What can we learn from news about the stock market?

Angelos Pelecanos, Caleb Noble, Dimitris Koutentakis
{apelecan, cnoble, dkout}@mit.edu

Abstract

In this paper we are exploring the correlation between news headlines and the movement of the stock market. This exploration will mostly focus on topic modeling of the news headlines and how the relevance of these topics varies over time. We expect that certain financial movements are either initiated or reported by the mainstream media. We hence expect that examining how the topics that the media covers change over time, we can actually recognize and reason behind some of the changes in the stock market.

1 Introduction

News affects headlines, but how much and in what way? A skilled trader will be able to quickly determine whether or not a story will have an effect and to what degree, but could a machine learn to do the same? Our project aims to explore what semantically meaningful topics appear in headlines and how they correspond to stock changes. To achieve this, we took large datasets of news headlines and tweets, and experimented with different topic modeling techniques to determine the best approach for extracting topics. From there we applied the topic models to tweets from Elon Musk and compared how the topics and stock values changed over time.

2 Datasets

2.1 News Headlines

For the first part of our project, we utilized a Kaggle dataset of Reddit World News headlines [1]. The data consisted of the 25 top news headlines per day (ranked by user votes) from 2008-06-08 to 2016-07-01 and corresponding Dow Jones Industrial Average stock changes of the next day (including, open, close, and volume).

2.2 Elon Musk Tweets

To perform a topic modeling experiment on a specific company (discussed in section 5), we used the Kaggle dataset [2] of 3218 tweets from Elon

[1] Daily News for Stock Market Prediction, *https://www.kaggle.com/aaron7sun/stocknews*

[2] Elon Musk's Tweets, *https://www.kaggle.com/kulgen/elon-musks-*

Musk, CEO of SpaceX and Tesla, spanning November 16, 2012 through September 29, 2017.

3 Data Preprocessing

3.1 General Methods

When it comes to dealing with textual datasets, we believe that preprocessing is crucial, considering the very noisy nature of textual data. Punctuation, typos and irregular use of whitespaces among others can all be sources of noise that will affect the performance of any model that we develop. This is why our general preprocessing procedure was quite strict.

Using regular expressions, we are tokenizing only words and numbers and then concatenating everything together using a single space as a delimiter. This has the disadvantage of losing the main structure of our data (such as sentences and commas), but it is a tradeoff that we preferred.

3.2 Stemming

Stemming is a widely used process that converts words to their linguistic root (stem) [Len+81]. For example "learning", "learner" and "learned" can be converted to their common word root, which is "learn". This has the benefit of restricting the size of our vocabulary while not losing significant information in the process.

tweets

Additionally, when it comes to topic modeling, including stemming should (theoretically) help with clustering multiple words with similar semantic features together and support the algorithms in finding the more broad topics covered. The case that we would like to avoid is having multiple different topics that refer broadly to the same topic, but using different words that are semantically close.

Throughout this paper we are using the Snowball Stemmer from NLTK [3].

3.3 Stop Word Inclusion

Stop words are defined as commonly used words that offer little to no content in a document or corpus. Examples of such words include: "a", "an", "with", "the".

It is a very common practice for stop words to be excluded from a corpus before any kind of Natural Language Processing. One reason is computational efficiency. In addition, the dimensionality of our data is reduced with no significant information loss, hence denoising our data [SR03].

Throughout this paper, we have been using the english stop words from NLTK [4].

3.4 tf-idf

Term Frequency - Inverse Document Frequency (tf-idf) is a numerical metric that expresses the significance of

[3] *http://www.nltk.org/howto/stem.html*

[4] https://www.nltk.org/book/ch02.html

a word in a document of a corpus [SY73]. This metric increases as the specified word appears more and more times in the document. It also decreases the more this word appears in other documents of the corpus.

The frequency of term t in document d from a corpus $\mathcal{D}$ is defined as

$$\text{tf}(t, d, \mathcal{D}) = \frac{\#(t \text{ in } d)}{\#(t \text{ in } \mathcal{D})}$$

Similary, inverse document frequency of term t in a coprus $\mathcal{D}$ is

$$\text{idf}(t, \mathcal{D}) = \log\left(\frac{|\mathcal{D}|}{|\{d \in \mathcal{D} : t \text{ in } d\}|}\right)$$

The result is obtained by multiplying the two quantities

$$\text{tf-idf}(t, d, \mathcal{D}) = \text{ tf}(t, d, \mathcal{D})\ \text{idf}(t, \mathcal{D})$$

Throughout this paper, we have been using Python's *TfidfVectorizer*. This module also gives us the opportunity to limit the number of features/words included in our corpus using the max_features argument. This argument was set to 10000 for the entirety of the project.

4 Methods

As topic modeling is unsupervised, no general "theme" is assigned to topics by any algorithm. What we will do to infer the topics learned, is to inspect the most probable words in every topic.

4.1 NMF

Non-negative Matrix Factorization (NMF) is a matrix decomposition technique, that given a non-negative matrix $M \in \mathbb{R}^{n\times m}$, it computes the non-negative matrices $A \in \mathbb{R}^{n\times k}$ and $W \in \mathbb{R}^{k\times m}$ such that $M = AW$. As per the following paper by Lee and Seung [LS01], the way this is achieved is by minimizing the following loss function:

$$\begin{aligned}\mathcal{L}(M, A, W) &= ||M - AW||_F^2 \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}\left(M_{ij} - (AW)_{ij}\right)^2\end{aligned}$$

The loss function is minimized by alternating between fixing the A matrix and performing the multiplicative update rule:

$$W_{a\mu} \leftarrow W_{a\mu}\frac{(A^T M)_{a\mu}}{(A^T AW)_{a\mu}}$$

Then, the algorithm fixes the W matrix and similarly updates A:

$$A_{ia} \leftarrow A_{ia}\frac{(MW^T)_{ia}}{(AWW^T)_{ia}}$$

For topic modeling, suppose we have a set D of documents from a vocabulary V. Define $M \in \mathbb{R}^{|V|\times|D|}$, where M_{ij} is the frequency of word w_i in the document d_j. Let k be the number of topics. Then we will have $A \in \mathbb{R}^{|V|\times k}$, where every row of A will represent the frequency of word w_i in the topics. Similarly $W \in \mathbb{R}^{k\times|D|}$,

Topic 0	Topic 1	Topic 2	Topic 3
year	korea	israel	kill
world	north	gaza	attack
new	south	isra	pakistan
china	korean	palestinian	strike
govern	nuclear	war	bomb
protest	missil	hama	peopl

Table 1: Topics from NMF, stemmed and no stopwords

Topic 0	Topic 1	Topic 2	Topic 3
police	korea	israel	russia
new	north	gaza	says
world	south	israeli	ukraine
year	korean	palestinian	iran
people	nuclear	war	russian
government	kim	hamas	putin

Table 2: Topics from NMF, unstemmed and no stopwords

with every column of W representing the frequency of topics for the given document.

We perform the NMF decomposition on the M matrix using Python's *sklearn.decomposition.NMF*. Using this decomposition, we have our topics represented in the columns of A and rows of W.

We can also sketch these topics and how popular they are in our documents over time. For the visualization, the process that we followed was to get the average of relevance of the topics for every document over a period of 40 consecutive days.

Topic 0	Topic 1	Topic 2	Topic 3
the	to	in	korea
is	for	killed	north
and	be	at	south
that	and	for	korean
has	on	and	nuclear
world	us	after	kim

Table 3: Topics from NMF, unstemmed and with stopwords

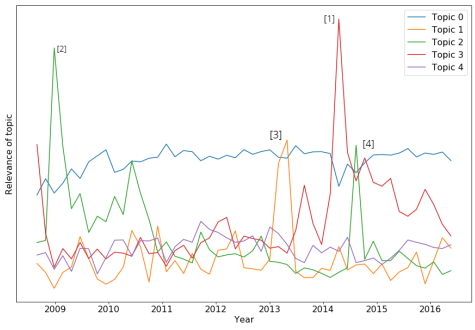

Figure 1: NMF Topics (unstemmed / no stopwords) over time

Figure 2: NMF Topics (unstemmed / with stopwords) over time

From the above graphs we can recognize the following events

1. Annexation of Crimea by the Russian Federation (2014)
2. Gaza War (2008-2009)
3. Korea's nuclear missile test (2013)
4. Israel-Gaza Conflict (2014)

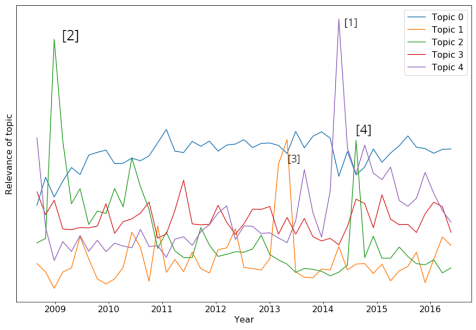

Figure 3: NMF Topics (stemmws / no stopwords) over time

We can see that NMF without stopwords (after removing stopwords) is pretty similar when it comes to topic recognition and the sketching of the topics over time. When it comes to NMF without removing stopwords, we can see that some of the topics make sense, but there is a significant portion of stop words that make the topics too general. This is also obvious from the graph, where most of the topics have a constant relevance.

4.2 LSA

Latent Semantic Analysis (LSA) performs matrix decomposition to extract topic models from a corpora of documents [Dum].

Given a document matrix M of word counts per article, LSA uses singular value decomposition to reduce M into separate document-topic matrix and topic-term matrix. Truncating to the top k singular values and corresponding singular vectors, yields k topics and the weights for the corresponding terms. This idea is very similar to NMF and also reminded the team of Singular Value Thresholding used in Recommender Systems.

Topic 1	Topic 2	Topic 3	Topic 4
says	korea	israel	russia
israel	north	gaza	ukraine
korea	south	israeli	says
north	korean	palestinian	iran
china	jong	north	putin
russia	china	hamas	military

Table 4: LSA top topics.

Figure 4: LSA topic evolution

From there we used Python's *sklearn.decomposition.TruncatedSVD* package to perform the topic modeling.

After extracting 10 topics, we can recover the most relevant words for each topic. Table 4 shows the top four topics, and Figure 4 shows the evolution of 5 topics when LSA is run with only 5 topics.

4.3 LDA

As described by [Dav03], LDA or Latent Dirichlet Allocation is a generative probabilistic model of a corpus

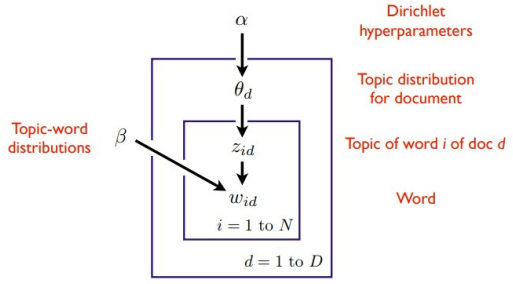

Figure 5: (Plate) Graphical model representation of LDA (from lecture slides)

of documents. The documents in our corpus are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. LDA's generative process for each document w in a corpus D with per document topic distribution parameter α and per topic word distribution parameter β is as follows:

1. Choose length of document: $N \sim \text{Poisson}(\xi)$
2. Choose topic distribution: $\theta \sim \text{Dir}(\alpha)$
3. For each of N words w_n:
   (a) Choose topic: $z_n \sim \text{Multinomial}(\theta)$
   (b) Choose word: $w_n \; p(w_n|z_n, \beta)$

The generative process is vizualized in Figure 5.

In order to implement LDA to our corpus of documents, we first build a tf-idf matrix of the corpus, as defined above and then build the LDA model using the `gensim` model for python [ŘS10] [5].

Topic 0	Topic 1	Topic 2	Topic 3
china	french	killed	isis
canada	egypt	syria	dead
australia	officials	ukraine	putin
protests	afghanistan	drug	australian
scientists	china	russian	turkish
years	state	attack	state

Table 5: Topics from LDA on reddit headlines dataset

Figure 6: Reddit topics via LDA vs time

For the reddit headlines dataset, we get the word distribution of the topics through LDA. The topics can be seen in Table 5.

Furthermore, we can see how LDA models the change of topics over time when run with only 5 topics in Figure 6.

5 Modeling Tweets

Applying the results of testing various topic modeling techniques to a spe-

[5] https://radimrehurek.com/gensim/models/ldamodel.html

Topic 0	Topic 1	Topic 2
spacex	model	good
launch	tesla	point
falcon	teslamotors	chance
dragon	elonmusk	looks
landing	like	piece
rocket	cars	rocket

Table 6: Top 3 topics from tweets.

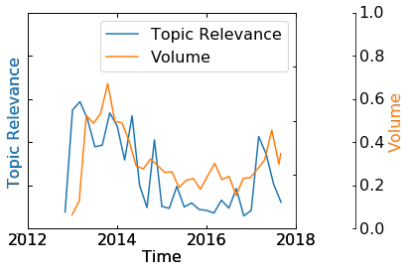

Figure 7: Sum of topic 1 relevance over time and volume of TSLA

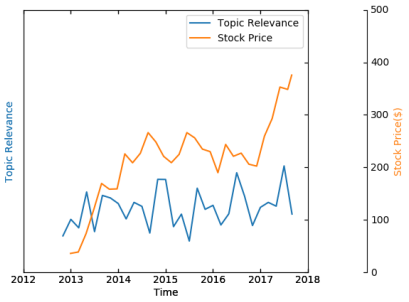

Figure 8: Average of topic 1 relevance over time and stock price of TSLA

cific company helps gain insight for how company specific news correlates to stock fluctuations. The team used Elon Musk tweets and ran the NMF topic model with removing stopwords to generate five topics, with the top three listed in Table 6.

From the first two topics, the model was able to separate terms related to spacex and teslamotors. Applying this model to the tweets then allows us to compute the relevance of each topic to that tweet's content. As before, we were able to visualize how the average relevancy of the topics changed over time; however, the tweets, unlike the news articles, varied in frequency over time, so it was also interesting to look at how the sum of relevance over all tweets in a given window changed over time. The data was grouped over two month periods and the sum and average relevance of Topic 1 was computed. We then compared these results to the volume and closing stock price data of TeslaMotors (TSLA) in Figure 7, 8.

Pearson correlation coefficients [Ped+11] between summed/average tweet relevance and stock volume/price, as shown in table 7, reveal that the volume is correlated with the sum of relevance of the tesla topic over tweets in the same time period. This makes sense because the more excitement there is around the company, the more the stocks will be traded and the more people will be tweeting. The less obvious result was that the average relevance of the tesla topic is correlated to the stock price. The average relevance is related to how focused the tweets are about Tesla, and the more focus Elon Musk is giving to Tesla could be related to how well the company is doing.

	Averaged	Summed
Stock	0.258	-0.356
Volume	0.073	0.561

Table 7: Pearson correlation coefficient between tweets and stock

6 Previous Attempts

Before looking into topic model, the team attempted to use the Reddit World News headlines to make a binary classifier of whether the Dow Jones Industrial Average would increase or decrease the next day. Bag of words was used to process the headlines into vectors representing word ngrams of up to length 3. Then using linear regression, they found the best hyperparameter for L1 regularizer based on validation accuracy of 0.05. From there, the ROC AUC score was calculated using sklearn roc_score function[Ped+11] and found to be 0.524. This value revealed that there was hardly any signal from the headlines, and based on this conclusion, the team decided to look into topic modeling.

7 Conclusion

Overall, the team has explored the effectiveness of various topic modeling techniques, concluding that removing stopwords and running NMF provides well separated topics. Additionally, the authors showed how these methods could be applied to real-world world data about a company to, and that the volume and stock prices are correlated to news about the company.

We understand that correlation does not imply any kind of causation in our case. We tried testing causation of Tesla's stock volume and price based on the Granger Causality Test[Gra69]. However, we were not able to reject the null hypothesis of the Tesla topic Granger causing the stock price or volume to a significant degree. Stock market data tends to be extremely noisy and hence doing simple topic modelling from a single Twitter account is not the best setup.

The team has examined several other ideas, such as scraping tweets that are related to Elon Musk and his companies and using a more appropriate model, such as Topics over Time [Xue06]. Additionally, the team has experimented with the LDA model with worse results and we believe that this is due to not optimal hyperparameter choices.

8 Division of Labor

Angelos focused on the NMF experimentation, Caleb on LSA experimentation and Dimitris on LDA. For the final part with the tweet modeling, all members worked closely together. At all times, every member was aware of what the other individuals were working on.

References

[Gra69] C. W. J. Granger. "Investigating Causal Relations by Econometric Models and Cross-spectral Methods". In: *Econometrica* 37 (1969), pp. 424–438. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912791.

[SY73] Gerard Salton and C.S. YANG. "On the Specification of Term Values in Automatic Indexing". In: *Journal of Documentation* 29 (Dec. 1973), pp. 351–372. DOI: 10.1108/eb026562.

[SY76] Gerard Salton and C.S. Yang. "On the Specification of Term Values in Automatic Indexing". In: *Cornell University Computer Science Technical Reports* (June 1976).

[Len+81] Martin Lennon et al. "An evaluation of some conflation algorithms for information retrieval". In: *Journal of Information Science* 3.4 (1981), pp. 177–183. DOI: 10.1177/016555158100300403. eprint: https://doi.org/10.1177/016555158100300403. URL: https://doi.org/10.1177/016555158100300403.

[LS01] Daniel D. Lee and H. Sebastian Seung. "Algorithms for Non-negative Matrix Factorization". In: *NIPS* (2001).

[Dav03] Michael I. Jordan David M. Blei Andrew Y. Ng. "Latent Dirichlet Allocation". In: *Journal of Machine Learning Research* 3 (2003), pp. 993–1022.

[SR03] C. Silva and B. Ribeiro. "The importance of stop word removal on recall values in text categorization". In: *Proceedings of the International Joint Conference on Neural Networks* 3 (2003), pp. 1661–1666.

[Xue06] Andrew McCallum Xuerui Wang. "Topics over Time: A Non-Markov Continuous-Time Model of Topical Trends". In: (2006).

[Wu+08] H. Wu et al. "Interpreting TF-IDF term weights as making relevance decisions". In: *ACM Transactions on Information Systems* 26 (2008), p. 3.

[ŘS10] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling

with Large Corpora". English. In: (May 2010). http://is.muni.cz/publication/884893/en, pp. 45–50.

[Ped+11] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[Her+15] Karl Moritz Hermann et al. "Teaching Machines to Read and Comprehend". In: *CoRR* abs/1506.03340 (2015). arXiv: 1506.03340. URL: http://arxiv.org/abs/1506.03340.

[Dum] Susan T. Dumais. "Latent semantic analysis". In: *Annual Review of Information Science and Technology* 38.1 (), pp. 188–230. DOI: 10.1002/aris.1440380105. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/aris.1440380105. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440380105.