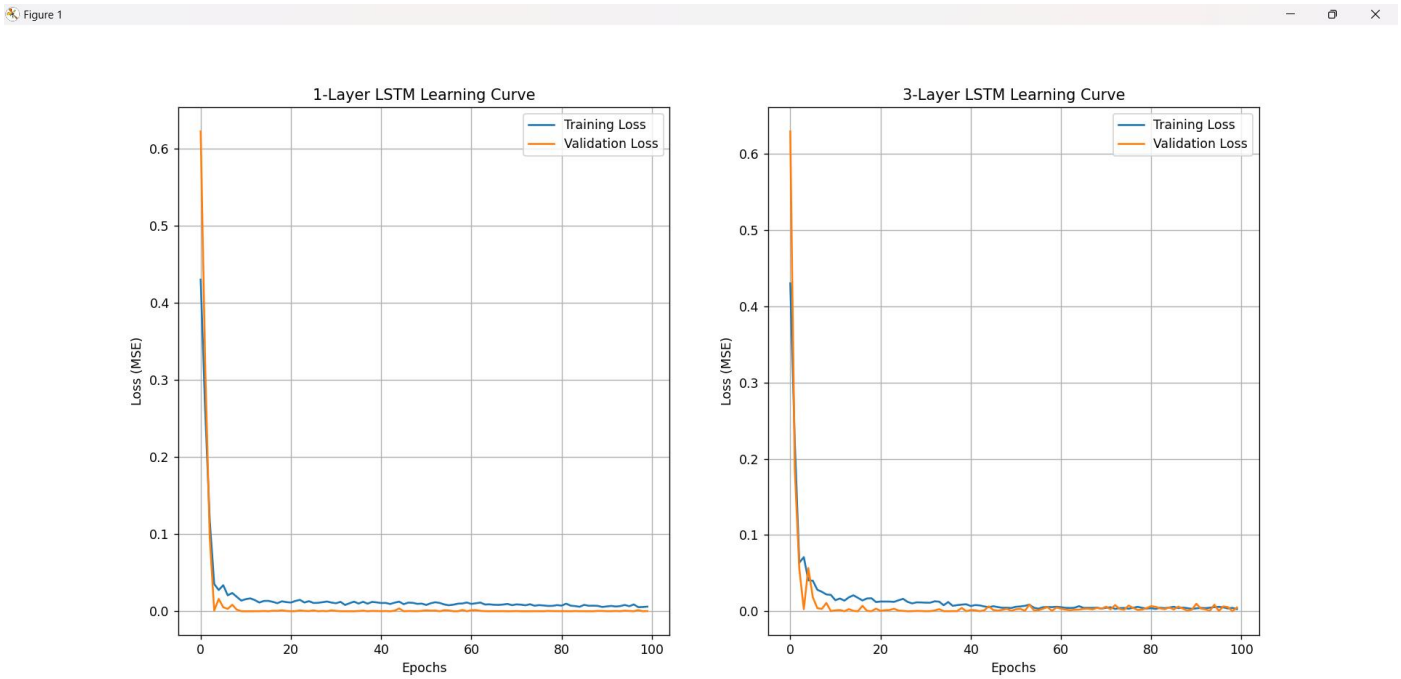
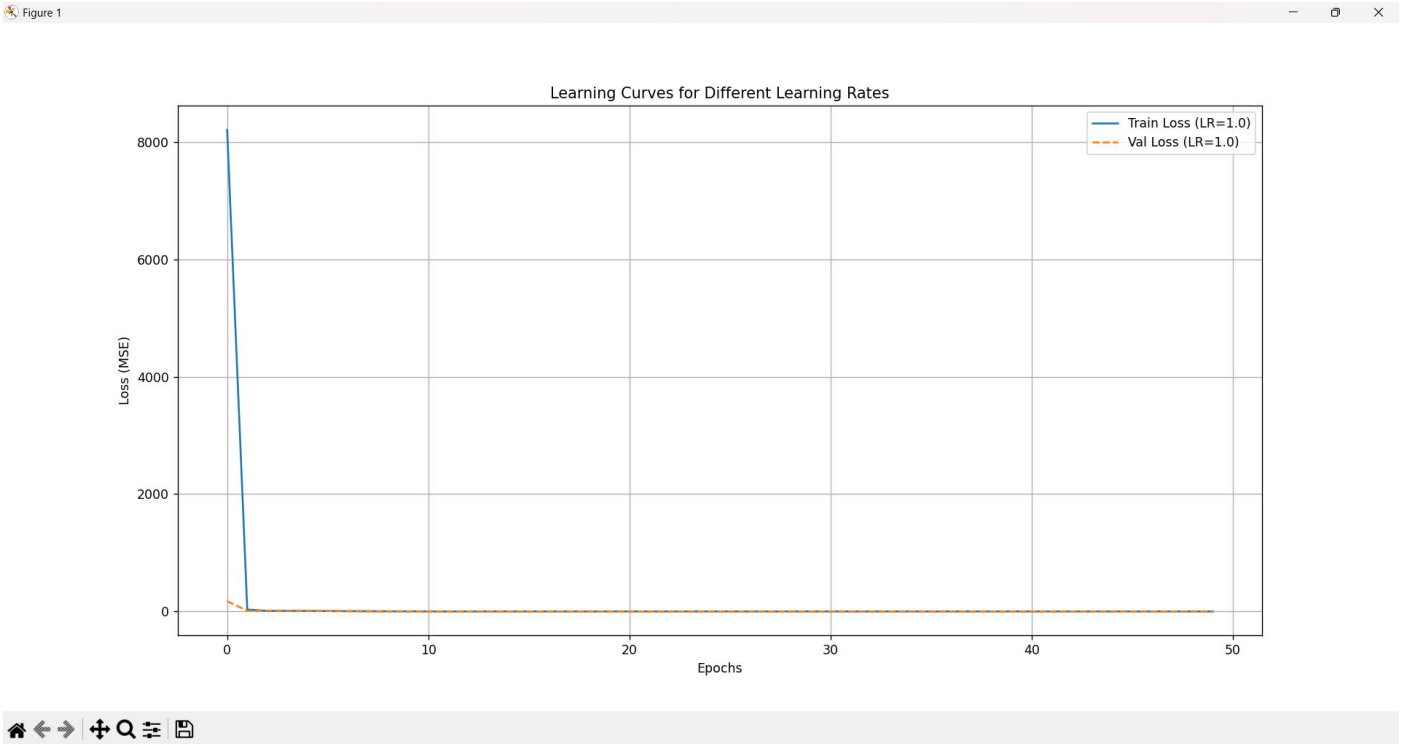
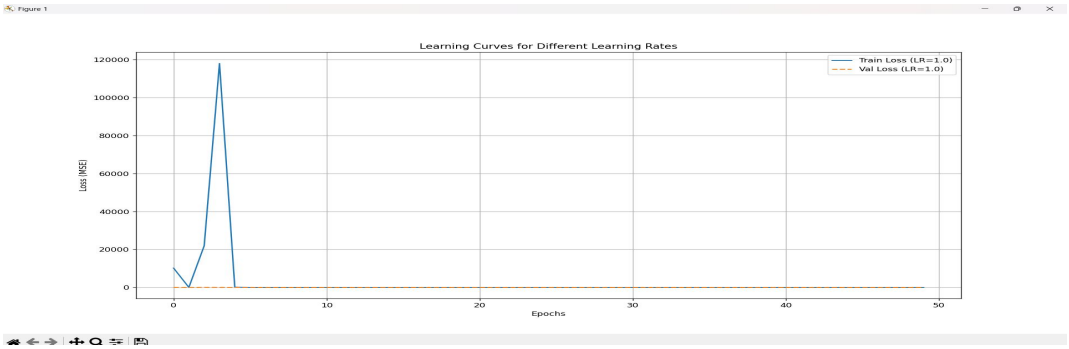
1 layer vs 3 layer LSTM



LR =1 , UNITS = 50 ; EPOCHS = 50 ;



LR = 0.1

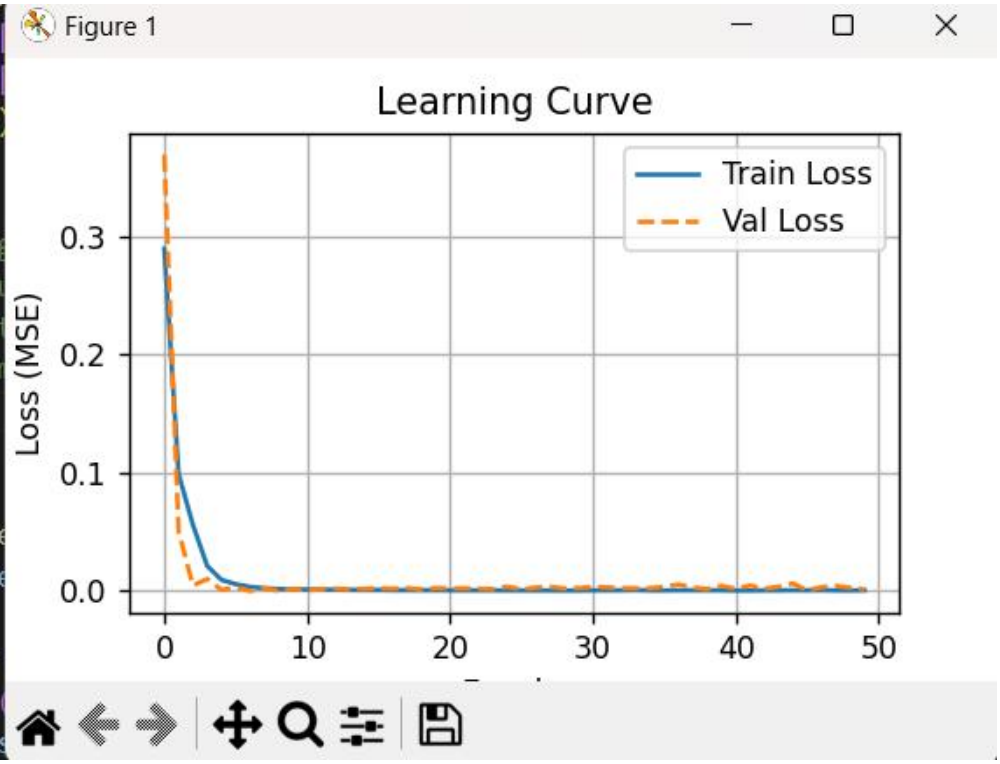


```
        X.append(data
        y.append(data
    return np.array(X)

# Hyperparameters
seq_length = 5  # Leng
lstm_units = 100  # Nu
batch_size = 8   # Bat
epochs = 50      # Nur
learning_rate = 0.01

# Prepare data
X, y = create_sequence
X = X.reshape((X.shape

# Split into training
split = int(0.8 * len(
X_train, X_test = X[:s
```

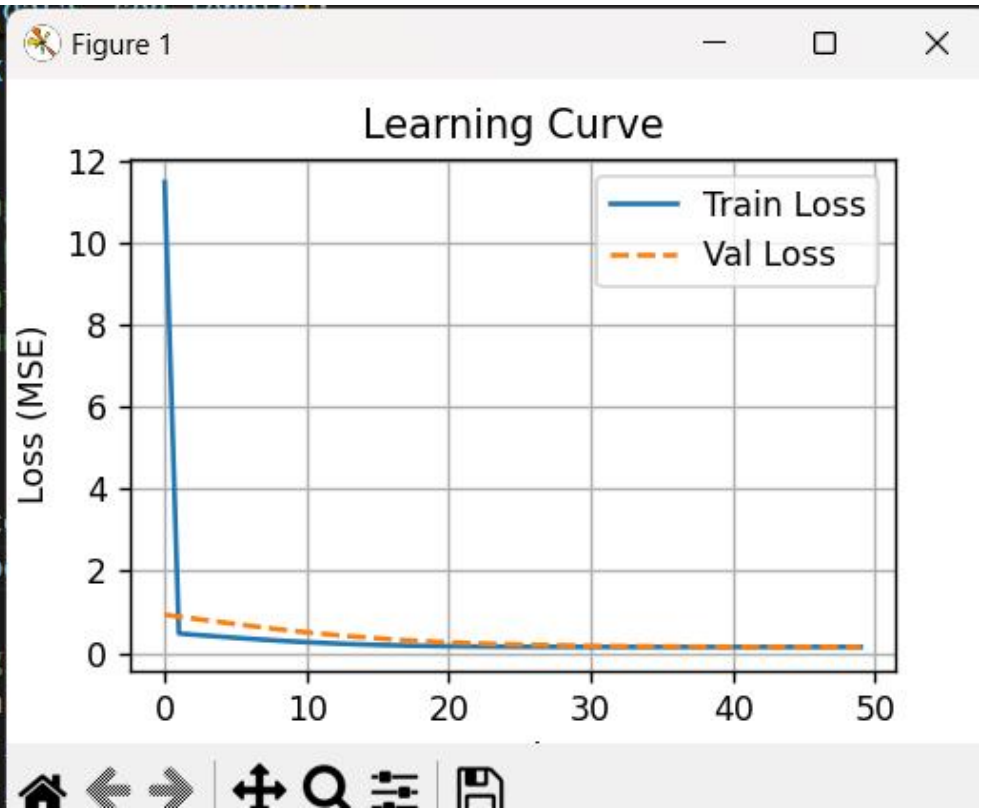



```
def create_sequences(
    return np.array(X

# Hyperparameters
seq_length = 5  # Len
lstm_units = 1000  #
batch_size = 8   # Ba
epochs = 50      # Nu
learning_rate = 0.01

# Prepare data
X, y = create_sequenc
X = X.reshape((X.shap

# Split into training
split = int(0.8 * len
X_train, X_test = X[:
y_train, y_test = y[:
```



```
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step - loss: 0.

🔹 Input Gate (i) Weights:
  - W_x: [-0.47492805]
  - W_h: [-0.671598]
  - Bias: [-0.23100398]

🔹 Forget Gate (f) Weights:
  - W_x: [0.53584105]
  - W_h: [-0.30135596]
  - Bias: [0.75140667]

🔹 Output Gate (o) Weights:
  - W_x: [-0.4802041]
  - W_h: [-0.17660163]
  - Bias: [0.28542095]

🔹 Cell State (g) Weights:
  - W_x: [0.23318675]
  - W_h: [-0.13485333]
  - Bias: [-0.18942383]
```