Predict the Movement of Stock Price Using News Headlines

(CSYE 7374 Parallel Machine Learning & AI Final Report)

Introduction

Nowadays, stock market prediction is crucial to an entire industry. Stock price is determined by the behavior of human investors, and the investors determine stock prices by using publicly available information to predict how the market will act or react. It is not difficult to find there must be some relationships between news and stock price movement.

According to our accumulated common sense, Negative news will normally cause people to sell stocks. Positive news will normally cause individuals to buy stocks. Good earnings reports, an announcement of a new product, a corporate acquisition, and positive economic indicators all translate into buying pressure and an increase in stock prices.

Such phenomena arouse our curiosity. we tried to find the specific relationship between news and stock price movement. As we know, news is text data, while stock prices are digital data, so finding a way to connect numbers and text is necessary for us right now.

Description of Dataset

### Stock: Dow Jones (1992.01.01 - 2019.12.31)

* Source: [https://finance.yahoo.com/quote/%5EDJI?p=^DJI&.tsrc=fin-srch](https://finance.yahoo.com/quote/%5EDJI?p=%5eDJI&.tsrc=fin-srch)
* Data Type: csv
* Description: we choose Dow Jones as our stock price dataset, The Dow Jones Industrial Average (DJIA), Dow Jones, or simply the Dow, is a [stock market index](https://en.wikipedia.org/wiki/Stock_market_index) that measures the stock performance of 30 large companies listed on [stock exchanges](https://en.wikipedia.org/wiki/Stock_exchange) in the United States. Although it is one of the most commonly followed equity indices, many consider the Dow to be an inadequate representation of the overall U.S. stock market compared to broader market indices such as the [S&P 500 Index](https://en.wikipedia.org/wiki/S%26P_500_Index) or [Russell 3000](https://en.wikipedia.org/wiki/Russell_3000_Index). It is one of the most commonly followed equity indices, and many consider it to be one of the best representations of the U.S. stock market.

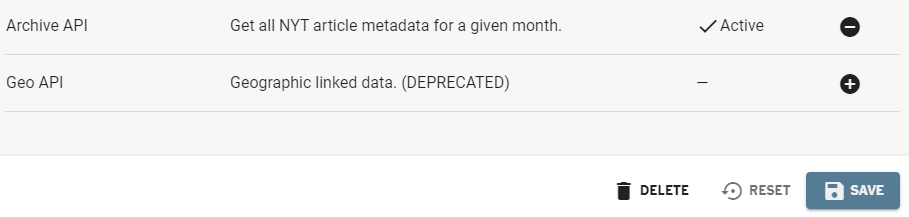
### News Headline: [HuffPost](https://www.huffingtonpost.com/)(2012 - 2018)

* Source: <https://www.kaggle.com/rmisra/news-category-dataset>
* Description: This dataset contains around 200k news headlines from the year 2012 to 2018 obtained from [HuffPost](https://www.huffingtonpost.com/).

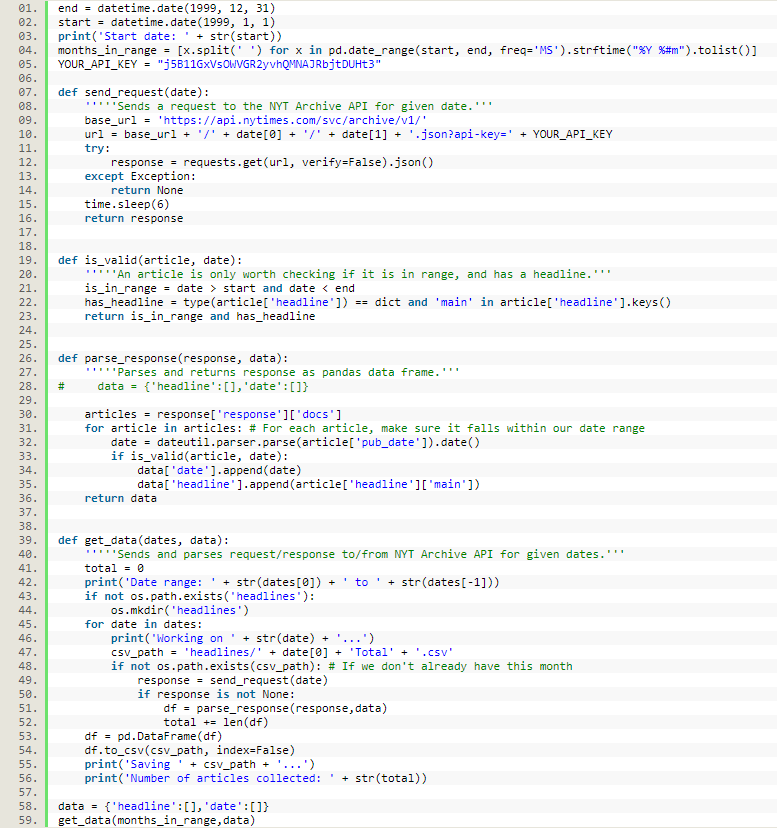
### News Headline: The New York Times(1992 - 2019)

* Source: <https://developer.nytimes.com/docs/archive-product/1/overview>)

Description: The Archive API returns an array of NYT articles for a given month, going back to 1851. Its response fields are the same as the Article Search API. The Archive API is very useful if you want to build your own database of NYT article metadata. You simply pass the API the year and month and it returns a JSON object with all articles for that month.



* Code of scraping NYT news headlines with python:



We used nearly 30 years of news headline and total size of over 500MB. After jointed our news headline with our Dow Jones stock data (deleted weekend and over long data), the file size is nearly 100MB, which is considerable large for a plain text dataset. Hope this will improve our accuracy and reflect the advantage of parallel programming.

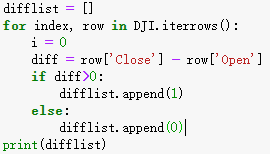
Methodology

We choose to use Jupyter notebook rather than .py file, that because Jupyter notebook makes our result more intuitive and we don’t need to re-run our whole program while debugging. Considering we are using a plain text dataset, my desktop should be able to handle this with out the boost from Discovery clusters.

We also tried to predict stock trend only using news from business category, but the dataset is too small for parallel program, and it does not improve our accuracy, so we discarded that part.

### Market up/down indicator:

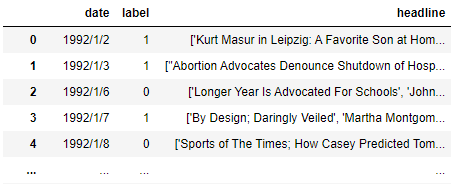
The raw data we got from Yahoo Finance includes S&P 500’s date, open price, high price, low price, close price, adj close, volume. In order to indicate the movement of stock prices more intuitively, we set the market up/down indicator:



* 1 means market goes up in that day
* 0 means marker goes down in that day

### Data joint

Due to News happens everyday but stock market only has 5 trading days per week, so tight up news and stock by date and delete the data of weekends when stocks are not traded.



### Divide data into test set and training set by date

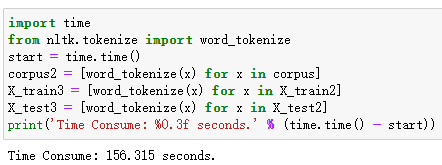


### Separate sentence into word

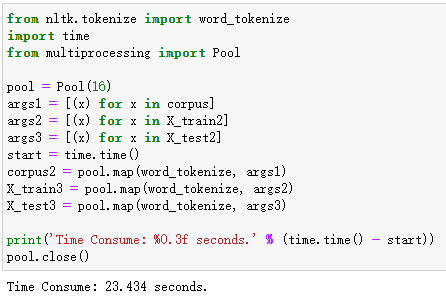
This part, we tried general method and parallelized method. We printed the time consumption in the end to help us visually compare the advantages of parallelization.

First, **nltk** package is required. Use pip install nltk to install it.

1. General method:



1. Parallelized method:

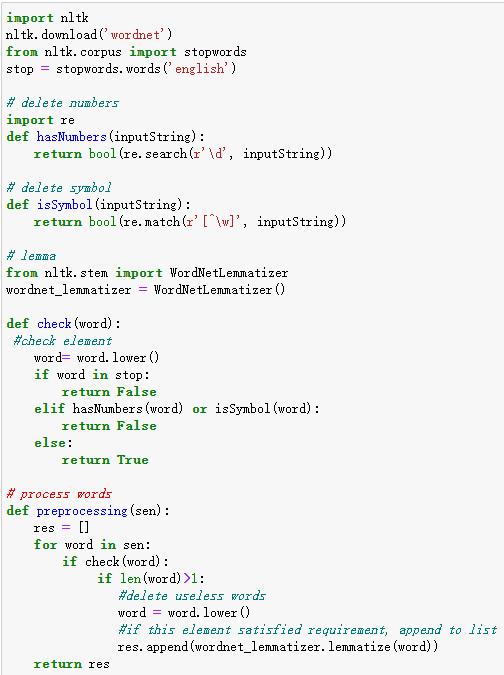


### Data cleaning

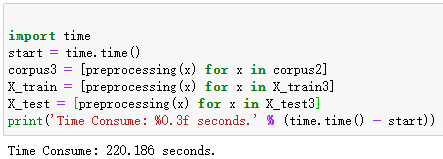
We still have general and parallelized methods to compare!

Delete redundant word: Lowercase word, stopword, number and character and drop N/A

We write a function to help us clean data:

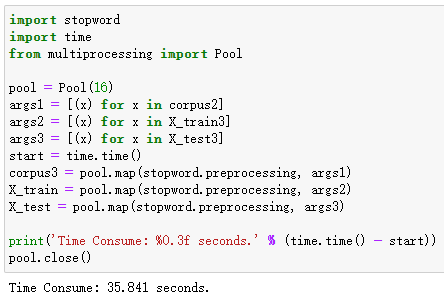


1. General method:



1. Parallelized method:

We write above methods into.py file and import them. By doing this way, we can avoid code stuck while using multiprocessing in interactive interpreter(SUCH AS Jupyter!). And made our result more intuitive

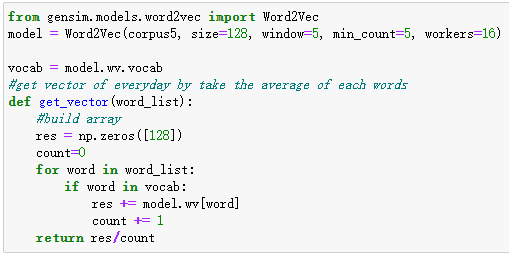


### Word vectorization

This is the key step of our project, word vectorization. Our data is news composed of words; we definitely cannot put text directly into our prediction model. Therefore, we need to vectorize the news to make the text into a number with a certain connection and meaning. So, we use a this package: Word2Vec

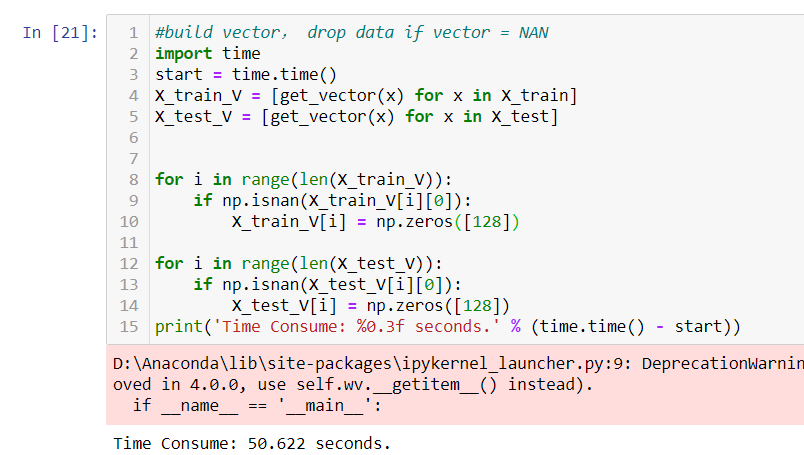
We are going to build our Vectorization Model in this step. It has a built-in parallel method by setting workers parameter. Let's set workers = 16 since I have 16 cores on local machine.

After trying lots of size from 32 to 1024, it seems set vector array size to 128 got a nice balance between running speed and accuracy. If we significantly increase this size, we may get a better result, but it seems not so good for my poor desktop.

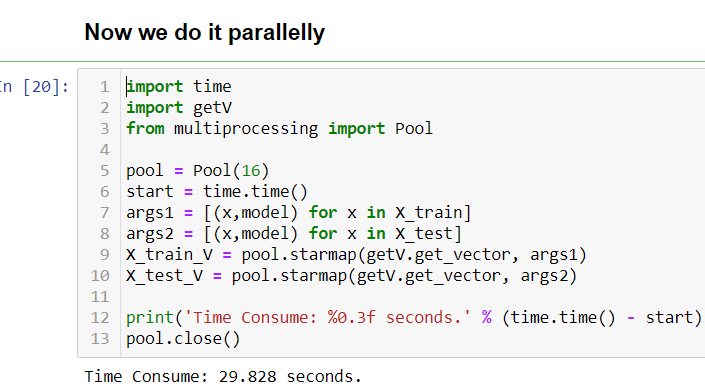


Now we vectorize our train and test data using model above

1. General method:



1. Parallelized method:



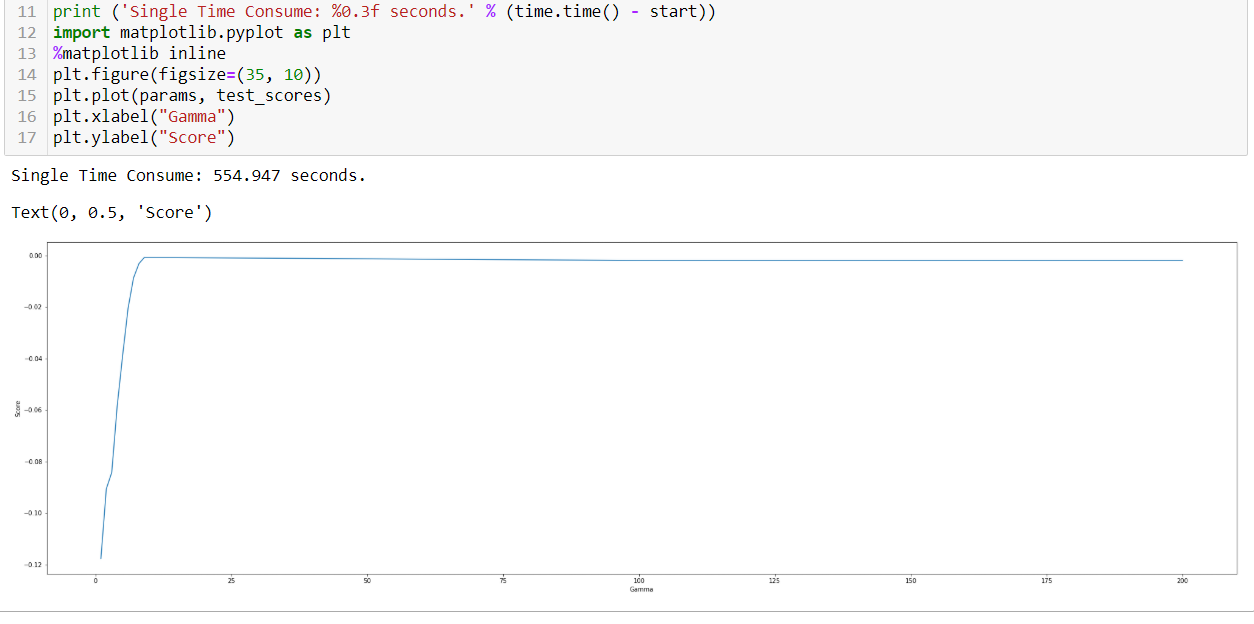
### Predict model

After tried lots of machine learning classifier, these models seems has a good fit with our dataset.

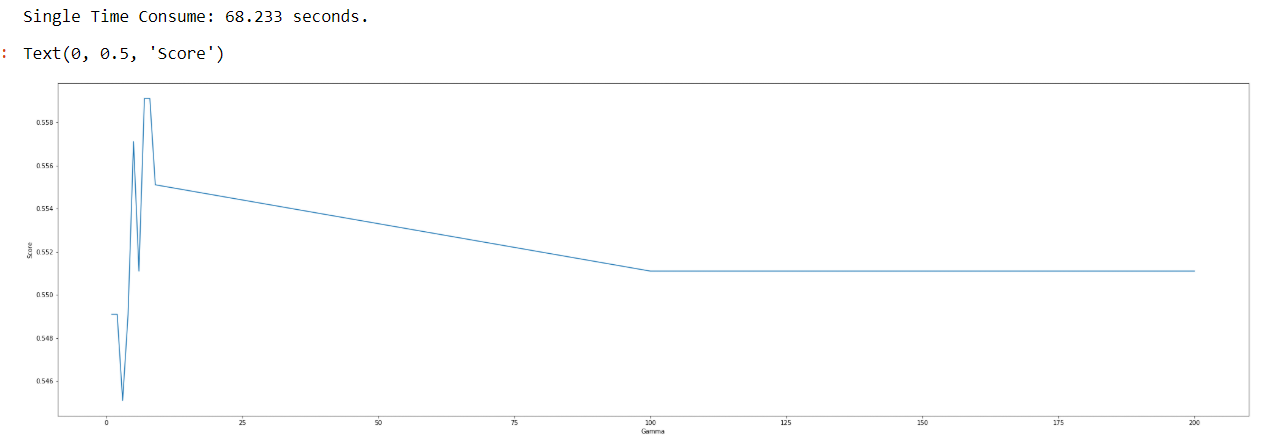
1. Support Vector Machine (SVR, SVC)
2. Naïve Bayes
3. Logistic Regression
4. Random Forest

Results and Analysis

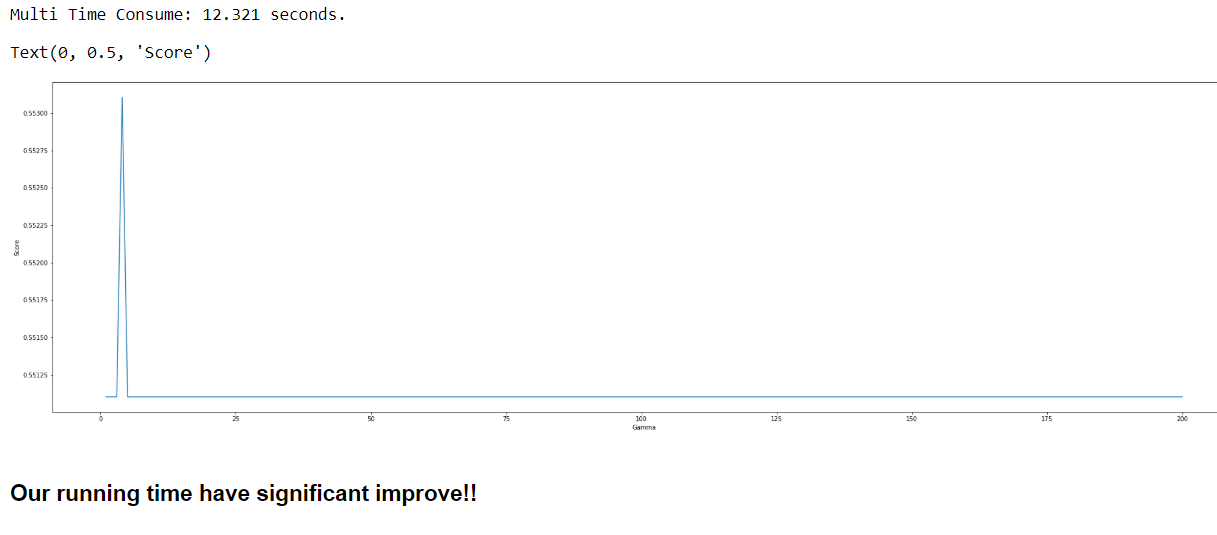
1. SVR



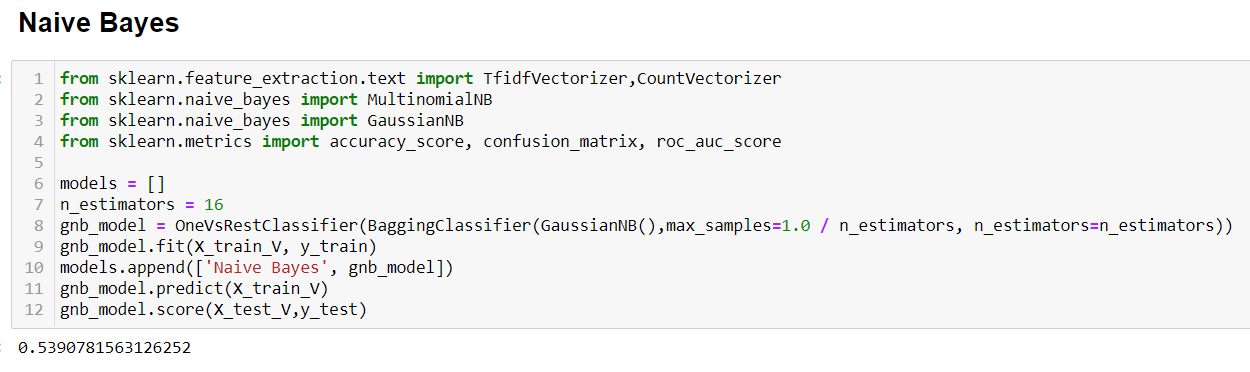
1. SVC



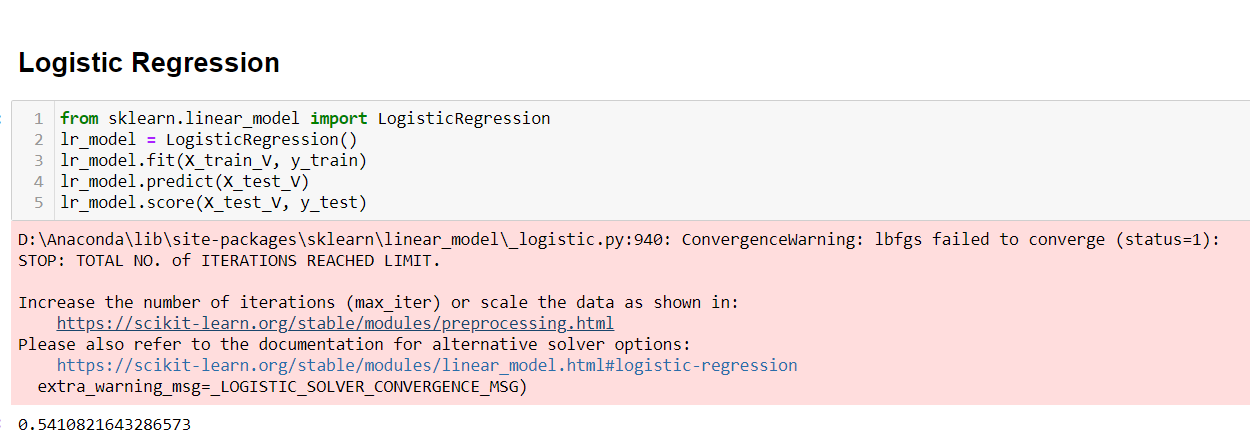
1. SVC with parallel



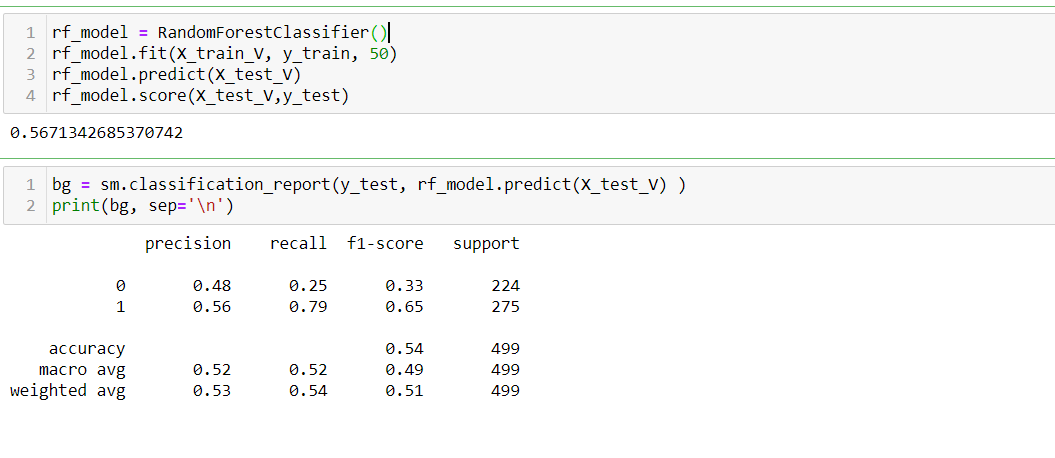
1. Naïve Bayes



1. Logistic Regression



1. Random Forest



Conclusion

From the results we listed above, we could find out that results rarely improved after testing several combinations of embedding and algorithms. No matter what input we provided, these predicted values generally around 56%. Machine learning algorithms are especially difficult to figure out connection between the word vectors and stock prices.

So why did this happen? We obviously adopted strict preprocessing of the data and calculated the most suitable model parameter values through multiple tests, but the results are still not optimistic.

We concluded the following reasons to help us explain:

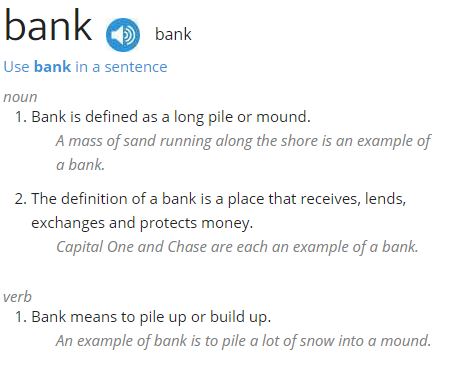
### Stock market itself is unpredictable

“Because if you could, then everybody would.” As we mentioned in our proposal : “Nowadays, there are many factors that affect the movement of stock prices, including news.” It is right, there are too many factors to take into account which affect stock prices. It is really impossible to build a precise model which would rely on all of those factors, and one of the main reasons is that most of the factors are not known beforehand: even if some events affecting stock market have happened in the past, you never know what else would happen in the future.

### The connection between text and digital

Word2vec is a method to efficiently create word embeddings and has been around since 2013. What is the word embedding? It is the conversion of "incomputable" and "unstructured" words into "computable" and "structured" vectors. This step solves that transforming real problems into mathematical problems, which is very critical step for artificial intelligence.

But word2vec ignore the influence of words order, and the relationship between words and vectors is one-to-one, which cannot solve the problem of polysemous words well. For example:



### News headlines are potentially biased

The core concepts behind the data we used are biased. The goals of news media are to attract readers and sell newspapers. Thus, their reporting is oriented in that way. Humans in general have a negativity bias and papers aim to use this to their advantage by publishing negative stories, such as a war breaking in the middle east, a fact reported every few years between 2008 and 2015. Most headlines are objective in phrasing, so there are few samples where subjectivity is not null. The few headlines that are not objective mostly contain negative sentiments, which skews the data towards a negative sentiment.

Reference

<https://blog.csdn.net/weixin_43612023/article/details/101475460>

<https://arxiv.org/pdf/1806.09533.pdf>

<https://www.quora.com/Why-is-the-stock-market-so-difficult-to-predict>

<https://towardsdatascience.com/collecting-data-from-the-new-york-times-over-any-period-of-time-3e365504004>