**import numpy as np //1MulticlassOCR**

from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense, Dropout from tensorflow.keras.optimizers import RMSprop from tensorflow.keras.datasets import mnist

import matplotlib.pyplot as plt from sklearn import metrics

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

plt.imshow(x\_train[0], cmap='gray') plt.show()

print(x\_train[0] )

print("X\_train shape", x\_train.shape) print("y\_train shape", y\_train.shape) print("X\_test shape", x\_test.shape) print("y\_test shape", y\_test.shape)

x\_train = x\_train.reshape(60000, 784) x\_test = x\_test.reshape(10000, 784) x\_train = x\_train.astype('float32') x\_test = x\_test.astype('float32') x\_train /= 255

x\_test /= 255

num\_classes = 10

y\_train = np.eye(num\_classes)[y\_train] y\_test = np.eye(num\_classes)[y\_test]

model = Sequential()

model.add(Dense(512, activation='relu', input\_shape=(784,))) model.add(Dropout(0.2)) # DROP OUT RATIO 20%

model.add(Dense(512, activation='relu')) #returns a sequence of vectors of dimen model.add(Dropout(0.2))

model.add(Dense(num\_classes, activation='softmax'))

model.compile(loss='categorical\_crossentropy', # for a multi-class classificati optimizer=RMSprop(),

validation\_data=(x\_test, y\_test))

score = model.evaluate(x\_test, y\_test, verbose=0) print('Test loss:', score[0])

print('Test accuracy:', score[1])

metrics=['accuracy'])

model.compile( optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'] )

batch\_size = 128 # batch\_size argument is passed to the layer to define a batch epochs = 20

history = model.fit(x\_train, y\_train,

batch\_size=batch\_size, epochs=epochs,

verbose=1, # verbose=1 will show you an animated progress ba