Stock's Daily Price Movement's Correlation with and Predictability from Financial News

Ziyi Xie

May 5, 2022

1 Introduction

Stock market movement is often less predictable than many other things since there are many driving forces behind it. One of the most important driving force is people's emotion or perspective about the stocks. These feelings toward certain stocks are often influenced by information obtained elsewhere rather than produced inside a person's mind. News is a major part of of these information, so news should play a role in the movement of stock prices. Therefore, the main purpose of this project is to study how news contents cause stock movement, and if we can use news to predict stock movement. The stock we look at is Apple.Inc stock, because of the good liquidity and completeness of data for such stock.

We use bag of words and topic model to convert the news contents into counts, and we consider the proportion of word or topic frequencies as features. The prediction target is binary, whether a stock goes up or down. We look at the accuracy of several binary classification models, compare it to the baseline accuracy (most prevalent class), and analyze the distribution of the predictions and the important features in the models.

As for previous work, since such study is very popular, there have been multiple papers discussing the predictability of stock movement from news. Kari Lee et al. [1] used bag of words to predict stock movement, but they focused on trading return instead of prediction accuracy. In other papers, such as the one by Jingyi Shen et al. [2], the prediction model is a deep learning model. Such models are complex and requires a lot of computing resources. This project unique in its light weight prediction model and the novel feature engineering of proportions of word frequency or topics frequencies. We think proportion based method is better than frequency based method as proportions are normalized to be somewhat a composition, which intuitively make more sense than frequencies.

2 Design

2.1 Data

The news dataset we use is the 'Historical financial news archive' dataset from kaggle[3], by GENNADIYR. This dataset contains 12 years of news archive for US equities. The sources of the news are mainly Zacks Investment Research (45%) , Reuters (20%) , Investing.com (5%) , Seeking Alpha (4%) , Bloomberg (1%), and other individual financial column writers (25%) . The dataset labels which stock the news are for, so it is very simple to filter out the news for the Apple stock. The dataset also labels the date of the news, which is useful as we process the news at daily level.

The stock price data is obtained from the tidyquant package in R. We use the close price as the daily price of the Apple stock. From the daily close price we calculate daily price changes of the stock, and create a binary label that indicates if the daily change is positive or negative. This is our prediction target.

2.2 Method

First, we filter out the news only for Apple stock, and group the news by date to have a daily news collection. Then we preprocess the text data into dfm format by converting to lower case, remove

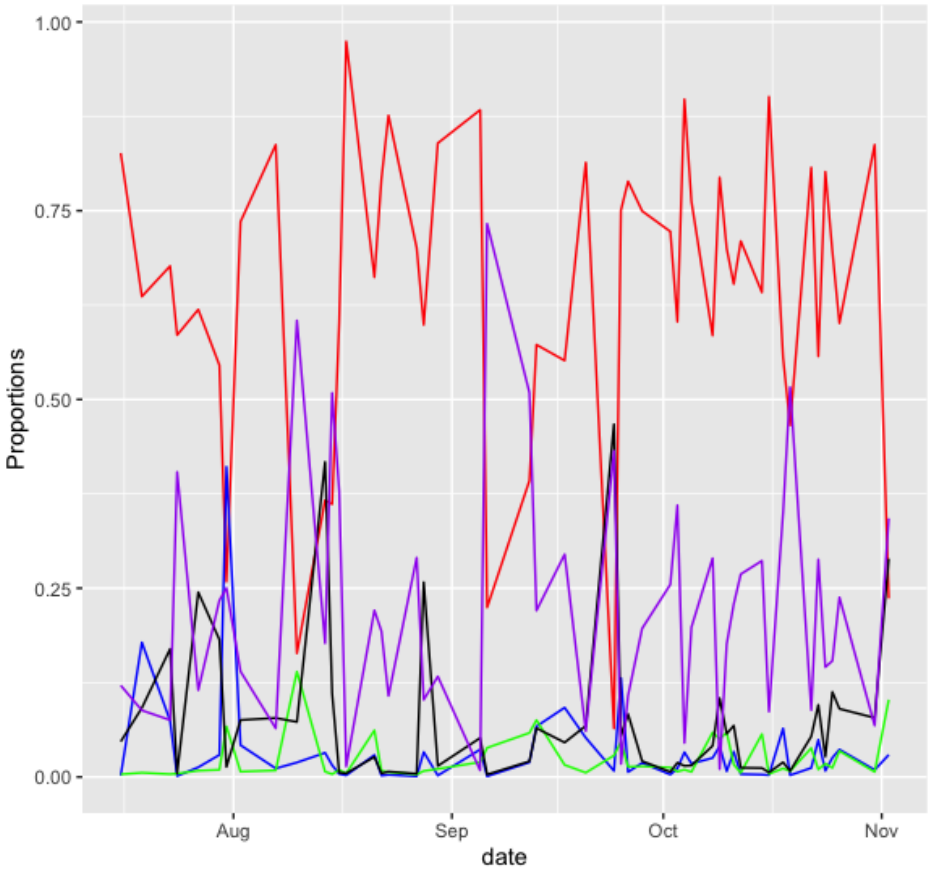

Figure 1: Topic/word proportion timeseries demonstration example

usually meaningless elements like stop-words, symbols, numbers, and stemming. So we have a daily word frequency matrix. Then we use two methods to convert the text data into time-series of proportion of counts.

The first method is simply bag of words. We find the top word frequency in the processed dfm for Apple stock, and calculate the daily proportion of the word frequencies, where the proportion sums up to 1 for each day. We consider words from top 25, 50, and 100 word frequencies as the features. Here we will also check the Kendall's correlation between the daily word frequency proportions and the daily closing stock price at lag 1.

The second method is topic modeling. We use the STM package in R to model 5,10 or 25 topics in the news. Then we obtain the proportion of the topic of each day and use that as features.

In figure 1, we demonstrate how the processed text data looks like. This is the proportion time-series for approximately 3 months, and with 5 topics/words. Here the red topic/word dominates the proportion most of the time, but can reach a very low level near start of September, where purple rises up hugely. Green topic/word stays low the entire time.

We then compare the performances of SVM, Naive Bayes, KNN, and Random Forest on predicting the target movement from previous day's news using the two kinds of features on the test set. The models will be trained using 5-fold cross validation model parameter tuning and a 80/20 train/test split on sorted time-stamps. In the end we pick the best performing model and number of features as our best models in both methods. We analyze the features and the prediction distribution to characterize how we can use the ratio of the news features to predict next day's stock movement.

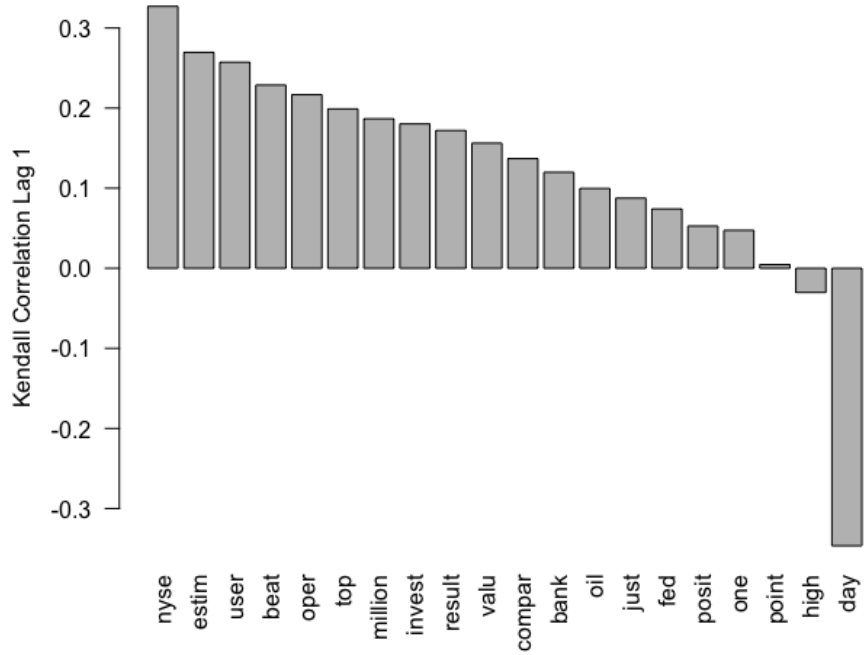

Figure 2: Kendall's rank correlation between word frequency proportions and next day's stock movement of Apple

3 Results

In figure 2 is the Kendall's rank correlation between word frequency and stock price with lag 1 day. We have a noticeable positive correlation with the word 'nyse', 'estim', 'user', and 'beat', and the most negative correlation is with the word 'day'. The word 'estim' (estimate,estimation), 'beat', and 'user' ofter comes with a positive financial report, so it makes sense for them to correlate with positive stock movement. On the other hand, we are not sure why the word 'day' has the most negative correlation with stock price. However, we can be sure that the word frequency feature can certainly contain stock movement information.

In table 1 and table 2 we present the accuracy and the corresponding 95% confidence interval for four models and 6 kinds of input features. We define the baseline accuracy to be the majority class proportion, which is around 0.555. In most cases, the accuracy are below the baseline accuracy, but such comparison is not statistically significant.

The highest accuracy achieved using word frequency proportion feature is with Naive Bayes model and top 50 term frequency proportions, around 0.546 . The highest accuracy achieved using topic proportion feature is the SVM model with 5 topics, around 0.557 . This topic model accuracy is the only one that is higher than the baseline accuracy. It looks like with topic models, we can use less number of features and achieve higher accuracy. We continue to analyze the both model in terms of their feature importance. The top 10 feature of the Naive Bayes model with top 50 word frequency are: 'googl','quarter','current','sector','ratio',consensus','service','compni','earn' and 'nyse'. This words are from perspectives like quarterly earning report, overall sector performance, and similar-stock correlation. These are certainly the perspectives human traders look at as well. For the SVM model with 5 topic frequency proportions, we have the top words in each topic in table 3. The importance of the 5 topics are ranked as 2,3,1,4,5. We can try to interpret the possible thesis of the 5 topics, and they are

1. Topic2: Stock's quarterly and yearly performances against their estimates

Baseline Acc = 0.5545977	Accuracy	AccuracyLower	AccuracyUpper
Model with top 25 Term Frequency Proportion Features			
KNN	0.5316092	0.4776672	0.5850088
SVM	0.4827586	0.4291624	0.5366506
Naive Bayes	0.4971264	0.4433768	0.5509254
Random Forest	0.5258621	0.4719349	0.5793455
Model with top 50 Term Frequency Proportion Features			
KNN	0.5143678	0.4604910	0.5679982
SVM	0.5172414	0.4633494	0.5708376
Naive Bayes	0.5459770	0.4920284	0.5991367
Random Forest	0.5000000	0.4462249	0.5537751
Model with top 100 Term Frequency Proportion Features			
KNN	0.5114943	0.4576343	0.5651570
SVM	0.5143678	0.4604910	0.5679982
Naive Bayes	0.5172414	0.4633494	0.5708376
Random Forest	0.5057471	0.4519261	0.5594695

Table 1: Accuracy of Models with Word frequency proportions as feature

Baseline Acc = 0.5545977	Accuracy	AccuracyLower	AccuracyUpper
Model with 5 Topic			
KNN	0.5287356	0.4748002	0.5821780
SVM	0.5574713	0.5035487	0.6104076
Naive Bayes	0.5258621	0.4719349	0.5793455
Random Forest	0.5028736	0.4490746	0.5566232
Model with 10 Topic			
KNN	0.5402299	0.4862787	0.5934908
SVM	0.5258621	0.4719349	0.5793455
Naive Bayes	0.5143678	0.4604910	0.5679982
Random Forest	0.5172414	0.4633494	0.5708376
Model with 25 Topic			
KNN	0.4913793	0.4376859	0.5452206
SVM	0.5316092	0.4776672	0.5850088
Naive Bayes	0.5402299	0.4862787	0.5934908
Random Forest	0.5086207	0.4547794	0.5623141

Table 2: Accuracy of Models with Topic proportions as feature

	More frequent word in the topic ->less frequent word in the topic							
Topic 1	"market"	"trade"	"stock"	"week"	"nasdaq"	"day"	"index"	"rate"
Topic 2	"year"	"zack"	"compani"	"earn"	"stock"	"quarter"	"estim"	"rank"
Topic 3	"said"	"compani"	"appl"	"china"	"reuter"	"inc"	"year"	"billion"
Topic 4	"compani"	"year"	"zack"	"quarter"	"stock"	"nasdaq"	"revenu"	"growth"
Topic 5	"appl"	"iphon"	"compani"	"market"	"year"	"new"	"can"	"time"

Table 3: Top words in the 5 topics used in the SVM model

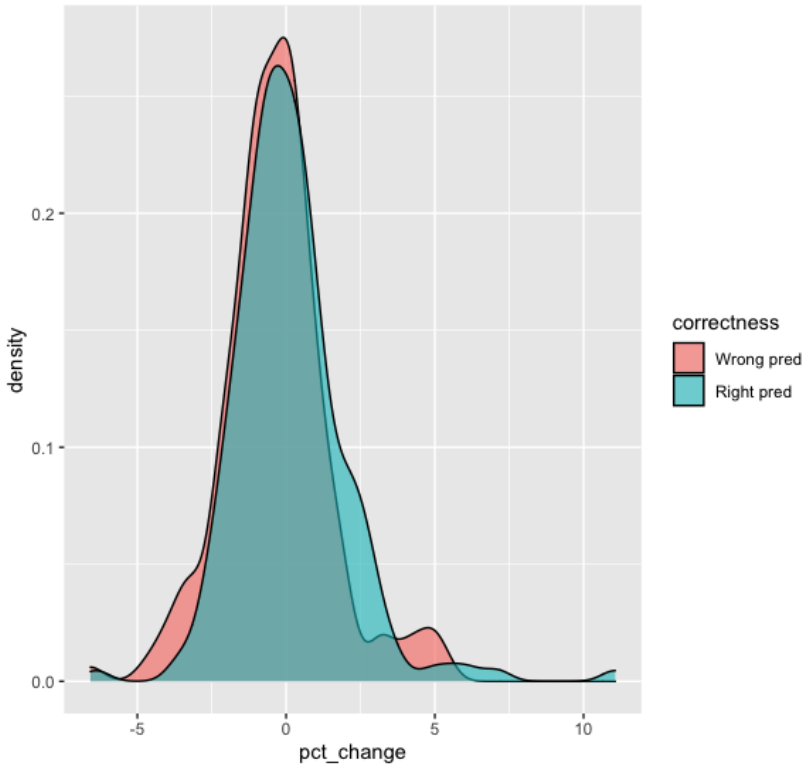

Figure 3: SVM prediction from 5 topic frequency proportions.

2. Topic3: Apple's business in China and their yearly revenue
3. Topic1: General market and index performance and Fed's rate adjustment
4. Topic4: Company's revenue growth over time
5. Topic5: Apple's market of iPhone sales

We notice that Topic 2 and Topic 1 appeared in the most important input feature of the Naive Bayes model as well. So features from both text processing methods share prediction perspectives.

In figure 3 and figure 4 we plot the prediction distribution density for correct and incorrect predictions from the two best models we picked above, against the percent change of the Apple stock of the corresponding prediction target (rather than binary). We notice that the NB model has significantly more correct predictions at small negative percentage change than incorrect predictions, while there is not much difference from the SVM model. The SVM model has much more correct predictions at medium percentage changes ($\sim 2.5\%$) than incorrect predictions. Also, both model is able to predict well when the stock movement is large, represent by large percentage change.

4 Conclusion and Discussion

In conclusion, we compared two text processing methods applied on news data to predict the stock price trend of Apple as a binary variable. We achieved above baseline accuracy with SVM model with topic frequency proportions feature. The best accuracy from using word frequency proportion feature is little below the baseline accuracy. We have analyzed the important features in the two best models using both models, and they share some popular perspectives in stock price speculation even though one uses words and the other uses topics. In the end, we analyzed the prediction distribution using the two best performing models with the 2 kinds of input features and points out the performance difference at different percentage change levels.

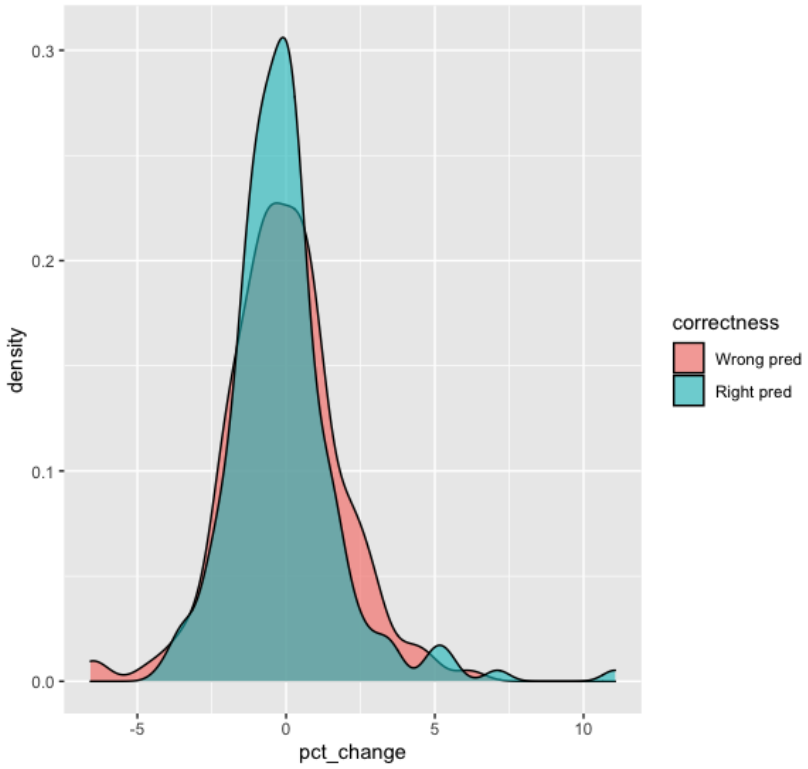

Figure 4: Naive Bayes prediction from top 50 term frequency proportions.

5 Improvements needed and Future Works

There is some aspects that are still to be explored. We should build the pipeline so we could do the same analysis for more stocks, not only the Apple stock. Also, it would be useful to test the existing methodology on smaller time spans. The time span in this project have the unit of years, but we might find more interesting aspects on time scales such as hours and minutes as the stock market could change both on big scale and extremely small scales. We should also set up more models as a comparison, especially neural network models, which are proved to perform well when there is enough data available (we have more than enough news data).

6 Reference

[1]https://nlp.stanford.edu/courses/cs224n/2007/fp/timmonsr-kylee84.pdf
[2]https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00333-6#Sec18
[3]https://www.kaggle.com/datasets/gennadiyr/us-equities-news-data

7 Github

https://github.com/messixieziyi/text_as_data_project
(not officially commented code and results)