Dynamic and Automatic Lexicon Generation for Sentiment Analysis in the Business Domain

Justin Nafe
800 Lakeside Circle apt#1234
Lewisville TX, 75057
justinnafe@my.unt.edu

Abstract

News articles about a stock, and the resulting price of the stock, can be used to automatically determine the polarity (positive or negative sentiment) of terms in articles. The change in stock price, immediately after a publication about the stock, reflects investorsô sentiment toward the article. From that sentiment, measurable by the resulting change in stock price, a sentiment lexicon based on terms relating to business can be derived. With a corpus of multiple news articles about stocks and numerical data about the change in prices, key sentimental words can be extracted and given a sentiment weight. The weights are assigned by using information retrieval methods using various term frequencies (tf) and inverse document frequencies (idf). An analyzer then uses the generated lexicons as a tool to measure the sentiment of the next dayȏs news articles about stocks. To determine the accuracy of the generated lexicon, the determined polarity is compared with the closing price of the related stock. The results show that news articles may not be opinionated enough to determine direction consistently. The next step would be to use blogs for a corpus. If this method can be perfected this automatic generation of sentiment lexicons would eliminate the laborious and time consuming task of annotation. The resulting lexicons would also help investors decide on what stocks to buy or sell or help them decide on when to buy or sell.

1 Introduction

The automatic generation of a lexicon for the business domain would save time and money in annotating lexicons. The manual creation of sentiment lexicons has involved user interpretation, but if user interpretation is already known, as in stock price changes from the investors reading news, then the creation of lexicons would be automatic.

The automatic creation of the business lexicon would solve the problem of general sentiment lexicon not accurately reflecting the sentiment in business news articles. In the business domain, words other than the words found in the general sentiment lexicon, may have a positive or negative polarity; or terms found in the general sentiment lexicon may not have an effect within the news article. For example, the term "share" may have a weak-positive polarity in the general sentiment lexicon but not have any polar significance in news articles about stocks.

The idea of creating a lexicon dynamically came from reading Turneyȏs article *Thumbs Up or Thumbs Down* (Turney, 2002). Turneyȏs research consisted of unsupervised sentiment analysis of movie reviews. I thought that if I could find opinions about stocks and use a sentiment lexicon to determine the sentiment toward the stock, much like Turney, then I could predict the direction the stock price. In previous trials of predicting the direction of stocks, I had used an OpinionFinder sentiment lexicon and gathered slightly positive results, but not enough to invest (http://www.cs.pitt.edu/mpqa/opinionfinderrelease/). Before investing, I needed to know more about what determines stock price and to use a more business specific sentiment lexicon. Using information retrieval methods, I thought that I could achieve the above tools automatically.

One theory about what determines the price of stock is what information the investor knows about the stock (Schumaker & Chen, 2009). A popular theory about the relationship of how much and what investors know about a stock and of the price of the stock is the efficient-market hypothesis (EMH). This theory assigns three levels to what people know and how they know it, and this knowledge reflects how they fair in the stock market. The three levels are weak, semi-strong, and strong, whereas weak suggests that the investor knows only what is published after it is published, semi-strong is a little more knowledge, and strong is nearly inside trading (Schumaker & Chen, 2009). The basic idea is that the investor reads about a stock through the newspaper, web, blogs, or chats, and then the investor determines the price of the stock.

To create a more business specific lexicon, I used news articles from the popular investment website Reuters to pull business terms from (http://www.reuters.com). This site provides a single source of news articles about businesses and investments.

Some more recent research includes predicting the actual future price of a stock based on news articles twenty minutes after they are published (Schumaker & Chen, 2009). Schumaker et al. use textual approaches, such as bag of words, noun phrases, and named entities. The method I used resembles the bag of words method.

The bag of words method pulls out stopwords, such as *a*, *and*, *the*, *of*, etcé and the remaining words are used to represent the text (Schumaker & Chen, 2009). I also pull out stopwords, but my method separates positive and negative documents and assigns a weight to the remaining terms. To calculate the weight of the terms in the document, I use a variation of term frequency (tf) and inverse document frequency (idf). The idf calculation is a popular method in information retrieval. (Mitzler, 2008)

2 Methods

For the project, I gathered business related news articles from specific dates and generated four lexicons based on those articles and on stock prices.

2.1 Corpus Creator

I started by gathering the news articles and creating corpora based on news articles published dates. The following table shows the number of news articles per corpus (the corpora are named after the date that the articles within it were published).

Date (yyyymmdd)	Articles
20091101	299
20091102	298
20091103	292
20091105	55
20091108	296
20091109	297
20091110	296
20091111	290
20091112	300
20091115	298
20091116	300
20091117	300
20091119	300
20091122	300
20091123	299
20091124	299
20091126	298
20091129	300

Table 1: Number of news articles.

2.2 Lexicon Builder

After creating the corpora, I tagged the news articles as either positive or negative, based on a 5% price increase or price decrease of stock mentioned within the article. The lexicons are based on the following calculations:

1. The *tf* in positive articles multiplied by *idf* in positive articles.

2. The *tf* in positive articles multiplied by *idf* in negative articles.

3. The *tf* in negative articles multiplied by *idf* in positive articles.

4. The *tf* in negative articles multiplied by *idf* in negative articles.

The *tf* shows the importance of a term within a single document. The frequency is normalized to even out long documents. The following formula shows the *tf* calculation for document $_j$.

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

Figure 1: tf formula. Source: http://en.wikipedia.org/wiki/Tf%E2%80%93idf

The numerator is the count of term n_i in document $_j$. The denominator is the sum of all terms in the document. Within my experiment, I found the *tf* of the terms in the documents tagged as positive and in the documents tagged as negative.

The *idf* is the general importance of the term and is found by taking the log of the quotient of the total number of documents divided by the number of documents that contain the term. The following formula shows the *idf* calculation.

$$\mathrm{idf_i} = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

Figure 2: idf formula. Source: http://en.wikipedia.org/wiki/Tf%E2%80%93idf

The numerator is the count of documents. The denominator is the count of the documents containing term $_i$. Within my experiment, I found the *idf* for terms found in documents tagged as positive and in the documents tagged as negative.

The *tf-idf* is the weight given to the term. The following formula shows the *tf-idf* calculation.

$$(\mathrm{tf\text{-}idf})_{\mathrm{i,j}} = \mathrm{tf_{i,j}} \times \mathrm{idf_i}$$

Figure 3: tf-idf formula. Source: http://en.wikipedia.org/wiki/Tf%E2%80%93idf

The *tf-idf* calculation assigns a weight to the term. With the weights of the terms and the positive and negative buckets to put these terms in, we are able to determine what terms and weights are found in ñpositiveò documents and what terms and weights are found in ñnegative documents.

2.3 Stock Analyzer

The stock analyzer reads the ñnext dayôs corpusò and finds symbols based on the Nasdaq index, finds opinionated terms based on the generated lexicons, calculates scores based on the opinionated termsô weights, and tests the accuracy of the scores based on the direction of the stock for that following day.

2.4 Valance Shifters

The valance shifters determine if the polarity of the term needs to be shifted. If the program parses a polar word, then the valance is sought after. If the type of valance shift is ñnegate,ò then the score is given opposite of what was found. If the type is ñshiftneg,ò then the polarity shifts down one point. If the type were ñshiftpos,ò then the polarity shifts up one point.

3 Results

For the corpora, the crawler grabbed news articles from www.reuters.com and put them in files named after the date the articles were published. The corpora spans the month of November in 2009. All articles were pulled from www.reuters.com because the website was easy to crawl, parse, and figure out the date the articles were published.

The stocks used within this program are limited to the Nasdaq index.

The gold standard for this experiment is the analysis using the OpinionFinder and the actual stock direction.

The following table shows the results from using the original OpinionFinder.

Date	**Precision**	**Accuracy**
20091101	0.0354	0.4082
20091102	0.0663	0.6375
20091103	0.0296	0.2853

20091105	0.0244	0.5906
20091108	0.0669	0.5803
20091109	0.0409	0.3772
20091110	0.0439	0.4530
20091111	0.0227	0.2141
20091112	0.0556	0.5719
20091115	0.0793	0.6778
20091116	0.0523	0.5194
20091117	0.0383	0.3831
20091119	0.0533	0.5307
20091122	0.0523	0.4954
20091123	0.0348	0.3934
20091124	0.0377	0.4158
20091126	0.0465	0.5608
20091129	0.0481	0.5343
Average	**0.0460**	**0.4794**

Table 2: OpinionFinder

As you can see, the average is not significant enough to assume we can predict direction. Even if we take the standard deviation (0.12237) and remove all errors outside twice the standard deviation (11/11/2009), then the average would still be less than .5 (to be exact, the new calculated average would be 0.49)

The following table shows the results from using the *tf-neg * idf-neg* lexicon.

Date	**Precision**	**Accuracy**
20091101	0.0354	0.4082
20091102	0.0660	0.6344
20091103	0.0663	0.6395
20091105	0.0127	0.3071
20091108	0.0669	0.5803
20091109	0.0439	0.4042
20091110	0.0439	0.4530
20091111	0.0747	0.7034
20091112	0.0344	0.3545
20091115	0.0793	0.6778
20091116	0.0409	0.4065
20091117	0.0549	0.5487
20091119	0.0536	0.5340
20091122	0.0523	0.4954
20091123	0.0354	0.4007
20091124	0.0458	0.5054
20091126	0.0465	0.5608
20091129	0.0481	0.5343
Average	**0.0501**	**0.5082**

Table 3: tf-neg * idf-neg Lexicon

The average results from this lexicon are better, and if we calculate the standard deviation and remove possible errors, the average would be the same because according to the twice the standard deviation rule set earlier, there are no errors.

The following table shows the result from using the *tf-neg * idf-pos* lexicon.

Date	**Precision**	**Accuracy**
20091101	0.0354	0.4082
20091102	0.0283	0.2719
20091103	0.0666	0.6426
20091105	0.0247	0.5984
20091108	0.0669	0.5803
20091109	0.0448	0.4132
20091110	0.0471	0.4866
20091111	0.0630	0.5933
20091112	0.0344	0.3545
20091115	0.0793	0.6778
20091116	0.0523	0.5194
20091117	0.0549	0.5487
20091119	0.0487	0.4854
20091122	0.0523	0.4954
20091123	0.0481	0.5441
20091124	0.0426	0.4695
20091126	0.0465	0.5608
20091129	0.0481	0.5343
Average	**0.0491**	**0.5102**

Table 4: tf-neg * idf-pos Lexicon

If we remove 20091102 from the calculation because the accuracy is twice the standard deviation, then the new calculated average would be 0.524.

The following table shows the results of using the *tf-pos * idf-neg* lexicon.

Date	Precision	Accuracy
20091101	0.0354	0.4082
20091102	0.0643	0.6188
20091103	0.0653	0.6301
20091105	0.0127	0.3071
20091108	0.0669	0.5803
20091109	0.0439	0.4042
20091110	0.0439	0.4530
20091111	0.0757	0.7125
20091112	0.0344	0.3545
20091115	0.0793	0.6778
20091116	0.0409	0.4065
20091117	0.0549	0.5487
20091119	0.0533	0.5307
20091122	0.0523	0.4954
20091123	0.0481	0.5441
20091124	0.0458	0.5054
20091126	0.0465	0.5608
20091129	0.0481	0.5343
Average	**0.0507**	**0.5151**

Table 5: tf-pos * idf-neg Lexicon

All calculations for this lexicon fall within the error tolerance.

The following table shows the results from using the *tf-pos * idf-pos* lexicon.

Date	Precision	Accuracy
20091101	0.0354	0.4082
20091102	0.0283	0.2719
20091103	0.0666	0.6426
20091105	0.0244	0.5906
20091108	0.0669	0.5803
20091109	0.0448	0.4132
20091110	0.0471	0.4866
20091111	0.0227	0.2141
20091112	0.0344	0.3545
20091115	0.0793	0.6778
20091116	0.0523	0.5194
20091117	0.0549	0.5487
20091119	0.0487	0.4854
20091122	0.0523	0.4954
20091123	0.0481	0.5441
20091124	0.0461	0.5090
20091126	0.0465	0.5608
20091129	0.0481	0.5343
Average	**0.0471**	**0.4909**

Table 6: tf-pos * idf-pos Lexicon

The date 20091111 falls outside of the accepted tolerance, so if we remove that date, then the new average would be 0.507.

These results do not show a significant improvement from using the original OpinionFinder. Even when removing significant deviations from the average, the resulting averages were not significantly greater than 50%.

4 Discussion

The results show an average over the span of a month. The average does not show a significant difference between lexicons, nor do they show that any one lexicon can be used to predict direction. News articles may not be the most ideal documents to detect sentiment. Blogs may be a better choice since blogs usually express opinions about topics, but news articles are shown to be unbiased.

Reference

http://www.site.uottawa.ca/~diana/publications/ci.pdf.

http://acl.ldc.upenn.edu/P/P02/P02-1053.pdfwww.cs.jhu.edu/~mdredze/publications/sentiment_acl07.pdf

http://www.mccombs.utexas.edu/faculty/Paul.Tetlock/papers/TSM_More_Than_Words_JF_05_07.pdf

Ingvaldsen, J. E., Gulla, J. A., Laegreid, T., and Sandal, P. C. 2006. Financial News Mining: Monitoring Continuous Streams of Text. In *Proceedings of the 2006 IEEE/WIC/ACM international Conference on Web intelligence* (December 18 - 22, 2006). Web Intelligence. IEEE Computer Society, Washington, DC, 321-324. DOI=

http://dx.doi.org/10.1109/WI.2006.80 (Ingvaldsen, Gulla, Laegreid, & Sandal, 2006)

Metzler, D. 2008. Generalized inverse document frequency. In *Proceeding of the 17th ACM Conference on information and Knowledge Management* (Napa Valley, California, USA, October 26 - 30, 2008). CIKM '08. ACM, New York, NY, 399-408. DOI= http://doi.acm.org/10.1145/1458082.1458137

OpinionFinder http://www.cs.pitt.edu/mpqa/opinionfinderrelease/

P. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In Proceedings of the Association for Computational Linguistics (ACL), 2002, pages 417--424.

Schumaker, R. P. and Chen, H. 2009. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. *ACM Trans. Inf. Syst.* 27, 2 (Feb. 2009), 1-19. DOI= http://doi.acm.org/10.1145/1462198.1462204