Ex: 2 a

BUILD A SIMPLE NEURAL NETWORKS

Aim:

To build a simple neural network using Keras/TensorFlow.

Algorithm:

1. Download and load the 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:

```
from tensorflow import keras
from tensorflow.keras.datasets import mnist
import numpy as np
# load dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()
# flatten the data
x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
# one-hot encode the labels
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
# build the model
model = keras.Sequential([
    keras.layers.Dense(128, activation="relu", input_shape = (784, )),
    keras.layers.Dense(10, activation="softmax")
])
```

```
# compile the model
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
# train the model
model.fit(x_train, y_train, epochs=5)
# evaluate the model
model.evaluate(x_test, y_test)
predictions = model.predict(x_test)
# other metrics
from sklearn.metrics import confusion_matrix, classification_report
# get predicted class labels(argmax)
predicted_classes = np.argmax(predictions, axis=1)
# calculate confusion matrix
cm = confusion_matrix(y_test.argmax(axis=1), predicted_classes)
print("Confusion Matrix: \n",cm)
```

Output:

```
Epoch 1/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 7s 3ms/step - accuracy: 0.8096 - loss: 6.8032
Epoch 2/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9061 - loss: 0.3990
Epoch 3/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 10s 3ms/step - accuracy: 0.9264 - loss: 0.2866
Epoch 4/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 4s 2ms/step - accuracy: 0.9376 - loss: 0.2351
Epoch 5/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 5s 3ms/step - accuracy: 0.9442 - loss: 0.2221
```

```
# evaluate the model
model.evaluate(x_test, y_test)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9293 - loss: 0.2863
[0.246164470911026, 0.9415000081062317]
```

```
predictions = model.predict(x_test)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```

```
# calculate confusion matrix
cm = confusion_matrix(y_test.argmax(axis=1), predicted_classes)
print("Confusion Matrix: \n",cm)
```

```
Confusion Matrix:
 [[ 938    0    6    2    1    1    7    2   23    0]
 [   0 1111    2    2    1    2    2    1   14    0]
 [   2    4  957   22    3    1    0   10   33    0]
 [   0    0    8  972    0    6    0    3   17    4]
 [   0    0    1    0  929    0    9    0    9   34]
 [   3    0    0   32    2  771   28    4   43    9]
 [   2    3    4    0    4   21  908    1   15    0]
 [   1    4   21   18    4    0    0  952    7   21]
 [   1    1    3   13    4    2    1    1  945    3]
 [   4    5    1   15   22    4    0    2   24  932]]
```

Result:

Thus the program to build a simple neural network using Keras/TensorFlow has been executed successfully.