Deep Learning for Finance: LSTM Time Series Forecasting

Task 1: Building and Benchmarking an LSTM Predictor (100 Points)

This assignment requires you to design, implement, and evaluate a Long Short-Term Memory (LSTM) network to forecast a real-world financial time series.

Part A: Data Acquisition and Preprocessing (30 Points)

1. Select a Time Series: Choose one continuous, historical time series of at least 5 years of daily data (or equivalent monthly/weekly data) related to a financial application. Examples include:
    - Stock Prices: Adjusted Close Price for a major equity (e.g., AAPL, TSLA, S&P 500 Index).
    - Sales/Demand: Daily or Monthly Sales for a major retailer (using publicly available or synthetic data).
    - Interest Rates: A key economic indicator like the 10-Year Treasury Yield or Federal Funds Rate.
2. Data Preparation:
    - Load the data and handle any missing values (e.g., using forward fill).
    - Normalization: Apply the MinMaxScaler to transform the data to a range of [0, 1]. Crucially, the scaler must be fitted ONLY on the training data to prevent data leakage.
    - Temporal Split: Split your dataset into Train (80%) and Test (20%) sets using a strictly temporal split (shuffle=False). The model must predict a future time period it has never seen.

3. Sequence Creation: Define a Lookback Window T (e.g., 60 days). Transform your 1D data into the 3D structure required by the LSTM: (Samples, T, Features).

Part B: LSTM Model Design and Training (40 Points)

1. Architecture Design (FinNet-T): Construct a sequential deep learning model in your chosen framework (Keras/TensorFlow or PyTorch) that adheres to the following structure:
    - Layer 1: An LSTM layer that returns sequences (e.g., 50 units).
    - Layer 2: A Dropout layer (e.g., 0.2) for regularization.
    - Layer 3: An LSTM layer that does NOT return sequences (e.g., 50 units).
    - Layer 4 (Output): A Dense layer with a single unit and linear activation (for regression).
    - Documentation: Provide a summary table or printout of the model architecture, including the output shape and parameter count for each layer.
2. Model Compilation and Training:
    - Compile: Use the Adam optimizer and Mean Squared Error (MSE) as the loss function.
    - Train: Fit the model on your training data for 50-100 epochs, using a suitable batch size.

Part C: Evaluation and Financial Conclusion (30 Points)

1. Prediction and Inverse Transform:
    - Generate predictions on the test set.
    - Inverse Scale both the predicted values and the actual test values back to their original scale (e.g., dollar amount or percentage).

2. Performance Metrics: Calculate and report the following on the unscaled test data:
    - Root Mean Squared Error (RMSE): Quantifies the magnitude of the errors.
    - Mean Absolute Error (MAE): Provides the average magnitude of error in the original units.
3. Visualization: Create a clear plot displaying the Actual vs. Predicted values over the test time period.
4. Financial Analysis: Write a short paragraph (100-150 words) interpreting the results from a financial perspective. Specifically address:
    - Trend Capture: Does the model successfully capture the general trend and major turning points?
    - Lag: Does the prediction appear to lag the actual data? Why is this a problem for real-time trading/forecasting?
    - Risk: Based on the RMSE, what is the estimated financial risk (average error) if this model were used to execute a transaction?