 1

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 7

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 2

```
<Figure size 1600x1000 with 0 Axes>
```

Digit: 8

```
<Figure size 1600x1000 with 0 Axes>
```

```
<Figure size 1000x1000 with 0 Axes>
```

```
X_train = X_train.reshape((X_train.shape + (1,)))
X_test = X_test.reshape((X_test.shape + (1,)))
```

```
import numpy as np

y_train = np.array([5, 0, 4, 1, 9, 2, 1, 3, 1, 4, 3, 5, 3, 6, 1, 7, 2, 8, 6, 9], dtype=np.uint8)
y_train[0:20]
```

```
array([5, 0, 4, 1, 9, 2, 1, 3, 1, 4, 3, 5, 3, 6, 1, 7, 2, 8, 6, 9],
      dtype=uint8)
```

```
model = Sequential([
Conv2D(32, (3, 3), activation="relu", input_shape=(28, 28, 1)),
MaxPooling2D((2, 2)),
Flatten(),
Dense(100, activation="relu"),
Dense(10, activation="softmax")
])
```

```
from tensorflow.keras.optimizers import SGD
optimizer = SGD(learning_rate=0.01, momentum=0.9)
model.compile(optimizer=optimizer,
loss="sparse_categorical_crossentropy",metrics=["accuracy"])
model.summary()
```

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

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

 flatten (Flatten)           (None, 5408)              0

 dense (Dense)               (None, 100)               540900

 dense_1 (Dense)             (None, 10)                1010

=================================================================
Total params: 542,230
Trainable params: 542,230
Non-trainable params: 0
_________________________________________________________________
```

```
(x_train,y_train),(x_test,y_test) = mnist.load_data()
print(len(x_train), len(y_train))
```

```
60000 60000
```

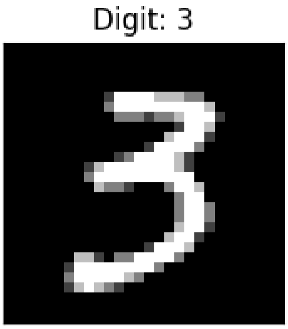
```
model.fit(X_train, y_train, epochs=10, batch_size=32)
```

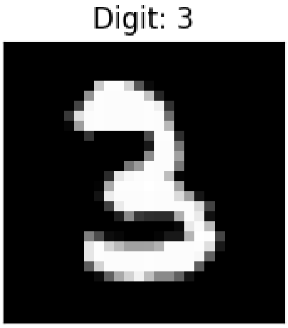
```
Epoch 1/10
1875/1875 [==============================] - 39s 20ms/step - loss: 0.2367 - accuracy: 0.9280
Epoch 2/10
```

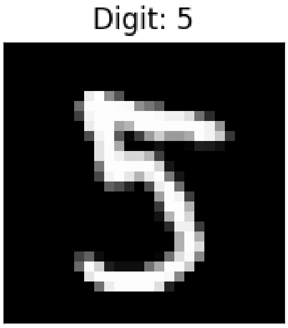
```
    1875/1875 [==============================] - 38s 20ms/step - loss: 0.0735 - accuracy: 0.9778
    Epoch 3/10
    1875/1875 [==============================] - 38s 20ms/step - loss: 0.0475 - accuracy: 0.9855
    Epoch 4/10
    1875/1875 [==============================] - 39s 21ms/step - loss: 0.0347 - accuracy: 0.9894
    Epoch 5/10
    1875/1875 [==============================] - 39s 21ms/step - loss: 0.0266 - accuracy: 0.9915
    Epoch 6/10
    1875/1875 [==============================] - 40s 21ms/step - loss: 0.0204 - accuracy: 0.9934
    Epoch 7/10
    1875/1875 [==============================] - 38s 20ms/step - loss: 0.0145 - accuracy: 0.9954
    Epoch 8/10
    1875/1875 [==============================] - 38s 20ms/step - loss: 0.0114 - accuracy: 0.9967
    Epoch 9/10
    1875/1875 [==============================] - 38s 20ms/step - loss: 0.0080 - accuracy: 0.9980
    Epoch 10/10
    1875/1875 [==============================] - 37s 20ms/step - loss: 0.0052 - accuracy: 0.9986
    <keras.callbacks.History at 0x79b831dbdd80>
```

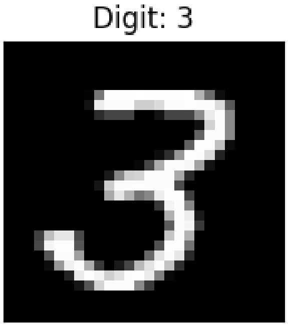
```
plt.figure(figsize=(16, 10))
for i in range(20):
 image = random.choice(X_test).squeeze()
 digit = np.argmax(model.predict(image.reshape((1, 28, 28, 1)))[0],
 axis=-1)
 plot_digit(image, digit, plt, i)
plt.show()
```

```
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 30ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 24ms/step
<ipython-input-14-d3a4bebdd4f8>:7: RuntimeWarning: More than 20 figures have been opened. Figu
  plt.figure(figsize=(16, 10))
```

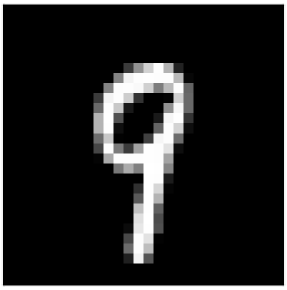




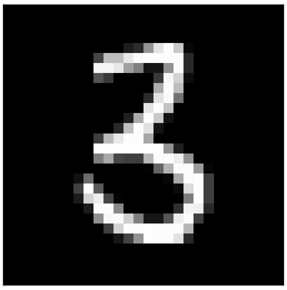




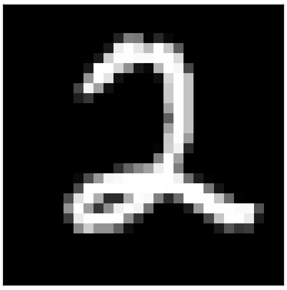
Digit: 9



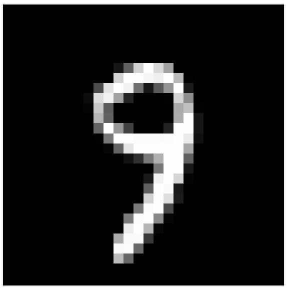
Digit: 3



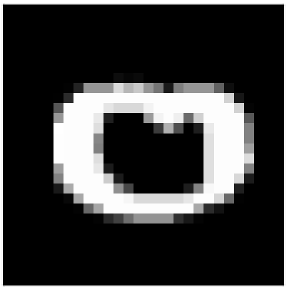
Digit: 2



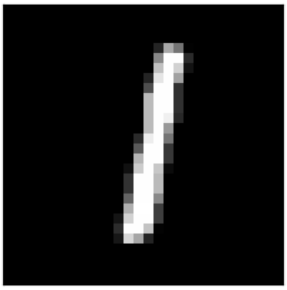
Digit: 9



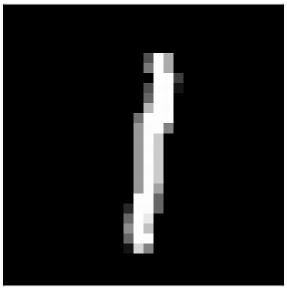
Digit: 0



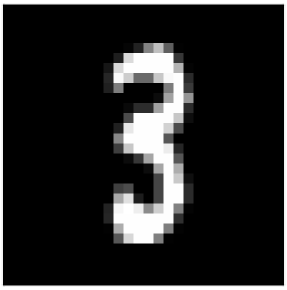
Digit: 1



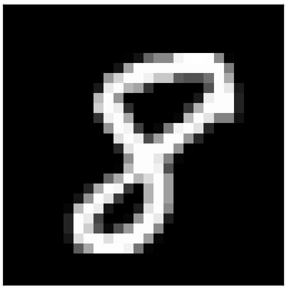
Digit: 1



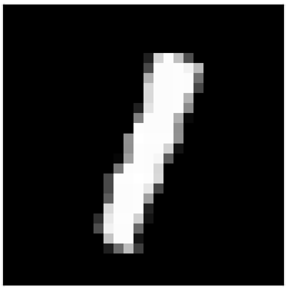
Digit: 3



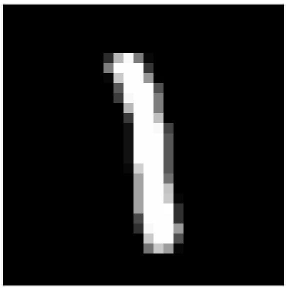
Digit: 8



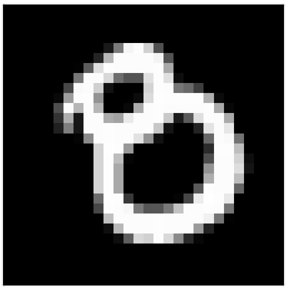
Digit: 1



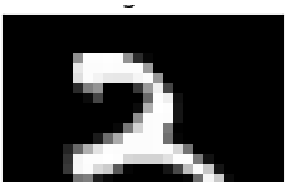
Digit: 1



Digit: 0



Digit: 3



```
predictions = np.argmax(model.predict(X_test), axis=-1)
accuracy_score(y_test, predictions)
```

```
313/313 [==============================] - 5s 16ms/step
0.9869
```

```
score = model.evaluate(X_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

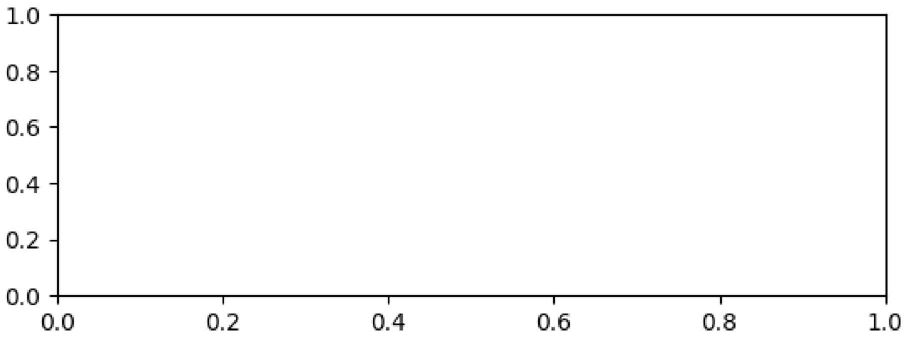
```
Test loss: 0.04557980224490166
Test accuracy: 0.9868999719619751
```

```
import os
import matplotlib.pyplot as plt
fig = plt.figure()
plt.subplot(2,1,1)
plt.plot(model_log.history['acc'])
plt.plot(model_log.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='lower right')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-83-a44c34ca7a33> in <cell line: 5>()
      3 fig = plt.figure()
      4 plt.subplot(2,1,1)
----> 5 plt.plot(model_log.history['acc'])
      6 plt.plot(model_log.history['val_acc'])
      7 plt.title('model accuracy')

AttributeError: 'dict' object has no attribute 'history'
```

SEARCH STACK OVERFLOW

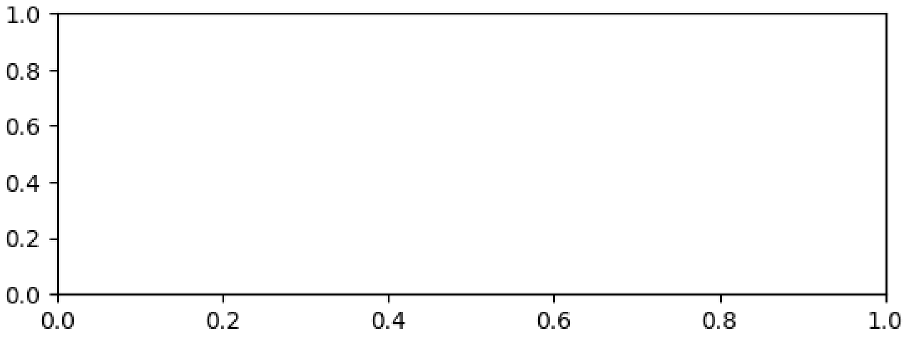


```
plt.subplot(2,1,2)
plt.plot(model_log.history['loss'])
plt.plot(model_log.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper right')
plt.tight_layout()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-81-fe1e2d4f04b1> in <cell line: 2>()
      1 plt.subplot(2,1,2)
----> 2 plt.plot(model_log.history['loss'])
      3 plt.plot(model_log.history['val_loss'])
      4 plt.title('model loss')
      5 plt.ylabel('loss')

AttributeError: 'dict' object has no attribute 'history'
```

SEARCH STACK OVERFLOW



```
model_digit_json = model.to_json()
with open("model_digit.json", "w") as json_file:
  json_file.write(model_digit_json)
# serialize weights to HDF5
model.save_weights("model_digit.h5")
print("Saved model to disk")
```

```
Saved model to disk
```