Nifty Stock Price Prediction with LSTM

1. Introduction

The project aims to predict the Nifty50 stock index's closing prices using a Long Short-Term Memory (LSTM) neural network model. LSTM is a specialized type of Recurrent Neural Network (RNN) optimized for sequence prediction problems.

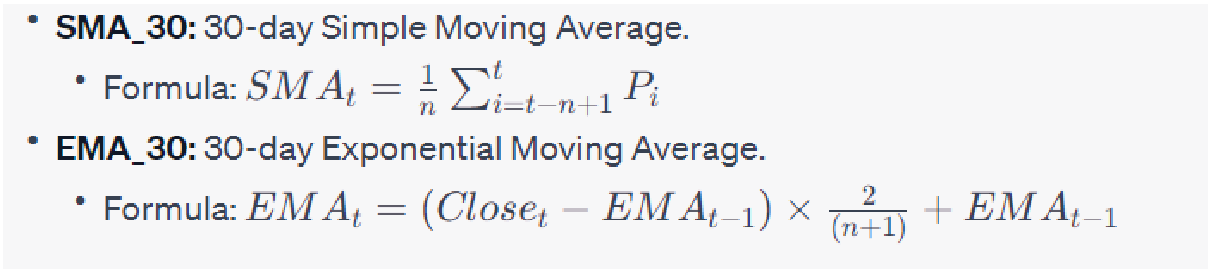
2. Data Acquisition and Features

2.1 Data Source

- Nifty50 Index: Historical stock market data representing the 50 most traded stocks of NSE India.

2.2 Data Features

- Date: Timestamp of recorded data.
- Close: Closing stock price for the day.



- **SMA_30:** 30-day Simple Moving Average.
  - Formula: $SMA_t = \frac{1}{n} \sum_{i=t-n+1}^{t} P_i$
- **EMA_30:** 30-day Exponential Moving Average.
  - Formula: $EMA_t = (Close_t - EMA_{t-1}) \times \frac{2}{(n+1)} + EMA_{t-1}$

3. Methodology

3.1 Data Collection

- Utilize the yfinance library to fetch historical data.
- Extract the "Close" prices to form the primary dataset.
- Calculate and append SMA_30 and EMA_30 to the dataset using their respective formulas.

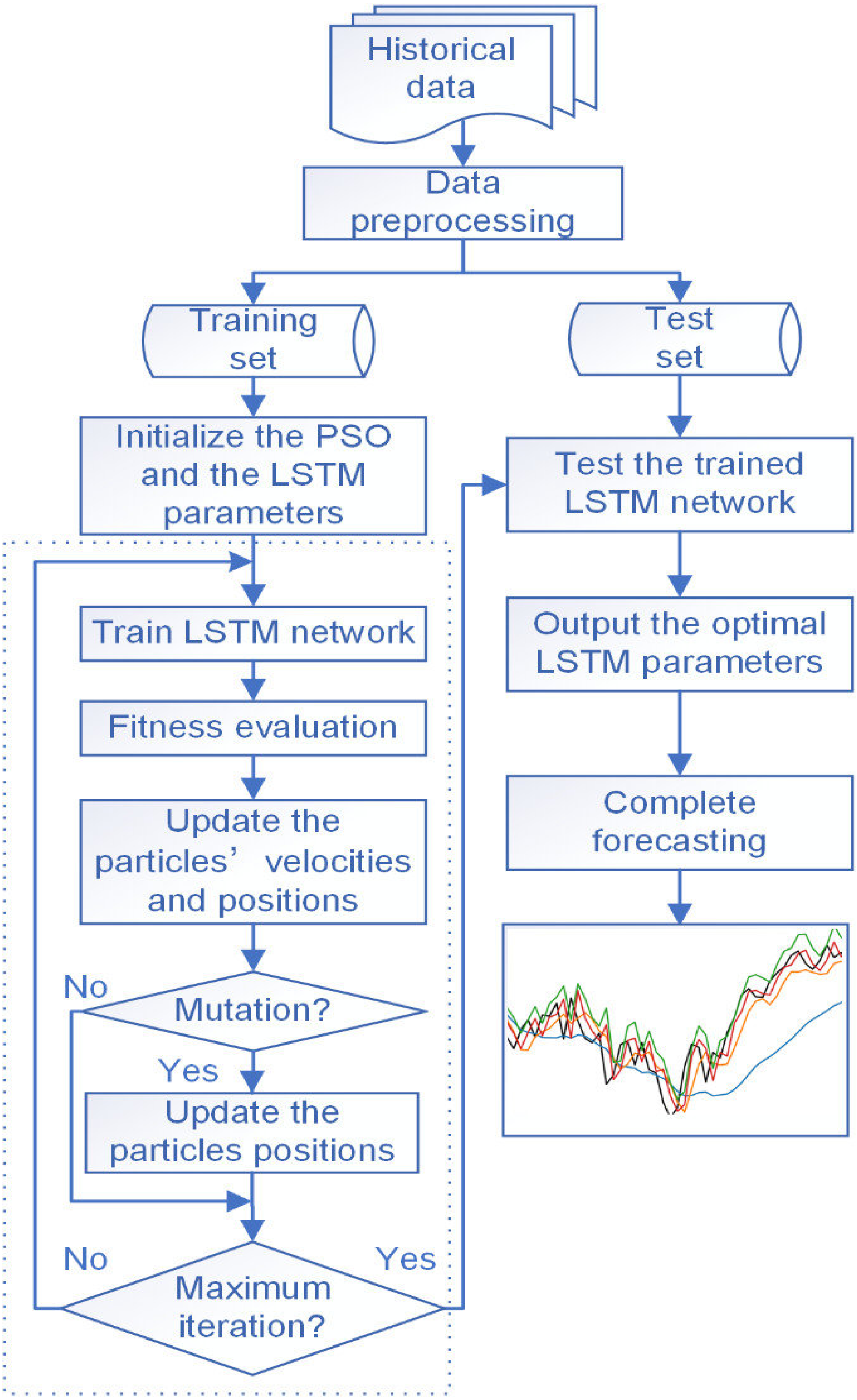
3.2 Data Preprocessing

3.2.1 Scaling

- Implement the MinMaxScaler from Scikit-learn.
- Transform the "Close" price data to fit within the range [0, 1].

3.2.2 Training and Test Data Preparation

- Allocate 75% of the scaled dataset for training the LSTM model.
- Utilize 25% for testing.
- Create overlapping sequences of 30 days to train the LSTM, as LSTM models require input sequences.


Historical data
Data preprocessing
Training set
Test set
Initialize the PSO and the LSTM parameters
Train LSTM network
Fitness evaluation
Update the particles' velocities and positions
No
Mutation?
Yes
Update the particles positions
No
Maximum iteration?
Yes
Test the trained LSTM network
Output the optimal LSTM parameters
Complete forecasting

3.3 LSTM Model Architecture

3.3.1 Input Layer

- Input shape: (30, 1) representing a sequence of 30 days.

3.3.2 LSTM Layers

- First LSTM Layer:
    - 50 units.
    - Return sequences: True.
    - Activation function: tanh (hyperbolic tangent).
    - Recurrent activation: sigmoid.
- Second LSTM Layer:
    - 50 units.
    - Return sequences: False.
    - Activation function: tanh.
    - Recurrent activation: sigmoid.

3.3.3 Dropout Layers

- Dropout rate: 0.2 (20% of neurons are randomly turned off during each epoch).

3.3.4 Dense Layers

- First Dense Layer: 25 units with 'relu' activation.
- Output Dense Layer: 1 unit to predict the stock price.

3.4 Model Training

- Optimizer: Adam optimizer with adaptive learning rates.
- Loss Function: Mean Squared Error (MSE) to minimize the squared differences between actual and predicted values.
- Epochs: 200
- Batch Size: 50

3.5 Model Evaluation

- Implement Root Mean Square Error (RMSE) to gauge the model's error magnitude.
- R^2 (Coefficient of Determination) to measure how well future samples are predicted by the model.

3.6 Visualization

- Use the plotly library for interactive visualizations.
- Plot historical vs. predicted stock prices.
- Visualize the trend with SMA_30 & EMA_30.
- Forecast future stock prices.

4. Technical Implementation

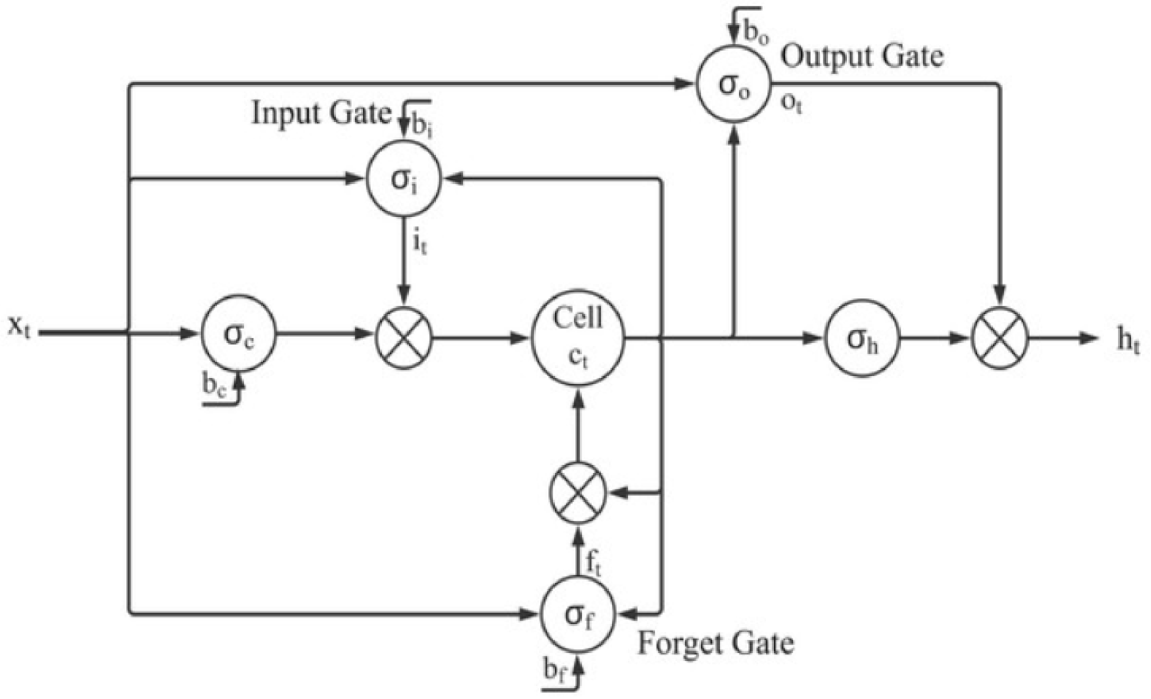
4.1 Libraries and Dependencies

- yfinance: Fetch stock data.
- Keras: Neural network framework.
- Scikit-learn: Data preprocessing.
- Plotly: Data visualization.
- Pandas and NumPy: Data manipulation.

4.2 LSTM Mathematical Background

- LSTMs use three gates: input, forget, and output, each with its activation functions.
- The cell state in LSTM allows for long-term memory.

- Equations:
  - Forget gate: $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
  - Input gate: $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
  - Update gate: $\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$
  - Final cell state: $C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t$
  - Output gate: $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$
  - Hidden state: $h_t = o_t \times tanh(C_t)$


b_o
Output Gate
σ_o
o_t
Input Gate
b_i
σ_i
i_t
x_t
σ_c
b_c
Cell
c_t
σ_h
h_t
f_t
σ_f
Forget Gate
b_f