Project Report

Forecasting Tomorrow: Navigating Chaos to Predict Stock Market Trends

Arushi Agarwal

May 20, 2024

1 Summary

This project delves into stock price prediction using deep learning techniques, particularly focusing on Recurrent Neural Networks (RNNs) due to their suitability for time series data. The study starts with data collection from S&P 500 companies, preprocessing the data, and balancing the training set. It then discusses why classification is chosen over regression, selecting RNNs over traditional models like OLS and kNN.

The model architecture includes a SimpleRNN layer followed by several dense layers, trained using binary cross-entropy loss and optimized with Adam. Evaluation metrics such as f1 score, accuracy, and ROC curve are employed to assess performance.

Results show a moderate level of accuracy in predicting stock price movements, with potential for further optimization and refinement. The study contributes insights into the effectiveness of deep learning models for stock price prediction, suggesting avenues for future research and model enhancements.

2 Introduction

In today's dynamic financial environment, the ability to accurately predict the movement of stock prices represents not just a desirable skill but a strategic advantage. Stock markets are influenced by a myriad of factors, including economic indicators, geopolitical events, company performance, investor sentiment, and inherent randomness.

Our project leverages historical stock data and analytical techniques to uncover meaningful patterns and relationships within the complex financial landscape. Through data-driven approaches and statistical learning techniques, we aim to identify and capture as many predictive patterns as possible.

This project's significance spans various aspects of the financial sector, providing precise predictions empowering investors and traders to make informed, data-driven decisions about their stocks. By developing a robust predictive model comparable to those used by financial institutions, we aim to enhance prediction accuracy and reliability. This effort contributes to improved decision-making and risk management practices in finance, showcasing the potential for innovation and actionable insights at the intersection of technology, data analytics, and finance.

3 Research and Related Work

The study of predicting stock prices has garnered significant attention, with a myriad of techniques, models, and approaches being explored and refined continuously. The existing research in time series forecasting and stock price prediction can be broadly divided into three approaches. The first group primarily employs multivariate regression on cross-sectional data. However, these models often lack accuracy due to their simplicity and reliance on linear assumptions. The second group employs econometric techniques such as ARIMA, Granger Causality Test, ARDL, and VAR to forecast stock prices, leveraging concepts from time series analysis. In contrast, the third group focuses on learning-based methods like machine learning, deep learning, and natural language processing for stock price prediction. Since the past few years, these deep learning techniques have gained a significant bit of attention.

3.1 Forecasting NIFTY 50 Stock Index Movements

The study by Sidra Mehtab, Jaydip Sen, and Abhishek Dutta [MSD21] focuses on forecasting NIFTY 50 stock index movements using machine learning and deep learning models.

The study leveraged NIFTY 50 stock index movements using daily historical data from December 29, 2014, to July 31, 2020. The data was sourced from Yahoo Finance. The data preprocessing phase involved normalization and the derivation of several key variables. These included high_norm (normalized highest price for a stock each day), low_norm (normalized lowest price for a stock each day), close_norm (normalized closing price for a stock each day), volume_norm (normalized volume of stock traded each day), and range_norm (normalized range, computed as the difference between the high and low values for each index record).

The study explored eight machine learning-based regression models: multivariate linear regression, multivariate adaptive regression spline (MARS), regression tree, bootstrap aggregation (Bagging), extreme gradient boosting (XGBoost), random forest (RF), artificial neural network (ANN), and support vector machine (SVM). Additionally, four LSTM-based deep learning regression models were evaluated, including univariate LSTM models with one and two

weeks of input data, encoder-decoder LSTM models with univariate data, and a multivariate encoder-decoder LSTM model.

Performance assessment utilized two main metrics: the product-moment correlation coefficient between actual and predicted open NIFTY 50 index values, and the ratio of RMSE to the mean of actual open values in the dataset. The dataset was divided into training and testing sets, with training data spanning from December 29, 2014, to December 28, 2018, and testing data from December 31, 2018, to July 31, 2020.

Among the machine learning models, multivariate regression and random forest exhibited superior accuracy. However, LSTM-based deep learning models demonstrated overall better performance, with the univariate LSTM model using one-week prior data input emerging as the most accurate and efficient in terms of execution time.

The study underscores the efficacy of deep learning models, particularly RNN-based ones, in predicting stock index movements. It also highlights the importance of selecting appropriate input data and model architectures for optimal results. For this project, a good starting point would be to use a CNN or an RNN, similar to those utilized in the study. Additionally, the specific choice of features used in the study is worth considering.

3.2 Forecasting Stock Prices Using LSTM with Genetic Programming

The research by Qi Li, Norshaliza Kamaruddin, Siti Sophiayati Yuhaniz, and Hamdan Amer Ali Al-Jaifi [LKY+24] focuses on forecasting stock price changes using LSTM neural networks with genetic programming.

This research delves into the complexities of stock market analysis, leveraging a dataset spanning 3558 stocks from the Chinese market over a two-year period, employing advanced feature engineering techniques and customized deep learning models to achieve superior prediction accuracy and uncovering avenues for future research in financial and technical domains.

The dataset consists of 3558 stocks from the Chinese stock market, collected through open-sourced API (Tushare) and web scraping from Sina Finance and SWS Research websites. It includes daily price data, daily fundamental data for each stock, suspending and resuming history, top 10 shareholders, etc. The dataset spans 2 years, chosen for analysis based on investor practices and analysis benefits from more recent data.

The study focuses on feature engineering and stock price trend prediction using Long Short-Term Memory (LSTM) models. A new algorithm component called feature extension is developed, combined with recursive feature elimination (RFE) and principal component analysis (PCA) for effective feature engineering. The LSTM model is customized and fine-tuned for prediction accuracy.

Evaluation includes comparing the LSTM model with other machine learning models, assessing prediction accuracy, feature engineering effectiveness, and model customization.

The proposed system achieves high prediction accuracy, outperforming leading models. The feature extension algorithm significantly improves model performance.

The research highlights the importance of feature engineering and model customization in improving prediction accuracy for stock price trends. It also again underscores the effectiveness of using RNN-based models in this type of problem.

3.3 Time series forecasting using a hybrid ARIMA and neural network model

In line with the observations from the above papers, nonlinear models based on Recurrent Neural Networks (RNNs) exhibit notably relevant results. Additionally, delving into hybrid model techniques presents a natural evolution to potentially capture more intricate patterns. G. Peter Zhang's study [Zha03] precisely delves into this by proposing a hybrid model combining linear and nonlinear components to better understand time-series behavior. This is particularly relevant for analyzing stock data, which also follows time-series patterns.

Zhang's research utilized three established time series datasets: Wolf's sunspot data, Canadian lynx data, and British pound/US dollar exchange rate data. These datasets were carefully chosen to showcase the effectiveness of the hybrid ARIMA (AutoRegressive Integrated Moving Average) and neural network model. The data underwent preprocessing steps such as differencing and power transformation to ensure stationarity where necessary.

The study primarily focused on the hybrid methodology, blending ARIMA models to handle linear aspects with ANNs (Artificial Neural Networks) to capture nonlinear patterns within the residuals of the ARIMA models. Evaluation of forecasting accuracy utilized metrics like mean squared error (MSE) and mean absolute deviation (MAD) across different time series datasets and forecasting horizons.

The findings consistently demonstrated that the hybrid ARIMA-ANN model outperformed standalone ARIMA and ANN models in terms of forecasting accuracy across various datasets and forecasting timelines. This indicates that the integration of both linear and nonlinear models can significantly enhance forecasting capabilities.

The implications of this study underscore the potential value of adopting a hybrid approach for time series forecasting in this project. By combining linear and nonlinear models effectively, it becomes possible to capture a broader range of patterns within the data. However, determining the optimal combination of these models for the specific dataset would require extensive analysis and exploration.

4 Problem Statement and Discussion

The objective of this study is to leverage deep learning to recognize, learn, and predict patterns in stock price time series. The Efficient Market Hypothesis suggests that consistently outperforming the market is improbable since prices adjust rapidly to new information. This would mean that a predictive model can not consistently perform better than a random walk model, which predicts the next value randomly based solely on the latest one. By analyzing a few weeks' worth of daily closing prices for a stock, the goal is to make predictions for the next day's closing price, and reject the EFH in the process.

Figure 1: Example of an observation of the time series data collected

The initial consideration was whether to treat this as a classification problem to predict the direction of movement or as a regression problem to predict the actual price. Initial efforts focused on regression. I evaluated the model against the efficient market hypothesis by generating next-day observations using a random walk model and assessing performance using RMSE. I trained an ordinary least squares (OLS) regression model and found that it consistently performed worse in terms of RMSE compared to a random walk model.

Upon further investigation, it was discovered that while the OLS model was better at predicting the direction of price movement, RMSE evaluates errors in both directions equally. This is not ideal in the context of our problem, as predicting the direction of the movement accurately is more critical than minimizing the error magnitude. Therefore, I decided to reframe the problem as a classification task instead.

Multivariate regression models, which rely heavily on linear assumptions, were initially considered but did not meet expectations. Similarly, k-nearest

neighbors (kNN) did not perform well, likely because it struggles to identify local patterns in time series data. This project thus focused on deep learning models, specifically Recurrent Neural Networks (RNNs), due to their suitability for handling the sequential nature of time series data.

RNNs are designed to recognize temporal patterns and dependencies in sequential data, making them an appropriate choice for stock price prediction. By capturing the temporal dynamics and patterns within the time series, RNNs can potentially provide more accurate and meaningful predictions regarding stock price movements.

In summary, the study transitioned from regression to classification due to the importance of direction prediction in stock price movements and the limitations of traditional models like OLS and kNN. Deep learning models, particularly RNNs, were chosen for their ability to handle the complexities and temporal nature of stock price time series data.

5 Methodology and Solution

5.1 Software Used

The following Python libraries and packages were utilized in the development and analysis of this project:

- Pandas: Used for data manipulation, analysis, and cleaning.
- BeautifulSoup: Utilized for web scraping tasks.
- Requests: Used for making HTTP requests to fetch data from web sources.
- yfinance: Used for fetching historical stock market data.
- NumPy: Utilized for numerical computations and array operations.
- Matplotlib: Used for data visualization, particularly plotting charts and graphs.
- Seaborn: Utilized for enhancing the visual aesthetics of plots.

For machine learning and statistical analysis:

- Scikit-Learn: Utilized for machine learning algorithms, metrics evaluation, and data preprocessing tasks.
- TensorFlow: Utilized for deep learning tasks, including building and training neural networks.

These libraries and packages were instrumental in conducting data analysis, visualization, model building, and evaluation throughout the project.

5.2 Data

The dataset for this study was derived from the stock prices of companies listed in the S&P 500 index. The ticker symbols for these companies were scraped from Wikipedia. Using these ticker symbols, the daily stock data was retrieved from Yahoo! Finance via the yfinance API, covering the period from January 1, 2020, to January 2, 2024.

The scraped data included the following fields for each observation – closing price, opening price, highest price, lowest price, and volume traded. For the purpose of this study, I focused on the closing price time series to evaluate its effectiveness as a predictor of future stock movements. The preprocessing steps included cleaning the data and arranging the daily closing prices into time series data spans of six weeks. This preprocessing step ensured that the data was structured to maximize the number of useful entries derived from the raw data.

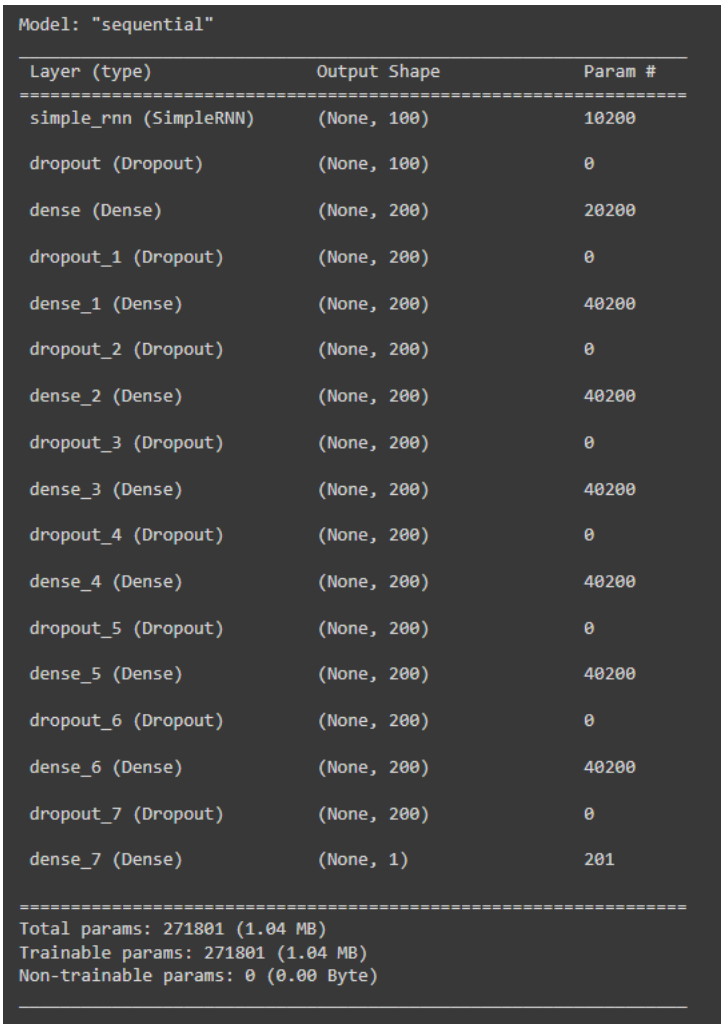
Once the data was organized, it was divided into training, validation, and test sets. A key challenge encountered was the imbalance in the training data; the number of observations where the stock price increased the next day differed from those where it decreased. Using imbalanced data to train a model may result in biases being built into the model. To address this, I balanced the training data by randomly oversampling the minority class, ensuring a more even distribution of observations for the model training phase.

5.3 Model Architecture

The model architecture includes a SimpleRNN layer, which is a type of recurrent neural network layer. The SimpleRNN layer operates as follows:

$$h_t = \sigma(W_{ih}x_t + W_{hh}h_{t-1} + b_h)$$
$$y_t = \sigma(W_{hy}h_t + b_y)$$

where h_t represents the hidden state at time step t, x_t is the input at time step t, W_{ih} and W_{hh} are the weight matrices for the input-to-hidden and hidden-to-hidden connections respectively, b_h is the bias vector for the hidden state, y_t is the output at time step t, W_{hy} is the weight matrix for the hidden-to-output connection, and b_y is the bias vector for the output.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn (SimpleRNN)      (None, 100)               10200

 dropout (Dropout)           (None, 100)               0

 dense (Dense)               (None, 200)               20200

 dropout_1 (Dropout)         (None, 200)               0

 dense_1 (Dense)             (None, 200)               40200

 dropout_2 (Dropout)         (None, 200)               0

 dense_2 (Dense)             (None, 200)               40200

 dropout_3 (Dropout)         (None, 200)               0

 dense_3 (Dense)             (None, 200)               40200

 dropout_4 (Dropout)         (None, 200)               0

 dense_4 (Dense)             (None, 200)               40200

 dropout_5 (Dropout)         (None, 200)               0

 dense_5 (Dense)             (None, 200)               40200

 dropout_6 (Dropout)         (None, 200)               0

 dense_6 (Dense)             (None, 200)               40200

 dropout_7 (Dropout)         (None, 200)               0

 dense_7 (Dense)             (None, 1)                 201

=================================================================
Total params: 271801 (1.04 MB)
Trainable params: 271801 (1.04 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 2: Chosen Model Architecture

The loss function utilized in training the model is binary cross-entropy loss, which is commonly used for binary classification tasks:

$$\text{Binary Cross Entropy Loss} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

where N is the number of samples, y_i is the true label, and $\hat{y}_i$ is the predicted probability for the positive class (in this case, the direction of stock price movement).

The optimizer used for training the model is Adam, which is an adaptive learning rate optimization algorithm. It combines the advantages of AdaGrad and RMSProp, providing efficient gradient descent optimization.

Training was conducted over 300 epochs using the training data. The model weights that resulted in the lowest validation loss across all epochs were selected for final evaluation and prediction.

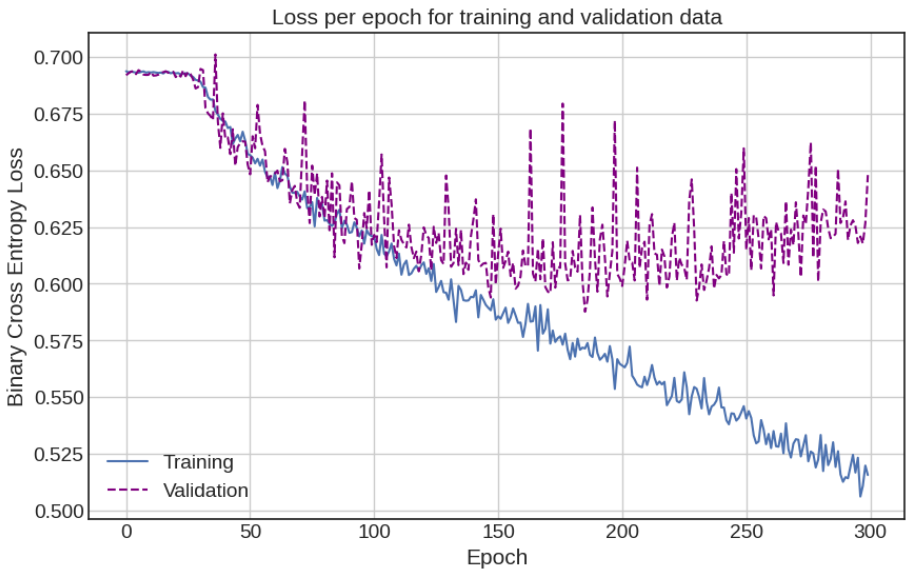


Figure 3: Training and validation loss versus epochs during the training process

5.4 Evaluation Metrics

This project involves a binary classification problem, and the metrics used for evaluation are as follows:

5.4.1 F1 Score

The f1 score is a metric that combines precision and recall into a single value, providing a balanced assessment of the classifier's performance. It is calculated using the following formula:

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

and TP (True Positives), FP (False Positives), FN (False Negatives) represent the counts of correctly classified positive samples, incorrectly classified positive samples, and incorrectly classified negative samples, respectively. The F1 score can range from 0 to 1, with higher values indicating better performance. A F1 score of 0 indicates that the model makes no correct predictions, while a score of 1 represents perfect predictions.

5.4.2 Accuracy

Accuracy measures the overall correctness of the classifier's predictions and is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where

$$TN \text{ (True Negatives)}$$

represents the count of correctly classified negative samples.

The accuracy metric ranges from 0 to 1, with higher values indicating better performance. An accuracy of 1 means all predictions are correct, while 0 indicates that the model makes no correct predictions.

5.4.3 ROC Curve and AUC

The Receiver Operating Characteristic (ROC) curve is a graphical representation of the classifier's performance across various threshold values. It plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at different threshold settings. The area under the ROC curve (AUC) quantifies the overall performance of the classifier. The AUC can range from 0 to 1, with higher values indicating better discrimination ability. An AUC of 0.5 suggests random guessing, while an AUC of 1 represents perfect discrimination.

These evaluation metrics provide a comprehensive assessment of the classifier's performance, considering both the positive and negative class predictions.

6 Results and Conclusions

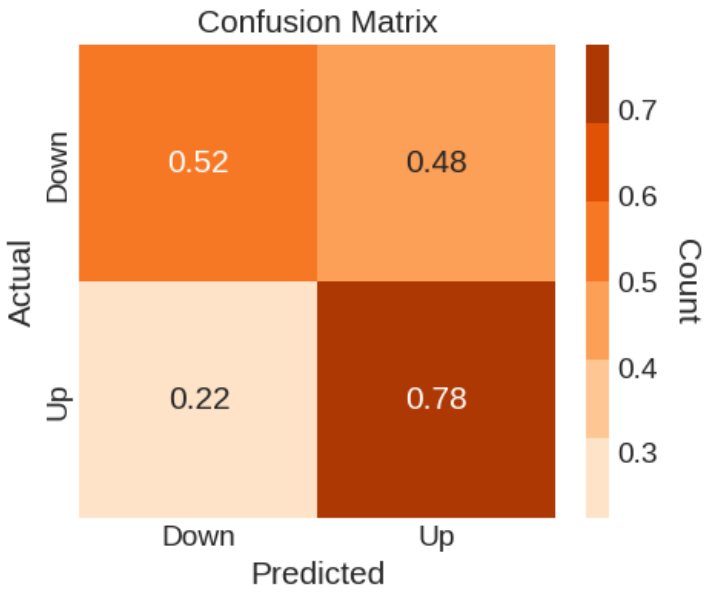
The model was trained, and the weights performing best on validation data were used. Here are some examples of its predictions (red means class 'Down' and green means 'Up') -


Closing price time series
Predicted
Actual
Price
Days


Closing price time series
Predicted
Actual
Price
Days


Closing price time series
Predicted
Actual
Price
Days

The model run on the test data gave the following results-



6.1 Evaluation

The test data had an accuracy of 0.67.

	Precision	Recall	F1 Score
class 'Up'	0.64	0.60	0.62
class 'Down'	0.69	0.72	0.71

Table 1: f1 score calculated for test daata

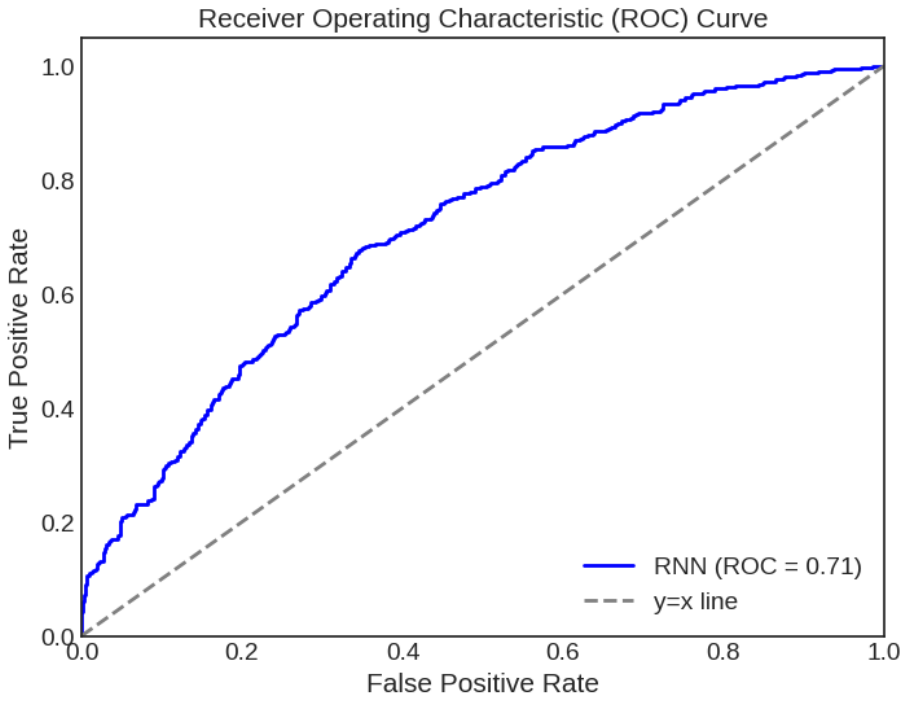


Figure 4: ROC curve for test data and AUC

6.2 Conclusions

Based on the results and evaluation metrics, the model achieved a moderate level of performance in predicting stock price movements. The precision, recall, and F1 score for both classes indicate a reasonable level of predictive accuracy. However, there is room for improvement, particularly in distinguishing class Down instances.

The results do suggest that the EFH can be rejected. The ROC curve with an AUC of 0.74 suggests that the model has some discriminative power, but further optimization and refinement may enhance its predictive capabilities. Overall, the study demonstrates the potential of deep learning models in predicting stock price movements, highlighting avenues for future research and model enhancements.

7 Project Reflection and Learnings

Throughout this project, significant time and effort were dedicated to various stages, each presenting unique challenges and valuable learning opportunities.

7.1 Data Collection

The data collection process was particularly time-consuming, primarily due to the amouunt of stock market data being scraped. Although the focus was on S&P 500 companies, there was initial consideration for including data from S&P 400 and 600 indices as well. However, resource constraints and time limitations restricted the scope to S&P 500 alone.

7.2 Model Architecture Exploration and Challenges

A significant portion of the project timeline was dedicated to exploring and testing different deep learning architectures. This included experimenting with various RNN configurations and understanding their internal workings. Another key concept learned was the necessity of rebalancing training data to address class imbalances, a crucial step in ensuring model robustness and performance.

Challenges were encountered in deciding between regression and classification approaches. The transition from regression to classification was required due to the additional complexity of the regression problem in this context and time-constraints.

Throughout the process, several interesting observations were made. Notably, the results indicate the presence of local patterns present in this time series data and refute the claim of the Efficient Market Hypothesis. Also, initially having initially attempting to train the model using the imbalanced training data led to a biased model highlighting the need for balancing.

The project provided an opportunity to deepen understanding and proficiency in using TensorFlow for developing and training machine learning models. Additionally, time was invested in exploring techniques for effective data visualization, enhancing insights into model performance and data patterns.

Overall, the project provided a comprehensive learning experience, from data collection and preprocessing challenges to model architecture exploration and insights into financial market dynamics. The journey highlighted the interdisciplinary nature of data science and its applications in addressing real-world problems in finance and beyond.

8 Future Work

While this project has laid a solid foundation for predicting stock price movements using deep learning techniques, there are several avenues for further exploration and enhancement.

8.1 Feature Engineering

Future iterations of this project could benefit from extensive feature engineering to extract more nuanced and relevant features from the dataset. By incorporating additional financial indicators, market sentiment metrics, and external economic factors, a more comprehensive and sophisticated predictive model can be

developed. Exploring techniques such as principal component analysis (PCA), feature selection algorithms, and domain-specific feature engineering can further enhance the model's predictive capabilities.

8.2 Hybrid Models and Ensemble Techniques

Investigating hybrid models that combine the strengths of multiple machine learning approaches could lead to improved prediction accuracy and robustness. Ensemble techniques, such as combining deep learning models with traditional statistical methods or incorporating reinforcement learning strategies, can be explored to leverage diverse sources of information and enhance predictive performance.

8.3 Sentiment Analysis Integration

Incorporating sentiment analysis of news reports, social media trends, and market sentiment indicators can provide valuable insights and additional predictive factors. By analyzing textual data for sentiment polarity, sentiment intensity, and event sentiment, the model can capture market sentiment shifts and incorporate them as features for more informed predictions.

8.4 Future Research Questions

This project opens the door to several intriguing research questions that can guide future work:

- Can the model be improved by incorporating additional financial indicators such as economic sentiment analysis and macroeconomic variables?
- How does the model perform across different market conditions, including bull markets, bear markets, and periods of high volatility?
- What is the impact of feature engineering on model performance, and which techniques (e.g., time series decomposition, lag features, technical indicators) are most effective?
- How does the model compare to traditional financial forecasting methods like ARIMA, GARCH, or regression models?
- Can the model be adapted for real-time stock price forecasting, and what are the challenges associated with real-time data ingestion and prediction?

References

[LKY+24] Q. Li, N. Kamaruddin, S. S. Yuhaniz, et al., *Forecasting stock prices changes using long-short term memory neural network with symbolic genetic programming*, Scientific Reports **14** (2024), 422.

[MSD21] Sidra Mehtab, Jaydip Sen, and Abhishek Dutta, *Stock price prediction using machine learning and lstm-based deep learning models*, Machine Learning and Metaheuristics Algorithms, and Applications (Singapore) (Sabu M. Thampi, Selwyn Piramuthu, Kuan-Ching Li, Stefano Berretti, Michal Wozniak, and Dhananjay Singh, eds.), Springer Singapore, 2021, pp. 88–106.

[Zha03] G.Peter Zhang, *Time series forecasting using a hybrid arima and neural network model*, Neurocomputing **50** (2003), 159–175.