Predicting Future Performance of S&P 500 Stocks Using Fundamental Analysis and Time Series Forecasting

Sanjiv Rao
Virginia Polytechnic
Institute and State University
Blacksburg, VA 24061-0002
sanjivr@vt.edu

Abstract

This project focuses on developing a machine learning model to predict the future performance of S&P 500 stocks using a hybrid approach that combines fundamental analysis and time series data. The model leverages an autoencoder for dimensionality reduction of fundamental features and an LSTM network to capture temporal dependencies in historical stock prices. The primary objective was to classify stocks into categories such as "Strong Buy," "Buy," "Hold," "Sell," and "Strong Sell" based on predicted future performance. Despite limitations in the dataset and challenges in accurately classifying certain stock categories, the model demonstrated strong performance in identifying "Strong Sell" stocks, providing potential value as a supplementary tool for decision-making in stock trading.

1. Introduction

The stock market is one of the most important financial institutions of the modern age. It facilitates the buying and selling of equity in publicly owned companies, allowing businesses to source capital for investments, and enabling individuals and institutions to own a percentage of these companies, while participating in their growth. Companies, organizations, and individuals around the world carefully consider the state of the market when making decisions. Being able to understand, analyze, and predict the movements of the stock market is a critical skill for investors, businesses, and policymakers alike. The stock market's performance is influenced by countless factors and generally reflects the overall economy, making it extremely difficult to predict how any individual stock or index will perform in the long term.

1.1. Motivation for the Project

The goal of this project is to develop a tool by which one can categorically predict how well a stock will perform over a set period of time in the future. Investors and firms alike have both attempted to generate such tools, with varying levels of success. As covered in a systematic review of artificial intelligence-based market prediction mechanisms by Lin and Marques [2], the main methods utilized in market prediction are Support Vector Machines (SVM), Long Short-Term Memory (LSTM), and Artificial Neural Networks (ANN). Time series analysis is by far the most prevalent method of these predictive models.

Despite the progression of artificial intelligence, the main obstacle facing advances in stock prediction is the efficient market hypothesis. This is a theory of financial economics that effectively states that future changes in market price cannot be predicted using historical information, as summarized by Kumbure et al. [1]. This is a product of the principle that the randomness of the market, and other external factors, are already incorporated in the price valuation of the stock. Overcoming this limitation is the main obstacle facing stock prediction. If successful, this project could be a steppingstone towards creating a larger stock prediction mechanism with applications in nearly any sector.

1.2. Background

When it comes to analyzing the stock market, two primary schools of thought tend to dominate: technical analysis and fundamental analysis. Technical analysis focuses on historical price data, volume, and other market metrics, seeking to identify trends, patterns, and statistical relationships that might predict future price movements. This is a quantifiable method by which to evaluate a stock, making it desirable for short term trading.

Fundamental analysis, on the other hand, involves evaluating a company’s intrinsic value by examining its financial health, revenue, earnings, industry position, and broader economic conditions. Investors who employ this approach study balance sheets, income statements, and cash flow to determine whether a stock is overvalued or undervalued. The aim is to invest based on the long-term growth potential of a company rather than short-term market fluctuations. With this project, the intention is to combine both techniques, with an emphasis on fundamental analysis, to predict whether a stock is projected to perform well over a medium-term period of 6 months.

The goal here is to evaluate whether or not market statistics, historical stock performance, and simple fundamentals can be used to project the performance of a stock in the future. Specifically, the project will focus on US stocks and specifically those in the S&P 500, a stock index that encapsulates the 500 most valuable companies in the USA. Since these stocks represent about 80% of US market capitalization, it will be used as a reflection of the overall market as a whole.

2. Methodology

The first step of the project was to define a clear data set by which the model could be trained on. For the two stages: fundamental and technical analysis, two different types of data were required. Fundamental analysis necessitated key financial metrics such as P/E ratio, EPS, earnings growth, etc. Technical analysis requires time series data over a decently long interval whereby the predictive model could be trained upon it. Various data sets were compared, but none possessed all the necessary values for model training. Therefore, a custom data set was needed for the specific use case of this project.

Alongside this, before data could be passed to the model it needed to be reshaped into the appropriate dimensions to be passed as an input. The two folders of fundamental and time series data were compressed and uploaded to Google Drive due to the experimentation environment being Google Colab. Upon loading the data into the main model file, it was cleaned and processed to search for inconsistencies which could affect model performance. One of the issues with the Yahoo Finance API as a data source is that oftentimes certain fields would be missing from the data. This may be an issue of the stock itself being delisted at the data collection time, or the company simply not making the information publicly available, but this resulted in holes in the dataset. To handle these cases, certain stocks were trimmed from the list or certain data points were omitted during preprocessing. Before passing the data to the model, all data used was cleaned and trimmed according to ideal expectations.

Following the data preprocessing, the data is split into training and testing sets. Alongside this, one-hot-encoding is applied to the data to interpret the categorical labels into a numerical representation that is understandable by the model. After passing through the multi-stage model, the data is validated on the testing set and experimental results are derived. A classification report with key information such as the precision, recall, f1-score, and support values are generated to provide an overview of model performance in terms of classification. Alongside this, the model loss and accuracy over the training and testing data is plotted. Through these results, how the model performs could be thoroughly analyzed. In terms of choice of technology, Tensorflow's Keras API was used for loading the sequential models used in this project.

2.1. Data Preprocessing

For data acquisition, a Python script was written that pulled various financial metrics and time series data from the Yahoo Finance API. A list of companies from the S&P 500 was generated, and then iterated over for each .csv file. Two files were generated per ticker symbol. The first was the ticker_fundamentals.csv file, which consisted of these financial metrics: trailing P/E ratio, earnings per share, the price to book ratio, return on equity, price to sales, market capital, debt to equity ratio, cash to debt ratio, and finally a label categorizing the stock. Initially, categorizing each stock manually was considered, but due to the time-consuming nature of the process, an alternative was chosen. Analyst recommendation data was available for the various stocks, so the average consensus among analysts was used to label each stock. The labels consisted of 5 variations which consisted of strongBuy, buy, hold, sell, and strongSell. Though these analyst recommendations are not the definitive best choice for a given stock, it provides a domain specific interpretation of the financial metrics that is difficult to recreate without the proper knowledge or experience. Especially considering the unpredictability of the market, this is a reasonable choice for a classification standard.

The second file generated for each stock was ticker_data.csv. This consisted of daily time series data for a stock over a 3-year period. Within this data, the date, opening price, and closing price were saved. This data will be passed as an input to the LSTM to predict future prices. The labels assigned to this data were the same classifiers as the previous section, except instead of using analyst recommendations, the price after 6 months of that given day was collected. Then, based on the percentage change in stock price, each data point was assigned a classifier.

Percent Change in Price (%)	Class Level Assigned
x >10%	Strong Buy
5% < x <= 10%	Buy
-5% < x <= 5%	Hold
-10% < x < 5%	Sell
x < -10%	Strong Sell

Figure 1. Table of boundaries used for classification of time series data points

2.2. Modeling Approach

The main design choice for this project that sets it apart from other stock prediction models is use of a hybrid model Compared to other mechanisms which may exclusively rely on time series data or financial metrics, this model employs a combination of both with a custom interpretation of the data. The model itself uses a two-stage process for combining these two approaches. First, the fundamental analysis is conducted using an autoencoder. Initially, the design approach was to use a multilayer perceptron model for the financial metrics, but an issue of dimensionality became prevalent as there was a vast difference between the dimensions of the time series data which collected 3 years' worth of data points compared to fundamental statistics which are collected once over a set financial period. Alongside this, when financial metrics were missing, it would cause issues with the modeling. To bridge this gap, an autoencoder LSTM was chosen as the best option for drawing meaningful understanding of the fundamental data and then being able to pass that as an input to the LSTM, despite a variability in length. Another reason behind this choice was the fact that fundamental analysis is applied on a company-by-company basis, making a general-based approach difficult. By incorporating an autoencoder, the encoder learns a general representation of all the stocks in the S&P 500 list, allowing for a general representation of the market behavior in reference to these metrics. Alongside this, the fundamentals data values were scaled down and weighted according to the experimenter's interpretation of relative importance between the metrics. This is where the model can be finetuned for different users, as altering these parameters affects the weights the model uses to interpret the fundamental data.

The outputs of the autoencoder are used then as inputs, alongside the time series data, for the LSTM model. Open and close prices were used to incorporate the historical data analysis alongside the fundamental understanding for prediction. A dropout layer was incorporated in both models to reduce overfitting and throw out random values. A combination of relu and softmax activation were used alongside the dense layer The loss function used in the model was categorical cross-entropy, since this is a multi-class classification problem. Alongside this, L2 regularization was included in the model with a lambda value of 0.01. Alongside this hyperparameter, others include the learning rate, dropout rate, number of epochs, and class weights. These were tuned using a combination of trial and error and domain understanding for the fundamental value weights. An Adam optimizer was included due to the size of the dataset and the noise present due to inconsistencies. This assisted the model in convergence.

Hyperparameter	Value
L2 Regularization Strength	0.01
Dropout Rate	0.2
Number of Dense Units (Encoder)	32
Number of LSTM Units	128
Learning Rate	0.001
Batch Size	32
Epochs	50
LSTM Lookback Window	30

Figure 2. Table of hyperparameters and values

An issue that became apparent during the preprocessing is the lack of variance in the labeling of the fundamental data. Due to the generally high performance of S&P 500 stocks, analysts' outlooks for these stocks are generally quite positive. A product of this issue is that the dataset that the model is being trained on is heavily biased for certain values and lacks a balanced perspective of market conditions. As demonstrated later in the results, this resulted in a model being overfitted for high performing stocks, making it a suboptimal choice for evaluating any stock outside of that performance range. This was not an issue for the time series data, as the 6-month price growth evaluation possessed a good variance of labeling, despite still demonstrating a slight bias towards high performing outlooks. This also does not factor in many other market conditions which could contribute to price drops in the short term.

3. Results

In terms of expectations, this model was not expected to be an utter success. One of the limitations listed previously was the dataset the model is trained upon. Though in design each step of the process is sensible, the model can only perform as well as the quality of data it is provided. The rudimentary nature of the dataset and the difficulty in classifying a stock for future performance resulted in a distinct difference in how well the model could evaluate specific class levels. Since there is no discrete method of evaluating whether or not to buy or sell a stock, alongside the limitless number of external factors influencing the price of a stock, simply using numerical analysis and recurrent neural networks are insufficient to fully grasp all the factors influencing how a stock will behave in the future. This is a root issue of the problem this project intends to address, and more work needs to be done in order to fully grasp the scope of the problem.

Alongside this, the requirement of needing to generate a custom dataset of this size resulted in the majority of the work for this project consisting of the data acquisition and preprocessing segments. Despite this, the model

demonstrates the capability to identify stocks that fall into strong sell and strong buy ranges, alongside a decent ability to identify stocks that should be held. In terms of the buy and sell class levels, the inherent nuance to these class levels made the model largely inaccurate in terms of classification. To conduct further experimentation, the model was passed inputs outside of the training and testing set to see if the general approach would be applicable. From the limited testing done with these data points, the same trends were observed of a bias for certain class levels.

3.1. Confusion Matrix Results

In terms of experimentation, the model was simply run on the validation data set which represented around 20% of the training set that was set aside for validation. In terms of future improvements and real-world application, the model could receive inputs in the form of specific stocks as opposed to general data. By passing the specific fundamentals for a given company, alongside historical data, the model can generate custom parameters for that one stock. The code written thus far provides a clear and customizable template to go about training the model on a better dataset.

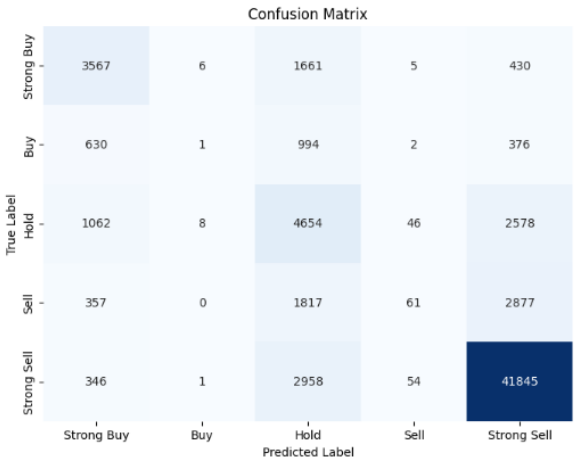

Figure 3. Confusion Matrix quantitatively demonstrating the model's classification capabilities

As demonstrated by this confusion matrix, quantitatively it can be assessed that the model performs best when classifying stocks which are a strong sell. Of the 45204 data points, the model was able to successfully classify 41845 of those correctly. In terms of a real-world application, this is useful in identifying when to sell a stock before future declines in value. This reinforces the model's usefulness as a supplement to efficient trading, and not a replacement for the decision-making process an investor or firm may make. Alongside the model's ability to identify when to sell a stock, it was also relatively competent in choosing strong buys and when to hold. Though accuracy has declined by a significant margin, with future improvements to the dataset and the weights, this could be significantly improved.

3.2. Classification Results

Class	Precision	Recall	F1-Score	Support
Strong Buy	0.60	0.63	0.61	5669
Buy	0.06	0.00	0.00	2003
Hold	0.39	0.56	0.46	8348
Sell	0.36	0.01	0.02	5112
Strong Sell	0.87	0.93	0.90	45204

Figure 4. Classification report for the validation data

Overall Metrics	Precision	Recall	F1-Score
Macro Average	0.46	0.42	0.40
Weighted Average	0.72	0.76	0.72
Accuracy	0.76		

Figure 5. Overall metrics and averages for classification results

The strong sell class had a high precision, recall, and F1-score, making it a valuable component of the model for real-world applications where timely selling decisions are crucial. However, the buy and sell categories exhibited poor recall, precision, and F1-scores, indicating that the model struggles to distinguish stocks that fit into these categories. The hold class had a more balanced performance, with a reasonably good recall rate, though still falling short of perfect accuracy. As demonstrated by the support values, the data set used for training and validation were very biased for certain classes.

3.3. Loss and Accuracy

Alongside these quantitative evaluations of model performance, included in the results are graphs plotting the training and validation loss as well as the training and validation accuracy over the course of the training process. These graphs provide valuable insight into how well the model is learning and generalizing over time, as well as identifying trends which can indicate the presence of overfitting within the data.

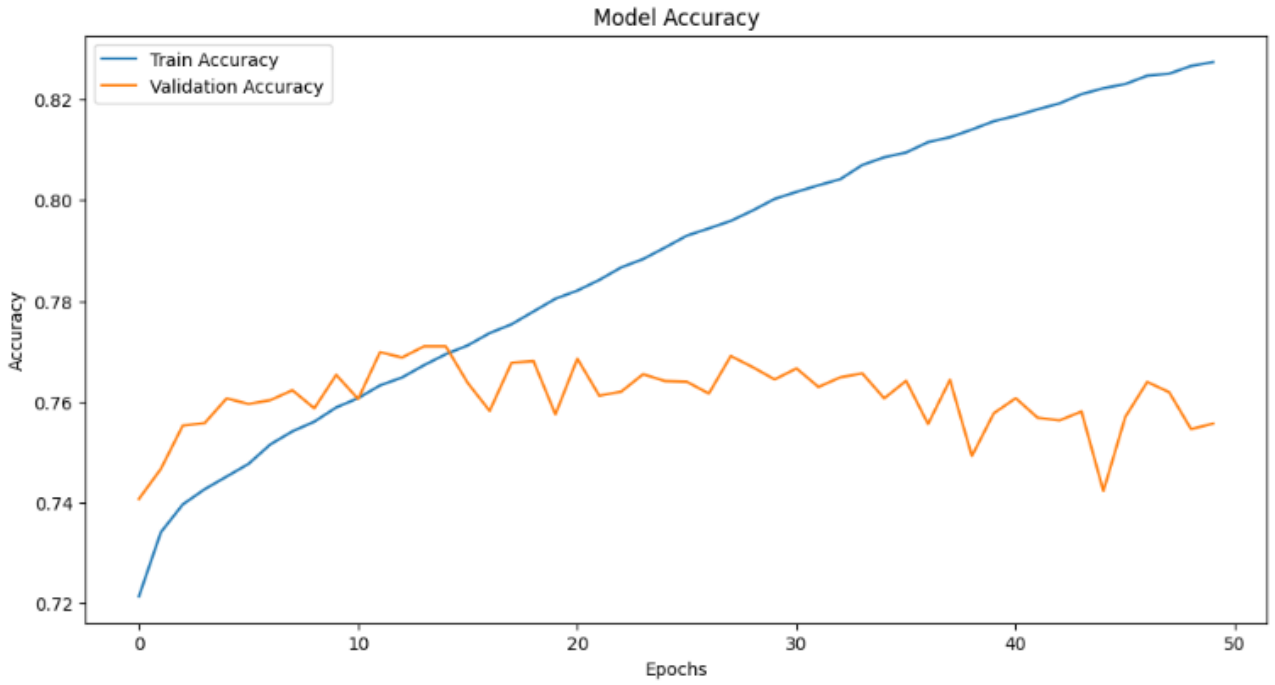

Figure 6. Model Accuracy for Training and Validation in terms of Epochs

The accuracy graph shows a steady increase in training accuracy, with the validation accuracy following a similar trend. This indicates that the model was successfully learning to classify the data, though the slight gap between training and validation accuracy suggests some overfitting. The training accuracy continually improves, but the validation accuracy plateaus at a slightly lower value, reinforcing the observation that the model is better suited to the training data compared to the unseen validation data.

Similarly, the loss graph illustrates a steady decline in the training data, but loss trends upwards at around 15 epochs for the validation data. This once again exhibits the fact that the model performance is less than ideal and requires more finetuning to be applied. The training loss decreases more sharply than the validation loss, suggesting that the model fits the training data very well. However, the validation loss decreases at a slower rate, indicating that while the model is improving, it might not be generalizing perfectly to unseen data.

4. Future Implementation

Future implementations should remedy the overfitting and data set quality by not only providing a larger and more vast data set for model training, but also using real time metrics for price and fundamentals as inputs to the system. This could be evaluated over a 1–2-month period to evaluate how accurate the model can be in real time

Along with these improvements, the model could benefit from further finetuning as well as incorporating the knowledge and understanding of a domain professional into the designation of the weight values for the autoencoder. Future users can reproduce the results by following this methodology alongside the guidelines listed in the codebase. The naming terminology in the codebase is as easy to understand as possible. The model parameters are also included clearly.

4.1. Availability and Reproducibility

The code for this project is available at the GitHub repository link in the following section. All software used for this project is open source, and the repository itself is licensed by the GNU General Public License. Alongside this, the data files and the scripts used for data acquisition are included in the repository, with full access and free to use for the public. This code incorporates some ideas and code from a Kaggle user who released their script for financial data acquisition freely on the internet. The script was adapted and updated for modern day packages and the specific use case of this project. It provided a starting point for the project, but the majority of the script for data acquisition is original.

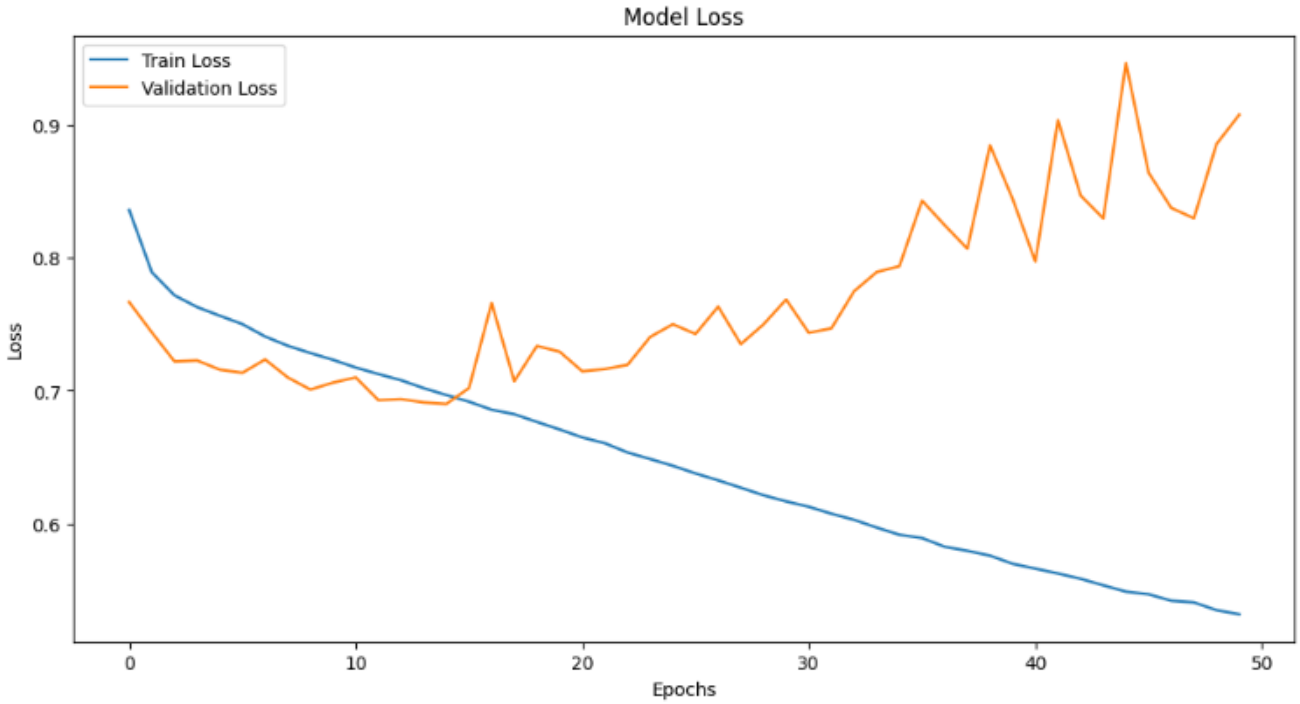

Figure 7. Model Loss for Training and Validation in terms of Epochs

Links

https://github.com/sanjivsrao/Autoencoder-LSTM-Stock-Performance-Predictor

https://github.com/CNuge/kaggle-code

References

[1] Kumbure, M. M., Lohrmann, C., Luukka, P., & Porras, J. (2022). Machine learning techniques and data for stock market forecasting: A literature review. Expert Systems with Applications, 197, 116659. https://doi.org/10.1016/j.eswa.2022.116659

[2] Lin, C. Y., & Lobo Marques, J. A. (2024). Stock market prediction using artificial intelligence: A systematic review of systematic reviews. Social Sciences & Humanities Open, 9,100864. https://doi.org/10.1016/j.ssaho.2024.100864

[3] Mintarya, L. N., Halim, J. N. M., Angie, C., Achmad, S., & Kurniawan, A. (2023). Machine learning approaches in stock market prediction: A systematic literature review. Procedia Computer Science, 216, 96–102. https://doi.org/10.1016/j.procs.2022.12.115

[4] Strader, T. J., Rozycki, J. J., ROOT, T. H., & Huang, Y.-H. J. (2020). Machine Learning Stock Market Prediction Studies: Review and Research Directions. Journal of International Technology and Information Management, 28(4), 63–83. https://doi.org/10.58729/1941-6679.1435

[5] Wafi, Ahmed. S., Hassan, H., & Mabrouk, A. (2015). Fundamental analysis models in financial markets – review study. Procedia Economics and Finance, 30, 939–947. https://doi.org/10.1016/s2212-5671(15)01344-1