

```
In [1]: !pip install xgboost

In [9]: # Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dense, SimpleRNN, Dropout, Input
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLROnPlateau
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import seaborn as sns
from sklearn ensemble import RandomForestRegressor
from xgboost import XGBRegressor
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split

# Set the global font to SF Pro Text
matplotlib.rcParams['font.family'] = 'SF Pro Text'

In [2]: # Loading the Sales.csv data
df = pd.read_csv("dataset/Sales.csv")
df.head()

Out[2]:
   Order Date  Sales
0  2011-01-01  2425.12800
1  2011-01-02  942.44000
2  2011-01-03  13509.77400
3  2011-01-04  8426.74040
4  2011-01-05  10986.82000

In [3]: df["Sales"].plot(figsize=(12,5))
plt.title("Sales for each day")
plt.show()

Sales for each day

In [8]: # sales data is highly volatile.
# Using rolling mean smoothing of 24 day window to reduce the noise.
df["Sales"].rolling(window=24).mean().plot(figsize=(12,5))
plt.title("Sales for a rolling window of 24")
plt.show()

Sales for a rolling window of 24

We can see the noise has been reduced with 50 day moving averages of sales are used.

In [9]: df_smoothed = df.copy()

In [11]: df_smoothed["Sales"] = df["Sales"].rolling(window=50).mean()
df_smoothed.head(60)

Out[11]:
   Order Date  Sales
0  2011-01-01  NaN
1  2011-01-02  NaN
2  2011-01-03  NaN
3  2011-01-04  NaN
4  2011-01-05  NaN
5  2011-01-06  NaN
6  2011-01-07  NaN
7  2011-01-08  NaN
8  2011-01-09  NaN
9  2011-01-10  NaN
10 2011-01-11  NaN
11 2011-01-12  NaN
12 2011-01-13  NaN
13 2011-01-14  NaN
14 2011-01-15  NaN
15 2011-01-16  NaN
16 2011-01-17  NaN
17 2011-01-18  NaN
18 2011-01-19  NaN
19 2011-01-20  NaN
20 2011-01-21  NaN
21 2011-01-22  NaN
22 2011-01-23  NaN
23 2011-01-24  NaN
24 2011-01-25  NaN
25 2011-01-26  NaN
26 2011-01-27  NaN
27 2011-01-28  NaN
28 2011-01-29  NaN
29 2011-01-30  NaN
30 2011-01-31  NaN
31 2011-02-01  NaN
32 2011-02-02  NaN
33 2011-02-03  NaN
34 2011-02-04  NaN
35 2011-02-05  NaN
36 2011-02-06  NaN
37 2011-02-07  NaN
38 2011-02-08  NaN
39 2011-02-09  NaN
40 2011-02-10  NaN
41 2011-02-11  NaN
42 2011-02-12  NaN
43 2011-02-13  NaN
44 2011-02-14  NaN
45 2011-02-15  NaN
46 2011-02-16  NaN
47 2011-02-17  NaN
48 2011-02-18  NaN
49 2011-02-19  9729.298287
50 2011-02-21  9549.902487
51 2011-02-22  10089.859447
52 2011-02-23  10080.936999
53 2011-02-24  10264.480011
54 2011-02-25  10248.32896
55 2011-02-26  10467.313212
56 2011-02-27  10067.450872
57 2011-02-28  9634.913152
58 2011-03-01  10540.033828
59 2011-03-02  10381.281494

In [12]: # Drop NaN values (first 50 rows will be NaN)
df_smoothed = df_smoothed.dropna(1).reset_index(drop=True)
df_smoothed.head()

Out[12]:
   Order Date  Sales
0  2011-01-01  9729.298287
1  2011-02-21  9549.902487
2  2011-02-22  10089.859447
3  2011-02-23  10080.936999
4  2011-02-24  10264.480011

In [13]: df_smoothed.shape
Out[13]: (1381, 2)

The data has 1381 rows and 2 columns

In [14]: df_smoothed.info()
Out[14]:
>
RangeIndex: 1381 entries, 0 to 1380
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype
---  --
 0   Order Date  1381 non-null    object
 1   Sales       1381 non-null    float64
dtypes: float64(1), object(1)
memory usage: 21.7+ KB

It can be observed that the datatype of Order Date is String. We need to convert to Datetime format.

In [15]: # Converting the Order Date column datatype from string to datetime.
df_smoothed["Order Date"] = pd.to_datetime(df_smoothed["Order Date"], format='%Y-%m-%d')

In [16]: # Check last few dates
print(df_smoothed["Order Date"].tail(15))
1366 2014-12-17
1367 2014-12-18
1368 2014-12-19
1369 2014-12-20
1370 2014-12-21
1371 2014-12-22
1372 2014-12-23
1373 2014-12-24
1374 2014-12-25
1375 2014-12-26
1376 2014-12-27
1377 2014-12-28
1378 2014-12-29
1379 2014-12-30
1380 2014-12-31
Name: Order Date, dtype: datetime64[ns]

In [17]: df_for_training = df_smoothed[["Order Date", "Sales"]].copy()

In [19]: # Split data before scaling
train_data, test_data = train_test_split(df_for_training, test_size=0.30, random_state=42, shuffle=False)

In [20]: train_data.tail()

Out[20]:
   Order Date  Sales
961 2013-11-02  32700.337844
962 2013-11-04  31663.978130
963 2013-11-05  32488.693970
964 2013-11-06  32219.521026
965 2013-11-07  31745.817162

In [21]: test_data.head()

Out[21]:
   Order Date  Sales
966 2013-11-08  32200.446236
967 2013-11-09  31526.586085
968 2013-11-10  30124.233202
969 2013-11-11  30409.693085
970 2013-11-12  30965.818909

In [22]: # Standardizing the sales values using StandardScaler()
# Fit scaler only on training data
scaler = StandardScaler().fit(train_data[["Sales"]])
# Transform both train and test data using the same scaler
train_scaled = scaler.transform(train_data[["Sales"]])
test_scaled = scaler.transform(test_data[["Sales"]])

In [23]: # Define past and future time steps
n_future = 1 # Number of days to predict in the future
n_past = 30 # Number of past days used to make a prediction

# Function to create sequences for time series forecasting
def create_sequences(data, past, future, dataset):
    X, y, data_list = [], [], []
    for i in range(past + len(data) - future + 1):
        X.append(data[i - past:i, 0])
        y.append(data[i + future - 1, 1])
        data_list.append((data[i - past:i, 0], data[i + future - 1, 1]))
    return np.array(X), np.array(y), np.array(data_list)

In [24]: # Prepare training and testing data
X_train, y_train, train_data = create_sequences(train_scaled, n_past, n_future, train_data[["Order Date"]])
X_test, y_test, test_data = create_sequences(test_scaled, n_past, n_future, test_data[["Order Date"]])

X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)

In [25]: # Printing the shapes of train and test datasets.
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
(965, 30, 1) (935,)
(316, 30, 1) (316,)

1. LSTM

In [26]: # Function to build and compile LSTM Sales Forecasting Model
def build_lstm_model():
    model = Sequential(name="LSTM_Sales_Forecasting_Model")
    model.add(Input(shape=(X_train.shape[1], X_train.shape[2]), name="Input_Layer_1"))
    model.add(LSTM(128, activation='tanh', return_sequences=True, name="LSTM_Layer_1"))
    model.add(LSTM(64, activation='tanh', return_sequences=False, name="LSTM_Layer_2"))
    model.add(Dense(128, name="Intermediate_Layer_1"))
    model.add(Dense(64, name="Intermediate_Layer_2"))
    model.add(Dense(1, name="Output_Layer"))
    model.compile(optimizer=Adam(learning_rate=0.0001), loss='mse')
    return model

In [27]: # Train LSTM model
lstm_model = build_lstm_model()
lstm_model.summary(show_trainable=True)

Model: "LSTM_Sales_Forecasting_Model"

In [28]: # Fitting the model
callbacks = [
    EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True, verbose=1), # Stops training if val_loss doesn't improve
    ModelCheckpoint(filepath='lstm_best_model.keras', monitor='val_loss', save_best_only=True, verbose=1), # Saves the best model
    ReduceLROnPlateau(monitor='val_loss', patience=5, min_lr=0.0001, verbose=1), # Reduces learning rate if val_loss doesn't improve
]

lstm_model.fit(X_train, y_train, epochs=50, validation_data=(X_test, y_test), callbacks=callbacks)

Epoch 1/50: val_loss improved from inf to 0.31061, saving model to lstm_best_model.keras
Epoch 2/50: val_loss improved from 0.31061 to 0.15894, saving model to lstm_best_model.keras
Epoch 3/50: val_loss improved from 0.15894 to 0.10394, saving model to lstm_best_model.keras
Epoch 4/50: val_loss improved from 0.10394 to 0.04878, saving model to lstm_best_model.keras
Epoch 5/50: val_loss improved from 0.04878 to 0.02035, saving model to lstm_best_model.keras
Epoch 6/50: val_loss improved from 0.02035 to 0.01628, saving model to lstm_best_model.keras
Epoch 7/50: val_loss improved from 0.01628 to 0.01238, saving model to lstm_best_model.keras
Epoch 8/50: val_loss improved from 0.01238 to 0.01124, saving model to lstm_best_model.keras
Epoch 9/50: val_loss improved from 0.01124 to 0.01042, saving model to lstm_best_model.keras
Epoch 10/50: val_loss improved from 0.01042 to 0.01029, saving model to lstm_best_model.keras
Epoch 11/50: val_loss improved from 0.01029 to 0.01016, saving model to lstm_best_model.keras
Epoch 12/50: val_loss improved from 0.01016 to 0.01012, saving model to lstm_best_model.keras
Epoch 13/50: val_loss improved from 0.01012 to 0.01011, saving model to lstm_best_model.keras
Epoch 14/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 15/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 16/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 17/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 18/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 19/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 20/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 21/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 22/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 23/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 24/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 25/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 26/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 27/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 28/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 29/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 30/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 31/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 32/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 33/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 34/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 35/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 36/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 37/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 38/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 39/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 40/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 41/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 42/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 43/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 44/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 45/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 46/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 47/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 48/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 49/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 50/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras

Epoch 11: ReduceLROnPlateau reducing learning rate to 8.9999972640876e-05.
Epoch 12/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 13/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 14/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 15/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 16/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 17/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 18/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 19/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 20/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 21/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 22/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 23/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 24/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 25/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 26/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 27/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 28/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 29/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 30/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 31/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 32/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 33/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 34/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 35/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 36/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 37/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 38/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 39/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 40/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 41/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 42/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 43/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 44/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 45/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 46/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 47/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 48/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 49/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 50/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras

Epoch 11: val_loss did not improve from 0.01011
Epoch 12/50: val_loss did not improve from 0.01011
Epoch 13/50: val_loss did not improve from 0.01011
Epoch 14/50: val_loss did not improve from 0.01011
Epoch 15/50: val_loss did not improve from 0.01011
Epoch 16/50: val_loss did not improve from 0.01011
Epoch 17/50: val_loss did not improve from 0.01011
Epoch 18/50: val_loss did not improve from 0.01011
Epoch 19/50: val_loss did not improve from 0.01011
Epoch 20/50: val_loss did not improve from 0.01011
Epoch 21/50: val_loss did not improve from 0.01011
Epoch 22/50: val_loss did not improve from 0.01011
Epoch 23/50: val_loss did not improve from 0.01011
Epoch 24/50: val_loss did not improve from 0.01011
Epoch 25/50: val_loss did not improve from 0.01011
Epoch 26/50: val_loss did not improve from 0.01011
Epoch 27/50: val_loss did not improve from 0.01011
Epoch 28/50: val_loss did not improve from 0.01011
Epoch 29/50: val_loss did not improve from 0.01011
Epoch 30/50: val_loss did not improve from 0.01011
Epoch 31/50: val_loss did not improve from 0.01011
Epoch 32/50: val_loss did not improve from 0.01011
Epoch 33/50: val_loss did not improve from 0.01011
Epoch 34/50: val_loss did not improve from 0.01011
Epoch 35/50: val_loss did not improve from 0.01011
Epoch 36/50: val_loss did not improve from 0.01011
Epoch 37/50: val_loss did not improve from 0.01011
Epoch 38/50: val_loss did not improve from 0.01011
Epoch 39/50: val_loss did not improve from 0.01011
Epoch 40/50: val_loss did not improve from 0.01011
Epoch 41/50: val_loss did not improve from 0.01011
Epoch 42/50: val_loss did not improve from 0.01011
Epoch 43/50: val_loss did not improve from 0.01011
Epoch 44/50: val_loss did not improve from 0.01011
Epoch 45/50: val_loss did not improve from 0.01011
Epoch 46/50: val_loss did not improve from 0.01011
Epoch 47/50: val_loss did not improve from 0.01011
Epoch 48/50: val_loss did not improve from 0.01011
Epoch 49/50: val_loss did not improve from 0.01011
Epoch 50/50: val_loss did not improve from 0.01011

Epoch 11: early stopping
Restoring model weights from the end of the best epoch: 8.

Out[28]: <keras.src.callbacks.history.History at 0x258f1e3050>

In [29]: # Plot the losses for train and test data
plt.figure(figsize=(8,4))
plt.plot(lstm_model.history.history['loss'], label='Train loss')
plt.plot(lstm_model.history.history['val_loss'], label='Test loss')
plt.xlabel('Epoch')
plt.ylabel('Loss (MSE)')
plt.title('Loss for each epoch')
plt.legend()
plt.show()

Loss for each epoch

In [30]: # Function to make predictions and scale back results
def predict_and_scale_predictions(model, X, scaler):
    predictions = model.predict(X)
    predictions = scaler.inverse_transform(predictions, 1, axis=1) # Ensure correct shape
    if predictions.shape[0] != 1:
        predictions = predictions.reshape(-1, 1)
    return scaler.inverse_transform(predictions, 1, 0)

In [31]: # Predict sales for train and test datasets
y_pred_train = predict_and_scale_predictions(lstm_model, X_train, scaler)
y_pred_test = predict_and_scale_predictions(lstm_model, X_test, scaler)

df_comparison_train = pd.DataFrame({'Order Date': train_data['Order Date'], 'Actual Sales': scaler.inverse_transform(y_train.reshape(-1, 1)), 'Predicted Sales': y_pred_train})
df_comparison_test = pd.DataFrame({'Order Date': test_data['Order Date'], 'Actual Sales': scaler.inverse_transform(y_test.reshape(-1, 1)), 'Predicted Sales': y_pred_test})

Epoch 30/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 31/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 32/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 33/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 34/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 35/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 36/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 37/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 38/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 39/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 40/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 41/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 42/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 43/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 44/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 45/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 46/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 47/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 48/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 49/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 50/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras

Epoch 11: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 12/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 13/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 14/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 15/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 16/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 17/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 18/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 19/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 20/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 21/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 22/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 23/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 24/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 25/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 26/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 27/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 28/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 29/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 30/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 31/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 32/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 33/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 34/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 35/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 36/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 37/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 38/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 39/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 40/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 41/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 42/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 43/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 44/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 45/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 46/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 47/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 48/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 49/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 50/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras

Epoch 11: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 12/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 13/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 14/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 15/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 16/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 17/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 18/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 19/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 20/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 21/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 22/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 23/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 24/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 25/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 26/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 27/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 28/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 29/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 30/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 31/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 32/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 33/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 34/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 35/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 36/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 37/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 38/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 39/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 40/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 41/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 42/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 43/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 44/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 45/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 46/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 47/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 48/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 49/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 50/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras

Epoch 11: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 12/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 13/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 14/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 15/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 16/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 17/50: val_loss improved from 0.01011 to 0.01011, saving model to lstm_best_model.keras
Epoch 18/50
```