Ex No: 7 **BUILD AUTOENCODERS**

AIM:

To build autoencoders with Keras/TensorFlow.

PROCEDURE:

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.

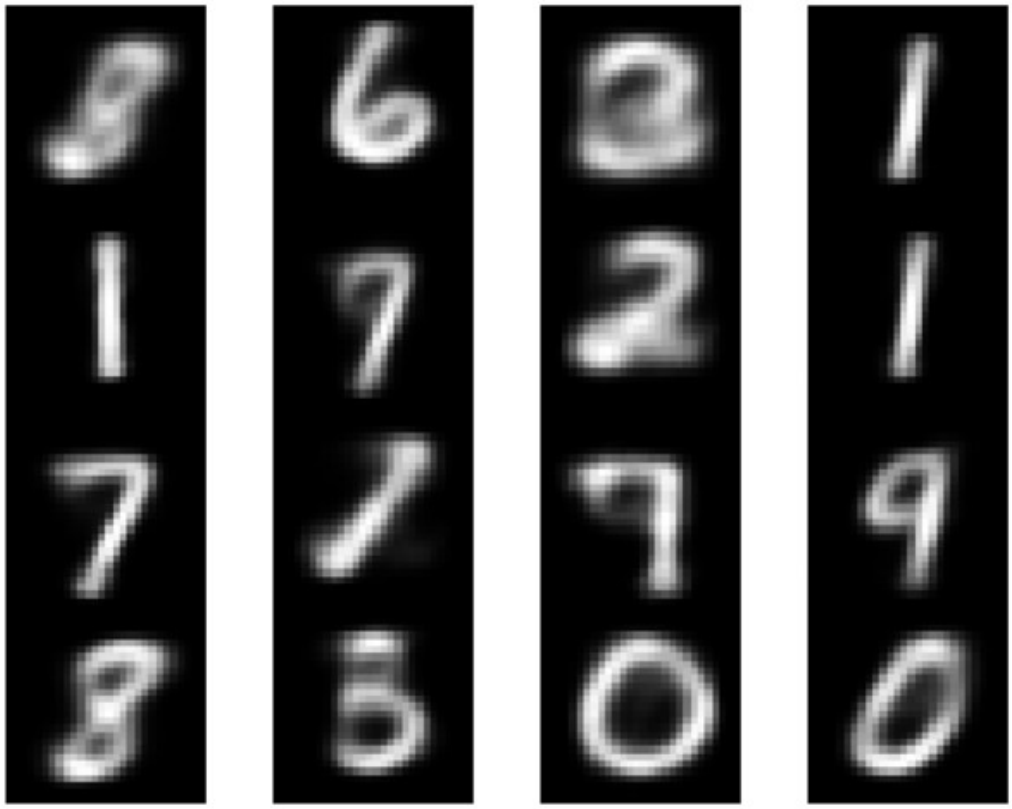
PROGRAM:

```
#import tensorflow.keras as keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.layers import Dense, Input, Flatten,Reshape, LeakyReLU as LR,Activation, Dropout
from tensorflow.keras.models import Model, Sequential
from matplotlib import pyplot as plt
from IPython import display # If using IPython, Colab or Jupyter
import numpy as np

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train = x_train/255.0
x_test = x_test/255.0

# Plot image data from x_train
plt.imshow(x_train[0], cmap = "gray")
plt.show()

LATENT_SIZE = 32
```

```
encoder = Sequential([
        Flatten(input_shape = (28, 28)),
        Dense(512),
        LR(),
        Dropout(0.5),
        Dense(256),
        LR(),
        Dropout(0.5),
        Dense(128),
        LR(),
        Dropout(0.5),
        Dense(64),
        LR(),
        Dropout(0.5),
        Dense(LATENT_SIZE),
        LR()
])

decoder = Sequential([
        Dense(64, input_shape = (LATENT_SIZE,)),
        LR(),
        Dropout(0.5),
        Dense(128),
        LR(),
        Dropout(0.5),
        Dense(256),
        LR(),
        Dropout(0.5),
        Dense(512),
        LR(),
        Dropout(0.5),
        Dense(784),
```

```
        Activation("sigmoid"),
        Reshape((28, 28))
])

img = Input(shape = (28, 28))
latent_vector = encoder(img)
output = decoder(latent_vector)
model = Model(inputs = img, outputs = output)
model.compile("nadam", loss = "binary_crossentropy")

EPOCHS=15
for epoch in range(EPOCHS):
    fig, axs = plt.subplots(4, 4)
    rand = x_test[np.random.randint(0, 10000, 16)].reshape((4, 4, 1, 28, 28))
    display.clear_output() # If you imported display from IPython
    for i in range(4):
      for j in range(4):
        axs[i, j].imshow(model.predict(rand[i, j])[0], cmap = "gray")
        axs[i, j].axis("off")
    plt.subplots_adjust(wspace = 0, hspace = 0)
    plt.show()
    print("-----------", "EPOCH", epoch, "-----------")
    model.fit(x_train, x_train)
```

OUTPUT:



```
----------- EPOCH 14 -----------
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 43s 23ms/step - loss: 0.1906
```

RESULT:

Thus autoencoders with Keras/TensorFlow are built.