STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **9**, December 2021, pp 769–788.
Published online in International Academic Press (www.IAPress.org)

S&P 500 Stock Price Prediction Using Technical, Fundamental and Text Data

Shan Zhong *, David B. Hitchcock

Department of Statistics, University of South Carolina, USA

Abstract We summarized both common and novel predictive models used for stock price prediction and combined them with technical indices, fundamental characteristics and text-based sentiment data to predict S&P stock prices. A 66.18% accuracy in S&P 500 index directional prediction and 62.09% accuracy in individual stock directional prediction was achieved by combining different machine learning models such as Random Forest and LSTM together into state-of-the-art ensemble models. The data we use contains weekly historical prices, finance reports, and text information from news items associated with 518 different common stocks issued by current and former S&P 500 large-cap companies, from January 1, 2000 to December 31, 2019. Our study's innovation includes utilizing deep language models to categorize and infer financial news item sentiment; fusing different models containing different combinations of variables and stocks to jointly make predictions; and overcoming the insufficient data problem for machine learning models in time series by using data across different stocks.

Keywords Stacking, LSTM, Random Forest, Text Sentiment

AMS 2010 subject classifications 62P05, 68U15

DOI: 10.19139/soic-2310-5070-1362

1. Introduction and background knowledge

Modeling and predicting future stock prices have long been important but difficult problems for researchers [1][2]. Accurate prediction of future stock prices provides a foundation for the risk analysis and investment decision making that are necessary in stock trading. On the other hand, recent studies [3][4] have shown stock markets are predictable with more than 50 percent accuracy and have exceeded the market growth rate, which is a breakthrough that may potentially belie the statement in the Efficient Market Hypothesis (EMH)[5] that stock prices are unpredictable. Then the questions remaining are how much accuracy can we achieve with state-of-the-art models, and how can we integrate these new predictive models together with different kinds of data. Surveys [6][7] of recent studies showed that many new methods have been proposed for prediction of stock prices, but few approaches have combined and integrated them together into one composite system efficiently, from a statistical point of view.

There are two popular investment strategies. One is trend following: The trend following strategies focus more on the trend movements of individual stocks and markets. The most basic trend following strategy assumes a time series with a type of momentum that suggests that people buy those stocks with recent positive returns and sell those stocks with recent negative returns [8]. The other popular investment strategy is value investing: Value investing focuses more on the intrinsic value of the companies and market. One value investing example would be buying those stocks with a high book-to-market ratio, which are the companies with high equity (company asset minus

*Correspondence to: Shan Zhong (Email: zhongs@email.sc.edu). Department of Statistics, University of South Carolina. Columbia, SC, United States (29208).

ISSN 2310-5070 (online) ISSN 2311-004X (print)

debt) to market value ratio [9]. In our study, we combined the strengths of the ideas and the data associated with these two investment strategies, integrating them into one model.

Traditionally according to the EMH, stock prices reflect all past information, since new information is unpredictable and the stock prices will be adjusted immediately after information is made public, implying that the stock price follows a random walk [10]. Thus the stock prices themselves and the trading volume are enough to predict future price movements. Based on assuming the stock price follows a random walk, or more generally, a stochastic process, analysts usually use traditional time series modeling techniques such as the Autoregressive Integrated Moving Average (ARIMA) model to help predict stock price. The ARIMA model has its strength in the robustness and efficiency in terms of short-run forecasting [11][12][13]. However, most ARIMA models are univariate, and the data involved in these ARIMA models are usually only the historical time series stock price itself [14][11]. For example, if we want to do daily forecasting, we lose the daily high, low, open, close and volume information. That means that we are unable to utilize the technical indicators which are derived directly from this stock price information that many traders use to predict the trend from a supply-demand perspective. At the same time, with machine learning models becoming popular, some researchers utilize the flexibility of machine learning algorithms to incorporate technical indicators into their model. For example, one can generate a list of related technical indicators to feed into a Random Forest (RF) or a Long Short-Term Memory (LSTM) model for directional prediction [15][16]. Nevertheless, when using these machine learning models which need more parameters to fit, usually there is not enough data available, as the available data is limited based on the chosen time period. Our model resolves this issue by the following techniques: carefully selecting the corresponding predictor variables to control the number of input variables; training the machine learning algorithm across different stocks simultaneously then fine-tuning on each individual stock; creating an ensemble of models trained with different groups of stocks; and allowing multiple input and output periods for the LSTM model for more recursive training.

On the other hand, since the EMH was introduced, some researchers [17][18][3] have disputed the notion and suggested that the market is inefficient and the stock prices actually do not follow random walks exactly, but rather approximate random walks. It is possible that a stock is mispriced by the market in the short run but reverts to its "correct" price eventually. Therefore they look at fundamental characteristics such as the price-earning ratio to check whether a stock is underpriced. There is no standard or commonly accepted baseline about what model or variables we should use for a value-investing approach. Researchers use a variety of different variables such as cash flow, interest rates, etc., in regression or empirical models [19][20][21]. In our study, to predict the intrinsic value of stock prices, we utilized some ratios suggested by both empirical rules and from the quarterly reports that public companies must produce based on generally accepted accounting principles (GAAP) and included them into our ensemble model.

Besides numerical information, since 2000, the growth of the internet has greatly sped up the spread of textual information. However, for large sets of stock price data, little research has incorporated jointly into one model text data related to news items together with other factors that may affect future stock prices. In our dataset which contains over 500 stocks and 3.2 million news items, we link the stocks to the related entities mentioned in the news. The method we propose is to first recognize the news as related to each company involved, and then calculate a sentiment score of the news which can measure the negative or positive impact on the stock. As these steps can be done independently for each article, this can be accomplished efficiently. We used the Longest Common Subsequence [22] distance metric and word embedding [23] techniques to find the entities related to each stock that were mentioned in the news articles. Then a pretrained Bidirectional Encoder Representations from Transformers (BERT)[24] model with finance domain knowledge [25] was used to calculate the sentiment of each article.

2. Literature review

There are several different studies with the goal of predicting the S&P 500 index and prices of stocks in the S&P 500 index, as summarized in Table 1. Jiang et al.[16] utilized an ensemble stacking with logistic regression to predict stock price direction after one month from 2012 to 2019. They used a list of 24 technical and macroeconomic variables together with eight different models to do weighted shrinkage. Yu and Yan[26] used a Long Short-Term

Memory (LSTM) model on daily S&P closing prices along with wavelet smoothing to predict the next day's closing prices. They reconstructed the LSTM series to allow for multiple outputs to help train the model. Both Jiang et al. and Yu et al. utilized the LSTM model in their research, however, the effectiveness of their LSTM model was limited by the insufficient number of observations: 4,054 and 2,518 observations compared with over 300,000 and 12,000 parameters needed to fit. Ding et al.[27] utilized text embedding with convolution to predict daily S&P 500 closing for a 10-month period in 2013. Their research showed the effect of news information gradually weakened over time, and that using news to predict daily data performs better than for weekly or monthly data, although news still has an effect in models for weekly and monthly data. Gorenc Novak and Velušček[32] used technical indicators with a linear Support Vector Machine (SVM) to predict the daily high for 370 S&P 500 companies from 2004 to 2013. They chose a 500-day rolling window period to train their model as a compromise between a smaller stationary set and a larger but less stationary set. They also did a grid search for the best model refresh period with options for 5, 10, 20, 40, 80 and 160 days, with 20 days yielding the highest accuracy. Across these distinct approaches, differences in the choice of period, preprocessing steps, and dependent variables make comparisons between different methods hard.

Table 1. Summary of different methods

Author	Period	Variables Used	Method	Directional Accuracy	Predicting
Jiang et al.[16]	2012-2019	Technical and Macroeconomic indicators	Ensemble stacking	69.17%	One month forward direction of S&P 500 closing
Yu and Yan[26]	2010-2017	Stock Price itself	LSTM	58.07%	Daily S&P 500 closing with wavelet smoothing
Ding et al.[27]	2013	Text News	Word Embedding with convolution	64.21%	Daily S&P 500 closing
Gorenc Novak and Velušček[32]	2004-2013	Technical indicators	Linear SVM	61.16%	Daily high for 370 S&P 500 companies

One important aspect of our research is the use in our model of text data from news items. Ding et al.[27] categorized news into long-term, mid-term, and short-term events based on events that happened one month, one week, and one day ago, respectively. Then they reduced the news item into action and actors, converted the text into word embedding, and fed them into convolution neural networks for direct inference about upward or downward movements. Akita et al.[29] used textual information from news headlines to attain inference about future stock prices. They first converted news headlines to a numerical vector representation. Then they concatenated historical price information and news headline vectors for correlated groups of companies. The combined vectors were fed into one LSTM layer with 20 nodes. The predicted price for these correlated companies were then attained by regression on these 20 nodes. Bollen et al.[4] extracted public sentiment from social media texts, without directly inferring the meaning of these texts, to analyze public mood. They showed sentiment can help explain investor behavior and eventually help to predict the Dow Jones Industrial Average (DJIA). Following these research ideas, we also utilized the predictive power of textual information and combined it with numerical data in our hybrid model. Distinctions between our approach and others' include that Ding et al. and Akita et al. used short phrases to fit word embedding models to get inference from text, while we used the BERT[24] model to get accurate inference from long paragraphs of words, and we incorporated other technical and fundamental variables into the model, whereas Bollen et al. and Ding et al. only used news as inputs.

3. Data Selection and Preprocessing

3.1. Stock Price Data

To limit the complexity of our data set, we first focused on the 505 common stocks issued by 500 large-cap companies that traded on American stock exchanges. The S&P 500 index, which measures the performance of these companies, is one of the most commonly used indices to represent the U.S. stock market. However, the S&P

500 index is a dynamic composition of the 500 currently most valuable companies. If we just analyze the 505 stocks which are currently on the list, we may then be biased in a sense that we already selected "winners", so we also included the historical S&P 500 companies to mimic a realistic environment. In addition, to ensure the accuracy and reliability of the stock price data, we collected stock data from two different sources on the same dates, namely Yahoo Finance[30] and Alpha Vantage[31], and controlled the adjusted closing prices across the two sources to within 2% error. The remaining stocks after data cleaning include 451 current and 67 past S&P 500 companies, due to the cases of acquisition, split-up, privatization, bankruptcy, and missing values, etc. The detailed data cleaning steps are in the appendix. For the preprocessing after the data cleaning, we converted the adjusted closing prices into a log return format, which is the difference in the weekly average log adjusted closing price. The relation between the weekly average log adjusted closing price $log(p_t)$ and the log scale return R_t is defined as:

$$R_t = log(p_t) - log(p_{t-1})$$

The log difference format allows easier comparisons among different stocks, and helps these time series to appear stationary. The final processed data contained stock information from January 1, 2000 to December 31, 2019, with the starting day for different stocks varying, as shown in Table 2.

Table 2. A summary for the weekly return data for the composite S&P 500 index, as well as 518 individual S&P 500 stocks, from January 1, 2000 to December 31, 2019.

	S&P 500 index	S&P 500 stocks
# of observations	1,043	427,255
percentage increasing	58.19%	55.03%

3.2. Technical Indicators and Finance Report Data

Table 3. A summary for the technical indicators we used in the model. These indicators are all calculated based on the daily high, low, open, close, dividend, and volume information.

Indicator Name	Quick Introduction	Window period
CCI	The Commodity Channel Index can help to identify price reversals, price extremes and trend strength.	20 days
MACDH	The Moving Average Convergence Divergence Histogram is the difference between MACD and the MACD signals.	12, 26, and 9 days
RSI	The Relative Strength Index, which measures the average of recent upward movement versus the upward and downward movements combined, in a percentage form.	14 days
KDJ	The Stochastic Oscillator measures where the close is relative to the low and high.	14, 3, and 3 days
WR	The Williams %R index is another index for buy signals by measuring the current price in relation to the past N periods.	14 days
ATR	The Average True Range provides an indicator for the volatility of price. We converted ATR into percentages.	14 days
CMF	The Chaikin Money Flow provides an indicator related to the trading volume.	20 days

Besides the daily adjusted closing price variable, which we used to calculate our dependent variable, return, we extracted the daily high, low, open, close, dividend, and volume information in each trading day for the cleaned data of the 518 current and past S&P 500 stocks. We transformed them into composite technical indicators that are calculated based on these variables. After careful research, we added several indices to measure the trend, momentum, volatility and volume of each stock. As Table 3 shows: The CCI and MACD measure the trend, while RSI, KDJ and WR measure the momentum, the ATR measures the volatility, and the CMF measures the volume of the stock. To remove extreme values, we capped each variable after the preprocessing steps we introduced in later sections. Finally, we took the median of each week as the variable value for our weekly data.

Table 4. A summary for the financial report variables we used in the model. These variables are calculated using the most recent stock price and quarterly report.

Variable	Quick Introduction
PE	The price to earning ratio is calculated as the share price divided by the earnings per share.
PB	The price to book ratio is calculated as the share price divided by the book value per share.
PS	The price to sales ratio is calculated as the share price divided by the revenues per share.

In addition to the stock price information, financial accounting information such as Revenue, Gross Margin, Assets and Liability for public companies can be found at the 10-Q quarterly report from the U.S. Securities and Exchange Commission. However, for convenience, we chose to use the financial reports data that Yahoo Finance collects. Also, because the collected financial reports dataset included several hundred variables, as an alternative, we instead utilized three commonly used valuation measurement indices to help to predict the return from the earning, booking value, and revenue perspective, as shown in Table 4. As with the technical indicators, we capped each variable after preprocessing, and took the median value of each week as the variable value for our weekly data.

3.3. Text News Data

Table 5. An example of NYT text data after selecting the variables for display and modifying the structure

Snippet	"Chris Cox, who quit Facebook last year after differences with the company's chief executive, Mark Zuckerberg, is returning as chief product officer."
Web url	"https://www.nytimes.com/2020/06/11/technology/facebook-chris-cox.html"
Lead paragraph	"SAN FRANCISCO — Facebook said on Thursday that Chris Cox, a former top executive, was returning to the company as chief product officer."
Headline	{main: "Facebook Brings Back a Former Top Lieutenant to Zuckerberg", sub: None }
Keywords	[{name: "subject", value: "Social Media", rank: 1 }, {name: "subject", value: "Mobile Applications", rank: 2 }, {name: "subject", value: "Appointments and Executive Changes", rank: 3 }, {name: "organizations", value: "Facebook Inc", rank: 4 }, {name: "persons", value: "Cox, Chris (1982-)", rank: 5 }, {name: "persons", value: "Zuckerberg, Mark E", rank: 6 }]
Other variables	{document type: "article", type of material: "News" , news desk : "Business", section name: "Technology", word count: 370, publish date: "2020-06-11T19:16:33+0000" }

Apart from the numerical data, one important feature of our study is the utilization of text data pertaining to news items. There are several newspaper websites such as Financial Times and Thomson Reuters that provide Application Programming Interface (API) access to their database. We chose the New York Times (NYT) API, as for now it provides free access to all historical data, and the data are carefully annotated. The NYT data include the title, the author, the publication time, the type of article, the annotated related organization and person, and the abstract and first paragraph of the article. We selected the period from January 2000 to December 2019, which included 3.2 million collected articles, to align with our numerical datasets. In Table 5, we chose several key variables to show an example of the structure of the text data. Also, Figure 1 shows the NYT article monthly inflow by category for the 20-year period.

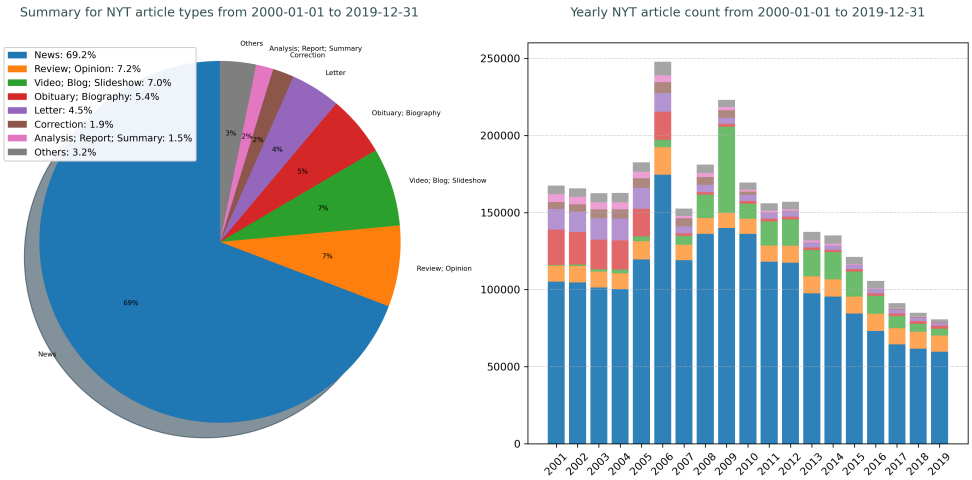

Figure 1. Summary for NYT news data from January, 2000 to December, 2019.

To link these news articles related to each specific stock to the rest of our data, we used the keywords annotated by the NYT database along with text processing techniques to classify which stocks an article refers to. Note that we do not have a direct matching from the S&P 500 stock tickers to the related keywords mentioned. Furthermore, the company names and the NYT keywords are not in a standard form, having many different expressions for the same thing, while the same entities have different expressions in terms of grammar and semantics. Thus we established a connection between them by first normalizing the NYT keywords and the tickers' company names, then matching them by calculating first edit distance similarity and next cosine similarity of word embeddings.

The edit distance is measured by the minimum number of operations needed to convert one string to another. In our preprocessing of strings, we made all characters lowercase (except for capitalizing the first letter of each word), cleaned all symbols and punctuations, and tightened the space between words to exactly one space. Then we set up decision rules using regular expressions to substitute one common word for words of similar meaning. For example, we replaced words like "corporation", "incorporated", and "company" with the "inc" symbol. We included as many replacement rules as possible to account for special cases and common synonyms. We then used the Longest Common Subsequence [22] metric, which only allows for insertion and deletion operations, and outputs the length of the longest common subsequence after insertion and deletion, to calculate the similarity between the ticker company names and the NYT organization keywords. We scaled the LCS length into percentages relative to the original string lengths for comparison. The portion with highest similarity was then sent to the word embedding steps, as embedding distance calculation takes more computation time.

For calculating word embedding [23] similarities, we chose to use a pre-trained 128-dimension token-level word embedding with a length-979,661 dictionary, trained on the English Google News 200 Billion corpus, provided by Google. We removed punctuation and split and tightened spaces, following the preprocessing standard of the pre-trained embedding model. For mapping strings into categories in the dictionary, we first checked whether the original string was in the dictionary, then the capitalized word, and next the lowercase word version. Finally for those remaining words, we separated them into pieces that were in the dictionary, by choosing the split with the least amount of splitting. For those words with the same amount of splits, we choose the one that has the lowest standard deviation of the lengths of separated pieces. As all 26 basic letters are in the dictionary, this ensured a mapping with every word in the dictionary. We did not replace words with regular expressions, as the word embedding itself had already learned the semantic similarity. Hereafter, the word level embedding was combined using the Sqrt N metric, which sums the token vectors and then divides by the square root of the number of tokens, to get the embedding for a short phrase and calculate cosine similarity.

We generated a list of the highest word embedding matches for each ticker for manual checking and were able to find the linked entities for the ticker companies accurately. A total of 170,779 related news items were obtained for analysis using our method, out of the 3.2 million articles. As expected, the news reports were heavily inclined towards the most renowned companies. Out of the S&P 500 stocks, there were only 163 that received more than 100 article reports during the 20-year period. Out of these 163 stocks, 36 received more than 1000, as shown in Figure 2.

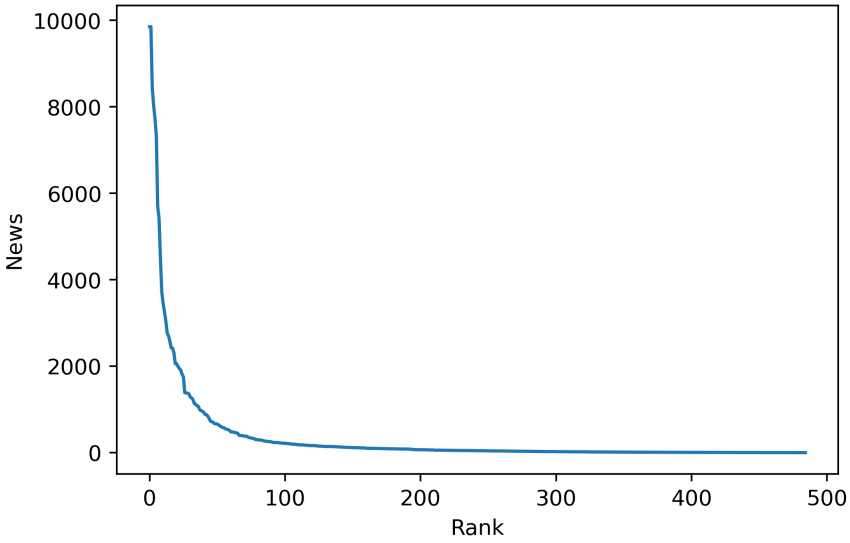

Figure 2. NYT news count, sorted by the number of news items for each company, from January 2000 to December 2019.

3.4. Sentiment Extraction for Text Data

With the link between tickers and NYT keyword organizations established, we calculated the sentiment of those related articles. The original BERT-BASE model[24] was obtained by unsupervised learning from a 800 million-word BookCorpus and the 2,500 million-word English Wikipedia dataset, with 12 encoder layers and a hidden size of 768. This BERT-BASE model is already trained by Google and is available publicly. The BERT model with finance domain knowledge[25] was then obtained by further unsupervised tuning of this BERT-BASE model using the Reuters TRC2 finance 29-million-word dataset. Finally, we used labeled finance data from Financial PhraseBank[33], which Araci[25] used to test the performance of their model, to train the BERT model to decide the sentiment of each article in the NYT dataset. The Financial PhraseBank data contains 4,840 sentences that were annotated by 16 people with finance backgrounds to judge positive, neutral, or negative tones. We considered subsets of the data for which 100% and over 66%, respectively, of people agreed in the judgements, and separated each subset into a training set of 80% of the observations and a test set of the other 20%. Using data for which the judgments of all 16 people agreed produced a higher accuracy of 0.97, but we chose to use the more extensive data for which more than 66% of the 16 people agreed in their judgments which yielded a 0.89 accuracy. As we used the fitted probability of positive opinion minus the probability of negative opinion as the sentiment score, a wider variety of opinions in human judgments would help us in estimating the sentiment.

With the model decided, we used the materials from the NYT news database to infer the sentiment. The NYT data contains one main headline, one print headline, a snippet and the first paragraph for each article. We combined them into a big paragraph for the analysis. The combined texts have an average length of 74.48 words with standard deviation of 31.06, with 94.6% of them being less than 128 words. We set the maximum input length to 128 words, so that sentences longer than 128 words would only have the first 128 words as input, while sentences shorter than 128 words would be padded to 128 words. In our preprocessing, we chose the "bert-base-uncased" tokenizer to

clean the text. We selected several samples to manually test the performance of the BERT model prediction, as Table 6 shows. Of the 170,779 extracted articles, 23,469 were positive, 37,426 were negative, and 109,884 were neutral. Again, we took the median sentiment each week for each company (with weeks ending on Fridays) as the sentiment variables in our weekly data.

Table 6. Examples for the BERT model judgment for NYT data.

Content concatenated from title, snippet, and part of first paragraph	Predicted Class Probabilities	sentiment
"MARRIOTT TO HALT 35 SENIOR-LIVING COMMUNITIES Marriott International to cancel up to 35 senior-living communities that are in early stages of developing, citing what it terms 'supply pressures in some markets' (S) Citing what it termed "supply pressures in some markets," Marriott International said yesterday that it would cancel up to 35 senior-living communities that are in early stages of development. The Washington-based hotel company, which currently operates 139 senior-living centers and is building an additional 15, also said slow sales at newly opened centers were expected to reduce earnings by 5 cents a share in the fourth quarter. The projects to be canceled represent about half of the senior-living projects in the company's development pipeline, Marriott said."	[0.0213, positive 0.9707, negative 0.008, neutral]	-0.9494
"Technology Briefing — Hardware: Advanced Micro Ships Faster Chips Advanced Micro Devicees begins selling faster computing Advanced Micro Devices, Intel's top rival in the market for personal computer processors, began selling faster models of its chips. The new versions of Advanced Micro's flagship Athlon chip run at 1.4 gigahertz, while its Durons, for less expensive home PC's, hit 950 megahertz, John Crank, its marketing manager, said. Compaq Computer and Gateway will use the Athlon chips in systems available immediately, he said."	[0.8819, positive 0.0226, negative 0.0955, neutral]	0.8593
"General Motors Chief to Hand Over Reins to Company's President General Motors Corp chairman-chief executive John F Smith Jr says he will hand his chief executive title and day-to-day responsibility for running company to G Richard Wagoner Jr, president and chief operating officer, starting in June; Smith will... John F. Smith Jr., the chairman and chief executive of General Motors announced today that he would hand his chief executive title and day-to-day responsibility for running the company, the world's largest automaker, to G. Richard Wagoner Jr., 46, the president and chief operating officer, starting in June."	[0.0136, positive 0.0025, negative 0.9839, neutral]	0.011

4. Preprocessing, Evaluation Metric and Separation of Training and Test Data

The training and test data were separated as follows: We used three years of information as training data. After two years of training data were input, the individual model predicts future observations iteratively, calculates the prediction errors and updates the model at the end of each year based on these errors. Also starting from the third year, the ensemble models were fitted using past years' prediction outputs as inputs to the ensemble model. In this way, we tracked the performance of different models at various regions of the time domain. As a preprocessing step, all predictor variables were transformed to standard Gaussian distributions via Yeo-Johnson transformations [34] on the training data, and we filled missing values with 0. In the test set, the absolute values of these preprocessed predictor variables were capped at 4.5 to prevent undue influence by extreme values. The dependent variable was still the return at time $t+1$. Also, we used several metrics to measure the performance based on the true return R and the predicted return $\widehat{R}$:

Direction prediction accuracy (DA) at time t for the past n periods:

$$DA_{n,t} = \frac{1}{n}\sum_{i=0}^{n-1} \mathbb{1}_{[\mathbb{1}_{[R_{t-i}\geq 0]}=\mathbb{1}_{[\widehat{R_{t-i}}\geq 0]}]}$$

Upward direction prediction accuracy (UDA) at time t for the past n periods:

$$UDA_{n,t} = \begin{cases} 1 & \text{, if } \sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} \geq 0]} = 0 \\ \frac{\sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} \geq 0]} \cdot \mathbb{1}_{[\widehat{R_{t-i}} \geq 0]}}{\sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} \geq 0]}} & \text{, otherwise} \end{cases}$$

Downward direction prediction accuracy (UDA) at time t for the past n periods:

$$DDA_{n,t} = \begin{cases} 1 & \text{, if } \sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} < 0]} = 0 \\ \frac{\sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} < 0]} \cdot \mathbb{1}_{[\widehat{R_{t-i}} < 0]}}{\sum_{i=0}^{n-1} \mathbb{1}_{[R_{t-i} < 0]}} & \text{, otherwise} \end{cases}$$

Root Mean square error (RMSE) at time t for the past n periods:

$$RMSE_{n,t} = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (R_{t-i} - \widehat{R_{t-i}})^2}$$

Mean absolute error (MAE) at time t for the past n periods:

$$MAE_{n,t} = \frac{1}{n} \sum_{i=0}^{n-1} |R_{t-i} - \widehat{R_{t-i}}|$$

For each individual stock, for all stocks combined, and for the S&P 500 index, we calculated these metrics for each year, and for all 20 years of data.

5. Interpretation of Variables

Before performing predictions for each individual company, we tested the existence of linear relationships among different variables and the dependent variable (return at time $t+1$) for the 427,255 observations from 518 stocks. We explored these relationships across different companies. Figure 3 shows scatterplots of the dependent variable return at $t+1$ against various preprocessed predictor variables at time t, from January 1, 2000, to December 31, 2019. From the graphs, possible linear relationships were observed for these variables: return$_t$, sentiment, CCI, RSI, KDJ, WR, MACDH and CMF. The relationships between the dependent variable and the variables PE, PS, PB, ATR do not appear linear.

Also, due to the sparsity of news items, 392,333 of the 427,255 of the observed variable pairs have missing values on the variable sentiment score. To check the relation between return and sentiment, we excluded missing values instead of filling them by 0, and plotted the sentiment versus the return at $t+1$ for each year, as Figure 4 shows. In general, we expect a positive news sentiment at time t produces a higher return at $t+1$, and the 18 positive slopes out of 20 aligned with our expectation. However, we observe a decreasing trend of the slope parameter over time, which might indicate that as the spread of information has become faster during the 20-year period, the effect of news on next week's stock price has become weaker, as Figure 5 shows. Note the weekly news data were delimited by each Friday, recognizing the fact that stocks do not trade on weekends.

For fundamental variables PE, PS and PB, we found that beginning in the year 2000, there was a linear trend showing a lower PE, PS, or PB ratio corresponding to a higher return at $t+1$. However, that linear trend diminished starting in 2003 and became negligible after 2007. We investigated the relationship between PE, PS, and PB and return at $t+1$ among different sectors, for each individual company. Figure 6 shows the fitted regression lines for each individual company for return at $t+1$ against PS, separate by the 11 industry sectors. Out of the 515 S&P 500 companies (3 excluded due to missing values), 86.6% have a negative slope, which aligns with our intuition that a lower price to sales ratio produces a higher potential return. When only considering year 2007 and hereafter, this percentage drops to 81.6%. When fitting regression lines of the return at $t+1$ against PE and PB, respectively, 67.7% and 78.1% of the slopes are negative, with this percentage varying across sectors.

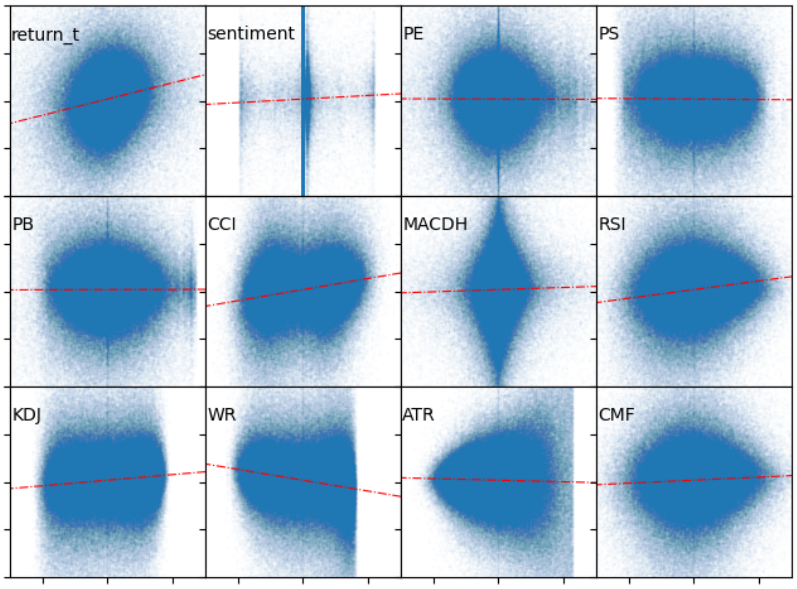

Figure 3. Relation of each predictor variable at t with the dependent variable return at $t+1$, from January 1, 2000, to December 31, 2019, where these variables were transformed to standard Gaussian and missing values filled by 0. These graphs share a x-axis range from -3 to 3 and y-axis range of -10% to 10%.

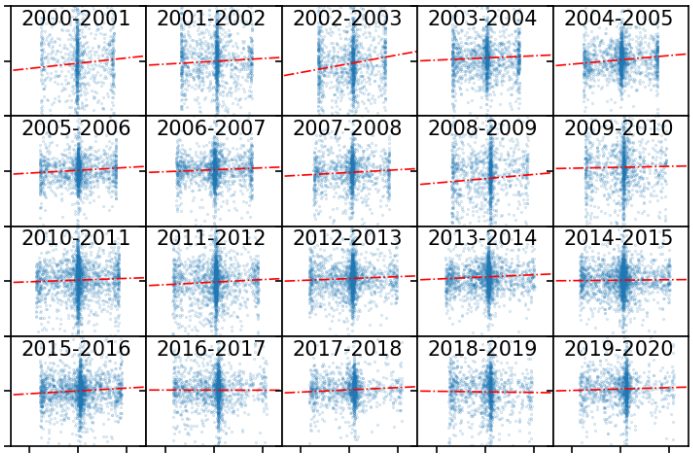

Figure 4. Scatter plot of sentiment at t on x-axis versus the dependent variable return at $t+1$ on y-axis, with the dashed line representing the simple fitted regression line for the two variables. These plots were generated every year, from January 1, 2000, to December 31, 2019, with missing values excluded. Out of the 20 years, 18 have a positive slope.

6. Applied Models

Our study of these technical, fundamental, and text-based sentiment variables revealed that different types of variables should enter our model in different forms. Because information on sentiment was sparse, sentiment

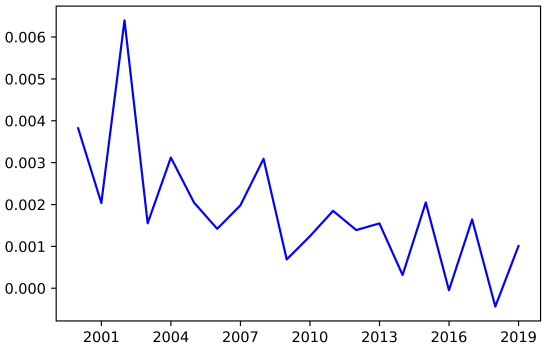

Figure 5. The parameter values of the 20 simple linear regressions fitted for each year sentiment at t versus the dependent variable return at $t+1$. Out of the 20 years, 18 have a positive slope. However, a decreasing trend in slope over time was also observed.

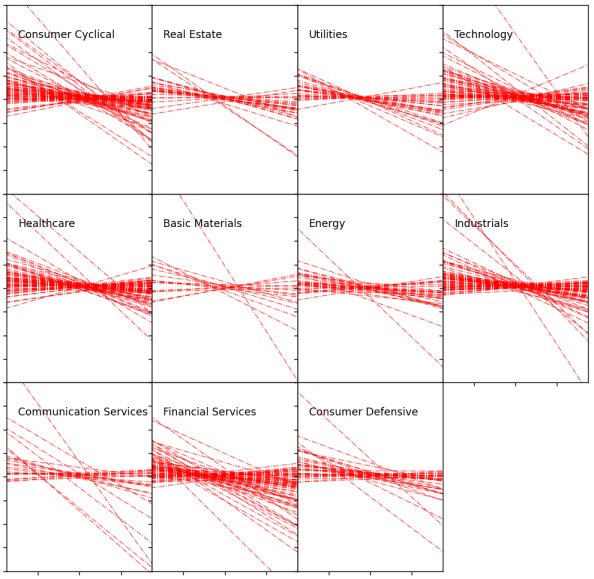

Figure 6. The fitted regression lines for each individual company for return at $t+1$ against PS, separate by the 11 industry sectors.

variables were best utilized in models using combined stock data. We included variables PE, PS, and PB only in models that did not assume a linear functional form. Also, although linear trends were detected with the other variables, the relationships are not totally linear, and variables like ATR have more to do with the volatility than mean trends in stock prices. To solve these somewhat conflicting requirements, we set up ensemble stacking models to combine different individual models together to make predictions. The models we considered in combination were as follows:

6.1. ARIMA Model

The Autoregressive Integrated Moving Average (ARIMA) model is a linear model that predicts a time-indexed observation using linear combinations of past values and errors[12]. We performed an automatic search for the best ARIMA model by setting p, q, d as tuning parameters to optimize: We first determined d by using both the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test and Augmented Dickey-Fuller test, choosing the larger d calculated by these two tests, but limiting the largest d to be 4. Then we performed an exhaustive search for the best choice of p and q in terms of the lowest Akaike Information Criterion (AIC), setting the max p and q to be 4 to limit the computation time. The ARIMA model response is univariate; we used the past returns $R_t \ldots, R_{t-p+1}$ to predict the next return R_{t+1}, for each individual company.

6.2. Linear Regression Model

The linear regression model has the advantage of being both easy to interpret and easy to extend to a multivariate form. We used a linear model for each individual company, each sector, and for the whole set of S&P 500 stocks. For each of these models, we used the variables return, CCI, MACDH, RSI, KDJ, WR, CMF and sentiment at time t to predict return at $t + 1$.

6.3. Tree Based Random Forest Model

The Random Forest[35] is a model based on decision trees that resolves the problem of overfitting existing in simple decision tree models by bagging on training data and variables: i.e., randomly sampling the training data with replacement and randomly selecting variables that are used to predict during each iteration. We performed a grid search of the number of estimators, maximum depth, and maximum number of variables to find the best fit. With variables return, CCI, MACDH, RSI, KDJ, WR, CMF, PE, PS, and PB, we applied the random forest for each individual company and each sector.

6.4. Feed Forward Neural Network Model

The Feed Forward Neural Network (FFNN)[36] is a neural network structure that links every input from the previous layer to the nodes in the next layer. The variables we used in the FFNN model are return, CCI, MACDH, RSI, KDJ, WR, CMF, ATR, PE, PS, and PB at time t. We utilized a model structure with one hidden layer of 48 nodes, relu activation[37], and a dropout [38] of 0.6 to construct our model. Each of the nodes in the hidden layer is calculated by linear combinations of the input variable values with the activation function applied, and then the predicted return is calculated by a direct linear combination of the 48 node values in the hidden layer. Also, dropout rate 0.6 means for each node $1, 2, \ldots, 48$, input elements are randomly set to zero with 0.6 probability during training to prevent overfitting.

6.5. Long Short-Term Memory Model

The Long Short-Term Memory (LSTM) model[39] has a flexible functional form that does not require assumptions on the error terms and incorporates temporal structures with a time window: Instead of putting inputs at every time point in the window period together in one function, LSTM uses a summary of previous inputs, along with the current input, to predict the next output recursively. LSTM also has a forgot-gate structure that allows filtering unimportant information and keeping key signals. The variables we used in the LSTM model were return, CCI, MACDH, RSI, KDJ, WR, CMF, ATR, PE, PS, and PB, from the previous three time periods of $t - 2$, $t - 1$, and t. We trained the model on multiple timesteps simultaneously by having input and output sequences of length three and predicting the response one time unit ahead. The LSTM layer we chose had the structure of return sequence, with 32 nodes, Tanh activation function, and dropout rate of 0.6. Similar to the FFNN model, the predicted return was calculated three times recursively by a direct linear combination of 32 LSTM node output values for time points $t - 1$, t, and $t + 1$. The predictions at $t - 1$ and t helped training, while the prediction at $t + 1$ really predicted the future. In this way, we increased the amount of data available for training. We also adopted two LSTM model structures: The first consisted of two LSTM layers stacked, while the second had just one LSTM

layer. The two-layer model was trained using all available stocks, while the one-layer model was first trained using all available stocks, then fine-tuned again using each individual stock's data for that stock's prediction. The fine tuning was controlled by keeping the LSTM layer parameters frozen, while only training and updating the linear combination parameters with a much smaller learning rate, and limited iterations. Although the fine-tuning led to a lower accuracy, the variability it created eventually boosted the ensemble model accuracy, which we discuss in later sections.

6.6. Summary of individual models

After carefully tuning each individual model's parameters and preprocessing options, we report the best results obtained with each category of models, by directional accuracy, as Table 7 shows. Other models that were also fitted, but not shown in the table were Support Vector Regression, Kernel Ridge Regression, ExtraTree Decision trees, and Gated Recurrent Units, we do not show them because either they have a relatively lower accuracy in stock prediction problems, or there are similar methods shown in the table that have a slightly better performance.

Table 7. Combined performance of several prediction models on all 518 current and past S&P 500 stocks, during the period January 1, 2003 to December 31, 2019, with each model updated and tested yearly.

Model	Input data	tuning parameters	MAE	RMSE	DA	UDA	DDA
ARIMA, yearly update all past data, for each stock	past return	max p = 4, max q = 4, d by hypothesis test	0.0254	0.00157	57.17	68.27	43.37
Linear Regression, all stock ten years rolling, yearly update	return, sentiment, CCI, MACDH, RSI, KDJK , WR, CMF from last period	minimize RMSE	0.0251	0.00152	59.72	65.18	52.99
Random Forest, for each sector ten years rolling, yearly update	return, sentiment, PE, PS, PB,CCI, MACDH, RSI, KDJ, WR, ATR, CMF from last period	max depth=8, max features=all, estimators=400, minimize MAE	0.0244	0.00144	60.35	74.90	42.20
Feed Forward Network, all stock ten years rolling, monthly update	return, sentiment, PE, PS, PB,CCI, MACDH, RSI, KDJ, WR, ATR, CMF from last period	one forward layer with 48 nodes, relu activation, Dropout 0.6, minimize MAE	0.0243	0.00142	60.92	78.53	38.95
LSTM, all stocks together, all past data, yearly update	return, sentiment, PE, PS, PB,CCI, MACDH, RSI, KDJ, WR, ATR, CMF from past three periods	2 LSTM layer, 32 nodes each, Dropout 0.6, minimize MAE, 3 outputs with one shift	0.0240	0.00140	61.69	74.07	46.13

In general, the old traditional ARIMA model performs worse than other models, and we view it as a baseline model. On the other hand, the traditional Linear Regression models trained together using all stock's data have an 2.55% DA accuracy higher on average than ARIMA, and we treated them as a baseline for multivariate models. Experimental results showed that the best linear regression models trained (1) separately on each individual stock; (2) separately on each sector; and (3) together using all available stocks data combined yield, respectively, 58.16%, 59.55%, and 59.72% overall DA accuracy, as Table 8 shows. This trend of accuracy increasing as the data scope grows was observed on other types of models as well, indicating that although different stocks might behave differently, they could be used in the same model with different variable values.

Table 8. Overall directional accuracy comparison of linear models trained on different levels.

Model	DA
linear regression models trained separately on each individual stock	58.16%
linear regression models trained separately on each sector	59.55%
linear regression models using all stocks data combined	59.72%

Using the linear model, we also examined different rolling window periods and found having a rolling period of ten years yields the highest accuracy, compared with using all past data. We then applied the ten-year period again to the Random Forest and FFNN model. We trained the Random Forest model on the subsets by sector level, since training the Random Forest using data from all stocks took much more computation time, and we wanted to have more variability in our ensemble steps. The FFNN and LSTM model were trained by having 10% of the training data randomly chosen as validation sets for early stopping [40]. We set the FFNN model to be updated monthly as it took less computational time compared to other machine learning methods. In summary, machine learning models, especially the LSTM, had better performance than traditional statistical models. The LSTM model outperformed other individual models in every aspect; one possible reason is that it incorporated an input period from the previous three weeks and thus contained more information. However, in Jiang et al.[16] the Random Forest model showed a higher accuracy by 3.99% compared with the LSTM. We think this difference in results came from the difference in sample size: our LSTM model had 14,113 parameters for the two-layer stacked model and 5,793 for the one-layer model, compared with our 427,255 observations. In contrast, Jiang et al. had a sample of size 4,054 but had a three-layer LSTM structure with even more parameters needed to fit it. The combined observations from 518 different stocks helped train these machine learning models, which usually work better on larger datasets. Figure 7 shows the average directional accuracy by year, for different models.

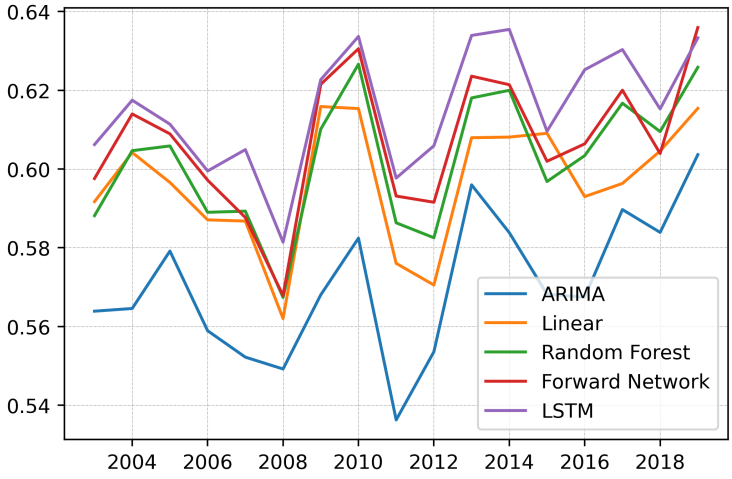

Figure 7. The average directional accuracy of different models by year.

7. Combining Models with Ensemble Stacking

7.1. Ensemble Stacking

After we obtained predictions from the individual models with different input variables, we combined these prediction results into an ensemble model. One natural way of combining different models is for these individual models to vote for whether the stock price will go up or down in the next period. The vote can either be a majority vote, or a weighted vote. Another natural way of combining different models is through choosing the best model for each year or for each stock, based on the past performance: although selecting models by cross validation generally cannot exceed the performance of the best model, the prediction for all S&P 500 stocks can be separated into different years and stocks, which when combined yields a better overall performance. Based on these two natural approaches, we utilized an ensemble stacking model to determine the weight of each individual model each year. Ensemble stacking [41] is a technique that uses several base models' predicted outputs as predictor variables for the final prediction. In our study, we used a linear regression as the meta-learner to learn the weight of each

base model, forcing the constant term in the regression to be zero, and all weights (coefficients) to be nonnegative during the combination stage for each year.

7.2. Summary of Ensemble Models

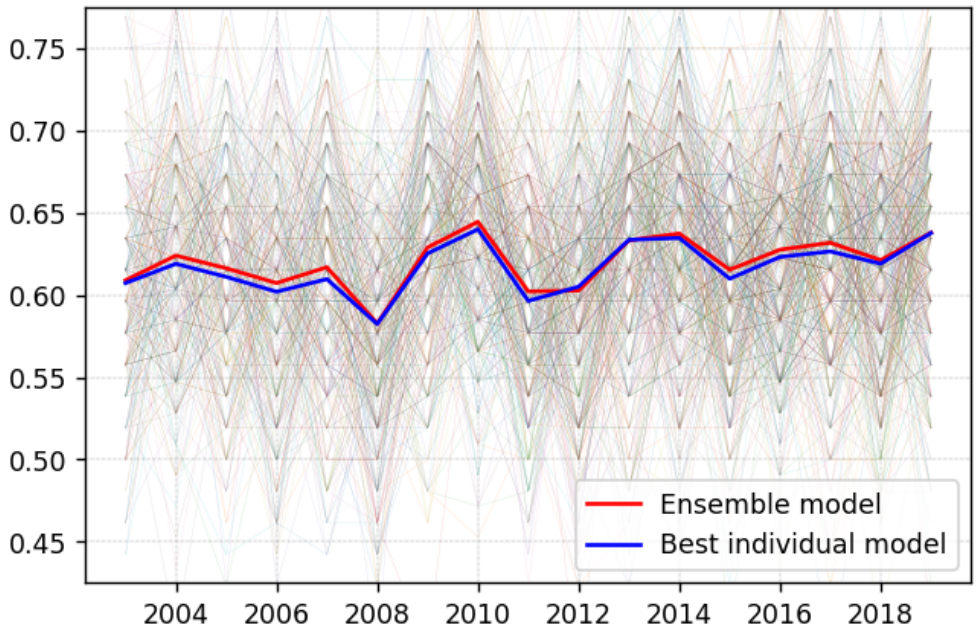

Figure 8. The yearly DA comparison between ensemble model and the best individual model for that year. The thin lines represent the ensemble DA of different individual stocks by year.

Model	Input data	Tuning parameters	MAE	RMSE	DA	UDA	DDA
Random Forest, for each sector ten years rolling, yearly update	return, sentiment, PE, PS, PB,CCI, MACDH, RSI, KDJ, WR, ATR, CMF from last period	max depth=8, max features=all, estimators=400, minimize MAE	0.0244	0.00144	60.35	74.90	42.20
Feed Forward Network, all stocks ten years rolling, monthly update		one forward layer with 48 nodes, relu activation, Dropout 0.6, minimize MAE	0.0243	0.00142	60.92	78.53	38.95
LSTM with all stocks together, fine tuning on each stock, all past data	return, sentiment, PE, PS, PB,CCI, MACDH, RSI, KDJ, WR, ATR, CMF from past three periods	1 LSTM layer, 32 nodes each, Dropout 0.6, minimize MAE, 3 outputs with one shift	0.0245	0.00146	61.07	67.75	52.61
LSTM, all stocks together, all past data, yearly update		2 LSTM layer, 32 nodes each, Dropout 0.6, minimize MAE, 3 outputs with one shift	0.0240	0.00140	61.69	74.07	46.13
Ensemble, yearly update, two year rolling	Individual Model outputs from above four model	Regression with constant fixed at 0, coefficient forced to be greater or equal than 0	**0.0239**	**0.00138**	**62.09**	73.71	47.61

Table 9. Combined performance of several prediction models, as well our ensemble model, on all 518 current and past S&P 500 stocks during the period January 1, 2003 to December 31, 2019.

Recall that our ensemble model was fitted using past years' prediction outputs as inputs, starting from the third year. The ensemble model we chose was based on a combination of four machine learning models: Random Forest model fitted for each sector, FFNN fitted monthly, two-layer stacked LSTM, and the LSTM model fine-tuned on each individual stock. Figure 8 shows the yearly directional accuracy comparison for the ensemble model versus the best individual model for that year. Table 9 shows the detailed structure for the four individual models we

chose for the ensemble, as well as the performance of the combined ensemble model built based on their outputs. We can see the ensemble model outperforms every individual model in terms of MAE, RMSE, and DA. The LSTM model further tuned using each stock's data boosted the ensemble model with a 0.3% increase in directional accuracy. Likewise, both the Random Forest and FFNN model contributed to a higher directional accuracy for the ensemble model, by a smaller margin. Adding other models, such as the Gated Recurrent Units (GRU) model into the ensemble, yielded still higher directional accuracy, but the improvement was minimal and GRU is very similar in structure to LSTM, so we preferred the simpler ensemble of four individual models.

Through repeated experimentation on different combinations of models, we found that forcing the intercept to be 0 was important for our ensemble stacking model to perform well. Holding other conditions unchanged, the ensemble model with no restriction on the intercept would have a 0.67% lower directional accuracy, causing the ensemble model to perform worse than the individual LSTM model. Also, if an individual model had a much lower directional accuracy than other models, including it in the ensemble would hinder the ensemble model performance. When the chosen individual models performed similarly well, a larger variability in terms of MAE or RMSE among the individual models helped the ensemble model to perform better. This need for variability led us to make different combinations of input data and models in our ensemble model. Also, we investigated the period needed to determine the individual model weights for the final ensemble model, and a grid search ranging from one month to several years revealed that a rolling period of two years and an update period of one year yields the best performance, with all choices of rolling period and update periods yielding a better result than the best individual model. Figure 9 summarizes the change across years in the weights placed on different individual models, for our ensemble model that predicts 518 S&P 500 stocks.

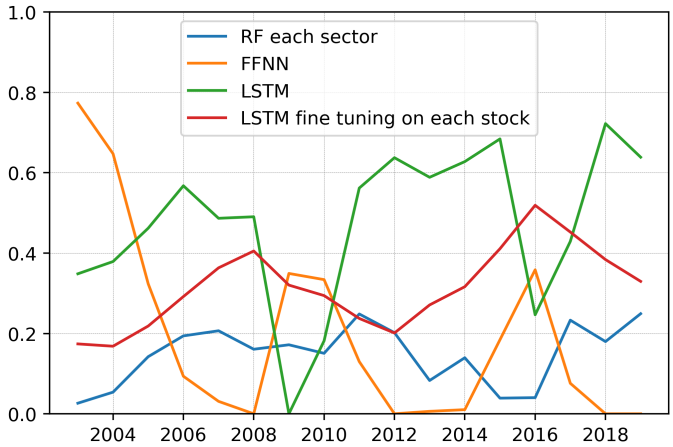

Figure 9. The weights placed on different individual models for the ensemble model predicting 518 S&P stocks, by year.

Table 10. Comparison of the directional accuracy for different models predicting the S&P 500 index, during the period January 1, 2003 to December 31, 2019. Because of the small size of the training data set, the FFNN and LSTM models trained using only past index data had unstable results with DA ranging from 55% to 61%.

Model	Input data	Tuning parameter	DA
Linear Regression	return, CCI, MACDH, RSI, KDJK , WR, CMF of the S&P 500 index from last period	minimize RMSE	58.62%
Random Forest	return, CCI, MACDH, RSI, KDJ, WR, ATR, CMF of the S&P 500 index, median PE, PB, PS from individual stocks and the sentiment of news about S&P 500 index from last period	max depth=8, max features=8, estimators=4000, minimize MAE	60.20%
Ensemble model	median of the 518 outputs from the previously shown linear regression, Random Forest, FFNN, and two LSTM models for individual stocks	Regression with constant fixed at 0, coefficients forced to be ≥ 0, rolling period one year	**66.18%**

Besides making predictions for each individual stock, we also built the ensemble model for predicting the S&P 500 index. We found that compared with models using only the past S&P 500 index data, the ensemble model built using all combined data coming from the 518 individual stocks had an incredible increase in predicting directional accuracy for the S&P 500 index. We used the medians of the 518 outputs from the previously shown linear regression, Random Forest, FFNN, and two LSTM models on the individual stock data as inputs for our ensemble model for the S&P 500 index, and chose a one-year rolling period. Table 10 shows the models that only used the past S&P 500 index data, as well as the ensemble model performance in DA accuracy.

Table 11. Comparison of the improvement in DA among different models. Our model for the S&P 500 index was fitted using the same ensemble method but with the median of the individual stock model outputs.

Author	model refresh and predicting period	Predicting	Predicted accuracy	Percentage of time stock price increased in test set	Improved accuracy compared to model only predicting upwards
Jiang et al.[16] for S&P 500 index	no refresh, 09/01/2012-04/01/2019	closing on next month	69.17%	67.35%	1.82%
Yu an Yan[26] for S&P 500 index	yearly update, 01/01/2010-12/29/2017	closing on next day	58.07%	54.59%	3.48%
Ding et al.[27] for S&P 500 index	no refresh, 02/22/2013- 11/21/2013	closing on next day	64.21%	59.68%	4.53%
Gorenc Novak and Velušček[32] for 370 S&P stocks	20 days update, 10/27/2005-06/14/2013	high on next day	61.16%	57.07%	4.09%
Our model for S&P 500 index	monthly & yearly mixed 01/01/2003-12/31/2019	closing on next week	66.18%	60.65%	**5.53%**
Our model for 518 S&P stocks	monthly & yearly mixed 01/01/2003-12/31/2019	closing on next week	62.12%	55.49%	**6.63%**

Note that stock prices have a general long-term tendency to increase over time. As a simple tool to assess accuracy, consider comparing proposed prediction models based on their improvements in directional accuracy over a hypothetical model that always predicts increases for stocks. Table 11 shows such an accuracy comparison between our model and the models proposed by the four authors introduced in the review literature. Compared with the 1.82% improvement by Jaing et al., 3.48% improvement by Yu et al., 4.53% improvement by Ding et al., and 4.09% improvement by Novak et al., our model has a larger 5.53% improvement for the DA of S&P 500 index, and 6.63% improvement for the DA of individual stocks, compared to a model that only predicts increases.

Figure 10. The spread of time when the ensemble model predicted an over 5% return, with y-axis the real return. Different colors indicate different stocks, and the red dashed line indicates the average real return of 5.15%, given the condition.

Although a 6.59% improvement in DA does not seem like a big improvement, in reality, we do not really need to make buy and sell decisions for every stock at every time point. Given realistic conditions, the ensemble model could help make investment decisions: 5.52% of the time the ensemble model predicted a 2% or above return in the next period, and given this prediction, 76.52% of the time the stock price went up in the next period, with an average real return of 2.80%. 6.08% of the time there existed a 5% loss in the next period, and given this condition, 66.42% of the time the ensemble model predicted the direction right, with an average predicted loss of 0.83%. We also found in our model that the past DA performance for specific stocks has little impact on the future DA accuracy for that stock. Figure 10 showed the spread of the real return from the individual S&P 500 companies, through January 1, 2003 to December 31, 2019, given the ensemble model predicted an over 5% return. We can see that the investment opportunities our model predicted exist among different years and different companies.

8. Conclusion

We summarized several different approaches to predict stock price, and then integrated those methods together to predict the S&P 500 stock data. Compared with similar works which apply their methods to a smaller set of data, our data set included text from news items, technical indicators, and fundamental reports for 518 current and former S&P 500 large-cap companies over a 20-year period to test and train our algorithm, which relates more closely to the real world market. Also, we integrated new machine learning methods such as the Random Forest and LSTM models in an ensemble setting to predict future stock prices with a higher accuracy and adaptability using our combined model compared to uni-method models.

Future research may extend beyond stock predicting: Another important aspect of stock trading is decision making. When outcomes of decisions are partly random, we can model them through the Markov Decision Process (MDP) framework with Reinforcement Learning (RL) techniques [42][43]. This is a self-taught learning method to solve MDP problems when the environment is partly unobservable [44]. In the stock trading problem, if we model agent states as the currently held stocks and at each week we buy, sell, or hold, then this can be modeled in the MDP framework to achieve goals such as maximizing the long-term return while keeping the risk as low as possible at the same time.

Appendix: Data Cleaning Steps

We actually collected the data on September 30, 2020, so the stocks we collected for our research were stocks listed in S&P 500 index as of September 30, 2020, plus the stocks that had been removed from the list but were still trackable. From January 1, 2000 to September 30, 2020, there were 247 times in which a common stock was once on the S&P 500 list but was removed due to reasons like market capitalization changes, merger and acquisition, or being spun off, etc. We summarize the reasoning for removal, as well as whether we decided to track the stock, in Figure 11. Here we justify the reasoning: 122 stocks were removed from the S&P 500 list due to being merged or acquired (M&A). Since the stock stops being traded in the stock exchange once the company associated with the stock gets acquired (it will be turned into cash or the shares of the purchasing company, depending on the buying terms), we decided not to track these companies. 102 stocks were removed from the S&P 500 list due to market reasons such as market value decline. Among these, we tracked 61 of the stocks. For the remaining 41 stocks, eventually 17 went bankrupt, 18 were acquired or merged, one was split, and five had too many missing values or insufficient data. For similar reasons, we tracked six additional former S&P 500 stocks from the other category, accounting our 67 former S&P 500 stocks.

For the missing values, we utilized the two data sources [30] and [31], and matched them by dates. Considering both current and past S&P 500 stocks, 69 have missing values. 47 out of them have just one day of data missing, while the other 22 stocks have an average of 2507 days missing. We first deleted missing dates and only considered dates both sources had data, and with a maximum of 10 consecutive days of gaps allowed. Then we deleted any questionable data after matching adjusted closing prices across the two sources to within 2% error, by choosing the [30] data as the base. Table 12 and Table 13 show a summary for the error rates. After these steps, stocks that

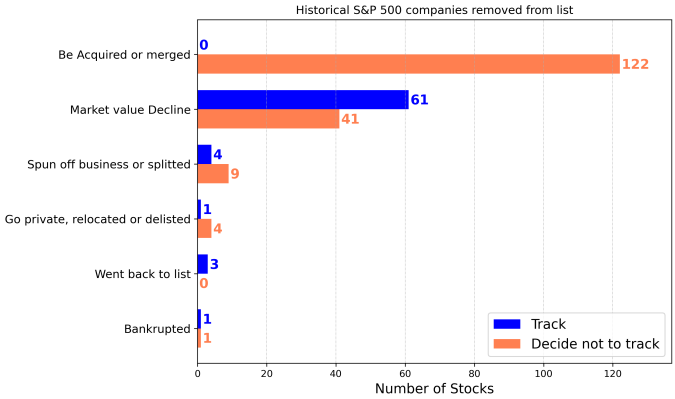

Figure 11. Summary of the reasons for removal from S&P 500 list, at the time the common stock was removed, for historical S&P 500 stocks, for the period from January 1, 2000 to September 30, 2020.

had less than 5 years of trading history after filtering were then also removed, since models training on individual stocks need sufficient data.

Table 12. Report on the reliability of stock price data, accounted for all the stocks we tracked, from November 1, 1999 to September 30, 2020, from the two different sources [30] and [31].

	Open	**High**	**Low**	**Close**	**Adj Close**	**Volume**
Percentage of data points within 1% error	97.07%	97.53%	97.53%	97.54%	77.11%	73.41%
Percentage of stocks have 99% quantile error less than 5%	96.01%	96.01%	96.01%	96.01%	71.06%	40.03%

Table 13. Report on the reliability of dividends and splits data, from [30] and [31]. The amount exactly matched and amount within 1% error are conditioned on dates being matched.

	Dividend	Split
Percentage of dates matched	98.08%	76.15%
Amount exactly matched	88.30%	96.30%
Amount within 1% error	91.98%	98.83%

REFERENCES

1. Mandelbrot, Benoit B *The variation of certain speculative prices*, Fractals and Scaling in Finance, pp. 371-418, 1997.
2. Samuelson, Paul A *Proof that properly anticipated prices fluctuate randomly*, The World Scientific Handbook of Futures Markets, pp. 25-38, 2016.
3. Qian, Bo and Rasheed, Khaled *Stock market prediction with multiple classifiers*, Applied Intelligence, vol. 26, no. 1, pp. 25–33, 2007.
4. Bollen, Johan and Mao, Huina and Zeng, Xiaojun *Twitter mood predicts the stock market*, Journal of Computational Science, vol. 2, no. 1, pp. 1–8, 2011.
5. Fama, Eugene F *Efficient capital markets: A review of theory and empirical work*, The Journal of Finance, vol. 25, no. 2, pp. 383–417, 1970.
6. Fischer, Thomas G *Reinforcement learning in financial markets - a survey*, FAU Discussion Papers in Economics, 2018.
7. Obthong, Mehtabhorn and Tantisantiwong, Nongnuch and Jeamwatthanachai, Watthanasak and Wills, Gary *A survey on machine learning for stock price prediction: algorithms and techniques*, 2020.

8. Hurst, Brian and Ooi, Yao Hua and Pedersen, Lasse Heje *A century of evidence on trend-following investing*, The Journal of Portfolio Management, vol. 44, no. 1, pp. 15–29, 2017.
9. Piotroski, Joseph D *Value investing: The use of historical financial statement information to separate winners from losers*, Journal of Accounting Research, pp. 1–41, 2000.
10. Fama, Eugene F *The behavior of stock-market prices*, The Journal of Business, vol. 38, no. 1, pp. 34–105 1965.
11. Adebiyi, Ayodele Ariyo and Adewumi, Aderemi Oluyinka and Ayo, Charles Korede *Comparison of ARIMA and artificial neural networks models for stock price prediction*, Journal of Applied Mathematics, vol. 2014, 2014.
12. Ariyo, Adebiyi A and Adewumi, Adewumi O and Ayo, Charles K *Stock price prediction using the ARIMA model*, 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, pp. 106–112, 2014.
13. Meyler, Aidan and Kenny, Geoff and Quinn, Terry *Forecasting Irish inflation using ARIMA models*, Central Bank and Financial Services Authority of Ireland, 1998.
14. Lux, Thomas and Kaizoji, Taisei *Forecasting volatility and volume in the Tokyo stock market: Long memory, fractality and regime switching*, Journal of Economic Dynamics and Control, vol. 31, no. 6, pp. 1808–1843 2007.
15. Nelson, David MQ and Pereira, Adriano CM and de Oliveira, Renato A *Stock market's price movement prediction with LSTM neural networks*, 2017 International Joint Conference on Neural Networks (IJCNN), pp. 1419–1426 2017.
16. Jiang, Minqi and Liu, Jiapeng and Zhang, Lu and Liu, Chunyu *An improved Stacking framework for stock index prediction by leveraging tree-based ensemble models and deep learning algorithms*, Physica A: Statistical Mechanics and its Applicationsl, vol. 541, pp. 122272 2020.
17. Butler, Kirt C and Malaikah, S Jamal *Efficiency and inefficiency in thinly traded stock markets: Kuwait and Saudi Arabia*, Journal of Banking & Financel, vol. 16, no. 1, pp. 197–210 1992.
18. Kavussanos, Manolis G and Dockery, Everton *A multivariate test for stock market efficiency: the case of ASE*, Applied Financial Economics, vol. 11, no. 5, pp. 573–579 2001.
19. Yao, Juan and Gao, Jiti and Alles, Lakshman *Dynamic investigation into the predictability of Australian industrial stock returns: Using financial and economic information*, Pacific-Basin Finance Journal, vol. 13, no. 2, pp. 225–245 2005.
20. Park, Young S and Lee, Jung-Jin *An empirical study on the relevance of applying relative valuation models to investment strategies in the Japanese stock market*, Japan and the World Economy, vol. 15, no. 3, pp. 331–339 2003.
21. Atsalakis, G and Valavanis, Kimon P *Surveying stock market forecasting techniques-Part I: Conventional methods*, Journal of Computational Optimization in Economics and Finance, vol. 2, no. 1, pp. 45–92 2010.
22. Hirschberg, Daniel S *Algorithms for the longest common subsequence problem*, Journal of the ACM (JACM), vol. 24, no. 4, pp. 664–675 1977.
23. Mikolov, Tomas and Sutskever, Ilya and Chen, Kai and Corrado, Greg S and Dean, Jeff *Distributed representations of words and phrases and their compositionality*, Advances in neural information processing systems, pp. 3111–3119 2013.
24. Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina *Bert: Pre-training of deep bidirectional transformers for language understanding*, arXiv preprint arXiv:1810.04805 2018.
25. Araci, Dogu *Finbert: Financial sentiment analysis with pre-trained language models*, arXiv preprint arXiv:1908.10063 2019.
26. Yu, Pengfei and Yan, Xuesong *Stock price prediction based on deep neural networks*, Neural Computing and Applications, vol. 32, no. 6, pp. 1609–1628 2020.
27. Ding, Xiao and Zhang, Yue and Liu, Ting and Duan, Junwen *Deep learning for event-driven stock prediction*, Twenty-fourth international joint conference on artificial intelligence, 2015.
28. Gorenc Novak, Marija and Velušček, Dejan *Prediction of stock price movement based on daily high prices*, Quantitative Finance, vol. 16, no. 5, pp. 793–826 2016.
29. Akita, Ryo and Yoshihara, Akira and Matsubara, Takashi and Uehara, Kuniaki *Deep learning for stock prediction using numerical and textual information*, 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS, pp. 1–6 2016.
30. Yahoo! Finance *Yahoo!*, https://finance.yahoo.com/, 2021.
31. Alpha Vantage *Alpha Vantage Inc.*, https://www.alphavantage.co/, 2021.
32. Gorenc Novak, Marija and Velušček, Dejan *Prediction of stock price movement based on daily high prices*, Quantitative Finance, vol. 16, no. 5, pp. 793–826 2016.
33. Malo, Pekka and Sinha, Ankur and Korhonen, Pekka and Wallenius, Jyrki and Takala, Pyry *Good debt or bad debt: Detecting semantic orientations in economic texts*, Journal of the Association for Information Science and Technology, vol. 65, no. 4, pp. 782–796 2014.
34. Yeo, In-Kwon and Johnson, Richard A *A new family of power transformations to improve normality or symmetry*, Biometrika, vol. 87, no. 4, pp. 954–959 2000.
35. Ho, Tin Kam *Random decision forests*, Proceedings of 3rd international conference on Document Analysis and Recognition, vol. 1, pp. 278–282 1995.
36. Svozil, Daniel and Kvasnicka, Vladimir and Pospichal, Jiri *Introduction to multi-layer feed-forward neural networks*, Chemometrics and Intelligent Laboratory Systems, vol. 39, no. 1, pp. 43–62 1997.
37. Brownlee, Jason *A gentle introduction to the rectified linear unit (ReLU)*, Machine Learning Mastery, vol. 6, 2019.
38. Srivastava, Nitish and Hinton, Geoffrey and Krizhevsky, Alex and Sutskever, Ilya and Salakhutdinov, Ruslan *Dropout: a simple way to prevent neural networks from overfitting*, The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958 2014.
39. Gers, Felix A and Schmidhuber, Jürgen and Cummins, Fred *Learning to forget: Continual prediction with LSTM*, Neural computation, vol. 12 no. 10, pp. 2451–2471 2000.
40. Prechelt, Lutz *Early stopping-but when?*, Neural Networks: Tricks of the trade, pp. 55–69 1998.
41. Wolpert, David H *Stacked generalization*, Neural Networks, vol. 5 no. 2, pp. 241–259 1992.
42. Sutton, Richard S and Barto, Andrew G *Reinforcement Learning: An Introduction*, 2018.
43. Puterman, Martin L *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 2014
44. Deng, Yue and Bao, Feng and Kong, Youyong and Ren, Zhiquan and Dai, Qionghai *Deep direct reinforcement learning for financial signal representation and trading*, IEEE transactions on neural networks and learning systems, vol. 28 no. 3, pp. 653–664 2016.