FORECASTING STOCK MARKET TRENDS USING LEARNING-BASED TIME-SERIES MODELING TECHNIQUES

Authors:

Mohammad Afrazi

Patrick Lo

Problem Description

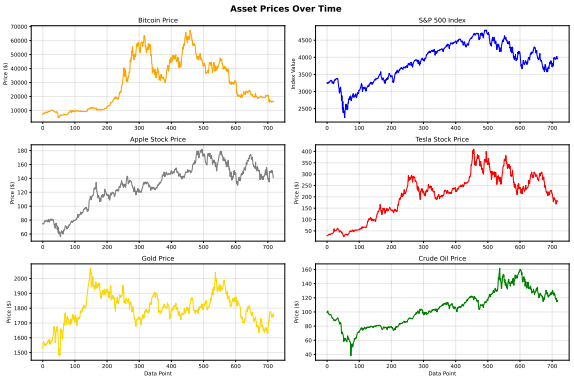
Predicting stock market movements is highly challenging due to the market's volatility, non-linearity, and chaotic dynamics [1]. Traditional time-series models such as ARIMA struggle to capture complex temporal dependencies in financial data, limiting forecasting accuracy and decision-making in finance [3].

We argue and hypothesize that learning-based time-series models - specifically Long Short-Term Memory (LSTM) networks, Temporal Convolutional Networks (TCNs), and Transformer- based models - can better model long-term dependencies in stock price data. This project will develop and compare these models for forecasting the daily closing price of the **S&P 500** index using historical price. Via evaluating LSTM, TCN, and Transformer approaches on a recent dataset of U.S. market data, we aim to advance financial forecasting methods and provide more reliable tools for investment and risk management [4].

Data Description

To address our research question, we utilize the "US Stock Market Dataset" from Kaggle [2], which contains over five years of daily stock and commodity data for a diverse set of U.S.-listed companies and assets. Each record includes the following features: *No, Date, Natural_Gas, Natural_Gas_Vol., Crude_oil, Crude_oil_Vol., Copper, Copper_Vol., Bitcoin, Bitcoin_Vol., Ethereum, Ethereum_Vol., S&P_500, Nasdaq_100_Price, Nasdaq_100_Vol., Apple, Apple_Vol., Tesla, Tesla_Vol., Microsoft, Microsoft_Vol., Silver, Silver_Vol., Google, Google_Vol., Nvidia, Nvidia_Vol., Berkshire, Berkshire_Vol., Netflix, Netflix_Vol., Amazon, Amazon_Vol., Meta, Meta_Vol., Gold, Gold_Vol.*

To mitigate the effects of major scale differences across time periods, we restrict our analysis to data from January 2, 2020, to November 29, 2022, yielding a total of 718 daily observations. Figures 1a and 1b provide visual summaries of the dataset, illustrating overall asset price dynamics and a sample of the input features used in our analysis.



(a) Asset price trends.

Data Preview - Total 718 rows

Date	Natural_Gas	Crude_oil	Copper	...	S&P_500	Apple	...
2/1/2020	2.122	99.18	2.825	...	3257.85	75.09	...
3/1/2020	2.13	101.05	2.787	...	3234.85	74.36	...
6/1/2020	2.135	101.27000000000	2.79	...	3246.28	74.95	...
7/1/2020	2.162	100.7	2.7935	...	3237.18	74.6	...
8/1/2020	2.141	97.61	2.812	...	3253.05	75.8	...
...	...	...	...	...	...	...	...

(b) Data preview.

Figure 1: Visualization of the dataset showing asset price dynamics and a sample of the input data.

Preprocessing

Log Return

To make the data more stationary and suitable for modelling, we first compute the **log returns** of each asset. This is to normalize price movements and stabilize variance across different scales. The log return for time k is calculated as

$$r_k = \ln\left(\frac{P_k}{P_{k-1}}\right),$$

where P_k represents the asset price at time t. It also helps mitigate the effects of large price discrepancies between assets, ensuring that model training focuses on relative changes rather than absolute price levels. Compared to figure 1a, below shows the log-return transformed of all asset prices.

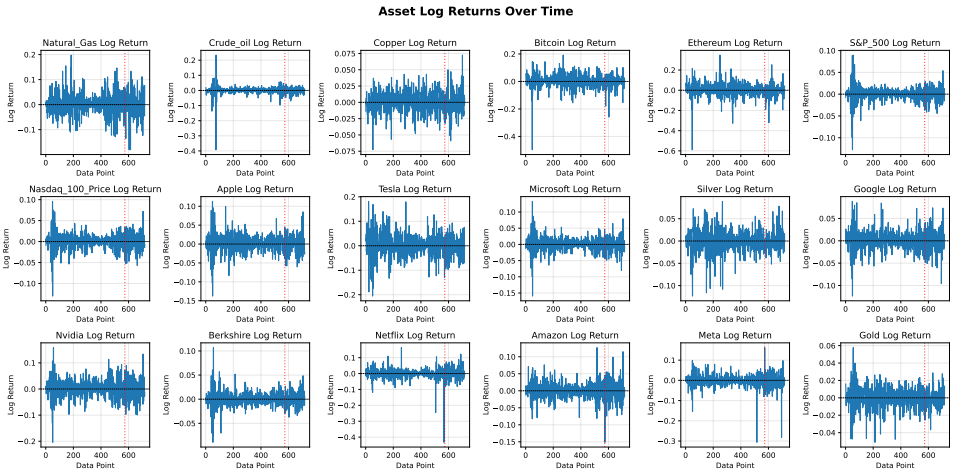


Figure 2: Log-return transformed asset price trends.

Dimension Reduction

Here we perform **lag-principal component analysis (Lag-PCA)** to identify the most informative features in the dataset and reduce dimensionality. Since we are dealing with time-series data, we create a lagged version by sliding a window of size $l = 70$ over the time series. For each data chunk, we perform PCA independently to capture local temporal patterns. We then apply a **majority voting** scheme across all chunks to determine the most significant principal components ($n = 10$) to retain for model training. The majority voting is updated via the following heuristic:

$$V_i \leftarrow V_i + n * \frac{1}{\text{rank}_i}.$$

Here, rank_i denotes the rank of the i^{th} component within a chunk, and n is the number of components considered. The components with the highest aggregated scores across all windows are selected as the final features for model training. Figure 3 shows the final result of the Lag-PCA.

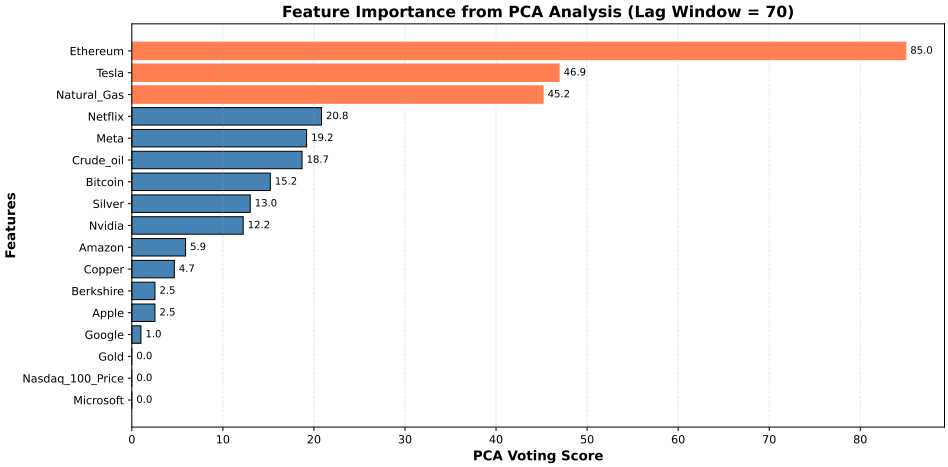


Figure 3: Lag-PCA feature importance.

Methods

In this project, we implement and compare three learning-based time-series models: Long Short-Term Memory (LSTM) networks, Temporal Convolutional Networks (TCNs), and Transformer-based models.

LSTM

We implement the LSTM model using PyTorch. The architecture consists of two LSTM layers followed by two fully connected layers. The model is trained using the Adam optimizer with a learning rate of 0.001 and Mean Squared Error (MSE) as the loss function. We employ early stopping based on validation loss to prevent overfitting.

Below shows the model architecture: **include figure here**

Preliminary Results

Here we include the preliminary results of the LSTM model. The training and validation loss curves are shown in Figure **??**. We observe that the training loss decreases steadily, while the validation loss starts to increase after a certain number of epochs, indicating potential overfitting.

Current Conclusions, Unexpected Challenges, Next Steps

- Successfully implemented the **LSTM** model for time-series prediction.
- Observed that the test loss is larger than the training loss, indicating possible **overfitting**. To mitigate this, we plan to gather and incorporate more data.
- With sufficient data, we intend to implement and compare the performance of **Temporal Convolutional Networks (TCN)** and **Transformer-based models** against the LSTM.

- Additionally, we will compare the results with traditional **regression models** to evaluate relative performance.
- We plan to investigate the effect of dimensionality reduction by examining whether reducing the number of **PCA components** significantly degrades model performance.
- Finally, we aim to train a model using the **raw price data** (without log-return transformation) and compare its results with the log-return–based model.

Work Breakdown:

- **Mohammad Afrazi:** Data preprocessing, feature engineering, writing, and model training/evaluation.
- **Patrick Lo:** Data preprocessing, feature engineering, writing, and model training/evaluation.

References

[1] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European journal of operational research*, 270(2):654–669, 2018.

[2] Saket K. "2019–2024 us stock market data: 5-year trends in crypto, stocks, metals, and energy markets". Kaggle Dataset, 2024. Accessed: 2025-11-11.

[3] Mehdi Khashei and Mehdi Bijari. A novel hybridization of artificial neural networks and arima models for time series forecasting. *Applied Soft Computing*, 11(2):2664–2675, 2011.

[4] David MQ Nelson, Adriano CM Pereira, and Renato A De Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. Ieee, 2017.