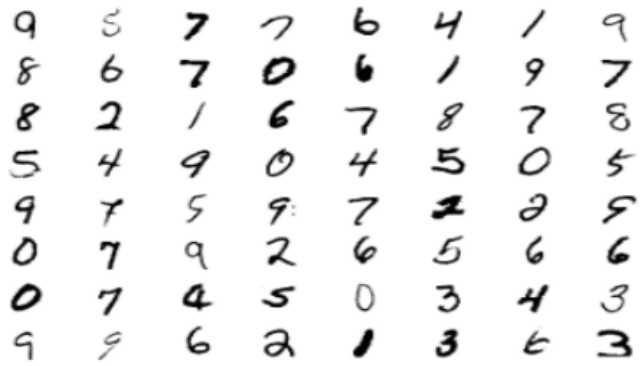
EXPERIMENT-5

AIM: Write a program to implement a Deep Neural Network and verify the performance on MNIST dataset.

CODE and OUTPUT:

```
import keras
from keras.datasets import mnist
from keras.layers import Dense
from keras.models import Sequential
from matplotlib import pyplot as plt
from random import randint

# Preparing the dataset
# Setup train and test splits
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Making a copy before flattening for the next code-segment which displays images
x_train_drawing = x_train

image_size = 784 # 28 x 28
x_train = x_train.reshape(x_train.shape[0], image_size)
x_test = x_test.reshape(x_test.shape[0], image_size)

# Convert class vectors to binary class matrices
num_classes = 10
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
```

```
for i in range(64):
    ax = plt.subplot(8, 8, i+1)
    ax.axis('off')
    plt.imshow(x_train_drawing[randint(0, x_train.shape[0])], cmap='Greys')
```



```
    model = Sequential()
    model.add(Dense(128, activation='sigmoid', input_shape=(image_size,)))
    model.add(Dense(128, activation='sigmoid', input_shape=(image_size,)))
    model.add(Dense(128, activation='sigmoid', input_shape=(image_size,)))
    model.add(Dense(128, activation='sigmoid', input_shape=(image_size,)))
    model.add(Dense(10, activation='sigmoid', input_shape=(image_size,)))
    model.summary()
```

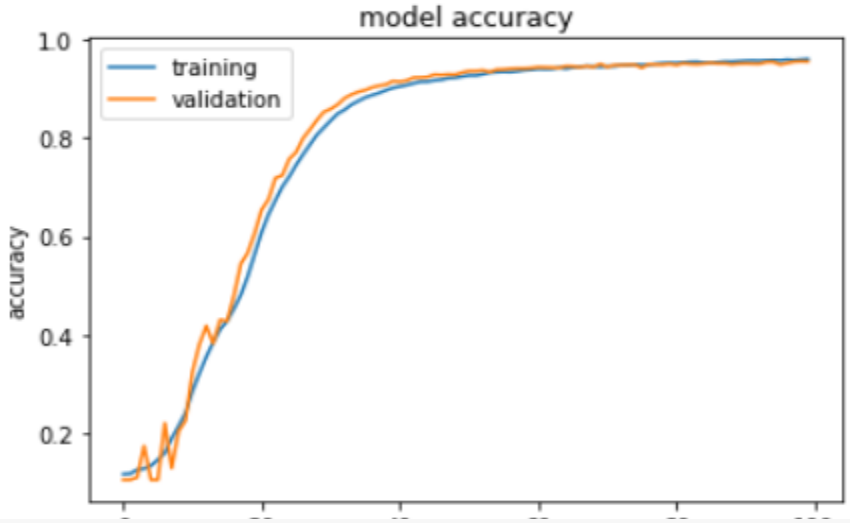
```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_5 (Dense)              (None, 128)               100480
_________________________________________________________________
dense_6 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_7 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_8 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_9 (Dense)              (None, 10)                1290
=================================================================
Total params: 151,306
Trainable params: 151,306
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(x_train, y_train, batch_size=128, epochs=100, validation_split=.1, verbose=True)
```

```
loss,accuracy  = model.evaluate(x_test, y_test, verbose=True)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.1932 - accuracy: 0.9469
```

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['training', 'validation'], loc='best')
plt.show()
```




```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('Loss')
plt.xlabel('epoch')
plt.legend(['training', 'validation'], loc='best')
plt.show()
```