9. Develop neural network-based time series forecasting model.

AIM:

To develop a neural network-based model to forecast future expenses based on past expense data using time series forecasting techniques..

PROCEDURE:

1. Import Libraries
2. Load and Preprocess Data
3. Visualize the Time Series
4. Normalize Data
5. Create Time Series Sequences
6. Build Neural Network Model (LSTM)
7. Train the Model
8. Forecast Future Values
9. Evaluate and Plot Results

CODE:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense


# 1. Load the data

df = pd.read_csv('/mnt/data/expenses.csv')


# 2. Check columns

print("Columns in dataset:", df.columns)
```

```
# 3. Convert date column if it exists
if 'Date' in df.columns:
    df['Date'] = pd.to_datetime(df['Date'])
    df.set_index('Date', inplace=True)

# 4. Select a numeric column for forecasting
# Change 'Total_Expense' below to the actual column name you want to forecast
target_column = df.select_dtypes(include='number').columns[0]
data = df[[target_column]].dropna().values

# 5. Normalize the data
scaler = MinMaxScaler()
scaled_data = scaler.fit_transform(data)

# 6. Create sequences
def create_sequences(data, seq_len=10):
    X, y = [], []
    for i in range(seq_len, len(data)):
        X.append(data[i-seq_len:i])
        y.append(data[i])
    return np.array(X), np.array(y)

SEQ_LEN = 10
X, y = create_sequences(scaled_data, SEQ_LEN)

# 7. Train/test split
split = int(0.8 * len(X))
X_train, X_test = X[:split], X[split:]
y_train, y_test = y[:split], y[split:]
```

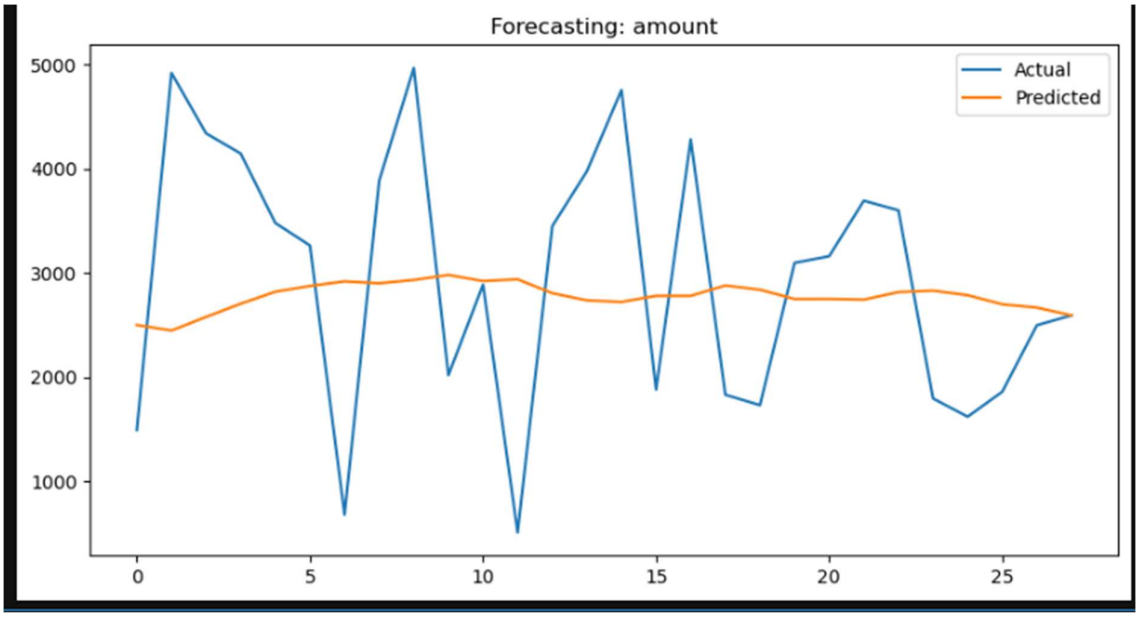
```
# 8. Build the model
model = Sequential([
    LSTM(64, input_shape=(SEQ_LEN, 1)),
    Dense(1)
])
model.compile(optimizer='adam', loss='mse')

# 9. Train the model
model.fit(X_train, y_train, epochs=20, batch_size=16, validation_data=(X_test, y_test))

# 10. Predict and inverse transform
predicted = model.predict(X_test)
predicted = scaler.inverse_transform(predicted)
y_test_orig = scaler.inverse_transform(y_test)

# 11. Plot
plt.figure(figsize=(10, 5))
plt.plot(y_test_orig, label='Actual')
plt.plot(predicted, label='Predicted')
plt.legend()
plt.title(f'Forecasting: {target_column}')
plt.show()
```

OUTPUT:



RESULT:

The program to develop a neural network-based model to forecast future expenses based on past expense data using time series forecasting techniques is implemented successfully.