#importing necessary libraries

import tensorflow as tf

from tensorflow import keras

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import random

%matplotlib inline

#import dataset and split into train and test data

mnist = tf.keras.datasets.mnist

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

plt.matshow(x\_train[1])

plt.imshow(-x\_train[0], cmap="gray")

x\_train = x\_train / 255

x\_test = x\_test / 255

model = keras.Sequential([

keras.layers.Flatten(input\_shape=(28, 28)),

keras.layers.Dense(128, activation="relu"),

keras.layers.Dense(10, activation="softmax")

])

model.summary()

model.compile(optimizer="sgd",

loss="sparse\_categorical\_crossentropy",

metrics=['accuracy'])

history=model.fit(x\_train,

y\_train,validation\_data=(x\_test,y\_test),epochs=10)

test\_loss,test\_acc=model.evaluate(x\_test,y\_test)

print("Loss=%.3f" %test\_loss)

print("Accuracy=%.3f" %test\_acc)

n=random.randint(0,9999)

plt.imshow(x\_test[n])

plt.show()

x\_train

x\_test

predicted\_value=model.predict(x\_test)

plt.imshow(x\_test[n])

plt.show()

# history.history()

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()

print(predicted\_value[n])

# history.history()

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()