Natural Language Processing: A Short Investigation

Cecilia Aubrun, Conor McIndoe, Francesco Sciacovelli

MSc Mathematics and Finance
Imperial College London
December 2019

Contents

Abstract

In lieu of the massive quantity of data the information age has made accessible, quantitative trading firms have increasingly sought to leverage artificial intelligence techniques to generate alpha from chaotic sources.

Natural-language processing is a ubiquitous approach, attempting to parse news headlines and social media content in real time to develop trade signals. Our paper investigates the ability of an NLP to produce profitable trading opportunities, by considering a specific architecture and testing its performance.

Inspired by The BlackRock Applied Finance Project : Using Natural Language Processing techniques for Stock Return Prediction, this paper presents our work on the utilisation of Natural Language Processing on headlines in order to predict Stock Return of the S&P500 companies. The first part describes the methodologies we used, then the second half present our partial and final results. The results were mixed. Indeed, we found that news headlines are statistically neither a good predictor for the direction of the returns neither for the magnitude of the evolution of the returns. Those results are largely due to some vagueness in the determination of the sentence's structure.

1 Introduction

Natural-language processing (NLP) is often thought to have begun in the early 1900s when Ferdinand de Saussure began to develop a theory of how meaning is created in language. Noam Chomsky's book *Syntactic Structures* (1957), considered suitable approaches to parse language content into computer-readable form [5]. Today, NLP is increasingly prevalent in the toolkit of quantitative trading desks. A recent *Deloitte* report found in the firms participating in the study, 70% have deployed artificial intelligence (AI) techniques, and 60% are using NLP [4].

Natural Language Processing is a field of artificial intelligence that attempts to algorithmically extract meaning from human language prose. If text is to be used in a machine learning framework, unstructured sentences must be parsed in a machine-readable form. The NLP practitioner has many methods to choose from; typically these will involve separating a phrase into word tokens, identifying some structure in the sentence, and inferring the semantics. The field is challenging: due to the complexity of human language, there are several possibilities in which one may express the same idea or concept.

Natural Language Processing is prevalent in many fields today. It has applications in translation services, smart search suggestions, chatbots and auto-correction to name a few [6].

The inspiration for the methodology used in the following paper has to be mainly traced back to the project carried out by BlackRock in association with the University of Berkeley [1]. Though the core of the methodology is kept, some techniques differ. We follow several suggestions directly provided by the authors of the project; indeed our model is trained strictly on financial news articles and not on general English corpus, therefore every word is *learned* in the same context in which will be found afterwards, avoiding in this way an ambiguity problem as well as increasing the chances of our algorithm of finding specific financial terms.

We also chose to avoid the implementation of the Name Entity Recognition (NER). This is a part-of-speech mapper which identified, amongst others, organisations and persons. We will eventually only consider subject and objects of sentences as potential companies, and since we check all such elements for validity of a company name, we will not be employing an NER to map organisations to company names prior to this step.

2 Problem Statement

We wish to investigate to what extent a news headline can predict the movement of a stock price. We consider the conjecture that real-time processing of news content can provide profitable trading strategies by providing information about future price movements.

In particular, we investigate the suitability of the NLP we generate using *subject, verb, object extraction* (SVO), Word2Vec vectorisation, and Ball Tree / Nearest Neighbour clustering as it applies to financial predictions.

3 Methodologies Overview

Methodology	Goal	Description
Tokenisation	Data Cleaning	Transforming sentences into lists; deleting some elements such as punctuation
Stop Word Removal	Data Cleaning	Removing words that do not provide information, ('and', 'to', ...)
Subject Verb Object	Structuring	Providing structure to phrases, parsing as vectors (subject, verb, object)
Name Entity Recognition	Organizing	Substituting organisation names for ticker values
Word2Vec	Structuring	Equipping the set of words with a vector space structure
Clustering	Structuring	Grouping vectors with similar coordinates in the vector space
Mapping	Meaning	Developing a semantic interpretation of vector clusters

Table 1: High-Level Methodologies Overview

3.1 Data Extraction

Drawing inspiration from the main article, [1], we made use of the *Wayback Machine Downloader* [9], which provides a convenient API for batch-downloading historical website snapshots. From these, we are able to collect the html page content, from which the Python library *beautiful soup* is able to extract key tags, such as *datePublished* and *title*.

We targeted several financial news outlets, and wrote scripts to parse the html output of each. In total. Our dataset contains some 65,000 headers initially.

3.2 Data Cleaning

Each headline is cleaned using a process made up of two main steps. First it is *tokenised*, meaning it is transformed in a list in order to delete any punctuation. *Stop words* are then removed; these words which do not provide information to the meaning of the sentence, and their removal aids part-of-speech extraction. The list of stop words was built from the *nltk* library, and extended with custom words which in our case are mostly auxiliary verbs.

3.3 Subject, Verb, Object (SVO)

To determine structure for a given headline, we extract the sentence subject, verb and object as a tuple we hope can retain most of the sentence's information content. Our main reference diverges from us on this point, parsing each headline through a Name Entity Recognition (NER) algorithms. This is used by the team from BlackRock and Berkeley [1] to determine the all "organizations" in a given sentence. Afterwards they apply the SVO algorithm to extract the subject, verb and object of the sentence and their analysis then just uses these three items for the following steps. As mentioned, since our analysis is SVO-based, we proceed instead by identifying sentence subjects and objects and then checking whether these can be mapped to a corresponding S&P500 ticker.

A company may be sensibly referred to by a news outlet in one of many ways. *American Express* can also be found in headlines as simply *AMEX*. This is challenging, since *Goldman* will almost always refer to *Goldman Sachs*, but *American* may be used in many contexts. In our report, we ended up insisting that either a ticker was present in a headline, or else the entire company name was present. We note that this is perhaps overly-limiting in many cases, but due to the noisy dataset, we would rather take a cautious route.

As an explicit example, the headline *American Express releases a new card* would, after this step, become (__AXP,release,card)

3.4 Phrase to Vector

This step constitute the transformation of data non quantitative (words) into figures. Using the *Word2Vec* Python package, we transform the words into vectors of coordinates which allow to place the word into an n-dimensional metric space that describes the words space where words that have a close meaning will be map into close vectors.

To do so, the model needs to be trained. Unlike to the authors who trained it with 2millions random words, we trained it with 48,000 financial headlines [2].

The Word2Vec documentation [11] provided by Google, the developer, explains that the Word2Vec package may be used via two methods. The first is the *continuous bag of words* model, the second is the *skip gram* model.

We decided to use the default method which is Continuous Bag Of Words. A separate database of financial headlines containing some 48,000 financial headlines [2] was used to train our Word2Vec model. The dates of the articles range from July to October 2015.

This methodology allows us to map a word to a coordinate in some high-dimensional $\mathbb{R}^d$. In order to parse the phrase itself as a vector, we use the concatenation methodology which, amongst those methods considered in the main reference, yields the most powerful results. That is, given a verb vector $V_{verb} = [x_1, \dots, x_n]$, and an object $V_{object} = [y_1, \dots, y_n]$, the phrase is assigned the vector

$$V_{verb+object} = [V_{verb} V_{object}] = [x_1, \dots, x_n, y_1, \dots, y_n]$$

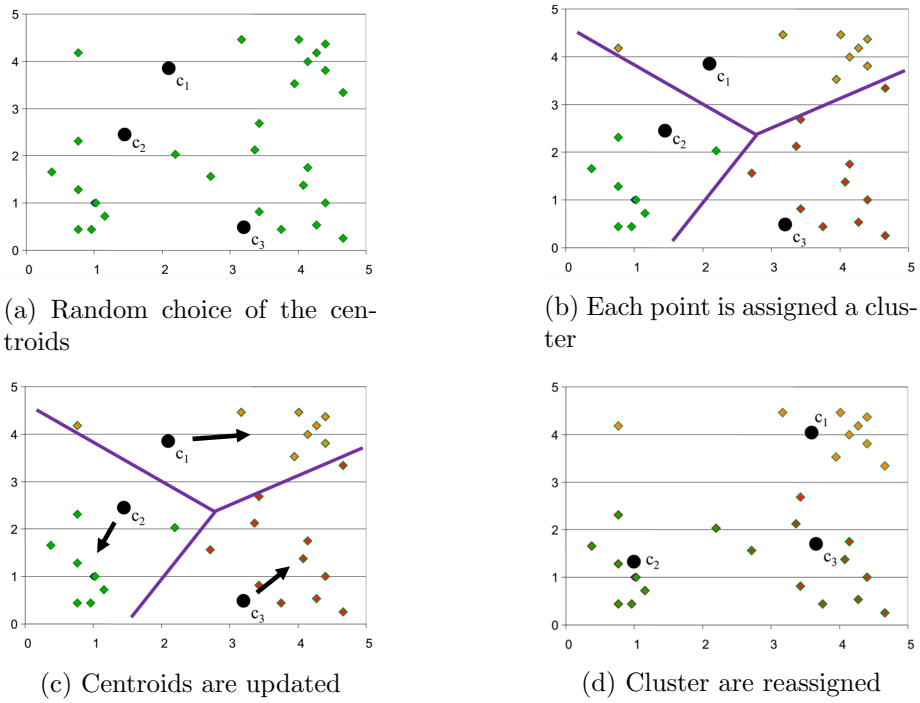
This final vector space we will denote $\mathfrak{F}$.

3.5 Clustering

When assigning a sentiment to each point in the verb-object vector space $\mathfrak{F}$ we have now developed, it is likely that each point will have only one corresponding headline from which it is mapped. We therefore have a problem of sample size. To address this, we cluster *close* vectors together, and declare sentiment homogeneity across the clusters.

We will use a K-nearest-neighbour algorithm to perform the clustering. The *KNeighborsClassifier* method of the *sklearn.neighbors* Python package provides the required functionality, with parameter set to *ball tree*.

The results of this part are the clusters with their centroids, which design the mean of each cluster. A visual example for a two dimensional space is given in the Figure 1 ([12]) where we can see the first iteration of the KNN algorithm from the first random selection of the centroids up to where the first clusters are created and the centroids adjusted. The whole process stops when the centroids are not updated at the end of an iteration.



(a) Random choice of the centroids

(b) Each point is assigned a cluster

(c) Centroids are updated

(d) Cluster are reassigned

Figure 1: First Iteration of KNN algorithm

3.5.1 Training Headlines Sentiment

In order to investigate how predictive headlines are of stock movements, we use the *Yahoo Finance* Python API to collect returns data for all relevant stocks over the time horizon our headlines are collected from. This is the past 10 years of trading.

In order to examine the effect an event on a stock's price, we consider the stock's *abnormal returns*. This is the returns of a stock over and above the expected gain (or loss). To this effect, we partition by industry classification, according to the *GICS Organisation Classification* [3]. For a given time period and industry classification, we can compute industry-wide performance. This dataset also provides industry *beta*, relative to the underlying industry, from which we can use the *Capital Asset Pricing Model* to infer the expected price movement for a given stock.

To compute expected returns of a stock in a given window, we make use of the Capital Asset Pricing Model (CAPM). Under this model, the expected returns are computed as:

$$E(r) = r_f + \beta[E(r_m) - r_f)]$$

where r_f is the risk free rate. As a proxy for this value, we took the 3-month Us Dollar LIBOR

rate. The abnormal returns we attribute to company-specific events, and hope that these can be inferred from news headlines.

For each cluster, we infer the constituent points' sentiment using this method, and take the average to define a *cluster sentiment*.

3.6 Interpolating a Sentiment Mapping

We now wish to extent to a full mapping $\varphi : \mathfrak{F} \to \mathbb{R}$ from the vector space of verb-object points onto a value representing a sentiment score. We want this map to be consistent with our observations $[(x_1, s_1), \ldots, (x_n, s_n)]$, and therefore require $\varphi(x_i) = s_i$ for all i in $\{1, \ldots, n\}$. We consider two approaches.

i For $x \in \mathfrak{F}$, set $\varphi(x)$ equal to the sentiment of the closest cluster

ii To use Shepard's inverse-distance interpolation method [10], fixing the observations.

Shepard's method for inverse-distance interpolation [10], prescribes a general method for generating a smooth function through a set of observations $[(x_1, s_1), \ldots, (x_n, s_n)]$, with contributions from each observation weighted according to a positive power of the inverse of the distance to that point.

Formally, given a point x in a metric space (X, d), and a parameter $p > 0$ we compute the weights

$$w_i(x) = \frac{1}{d((x, x_i)^p}$$

corresponding to the contribution from each observation. We want to normalise by the sum of the weights, so define $\tilde{w}_i = w_i / \sum_{i=1}^{n} w_i(x)$ and then the inverse-weighted interpolation is given by:

$$\varphi(x) = \begin{cases} \sum_{i=1}^{n} \tilde{w}_i(x) s_i & \text{if } d(x, x_i) > 0 \ \forall 1 \leq i \leq n \\ s_i & \text{if } d(x, x_i) = 0 \end{cases}$$

Here, we have $(X, d) = (\mathfrak{F}, ||\cdot||_2)$. Shepard's method has desirable properties which we would expect of our imputed function. We investigate both functions' performance over the test data in our results section.

4 Results

4.1 Data

After cleaning the 62526 headlines thanks to NLP methodologies, selecting the headlines concerning only one of the S&P500 companies, we ended up with 1093 Headlines: 876 training data + 217 tested data.

The frequency of appearance of the S&P500 companies in the headlines is illustrated in the figure 2

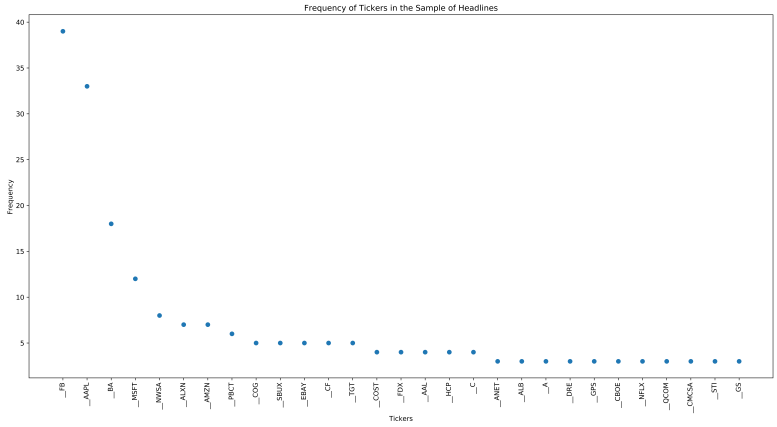


Figure 2: Frequency of appearance of the S&P 500 companies

The frequency of appearance of the S&P500 company's classification in the headlines is illustrated in the figure 3

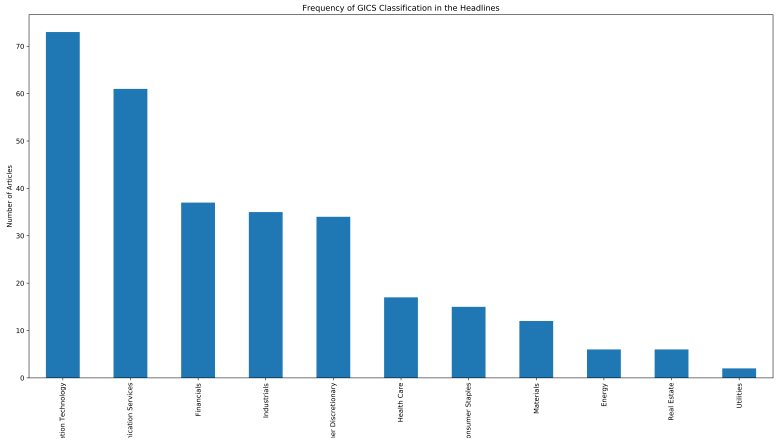


Figure 3: Frequency of appearance of the S&P 500 company's classification

4.2 Training our model

After cleaning, clustering and mapping the training headlines, we obtain clusters that can be illustrated through a python dictionary printed on figure 4.

Figure 4: Example of a cluster

Moreover, every time the code is run, a CSV file Clusters_dictionary.csv is created in the GitHub folder, thus, it allows to see all the clusters built in this part.

4.3 Tested Data

For each headline, we computed the average daily abnormal returns for several horizons that follow the headline date (3, 5 and 10 days in the IPython notebook). We can see in Figure 5 the first 10 entries of the DataFrame containing the test headlines with the corresponding abnormal returns for the 3 times horizons.

	date	headline	ticker	3_days	5_days	10_days
0	2018-11-26	boeings corporate giving exceed usd – financia...	__NWSA	-0.001875	-0.000117	-0.000998
1	2018-11-26	boeings corporate giving exceed usd – financia...	__NWSA	-0.001875	-0.000117	-0.000998
2	2018-12-01	boeing donates usd bush institutes military se...	__BA	-0.001875	-0.000117	-0.000998
3	2018-12-05	flipkart investors force walmart take company ...	__WMT	-0.001519	-0.003956	-0.001834
4	2018-12-05	flipkart investors force walmart take company ...	__WMT	-0.001519	-0.003956	-0.001834
5	2018-12-07	apple announces million china clean energy fund	__AAPL	0.023068	0.012959	0.007831
6	2018-12-07	comcast raises offer sky bid edge rupert murdo...	__CMCSA	-0.005675	-0.007035	-0.011692
7	2018-12-18	banks banned charging ripoff overdraft fees fc...	__STI	-0.001875	-0.000117	-0.000998
8	2018-12-19	boeing opens first completion plant china amid...	__BA	0.003004	-0.001120	-0.000769
9	2018-12-27	turkish airlines mandates aviation capital fin...	__AAL	-0.001875	-0.000117	-0.000998

Figure 5: Dataframe containing the test data

4.4 Evaluation of the results

4.4.1 Direction predictor

In this section, we investigate whether we have evidence that our model is able to select the direction of a stock's price movement with greater than 50% accuracy. That is, we do better than random guessing.

H_0 : The model predicts the correct direction no better than 50% of the time

H_1 : The model predicts the correct direction with accuracy greater than 50%

Under the null hypothesis, our model making a correct directional prediction is a Bernoulli trial with probability 0.5. Therefore, for N headlines, the total number of correct predictions X is a Binomial random variable, and we have

$$\mathbb{P}(X \geq x) = \sum_{i=x}^{n} \left[\binom{n}{i} \times (0.5)^i \right] \tag{1}$$

which is the p-value for our test, using N headlines. For each of our horizon times [3, 5, 10], we can compute this statistic in this way.

In order to check whether our prediction of price movement is correct, we multiply together the columns of our prediction of abnormal returns and the real abnormal returns for the three horizons; in this way any positive value at the end of this multiplication will indicate a correct prediction and any negative one will indicate a wrong prediction. Then the ratio of right prediction over the total number of predictions. The table 2 shows this ratio for the different time horizons for the two methods.

Horizon	Shepard's method	Closest cluster
3 days	0.5641	0.4324
5 days	0.4872	0.5676
10 days	0.4103	0.5946

Table 2: Direction prediction

We clearly see that neither of the two methods outperforms random guessing for all time horizons. The Shepard's method seems to be a sensible indication for the direction of stock's prices only in the closest future; while on the other hand the closest cluster method shows better results further away from the release of the headline.

4.4.2 Magnitude predictor

In this section, we investigate whether our model is able to predict the magnitude of the returns movement. To do so, we computed the relative gap between our prediction of abnormal returns and the real abnormal returns.

$$\epsilon_{relative} = \frac{abreturns_{prediction} - abreturns_{real}}{abreturns_{real}} \tag{2}$$

Instead of providing the whole data set enriched with relative errors for each headline, we show only the mean and standard deviation of these values for the different time horizons. Table 3 shows the performance of the closest cluster method while table 4 shows the one of the Shepard's method.

Horizon	Mean	Std deviation
3 days	-0.6288	2.929825
5 days	-0.3464	1.756542
10 days	0.006009	1.384301

Table 3: Closest cluster

Horizon	Mean	Std deviation
3 days	0.05179	1.294246
5 days	0.2099	1.013285
10 days	0.3671	1.092541

Table 4: Shepard's method

Clearly each method has smaller errors on the time horizons where the direction of the stock prices is predicted correctly most of the times. Indeed, the Shepard's method shows to be only 5% off in the prediction of the abnormal returns for the closest future, but then this errors grows up to 36% which is clearly too large to be used sensibly in a trading strategy. On the other hand the closest cluster method for a 10 days time horizon seems to show prediction really close to the actual value, with an error of only 0.6 %.

Focusing now on the standard deviation column of these table, we notice very large values. This shows that, even if the error's mean is small, there exists some headlines for which the predicted abnormal returns are very far from the true values.

5 Conclusion

The results shown are clearly mixed. Indeed we don't see our model outperforming random guessing all the times but only on specific time horizons. The model used to generate these results, can be clearly improved. We, in fact, see how the SVO algorithm applied to each headline is sometimes inaccurate. Furthermore also the Word2Vec model should be further improved since it should be able to recognize a greater variety of words. This two problems are in fact the main reason why, starting from a data set of over 60,000 headlines, we carry our analysis only on 1,000 of them. Our results tough are not to discard entirely. The model as it is, shows in fact the ability to predict price movements on specific time horizons most of the times, even if trained on a small data set. Thus, the most important factor to change should be the size of the data set and its nature. Probably considering also different kind of information, such as tweets, speeches etc. , could lead to a great improvement in the predictions. However, the complexity of the data to analyse could still lead to misinterpret the sense of the message and needs to be treated carefully.

References

[1] Ming Li Chew, Sahil Puri, Arsh Sood, Adam Wearne. 2017. BlackRock Applied Finance Project: Using Natural Language Processing techniques for Stock Return Predictions

[2] https://webhose.io/free-datasets/financial-news-articles/

[3] https://en.wikipedia.org/wiki/List_of_S%26P_500_companies

[4] https://www2.deloitte.com/us/en/insights/industry/financial-services/artificial-intelligence-ai-financial-services-frontrunners.html?id=us:2di:3em:4di4687:5awa:6di:MMDDYY:&pkid=1005552, published 2019/08/13

[5] https://www.dataversity.net/a-brief-history-of-natural-language-processing-nlp/

[6] https://www.wonderflow.co/blog/natural-language-processing-examples

[7] https://spacy.io/usage/models#download

[8] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pp. 363-370. http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf

[9] https://github.com/hartator/wayback-machine-downloader

[10] A two-dimensional interpolation function for irregularly-spaced data. Donald Shepard, 1968

[11] https://code.google.com/archive/p/word2vec/

[12] http://www.mit.edu/~9.54/fall14/slides/Class13.pdf