Stock Data Predictor with LSTMs

Anjani Bahl
B.Sc. Management and Technology
03767337
anjani.bahl@tum.de

Carolyn Vool
B.Sc. Management and Technology
03775411
carolyn.vool@tum.de

Stanisław Woźniak
B.Sc. Management and Technology
03777776
ge92taq@mytum.de

***Abstract*—This project implements a Long Short-Term Memory (LSTM) neural network to forecast next-day open stock prices using historical data from the Kaggle S&P 500 dataset. Data is normalised, segmented into rolling windows (10, 20, 50 trading days) and fed to a custom pytorch LSTM that fuses gate computations into two matrix multiplications for efficiency. We evaluate mean squared error (MSE) and coefficient of determination (R2) on a held-out test set. Results show that context windows up to ~20 days improve accuracy, with diminishing returns beyond that. We also outline practical pitfalls (data leakage, temporal splits, inverse scaling) and propose extensions (multi-feature inputs and attention).**

Keywords: LSTM, stock prediction, S&P 500, time series, neural networks

I. Introduction

Predicting stock market prices is a long-standing and challenging task due to the markets' volatile and nonlinear nature. Traditional linear models such as ARIMA are effective on stationary data, but deep learning models like LSTM networks excel at capturing both short- and long-term dependencies inherent in financial time series data. This paper documents the process and outcomes of building an LSTM-based predictor for future stock OPEN values, using the S&P 500 dataset.

II. Data Collection and Preprocessing

- Dataset: S&P 500 stock prices from Kaggle, Yahoo finance (yfinance library) focusing on OPEN prices.
- Data Loading: Read the data using a Python script; can be limited to a manageable subset for hardware constraints.
- Normalization: All features (e.g., OPEN, CLOSE, VOL) are scaled to the range using MinMaxScaler, a standard method for stabilizing neural net training.
- Windowing: The data is reshaped into overlapping sequences (windows) of lengths 10, 20, and 50. Each window sequence is paired with the subsequent OPEN value for forecasting.
- Train-Test Split: The dataset is divided into a training set (e.g., 80%) and a test set (10%) to ensure unbiased evaluation. In addition, 20% of the training set was selected as a validation set to prevent overfitting.

III. Methods

We implemented a custom long short-term (LSTM) network in pytorch to forecast next-day open prices from historical S&P 500 data (ticker's name is ^GSPC). The full training pipeline is implemented in python and follows these steps.

Data source and selection - we downloaded daily price series for the S&P 500 index using yfinance for the period 2020-01-01 through 2025-01-01. We focus on the open price as our target; other columns (close, high, low, volume) are used only during exploratory analysis and are not fed to the single-feature models reported here.

Preprocessing - for each experiment the series is converted into overlapping windows of fixed length (10,20,50). For a window size w, each example contains the previous w open values (features) and the subsequent open value as the target. To stabilise the training we scale the data using min-max normalisation.

Model architecture - the model is a compact custom LSTM implemented with two matrix multiplications per time step (input and recurrent matrices) and manual gate computations (input, forget, cell, output)[1]. The model uses a single recurrent layer with 8 hidden units and a final linear layer projection of the last hidden state to a single scalar output.

Training procedure - we train each model for 4 epochs using the Adam optimiser (lr=0.001) and mean squared error (MSE) loss. Batches of size 16 are used. To evaluate generalisation, we split the prepared dataset into train / validation / test groups (train/val split is performed via a random split of the training dataset in the current code).

Evaluation metrics and visualisation - models are evaluated on MSE as the optimisation target and reported with r2 on the hold-out test set to quantify explained variance. We also visualise predicted vs actual open prices and plot train/validation loss curves.

Implementation details - experiments are implemented in pytorch. Data preparation creates numpy arrays and then converts them to torch.tensors and dataloaders. The custom dataset class preserves the (sequence, feature) shape required by the model.

IV. Results

A. Training Performance

All window sizes exhibited rapid convergence with decreasing loss values:

Window	Final Train Loss	Final Val Loss
10	0.0067	0.0029
20	0.0020	0.0014
50	0.0021	0.0015

B. Prediction Accuracy

R^2 scores on test sets increased moderately with window size:

- Window 10: 0.94
- Window 20: 0.96
- Window 50: 0.95

C. Visual Results

Figures 1–3 (see Appendix) compare actual and predicted Open prices and show train loss for a given number of epochs for each window size. Models tracked general trends with greater stability at larger window sizes, while shorter windows captured short-term fluctuations better.

V. Discussion

The experimental results confirm that incorporating longer historical context (up to 20 days) enhances forecasting accuracy. The 50-day window did not significantly improve R^2 further, likely due to diminishing returns and model complexity.

Model limitations include underperformance during rapid market shifts, suggesting the benefit of augmented features (technical indicators, sentiment data) or advanced architectures (attention mechanisms).

VI. Conclusion

This work demonstrates that LSTM models, trained on appropriately windowed historical stock data, can reliably forecast S&P 500 next-day Open prices with strong accuracy. The study underscores the importance of window size in balancing short-term sensitivity and long-term trend stability.

Future work will explore multi-feature inputs and hybrid models to further enhance predictive power.

VII. Appendix & Citations

1. *piEsposito, "pytorch-lstm-by-hand: LSTM.ipynb," GitHub repository. Available:https://github.com/piEsposito/pytorch-lstm-by-hand/blob/master/LSTM.ipynb*

2. *Pilla, Prashant, and Raji Mekonen. "Forecasting S&P 500 Using LSTM Models." arXiv, 29 January 2025, https://arxiv.org/html/2501.17366v1.*

FIGURE 1

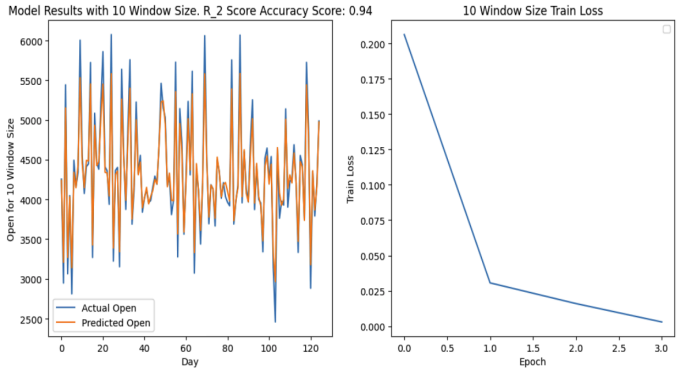

FIGURE 2

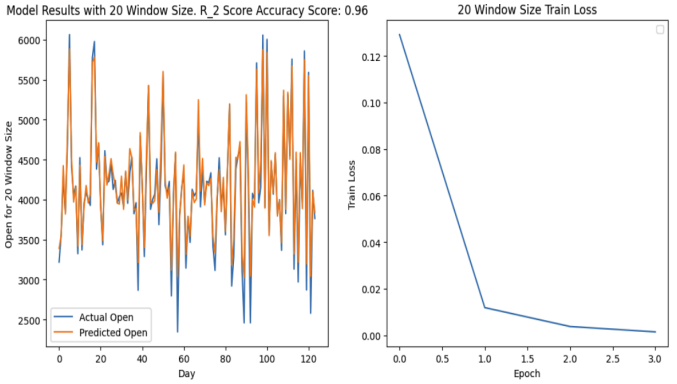

FIGURE 3