PAREENITA A.SHIRSATH B.E.A.I.&.D.S. ROLL.NO : 57

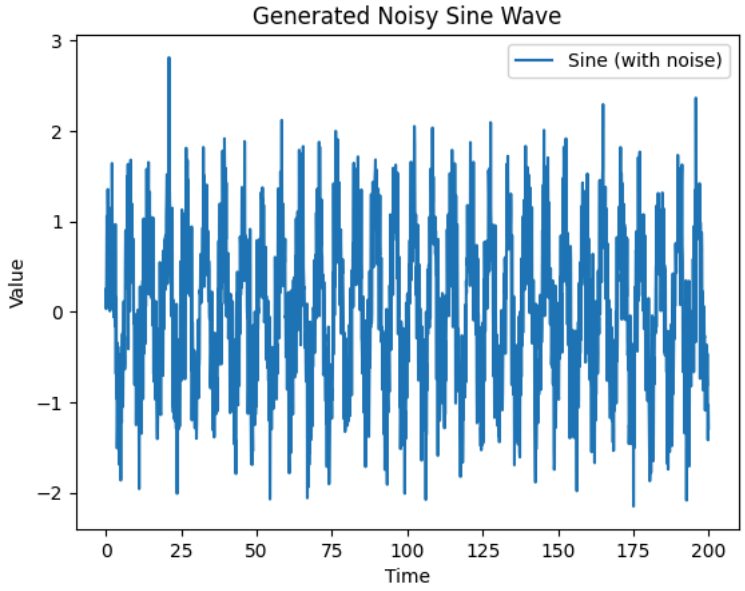
DL EXPERIMENT NO : 10

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Set random seeds for reproducibility
RANDOM_SEED = 42
np.random.seed(RANDOM_SEED)
tf.random.set_seed(RANDOM_SEED)
```

```
# Generate time series data
data_time = np.arange(0, 200, 0.1)
sin_values = np.sin(data_time) + np.random.normal(scale=0.5, size=len(data_time))

# Plot the data
plt.plot(data_time, sin_values, label='Sine (with noise)')
plt.xlabel("Time")
plt.ylabel("Value")
plt.title("Generated Noisy Sine Wave")
plt.legend()
plt.show()
```




```
# Create DataFrame
data_full = pd.DataFrame({"sine": sin_values}, index=data_time)

# Split into train and test sets
len_train = int(len(data_full) * 0.8)
train, test = data_full.iloc[:len_train], data_full.iloc[len_train:]

# Function to create sequences
def gen_data(X, y, num_steps=10):
    Xs, ys = [], []
    for i in range(len(X) - num_steps):
        Xs.append(X.iloc[i:(i + num_steps)].values)
        ys.append(y.iloc[i + num_steps])
    return np.array(Xs), np.array(ys)

# Generate sequences
num_steps = 10
trainX, trainY = gen_data(train, train['sine'], num_steps)
testX, testY = gen_data(test, test['sine'], num_steps)
```

```
# Define the LSTM model
lstm_model = keras.Sequential([
    layers.LSTM(128, input_shape=(trainX.shape[1], trainX.shape[2])),
    layers.Dense(1)
])
```

```
# Compile the model
lstm_model.compile(optimizer=keras.optimizers.Adam(0.001), loss='mean_squared_error')
```

```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/rnn/rnn.py:199: UserWarning: Do not pass an `input_shape`/`input_dim`
  super().__init__(**kwargs)
```

```
# Set early stopping criteria
early_stopping = keras.callbacks.EarlyStopping(monitor="loss", patience=3)

# Train the model
history = lstm_model.fit(trainX, trainY,
                         epochs=30,
                         batch_size=16,
                         validation_split=0.1,
                         shuffle=False,
                         callbacks=[early_stopping])

# Plot training history
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.title("Model Loss Over Epochs")
plt.legend()
plt.show()
```

```
Epoch 1/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - loss: 0.4047 - val_loss: 0.3595
Epoch 2/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3088 - val_loss: 0.3566
Epoch 3/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3078 - val_loss: 0.3548
Epoch 4/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3072 - val_loss: 0.3532
Epoch 5/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 2s 15ms/step - loss: 0.3066 - val_loss: 0.3514
Epoch 6/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - loss: 0.3057 - val_loss: 0.3492
Epoch 7/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3046 - val_loss: 0.3469
Epoch 8/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3034 - val_loss: 0.3452
Epoch 9/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 2s 16ms/step - loss: 0.3027 - val_loss: 0.3441
Epoch 10/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - loss: 0.3021 - val_loss: 0.3434
Epoch 11/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.3016 - val_loss: 0.3427
Epoch 12/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.3011 - val_loss: 0.3421
Epoch 13/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.3007 - val_loss: 0.3415
Epoch 14/30
```

```
# Evaluate the model
test_loss = lstm_model.evaluate(testX, testY)
print("Test Loss:", test_loss)
```

```
Epoch 17/30
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - loss: 0.3060
Test Loss: 0.3043603301048279
```

```
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2994 - val_loss: 0.3396
Epoch 18/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2991 - val_loss: 0.3391
Epoch 19/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.2987 - val_loss: 0.3387
Epoch 20/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.2983 - val_loss: 0.3383
Epoch 21/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2979 - val_loss: 0.3379
Epoch 22/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2974 - val_loss: 0.3375
Epoch 23/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2968 - val_loss: 0.3372
Epoch 24/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 9ms/step - loss: 0.2961 - val_loss: 0.3372
Epoch 25/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2953 - val_loss: 0.3375
Epoch 26/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - loss: 0.2944 - val_loss: 0.3381
Epoch 27/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 2s 16ms/step - loss: 0.2936 - val_loss: 0.3389
Epoch 28/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - loss: 0.2929 - val_loss: 0.3398
Epoch 29/30
90/90 ━━━━━━━━━━━━━━━━━━━━ 1s 10ms/step - loss: 0.2923 - val_loss: 0.3409
```