CS230

Natural Language Processing and Event-driven Stock Prediction

Cheryl Ji
Management Science and Engineering
Stanford University
yxji0123@stanford.edu

Jimmy Qin
Department of Statistics
Stanford University
qiqin@stanford.edu

1 Introduction

Predicting stock price movements has been of central interest to investors, public companies as well as regulatory agencies, and has attracted much attention from academia and business. Given the high volatility of stock markets during the financial crisis and recently starting from late 2017, it is essential to be able to estimate the changes of stock prices in order to manage portfolio profits and losses (P&L) w.r.t individual risk appetites.

On the other hand, stock prediction has been a challenging task. One theory that explains the underlying drivers, the Efficient Market Hypothesis (EMH) (Fama, 1965), states that asset prices fully reflect all available information and market prices only react to new information, e.g. financial news, social media blogs, etc. Given the major role of institutional investors and their information gathering conventions, we believe it is more important to focus on financial new events.

Therefore,our main goal is to build a neural network to analyze the predictive power of financial news headlines on the (volatile) movements of S&P 500 index, and investigate whether some types of information / some words have particular influences, e.g. "CEO resigned". And since stock markets feature high volatility, we believe news would be a better indicator for extreme changes in S&P 500. However, we understand that we may not achieve very accurate prediction results for the well-known randomness of stock prices.

2 Dataset and Features

2.1 Data description

Given the time constraint and complexity of text crawling, we use the available Reuters' news from October 2006 to November 2013 scraped by Ding et al. (2014). The raw dataset is .txt files each containing one news event with headline, main body and timestamp. We extract headlines and timestamps only. According to EMH, we assume that new information only has immediate effect on the next trading date. Thus we align each event with the next trading date and the corresponding index movement direction downloaded from Yahoo Finance Further, to model volatility, we add a response called "extreme" where we set it as 1 if the absolute change rate $> 0.7\%$ to make the data balance. Dropping 25 empty news headlines and nan trading data, we are provided with 92,063 news events, among them 42414 observations are positive.

The training, development and test sets are split according to time given the time-varying characteristics of data, i.e. some words may have stronger predictive power in one period of time than others. In order to make sure that we have enough data in the development and test sets, 80% of the total data are used as the training data, 10% for the development and test sets respectively. Among all the training data, 37509 of them are positive, and 38360 are negative. To make sure that the dev and test

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

sets have the same distribution, we shuffle and randomly split the rest 20% data into the dev and test sets. The splitting is summarized in Table 1.

Table 1: Data splitting

	Train	Dev	Test
Number of events	75869	8097	8097
Time interval	10/20/2006 - 10/06/2012	10/07/2012 - 05/03/2013	10/07/2012 - 05/03/2013

2.2 Embedding

We would like to have some control over the embedding process, thus we decide to use some low-level packages step by step instead of using keras. We tokenize each headline into a word list, unify each word in terms of tense, noun and case using the en package, and remove stopwords using nltk. Given the potential special semantics in the finance context, we decide to do transfer learning on our entire dataset using the pretrained Word2Vec model, embedding each word into a 300-d dense word vector. We also experimentally used the GLoVe embedding, which yielded less accurate results.

To this end, we engineer our input feature to be the sum and mean of the dense vectors representing words included in each headline, and normalize them. We understand that much information would be lost. Thus we also put word vectors in each headline in a sequence, and pad shorter observations with zero vectors to address variable lengths.

3 Methods

3.1 Evaluation metrics

3.1.1 Accuracy

The model's performance is tuned and evaluated by its classification accuracy. After adjusting our threshold of extreme changes, data shows that the event of extreme changes and non-extreme changes are approximately equally frequent. Thus we believe that accuracy is a good evaluation metric.

3.1.2 Receiver Operating Characteristic (ROC)

Another method to address data skew is Receiver Operating Characteristic (ROC). The ROC curve illustrates the relationship between True Positive and False Positive rates by setting different threshold values. We expect the AUC to be slightly over 0.5.

3.2 Bag-of-words

Our baseline model is "bag-of-words" using a standard 3-layer neural network, with 300-d sum/mean word as inputs. We use a standard neural network of 2 hidden layers using keras: LINEAR $\rightarrow$ RELU $\rightarrow$ LINEAR $\rightarrow$ RELU $\rightarrow$ LINEAR $\rightarrow$ SIGMOID. Batch-normalization is employed for both hidden layers and inverted drop-out on the first hidden layer. We have tuned hyper-parameters of learning rate, drop-out rate and batch size using grid search.

3.3 LSTM

Our sequence model is a many-to-one, quasi-sentiment analysis, with the response as the indicator of whether the S&P 500 index on the next trading day would change by over 0.7% or not. In order to implement this model, we pad shorter headlines to make sure that the timestep of each observation is exactly 18. (The longest news title consists of 18 words. We add a Masking() function to tell the model to learn to ignore zero vectors while training.

4 Results

4.1 Bag-of-words

Initial results of modeling extreme changes show we achieve 59.74% and 65.35% accuracy for the sum-vector model and mean-vector respectively, which we believe are quite promising for finance data. The training result is summarized below.

Table 2: Sum Model result

	Train	Test
AUC	0.5927	0.5224
Accuracy	0.5641	0.5974

Figure 1: The test AUC of the sum-vector model

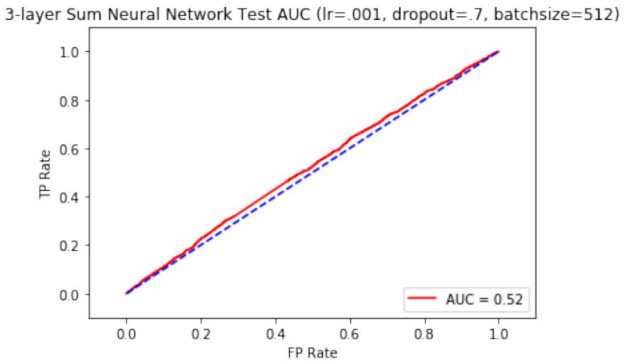
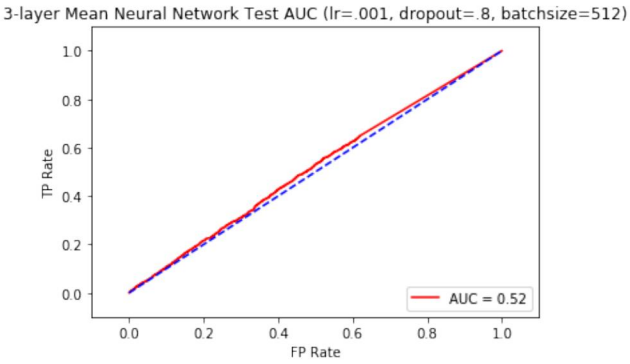


Table 3: Mean Model result

	Train	Test
AUC	0.5738	0.5165
Accuracy	0.5550	0.6535

Figure 2: The test AUC of the mean-vector model

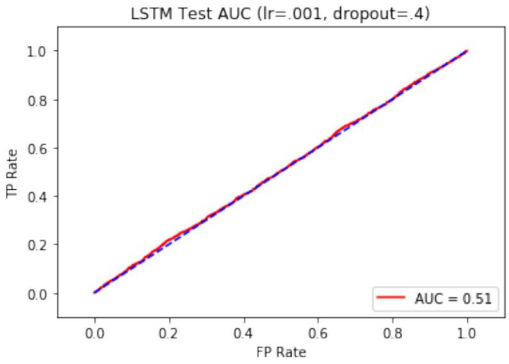


4.2 LSTM

Table 4: LSTM result

	Train	Test
AUC	0.6372	0.5136
Accuracy	0.5980	0.5934

Figure 3: The test AUC of the LSTM model

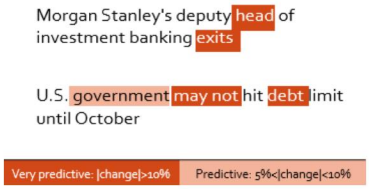


5 Contextual Decomposition

To this end, we have focused on prediction accuracy. Another problem of interest is to understand the influence of each individual word, by comparing changes in predicted probability before and after masking the word from the headline using the Bag-of-words Sum Model.

Though our predictive words search is not exhaustive, from the 2 examples above we can see that actions in financial news headlines, e.g. ‘exit’, etc., are likely to have strong influence on the volatility of S&P 500 index. So are some important named entities, e.g. ‘government’, ‘head’, etc. However, the joint influence, e.g. ‘head’ and ‘exit’ is not significant in our test.

Figure 4: Examples of News with Predictive Power from BOG



6 Conclusion

Despite well-known randomness of stock exchange, our study tries to capture (volatile) stock movement via financial news using natural language processing. Bag-of-words and LSTM perform equally well under "Accuracy" metric (0.65 for mean bag-of-words). Considering the nature of capital market, this result is more than satisfactory. However, LSTM model suffers more from overfitting post tuning.

We also preliminarily explore the predictive power of individual words. To improve our study, we would expand our dataset by scraping data and focus on training the pretrained word2vec . We expect to get better results.

7 Contributions

Cheryl: Cheryl reviewed relevant studies and implementations, preprocessed data, and built and tuned the prediction model.

Jimmy: Jimmy helped build the prediction model architecture and made the poster.

References

Fama, E. (1965). The Behavior of Stock Market Prices. Journal of Business, 38: 34–105.

Xiao Ding, Y. Z. (2014). Using structured events to predict stock price movement: An empirical investigation. Empirical Methods in Natural Language Processing.

Xiao Ding, Y. Z. (2015). Deep Learning for Event-Driven Stock Prediction. International Joint Conference on Artificial Intelligence.

W. James Murdoch, Peter J. Liu, Bin Yu (2018). Beyond Word Importance: Contextual Decomposition to Extract Interactions from LSTMs. International Conference on Learning Representations.

Sentiment-Analysis-in-Event-Driven-Stock-Price-Movement-Prediction

Word-vectors-game-of-thrones