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

(x_train, y_train), (x_test, y_test) = mnist.load_data()

plt.matshow(x_train[5])

plt.imshow(x_train[4], cmap="gray")

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()

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")
])

optimizer = SGD(learning_rate=0.01, momentum=0.9)
model.compile(
   optimizer=optimizer,
   loss="sparse_categorical_crossentropy",
   metrics=["accuracy"]
)
model.summary()

history=model.fit(x_train,
y_train,validation_data=(x_test,y_test),epochs=3)

n=random.randint(0,9999)
plt.imshow(x_test[n])
plt.show()

x_train
x_test
y_train

predicted_value=model.predict(x_test)
plt.imshow(x_test[n])
```

```
plt.show()
print(predicted_value[n])

score = model.evaluate(x_test,y_test)
print("loss =%.3f " %score[0])
print("accuracy =%.3f" %score[1])

history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()


history.history.keys()
# dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```