Stock Price Prediction via Financial News Sentiment Analysis

Lizi Chen
New York University
New York, United States
lc3397@nyu.edu

Chun-Yi Yang
New York University
New York, United States
chunyi.yang@nyu.edu

***Abstract*—**

There has been a steady trend in using big data and machine learning techniques to analyze how daily media affects the stock market. Upon takeaways from researches in macro market analysis and short twitter text analysis, we conduct a more granular stock and news data analysis. With by-minute U.S. stock market prices in addition to Reuters and Wall St. Journal news scrapped in full details for 45 days, we have a stock price predictive model trained from historical stock trend, news keyword, news content, and news title. [PROVIDE RESULTS FROM THE MODEL LATER]

***Keywords*—spark, big-data, sentiment-analysis, stock-prediction**

I. Introduction

The Big Data era has prospered the financial industry for many years. Banks and hedge funds now can fully leverage their huge amount of data to construct their robust strategies. Likewise, as the software and hardware technology springing up in the big data field, the capability of it rocks even further. The 2014 released Apache Spark framework has already been gradually taking the preeminence over its predecessor Hadoop. Its efficiency and capability have been canonized from technology to finance industries and even helps the evolution of Fin-Tech industry and Machine Learning.

The busy industry Fin-Tech and many of its upholding hedge fund companies have been pouring effort to improve the integration of big data and machine learning technology. In this project, we find the relationship between news that are published in popular business media and the stock price trend. With the drastic improvement of data process capability from Spark, we collect full text news and intra-day stock prices for all the U.S. stocks traded from NASDAQ, NYSE, and AMEX. By having news and stock prices mapped by exact minute timestamp, we provide high-granularity insight into the complex relationship between these two.

We construct our model according to various algorithms: TF-IDF, Word2Vec, Neural Network, [**ADD FURTHER WORK**]. After comparing and contrast with various predication times, we conclude that [**ADD RESULT**].

II. Motivation

Being able to selectively consume a large amount of information efficiently can be a major dominance in financial market. There are thousands of newly-published articles from various websites every day, and that can be millions of words in a lot of ways of narration. Investors have to choose from resources, such as Reuters website; then search for reports and articles that may relate to his or her portfolio positions, and then read and understand the information from these articles. This process will take a lot of time. Although there are investors who trade mostly based on stock data, charts and statistical analysis, getting a gist from the most recognized media adds more security to foresee a stock trend.

Most of news come with a topic, or a simple narration of an event, *[A subject] [B predicate] [C object]*. News will provide more detailed information yet the theme can be established in a much shorter paragraph. Although there are better machine learning algorithms that can provide read-able abstract paragraph based on a long article, we believe that a more concise indicator in a pattern of *[A stock] (will) [Drop / Rise]* is more efficient.

There have been a lot of stock prediction research with the help from social media, such as Twitter and Yelp, yet the training data were mostly EOD (End-Of-Day) stock prices. Many events have shown direct impact on related stocks after several hours if not minutes in the same trading day. We believe by having the news and stock trend matched by minute level can add versatility in model training as well as provide better predicative result.

Although social media have been increasingly reflecting and influencing behavior of other complex systems; such as the stock market, the collected data is very scattered. News website provide more main stream comprehensive narration from a less objective point of view.

III. Related Work

[**ADD INTRO PARAGRAPH OF THE RELATED WORK**]

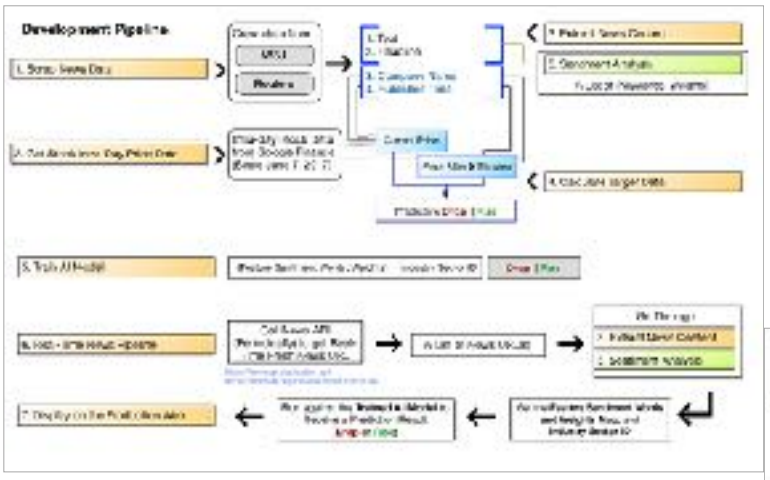
The tones and words used as content in the financial news are correlated to the performance of stock markets in several aspects (Robert et al. 2012). For example, through sentiment analysis on the financial articles, the negative sentiment articles perform better on positive ones on price direction trend (50.9% versus 50.0%). Moreover, an interesting finding is that the negative articles have 52.4% of time accompanied with the price increase as well as the positive articles have 53.5% of time accompanied price decrease. The authors concluded that it might be caused by the contrarian trading strategy of traders. [2]

A full-around survey about sentiment analysis algorithms, which was conducted by Walaa [3] provides insight on how each algorithm fits to certain scenario and which model provides better results. Within 9 models, the Neural Network

with Deep Learning gives the best result. [**ADD OUR RESULTS WITH COMPARISON**]

IV. DESIGN

Since the detailed data; full content news and intra-day minute-level historical stock prices, is neither public nor free throughout the web, we collect all our data from scratch. Below is the development process diagram, which contains 7 major steps.



1. Scrap News Data:

2. Get Stock Intra-Day Price Data:

To construct the model, we need two sets of data: Intra-day stock price data, and News from WSJ and Reuters. The use of it is described in the following.

3. Extract News Content: For each scrapped news, we have to extract useful information from the meta data by using Python's Beautiful Soup package. One piece of news will provide the following six feature data:

- Published Date
- Title
- Keywords *(May not be shown from all webs)*
- Sector *(May not be shown from all webs)*
- Content
- URL

Not all websites have these meta data written in the web HTML, but if so, our code will scrap them.

4. Calculate Target Data: For each news, there can be one or many related stocks. We will first extract the company name text from the news, then map to the ticker symbol used for its stock. With a ticker symbol and news publish time, we can call the historical stock price database to query the stock price at the time the news was published, in addition to prices at any time after that, 20 minutes, 1 hour, 2 hours or further. By comparing the current price of the stock and its future prices, we can see the trend of it after the publication of news, either *'Rise'* or *'Drop'*.

5. Train AI Model: Now that we have all target values calculated, we will have a pre-training set of data, of which the schema is shown below:

TITLE	CONTENT	DROP_RISE

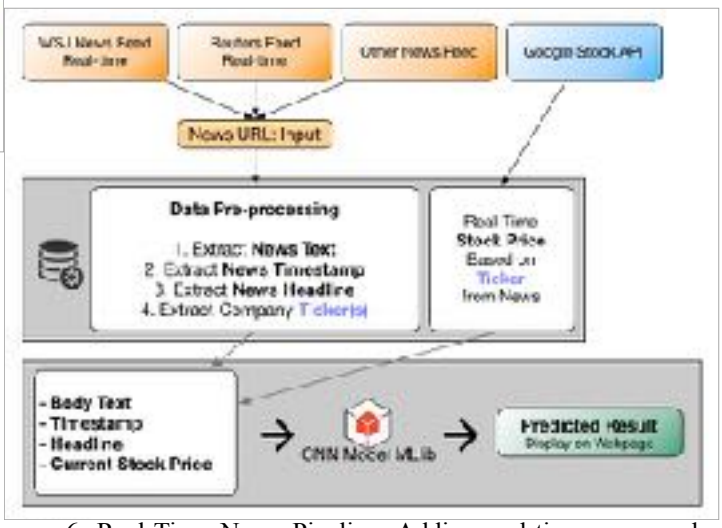
After the sentiment analysis process (which is described in the next paragraph), the training data set schema is shown below:

Weights of Word Feature	DROP_RISE

We will consider the news title and body content altogether in this case, since most of the news have both of these two attributes. The publication time, keywords, sector, and related company names and stocks are not part of the training data set but to determine the target value and to do the sentiment analysis job.

Sentiment Analysis: as part of the model building process, we need to analyze the sentiment for each piece of news. **[DESCRIBE THE FINAL SENTIMENT ANALYSIS APPROACH HERE AND REASONS]**

After having the AI Model training, we can feed the model with news pieces; however, it's better to provide real-time function that helps future investors to access the indicators directly from a web browser. Thus, we add steps 6 and 7 as shown in below:



6. Real-Time News Pipeline: Adding real-time news and stock data streaming as well as automate the whole data collection and model training process will help eliminate post-production maintenance work. This process begins with periodically calling the newsapi.com asking for new JSON data from target websites. For each newly published article webpage, a webpage extractor will collect information for the trained model so as to display a real-time predicative result. After the ground-truth comes out after 20 minutes, 1 hour, or 2 hours, we will compare it with the predicative result generated previously so as to improve our model.

7. Display on the Production Web: Below is the diagram for the components of the final product. We provide visualized web interface for clients to analyze and customize according to their portfolio interests.

V. EXPERIMENTS

(In this section, you can describe: *Your experimental setup, problems with: data, performance, tools, platforms, etc. Discuss your experiments, describe what you learned.* Discuss limitations of the application. Discuss what you would do to expand it given time - how would you improve it, etc.)

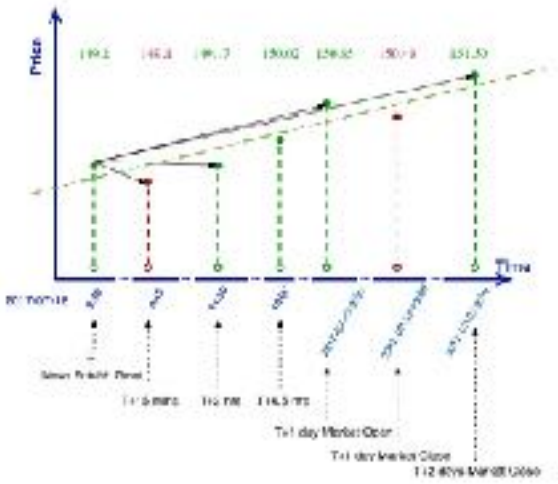
1. Data collection: The data is divided into information data and updated data by different functions. The information data such as company attributes are collected from Intrinio.com as well as the words of company alias are collected from CityFalcon.com. As for updated data like stock prices and financial news articles, the stock prices data are fetched on the minute basis, starting from 04:00 to 20:00. The period includes the pre-sale trading time, normal trading time and post-trading time. On the other hand, by utilizing the News API, a multi-media headlines metadata API, we check the WSJ (Wall-Street-Journal) and Reuters API every 5 minutes and apply crawlers to scrap the website content and timestamp we need.

2. Data processing: We first dealing with the missing value. Companies lacking information attributes are marked as Null. Next, to trace the influence of the articles to the stock price, we mark the company ticker to financial articles by pairing the articles keywords to bags of words of company alias names. All data are categorical and stored in JSON format.

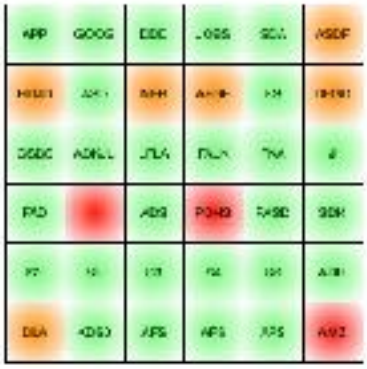
3. Data generation: Before generate data features, we need to know what company information does each article contain and used it to map stock price. There are 2 data mapping processes in our dataset. The first is article mapping with company ticker. A keyword list of each article from internet combined with a 4-gram word list from it's title is generated. Then we utilized a company alias name dictionary from CityFalcon database to mark company ticker to each article. The second the mapping the articles to stock price. The company ticker mark from last process is used to position its own stock price.

4. Feature generation: The data input attribute consists of context features and related company information. The TF-IDF (Term frequency-inverse document frequency) and PCA (Principal component Analysis) in Spark MLlib are applied on article content to generate weights of word features. As the result, a 400,000 feature vector from TF-IDF will decrease to 1,000 vector long to each element. The other features contains company category, group, and sector fields information from Intionio.com. A price trend The output is calculated by logistic regression to of specified time period to represent as price performance after certain period of time the press was released. If the stock price increases, the output mark as 1, otherwise 0.

5.



Industry	K Slope
Healthcare	K > 1
Energy	K > -1 and K < 0
Materials	K > 0 and K < 1
Government	K < -1
Services	K > 0 and K < 1
Technology	K > 1



VI. CONCLUSION

(One paragraph about the value, results, usefulness of your application.)

ACKNOWLEDGMENT

(This section is optional. It can be used to thank the people/companies/organizations who have made data available to you, for example. You can list any HPC people who were particularly helpful, if you used the NYU HPC. List Amazon if you used an Amazon voucher.)

REFERENCES

1. T. White. Hadoop: The Definitive Guide. O'Reilly Media Inc., Sebastopol, CA, May 2012.
2. Robert S, Yulei Z, Chen-Neng H, Hsinchun C. 2012, Evaluating sentiment in financial news articles. Decision Support System 53 : 458-464
3. Walaa M., Ahmed H., Hoda K., 2013, Sentiment Analysis Algorithms and Applications: A Survey.