NLP and Supervised Learning to Evaluate the Stock Market

By Patrick Gourdet

NLP Project
University of Florida

Dobbins, Peter Johnathan pjd@cise.ufl.edu

University of Florida, Gainesville, FL 32611

Date of Talk Sunday, August 5, 2018

Abstract

Using news headlines and articles obtained from "The Wallstreet Journal," we have set out to create a supervised neural network to predict the rise and fall of the stock market. I did not implement a feature vector using both the headline and the news article verifies said sentiments. Furthermore, the stock market indexes obtained from Kaggle.com, from which I chose a ticker at random. The news articles and headlines were processed using the NLKT library. The functions used are the word tokenizer, lemmatizer, stop word extract. The tokenizer separates each word within each article and headline. The tokenized cells then lemmatized, in this process, the synonym words consolidated into one representation for each word. Then the processed data is run through the sanitizing function, removing the stop words paving the way to sentiment analysis. There are multiple methods of analyzing data, but in this paper, we focus on the relative proportional differences bounded from -1 to 1. Using the respective positive and negative connotations of words within the scrapped data, denoted P and N, respectively, result in the sentiment of current world occurrences. Utilizing a sparse matrix populated with the resulting sentiments, to train the neural network to predict possible stock market outcomes. Logistic regression applied to the data points resulted in the baseline. The resulting baseline inferred a 55% accuracy before normalization, and 63% to 83%, depending on the ticker index. Following the configuration of the baseline, we then proceeded to train the Gaussian, Bernoulli, and Multi-Naive Bayes modules. The utilization of multiple models is known as "Random Forests," this, in most cases, issues more stable outcomes. The "Random Forests," achieved a constant accuracy of > 80%. To solidify my findings, I then used SVM (Support Vector Machine) classifier, a discriminatory algorithm which, is used to assign new values added to the kernel to the respective groups. Using variables with already known outcomes, I was able to further the confidence in the obtained results. Furthermore, to eliminate any possible bias, I then applied a cross-validation algorithm, obtaining a > 80% using 6-fold validation. Future steps are needed to train models for more specific stock indexes. An example would be to collect tech headlines and news articles and test it against tech companies such as Intel or AMD.

Introduction

NLP (Natural Language Processing) is utilizing the basses of information transfer, communication, and daily interactions between the human population to extract useful data applying such data to computer applications respectfully. Functions such as test completion and text to speech for the visually impaired are applications that have proven the usefulness of NLP in today's high paced world. NLP, in conjunction with neural networks, we can predict the possible outcome of various scenarios, such as voting results and the retention rate of clients and customers.

Problem Domain

The stock market is an entity that seems to have a life of its own. The daily rise and fall, unpredictable and making money almost like a gamble that a person must be willing to take to be successful within the stock market. It is evident; there is a need for risk management. The ability to process daily verbal interactions in the form of text has proven to be a useful tool. The language processing, in conjunction with neural networks indicators, can manage the risk of the stock market. The steps to reproduce the indicators are listed as follows.

- The implementation of a pyhton script and the octoparse API, I was able to collect past news from the “The Wallstreet Journal” web site. The Wall Street Journal website contains data from 2012 to the present.
- After completion of the data harvesting, 2190 days of news with 159 news headlines and articles.
- The stock market values obtained from Kaggel.com contained all the stock indexes of the global market.
- Using sing Excel I then sanitized and aligned the news text with the stock market readings, removing the lines which correlated with the weekends.
- Continued data processing by creating scripts using the NLTK library. The script tokenizes the words within an article, removes stop words, consolidates synonymous words, and removes any non-English characters.
    - Implemented the semantic analysis for the preprocessed data.
    - Implemented the baseline model using logistic regression.
    - Implemented classifiers and verification methods.

Previous Work

Previous work conducted in this field consisted of first creating feature vectors and utilizing said vectors to calculate the frequency of occurrences within a given document or news headline. Such works as [2] used feature vectors and applying the inverse document frequency to calculated vectors [1]. Using regression analysis to calculate the stock market outcome using feature vectors [0].

Corpus

In my corpus, I have described the alternative methods used to obtain and train models that can minder the risk when investing in the stock market. The pure and direct sentiment of each headline provides the ability to circumvent risk, as shown in the following corpus.

3.1 Headlines and Articles

Using octoparse to scrape was relatively straight forward, pointing the said program by inputting the URIs in question. Obtaining achieved news data for further processing. The operations implemented where: Loop following steps from last day to the first day of each year from December to January, ranging from 2011 to 2017. 1 Extract all top headlines from the last day of the year. 2 Move to the next day.

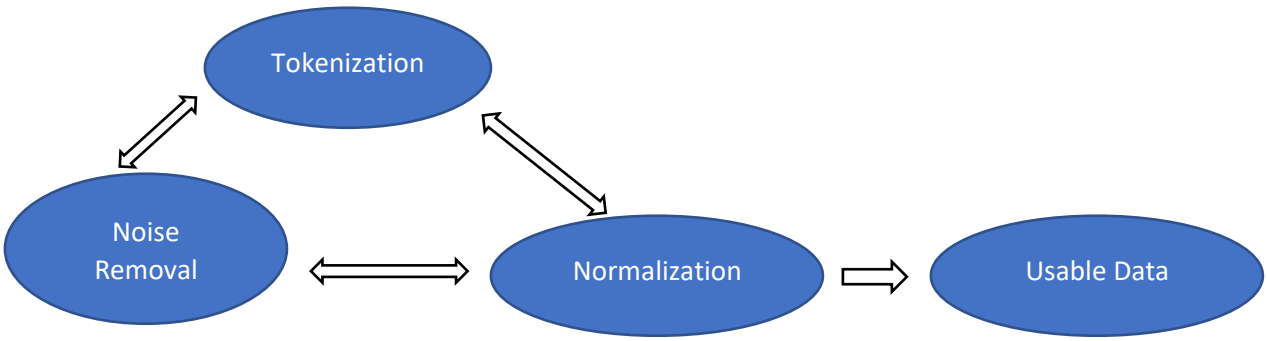
3.2 Stock market Index

Downloaded data set from Kaggle.com then first selecting one index at random and then later a global index. The values ranged between -1 and 1. The opening value depicted V1 and closing V2. Receiving the value by V2 – V1 = +-V3;

3.3 Preprocessing

Using the NLTK library, I preprocessed data in order to clean said data and prepare it for the following sentiment analysis. The tokenization, Normalization, and Noise removal.

3.4



3.4

The sentiment analysis procedure first used the TextBlob sentiment function. This function calculates the polarity. The polarity extrema -1 to 1. This function has three values of polarity subjectivity and intensity. The polarity of positive and negative words is denoted **P**, and the intensity from the modifier words denoted **I**.

We obtain the sentiment of an article or headline by:

$$\pm P_1 * \pm P_{i-1} * \pm I_i * \pm I_{i-1} = \text{Sentiment}$$

Populating these values into a sparse matrix:

$$A_{sparse} = \begin{bmatrix} 0 & A_{12} & A_{13} & 0 & 0 \\ 0 & A_{22} & 0 & 0 & 0 \\ 0 & 0 & A_{33} & 0 & 0 \\ 0 & 0 & A_{43} & A_{44} & 0 \\ 0 & A_{52} & 0 & 0 & 0 \end{bmatrix}$$

Then for the stock market index, denoted $Yi\ and\ X1 \cdots$ Xi the normalization process to remove the polarity obtaining values from 0 to 1. Using the normalization formula N = (X - Xmin)/

(Xmax-Xmax).

3.4

Model Creation

Using Logistic Regression:

The sigmoid function taking any real input and outputs a value between 0 and 1 or takes input odds and outputs the probability.

$$P = \frac{1}{1 + e^{-(-.5596 + 1.2528X)}}$$

Gaussian Naïve Bayes:

We deal with continuous data using the gaussian model. Using the mean of x compared to the entirety of class C and the variance sig in comparison to the class C, and computing the normal distribution.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Bernoulli Naïve Bayes:

The Bernoulli function uses Boolean occurrences rather than the frequency of the terms within a document.

$$p(\mathbf{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i} (1 - p_{ki})^{(1-x_i)}$$

Decision Tree:

Breaking down the data into smaller and smaller portions, leaving branches preceded by nodes which are refereed to as decision nodes.

$$Entropy = \sum_{i=1}^{C} -p_i * \log_2(p_i)$$

Multinomial Naïve Byes:

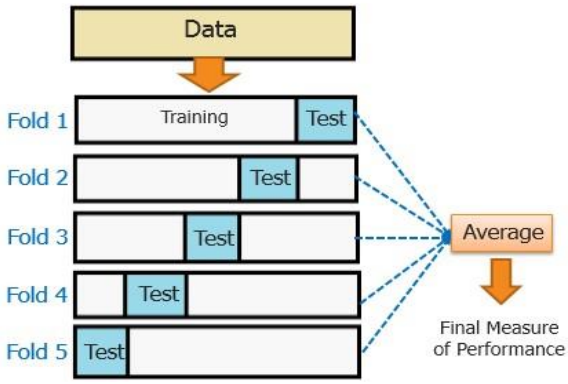
With multinomial Naïve Bayes which is uses the word frequency of a word within a text document.

$$p(\mathbf{x} \mid C_k) = \frac{(\sum_i x_i)!}{\prod_i x_i!} \prod_i p_{ki}^{x_i}$$

3.5

Verification Methods

Using leave one out cross-validation method, we partition the data into train and test at

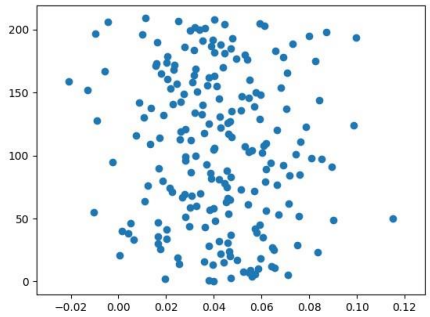


random and recalculate the accuracy in each iteration. Doing so validates the classifier for accuracy.

Results

To test the method, I created a subset of the data, consisting of 300 data points. Combined amounted to 2 years of stock market index recordings. After preprocessing 3.3 and calculation of the sentiment 3.4, The stock market index chosen at random was Facebook values for the

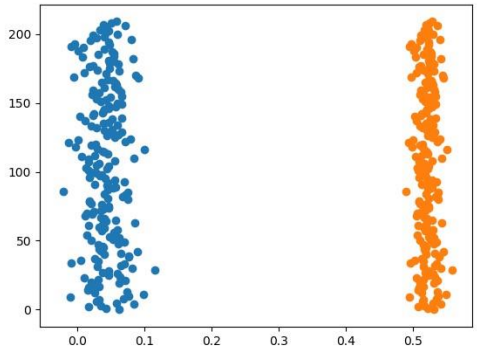
corresponding days of news. Taking the data points and visualizing the data without the stock market indexes depicted below.



As we can see, the data is evenly distributed; this posed a problem. We applied the above data to generate the baseline using regression analysis, which only gave us a 46 to 54 percent accuracy depending on how the training data is randomized. Furthermore, verify the outcome, the Gaussian and Bernoulli Naïve Bayes Classifiers algorithms processed the kernel, resulting in the outcome shown below.

Logistic Regression
Train Score: 1.0
Test Score: 0.5571428571428572
Gaussian Naive Bayes
Train Score: 0.8785714285714286
Test Score:0.5714285714285714
Bernoulli Naive Bayes
Train Score: 0.8285714285714286
Test Score: 0.5428571428571428

Un-normalized data as a predictor is, as shown above, not to be used in a final product, but gives a good baseline. After the normalization using the formula in 3.4 we received a more desirable outcome. After visualization, we see the results below.



Already after visualization we can see a distinct differentiation and applying the data to the prior models

Logistic Regression
Train Score: 1.0
Test Score: 0.8142857142857143
Gaussian Naive Bayes
Train Score: 0.9285714285714286
Test Score: 0.8285714285714286
Bernoulli Naive Bayes
Train Score: 0.9285714285714286
Test Score: 0.8571428571428571

We see after running the same methods a tremendous increase in accuracy this outcome thus allows us to apply the more substantial data set to the resulting models.

Large Data Set

After applying the preprocessing methods and normalization to the extensive data set and running the same calculations, I received the results depicted below.

Logistic Regression
Train Score: 1.0
Test Score: 0.575
Gaussien Naive Bayes
Train Score: 0.8455056179775281
Test Score: 0.575
Bernoulli Naive Bayes
Train Score: 0.7219101123595506
Test Score: 0.525

Looking at the resulting numbers I realized, I ran into the first problem, I then reiterated to see if the data may be corrupt. After taking a closer look, I found that the kernel is processed properly. Next, reevaluating the problem, I noticed I was using world news data. I was world indicators to predict an unrelated index. To predict a marker such as Facebook, the data set must be different. Thus I then decided to use the NASDAQ as an index. I reconfigured the data set and was able to rerun the models and received the results depicted below:

```
Logistic Regression
Train Score:  1.0
Test Score:  0.6401869158878505
Gaussien Naive Bayes
Train Score:  0.859122401847575
Test Score:  0.7336448598130841
Bernoulli Naive Bayes
Train Score:  0.8429561200923787
Test Score:  0.8738317757009346
```

After running the model about 10 times, I was able to calculate the average of 84% after 10 iterations.

Thus I decided to move to the next step and implement a Decision Tree and an SVM classifier for the data set in question and test this against brand new data obtained from the “The Wallstreet Journal” website.

```
Logistic Regression
Train Score:  1.0
Test Score:  0.6401869158878505
Gaussien Naive Bayes
Train Score:  0.859122401847575
Test Score:  0.7336448598130841
Bernoulli Naive Bayes
Train Score:  0.8429561200923787
Test Score:  0.8738317757009346
Multi Naive Bayes
Train Score:  0.8406466512702079
Test Score:  0.8785046728971962
Decission Tree
Train Score:  1.0
```

```
Test Score:  0.8084112149532711
```

This is a very satisfying outcome and can definitely allow for further evaluation of this implemented method. Before the completion of the results, I wanted to verify each classifier using the leave one out method to verify that the results were in fact stable. These are depicted below:

```
Multi Naive Bayes Cross Validation
[0.83333333 0.83333333 0.83333333 0.83333333 0.85106383 0.84782609]
Decission Tree Cross Validation
[0.6875     0.6875     0.75       0.66666667 0.80851064 0.67391304]
SVMCross Validation
[0.83333333 0.83333333 0.83333333 0.83333333 0.85106383 0.84782609]
```

We only see slight variations within the models and thus will deem this method to be valid and proven for said data sets.

Summary

After the implementation of this process, I was able to determine that the raw sentiment analysis or namely the polarity of a text is indeed indicative of the rise and fall of the stock market. I was also able to establish that not all data is suited to acquire a blanketed result as all stock market indexes react to industry-specific occurrences. Each stock market index requires news data specific to its range i.e.; global news is a great indicator for global index values.

Further steps

What is left to be shown is that with more specific news articles, we are also able to predict more specific stock market fluctuations? Such as technology data is needed to establish the rise and fall for the mean of all tech-related stock. Then to go further into detail news about specific markets with the tech industry, we could then predict even more finite stock indexes such as Facebook or Apple stock.

Standard and Constraints

The methods used in this project are the standard Python libraries such as the NLTK for language processing, Numpy, and Scikit Learn for statistical processing. Furthermore, the constraints posed upon Python 3.

References

[0] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. Smith, “Predicting Risk from Financial Reports with Regression.” [Online]. Available: https://homes.cs.washington.edu/~nasmith/papers/kogan levin routledge sagi smith.naacl09.pdf. [Accessed: 2018].

[1] S. Maskey, “https://pdfs.semanticscholar.org/presentation/1417/b6447cf4aee14e86a3262ebaed8faa64a18f.p df.” [Online]. Available: https://pdfs.semanticscholar.org/presentation/1417/b6447cf4aee14e86a3262ebaed8faa64a18f.pdf . [Accessed: 05-Aug-2018].

[2] H. lee, M. Surdeanu, B. MacCartney, and D. Jurafsky, “On the Importance of Text Analysis for Stock Price Prediction.” [Online]. Available: https://nlp.stanford.edu/pubs/lrec2014stock.pdf. [Accessed: 05-Aug-2018].

[3] Boyi Xie, Rebecca J. Passonneau, Germn Creamer, and Leon Wu. 2013. Semantic frames to predict stock price movement. In Proceedings of the 2013 Annual Meeting of the Association for Computational Linguistics.

[4] Bhard, R. Anshul, S. “Extracting Emotions from News Headlines ”, http://cs229.stanford.edu/proj2013/RaghavanSamar-ExtractingEmotionsFromNewsHeadlinesSemEvalAffectiveTextTask.pdf, Stanford University (as-of Dec 23 2013)

[5]Anon. “Computer Statistics in Python” https://people.duke.edu/~ccc14/sta-663/index.html (retrieved on 05/20/2018)

Anon. http://www.nltk.org/ (retrieved on 05/18/2018)

Sentdex “Practical Machine Learning Tutorial” Online video. You Tube. You Tube Apr. 10, 2016

[6] Nelson, P. “"Cruising the Data Ocean" Blog Series” , https://www.searchtechnologies.com/blog/natural-language-processing-techniques,(retrieved on 05/18/2018)

[7] Keles, S. “Theory and Aoolication of Cross-Validation in Linear Regression Models”, ftp://ftp.cs.wisc.edu/pub/users/keles/849_TEX/lecture_110308_CV.pdf , (retrieved on 05/18/2018)

Used stop words long from https://www.ranks.nl/stopwords

Git hub replace code https://gist.github.com/tushortz/9fbde5d023c0a0204333267840b592f9

https://www.investors.com/news/technology/fang-stocks-get-positive-reviews-ratings-with-newcoverage/

https://web.archive.org/web/20051231051202/http://www.bloomberg.com:80/news/regions/us.html