Ex No: 4 HANDWRITTEN DIGITS RECOGNITION WITH MNIST

Aim:

To build a handwritten digit's recognition with MNIST dataset.

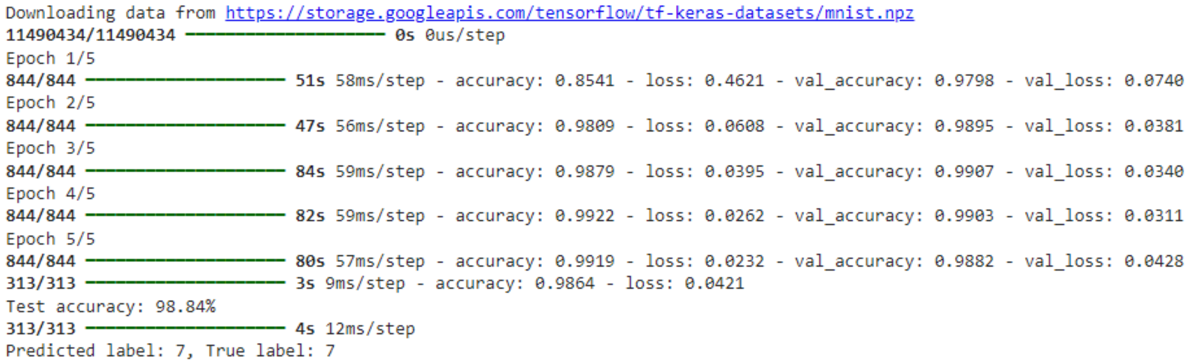
Procedure:

1. Download and load the MNIST dataset.
2. Perform analysis and preprocessing of the dataset.
3. Build a simple neural network model using Keras/TensorFlow.
4. Compile and fit the model.
5. Perform prediction with the test dataset.
6. Calculate performance metrics.

Program:

```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical
(X_train, y_train), (X_test, y_test) = mnist.load_data()
X_train = X_train.reshape((X_train.shape[0], 28, 28, 1))
X_test = X_test.reshape((X_test.shape[0], 28, 28, 1))
X_train, X_test = X_train / 255.0, X_test / 255.0
y_train = to_categorical(y_train)
y_test = to_categorical(y_test)
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(X_train, y_train, epochs=5, batch_size=64, validation_split=0.1)
test_loss, test_acc = model.evaluate(X_test, y_test)
print(f"Test accuracy: {test_acc * 100:.2f}%")
predictions = model.predict(X_test)
print(f"Predicted label: {predictions[0].argmax()}, True label: {y_test[0].argmax()}")
```

Output:



```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
Epoch 1/5
844/844 ━━━━━━━━━━━━━━━━━━━━ 51s 58ms/step - accuracy: 0.8541 - loss: 0.4621 - val_accuracy: 0.9798 - val_loss: 0.0740
Epoch 2/5
844/844 ━━━━━━━━━━━━━━━━━━━━ 47s 56ms/step - accuracy: 0.9809 - loss: 0.0608 - val_accuracy: 0.9895 - val_loss: 0.0381
Epoch 3/5
844/844 ━━━━━━━━━━━━━━━━━━━━ 84s 59ms/step - accuracy: 0.9879 - loss: 0.0395 - val_accuracy: 0.9907 - val_loss: 0.0340
Epoch 4/5
844/844 ━━━━━━━━━━━━━━━━━━━━ 82s 59ms/step - accuracy: 0.9922 - loss: 0.0262 - val_accuracy: 0.9903 - val_loss: 0.0311
Epoch 5/5
844/844 ━━━━━━━━━━━━━━━━━━━━ 80s 57ms/step - accuracy: 0.9919 - loss: 0.0232 - val_accuracy: 0.9882 - val_loss: 0.0428
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step - accuracy: 0.9864 - loss: 0.0421
Test accuracy: 98.84%
313/313 ━━━━━━━━━━━━━━━━━━━━ 4s 12ms/step
Predicted label: 7, True label: 7
```

Result:

Thus the program for building a simple convolutional neural network for MNIST dataset was executed successfully.