Predicting IPO Performance from Nearest Neighbors
Using TF-IDF Weighted Word Count Vectors

Roi Ceren
Department of Computer Science
University of Georgia
Athens, GA 30602
ceren@cs.uga.edu

Will Richardson
Department of Computer Science
University of Georgia
Athens, GA 30602
wdr525@uga.edu

Muthukumaran Chandrasekaran
Department of Computer Science
University of Georgia
Athens, GA 30602
mkran@uga.edu

Abstract

We introduce a novel approach to mining and leveraging data concerning stocks in order to predict the performance of new stocks following their initial public offering, a traditionally difficult task due to the lack of information and historical performance data. We collect a large corpus of articles for every existing stock between March 1st, 2014 and March 1st, 2015. We create weighted feature vectors by calculating the TF-IDF values for every word that appears in a document about a given stock. We then perform locality-sensitive hashing on these feature vectors and bucket similar stocks into a neighborhood. Using a variety of monotonically decreasing functions, we examine the relationship between 30-day price fluctuations of each IPOs neighbor by propagating each neighbor's 30-day change to the IPO relative to its distance. Locality-sensitive hashing uncovers interesting relationships between IPOs and nearby neighbors. We observe a strong correlation between neighbor stock distance and respective 30-day performance.

1 Introduction

Stock prediction is an extremely popular topic in many fields, including economics and computer science, and has been subject to a variety of approaches and perspectives, such as machine learning or simple statistical optimization. Popular hedge fund and large investment firms experience significant success and publish fairly accurate index predictions for stocks using a variety of well-known techniques. Popular financial news agency Bloomberg leverages statistical model optimization during events experts conjecture will have an effect on the financial market using a proprietary supercomputer called "The Terminal" [19]. Such demonstrative methods are common, as empirical comparison is far less complex than process models examining global economies [22].

Prediction of stocks relying on historical information is useful, but in some cases such empirical data are sparse or non-existent, such as in the case of stocks during their initial public offerings (also known as IPOs). Recent research has investigated using "prediction markets", or indicative domains that the IPO might be related to, as indicators for IPO performance in the absence of hard empirical data [1, 10]. Other approaches involve gauging public perception of an IPO as a predictor, but had minimal results [23]. The former approach is promising, but requires a significant quantity labeled data for learning. With sparse markets and relatively few stocks to choose from, an unsupervised approach is better suited for this domain.

A majority of data, particularly articles discussing various financial aspects of companies, is unlabeled, though it can be leveraged in an informative manner. There exists a vast quantity of data, but quite a bit of it is noise. Noise reduction and information retrieval can be performed by leveraging statistical techniques in a scalable manner using distributed frameworks such as MapReduce. Using such distributed frameworks allows the unbiased processing of these large corpora.

To this end, we propose a many-tiered, distributed approach to identifying relationships between IPOs and related stocks that have a richer history to exploit. Similar to the "hot market" strategy, we propagate the performances of stocks that are considered similar in market to their respective IPOs. In order to identify candidate neighbors, we first extract a large dataset of articles for every existing stock in our IPO and neighbor candidate sets using the Yahoo! Finance News API. After processing noise and uninformative words from the corpus, we generate TF-IDF feature vectors from the word counts for each stock. We create neighborhoods utilizing locality-sensitive hashing and calculate the distance between stocks using order 2 Minkowski distance.

Identifying the neighborhoods is the first step in predicting IPO performance. Understanding the impact each neighbor has on its related stock requires a distance metric, but must leverage some other statistical model to map the distance to some impact. One approach is to fit the varying distances to some decreasing function and normalize the outputs while controlling for some intrinsic variance. A far more complicated approach would be a heavily parameterized decreasing function whose shape is learned. For this work, we focus on the former approach, as the framework, including LSH and approximate bucketing techniques, already introduce a significant amount of noise.

The work proceeds as follows. We highlight the research most related to our approach in Sec. 2. We classify our approach as two disparate phases: first, creating feature vectors from articles and, second, leveraging those vectors to identify neighbors and propagate their performance. The former phase is discussed in Sec. 3, with the latter discussed in Sec. 4. We follow with experimental results in Sec. 5 and conclude with a discussion.

2 Related Work

In the literature, historical stock data have been used to predict future stock price or direction [4, 6]. By looking at past daily/weekly/monthly/yearly prices, the future stock price or direction can be predicted. More recently, researchers have identified a strong correlation between stock markets and financial news [21, 14]. With the information technology boom, abundant information about finance has been made available to us particularly through news websites including Yahoo! Financial News. Consequently, since most news articles, blogs, and financial news forums that found on the internet are semi-structured at best, we will need to employ Natural Language Processing and Data mining methods to create our feature vectors, the elements of which are nothing but words contained in the news document [20].

Recently, Gunduz and Cataltepe investigated the effects of financial news articles on the Istanbul Stock Exchange and attempted to predict its direction after the news articles were published [5]. Gunduz et. al go on to extract informative features from these news articles using Mutual Information and TF-IDF weighting methods. We use the latter for feature selection on bag-of-words features [15].

Mittermayer studied financial news articles 15 minutes after its publication (almost in real time) and used Support Vector Machines for the classification [12]. Gidofalvi, however, used a Nave Bayes Classifier to predict the direction of stock [12].

Very recently, researchers found that the mood of tweets in Twitter can be a good predictor of IPO performance [9]. The key insight here is that twitter sentiment could reflect the sentiment of an investor and hence mirror the IPOs opening day performance.

Berg and Neumann used prediction markets to forecast Google's post-IPO market capitalization [2] and assert that such markets could provide useful information for understanding the IPO process. Ljungqvist et al. also provide key insights into IPO pricing: IPOs exhibit positive first-day average returns and hence, seem to be undervalued [11]. However, IPOs are, more often than not, overvalued from the vantage of longer horizons because, in subsequent months, the initial price hike often diminishes or reverses. It is also interesting to note that according to Ritter, IPOs underperform

the index [17] and seem to perform no worse than *similar* small and high growth companies [3]. There are several other models that impact IPOs long-run performance in their corresponding market including *investor sentiment*. Intuitively, the investor's interest in an IPO fades, resulting in long-run underperformance. More companies were observed to go public when the investor sentiment was high [8]. We draw inspiration from several of such models in this work.

3 Phase 1: Extracting Feature Vectors

In this first phase, we introduce our methodology for representing stocks as feature vectors. We begin with a discussion about the initial representation of stocks as documents in a corpus, where we acquire the corpus, and what methodology we use for heavily weighting important topics within each stock.

3.1 Text-based Representations

The central idea behind the representation of the companies is that the language used to describe them is sufficient to uncover relationships between companies, and that the stock of related companies will perform in correlated ways. More specifically, similar companies will be described by similar distributions over a vocabulary of "informative" terms. This approach has two principle advantages: it can potentially detect subtle connections that might not be readily apparent, and it works in a completely automated way, given the corpus. There is no need for an expert to perform extensive research, analysis, and classification/annotation tasks.

One important aspect of the representation is that only "useful" terms should be included. This is a practical consideration both in assembling and processing the data (memory, network latency) and in learning from the data (the curse of dimensionality). The informativeness of any given term can be made along absolute and relative criteria. Some words' relative frequencies will not be informative under any circumstances; these are called stop words, and lists of such irrelevant words are often used in frequency-based natural language processing. Examples of common stop words include articles, pronouns, prepositions, and other terms that contribute to the structure of text but not necessarily its content. Given a list of such words, it is safe to ignore them wherever they are encountered regardless of their relative frequencies across articles, companies, etc. The other way of finding uninformative words is to examine their frequencies across companies' texts. In contrast with the stop words, which are largely a property of language, these words are ruled out due to the properties of the corpus itself. It's not that these words *could not* be informative, merely that they *are not* informative in the data used to construct the model. For instance, if a word appears the same number of times (perhaps approximately) in all documents, it cannot be informative and therefore can be safely discarded, reducing the dimensionality of the representations. Similarly, if a word only appears in a single document (and will not/does not appear in the unseen documents), it cannot provide any information and so may also be safely discarded.

Term frequencies within each stock's article text do not themselves constitute the vector values but instead are used to calculate the TF-IDF measures. TF-IDF is a widely used measure of term informativeness that is frequently used to determine the similarity of two documents (i.e. bags of words), and is regarded as a vital technique for many natural language processing applications[13, 16]. Although its theoretical underpinnings are not comprehensively understood, TF-IDF has demonstrated strong empirical performance in many domains and is justified by some probabilistic models of information retrieval [18]. In our application, each company constitutes a document whose word frequencies are calculated from their concatenated body of articles. TF-IDF has a number of appealing properties, not least of which is its ability to weight rare, informative terms highly. Simpler frequency-based approaches often lend disproportionate weight to uninformative, commonly-occurring terms. While filtering stop words may remove many such terms, it is still important to remove such bias. As suggested by the name, TF-IDF consists of two components: a term frequency (Equation 1) and an inverse document frequency (Equation 2). The TF-IDF measure is simply the product of these two components.

$$f_{t,d} = \log(1 + c(t, d)) \tag{1}$$

$$idf_t = 1 + \log\left(\frac{N}{\sum_{i=1}^{N} I(t, d_i)}\right) \tag{2}$$

In Equation 1, $c(t, d)$ is simply the raw word count of term t in document (company) d, and in Equation 2, N is the total number of documents, and $I(t, d)$ is an indicator function with value one if $c(t, d) > 0$ and zero otherwise. We use the smoothed versions of both the term frequency and the inverse document frequency in our implementation.

$$v(t, d) = f_{t,d} \cdot idf_t \tag{3}$$

3.2 Articles

One service Yahoo! News offers is the ability to retrieve a list of relevant articles for any given stock ticker symbol via RSS. The articles used to construct the corpus come from a wide variety of sources on the Internet, but they take the form of HTML web pages, which contain large amounts of markup as well as client-side scripts. After a great deal of trial-and-error, we were able to remove the tags and and other rendering/formatting content. Unfortunately, this still leaves a large amount of non-article content in the form of navigation, sidebar, advertising, and other extraneous text. Given that Internet readers with money to invest are a very desirable potential customer base, this content forms a relatively large portion of the non-markup text as advertisers compete for visits and content providers try to keep readers on their respective sites.

Fortunately, this extraneous content will be distributed relatively uniformly across the articles (and therefore, across the stocks). As mentioned above, part of the feature extraction removes words that appear uniformly across company texts; additionally, TF-IDF diminishes the importance of words that appear in all company texts, and as we shall see, locality-sensitive hashing further de-emphasizes uniformly distributed words. In short, even though the text contains a great deal of noise, its distribution, and the method of vector formulation and comparison ensure that the noisy text will have minimal effect on the company representation vectors.

Having determined how to get individual articles into a suitable format, we faced additional challenges in acquiring the full corpus *en masse*. Although Yahoo! makes their service freely available, placing too many requests from the same IP address within a relatively short interval of time results in refused connections. In order to circumvent this, we divided the task into sixty lists and distributed these tasks across sixty Amazon EC2 instances. This resulted in substantial speed-up through parallelization and simultaneously allowed us to partially circumvent the IP-related restrictions. The article-retrieval programs required monitoring and occasional manual restarts when the instance was cut off.

3.3 Representation Extraction

To couch this problem in the realm of "big data," and to ensure that the algorithm can produce good representations, it is necessary to assemble as large a corpus as possible. Accordingly, the process for constructing company representations must be scalable to handle a potentially huge body of articles. Since processing is largely independent across articles and companies, this task constitutes an ideal application for the MapReduce framework. We divide the formulation of stock representations into three sub-tasks.

The objective of the first MapReduce task is to process articles tagged with an associated company to produce raw word counts for each company over all its articles. The input records are articles keyed by their relevant stock-ticker symbol and the article contents have been filtered for punctuation, though they often contain a substantial quantity of noise: terms that were not part of the pertinent news article but ancillary web content such as navigation bars, advertisements, and links to other articles. To produce the word counts for each article, article contents are split by whitespace, converted to lower case, and all terms that do not appear in the list of stop words are stored in a hash map. When all terms in an article have been either counted or discarded, word counts are pushed to the reducer, keyed by the stock symbol. The reducer then produces the company's overall raw word counts by merging counts of identical terms.

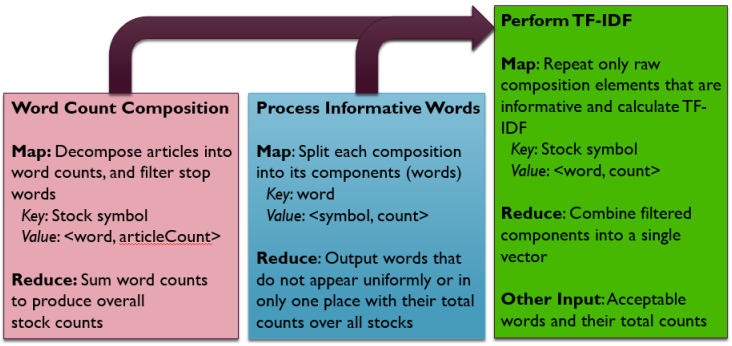

Figure 1: Workflow for text retrieval, importance pruning, and TF-IDF weight calculation in the first phase

Once the overall counts have been assembled for each company, it is possible to discard uninformative terms on the basis of their distribution across companies; this is the second sub-task. Using the output from the previous sub-task, the mapper produces records keyed by words whose value is a stock symbol-word count pair. Thus each reduce task is responsible for determining whether a given word should be used for the final representations. In the worst case, each reduce task receives one record for each stock, though the reducer is written in such a way that it requires a constant amount of space. The final output from the reducer is the term together with the number of companies it appears with.

Finally, having produced the raw word counts, the list of informative terms to retain, and the number of companies each term appears with, it is possible to produce the TF-IDF vectors for each company. The mapper in this task does most of the heavy lifting: it filters uninformative terms from the list of raw counts and calculates the TF-IDF value for each informative term. The list of informative terms and their company counts (i.e. document counts) are read in during the mapper setup phase and stored in a hash map. Once again, it is assumed that the vocabulary can fit in main memory. Mapper output records are keyed by the stock-ticker symbol and contain a word-value pair. The reducer simply concatenates these pairs together to form the final sparse vector representation for each company. A full representation of all three tasks in this phase are illustrated in Fig. 3.3.

4 Phase 2: Identifying and Inferring from Neighbors

In the second phase, we utilize the TF-IDF weighted feature vectors generated in phase 1 to create neighborhoods of stocks that are nearby in dimensionality. In order to reduce complexity and promote tractibility in calculating distances between stocks, we bucket stocks using locality-sensitive hashing prior to calculating order 2 Minkowski distance. We then implement a variety of monotonically decreasing functions that map the distance of stocks to their associated weights, with linearly decreasing being a direct mapping. Each stock's overall change in value over 30 days is propagated to each IPO in the bucket relative to its weight.

4.1 Locality-Sensitive Hashing

While more sophisticated dimensionality-reduction techniques exist, locality-sensitive hashing (LSH) is a very inexpensive technique that is highly parallelizable. Previous works identifying neighbors in high-dimensional space, such as distributed approximate spectral clustering [7], show remarkable computational gains by leveraging LSH.

Locality-sensitive hashing groups similar feature vectors into the same bucket by dividing dimensions at their most divisive point, where most of the data lies relatively equally on either side. It first probabilistically selects the most *important* features in the representation using the spans of each dimension. Equation 4 determines the probability that a dimension is selected to be a hyperdimension.

$$P(dim(i)) = span(i) / \sum_{d \in D} span(d) \tag{4}$$

For every hyperdimension chosen, a threshold for whether the candidate stock receives a 0 or 1 in its signature is chosen at the point of minimum representation. A user-defined amount of "bins" are chosen, and the span of the hyperdimension is broken up into that many bins. Each bin is populated with the count of how many stocks have a dimension that falls into that value. The bin with the lowest count is the point of minimum representation, and is selected as the threshold. Figure 4.1 illustrates the threshold selection. Note that the distribution of counts need not, and is unlikely to be, parabolic as illustrated.

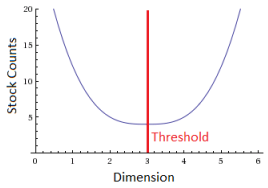

Figure 2: Threshold selection in LSH

We implement LSH in Hadoop similar to DASC [7]. The mapper is responsible for creating a hash from each TF-IDF vector over all selected hyperdimensions, giving a 1 for each hyperdimension m if it is within the threshold and 0 if it is without. Also like DASC, we approximate the bucketing by allowing the bit difference to be smaller than the selected hyperdimensions, some $P \leq M$. This occurs after the mapper and prior to the reducer. The reducer simply joins all stocks that have the same signature. Fig. 4.1 illustrates the entire LSH workflow.

4.2 Propagating Relative Performance from Neighbors

After arriving at neighborhood buckets by performing LSH on the stock TF-IDF vector dataset, we calculate a kernel matrix of order 2 Minkowski (Euclidean) distances between every IPO and its neighbor stocks. Since these distances are commutative, the resulting kernel matrix is symmetrical. For every non-IPO that is a neighbor of an IPO, the distance is fed into a monotonically decreasing function (several of which we compare in the results section) and its 30-day point change is propagated to the IPO relative to its weight. Equation 5 illustrates this weighted value approach.

$$val(IPO) = \sum_{s \in neighbor(IPO)} \frac{f(distance(s, IPO))}{\sum_{s' \in neighbor(IPO)} f(distance(s', IPO))} \cdot val(s) \tag{5}$$

The function used to map the distance to some weighted value need only be monotonically decreasing. A future avenue of work would be to create a single mixed parameter monotonically decreasing

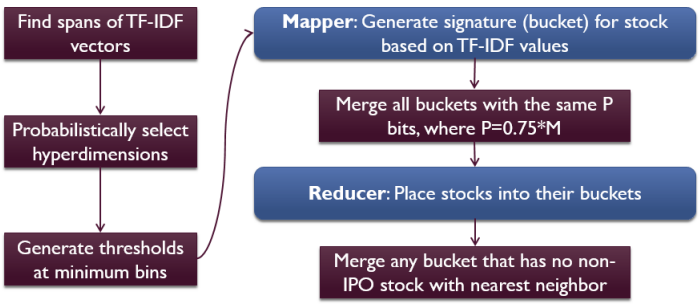

Figure 3: Workflow for approximate LSH bucketing in Phase 2

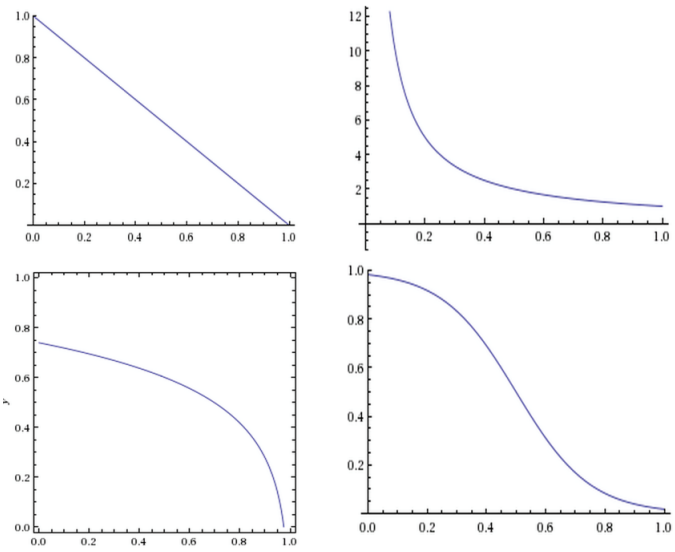

Figure 4: Models, clockwise from top left: linear, exponential decay, concave, sigmoidal

function and learn the shape. In this work, we explore four different functions: linear, exponential decay, sigmoidal, and concave. Figure 4.2 illustrates examples of these four models.

In addition to the value propagated from the neighboring stocks, we also examined if there was any intrinsic meta-performance of IPOs, such as generally decreasing or increasing after 30 days. We utilized central finite differences in exploring two new parameters, α, which indicates the weight given to some constant performance, and ϕ, the constant performance coefficient. Equation 6 describes the balance of constant performance and weighted propagation to be learned.

$$v(IPO) = (1 - \alpha)\phi + \alpha \cdot val(IPO) \tag{6}$$

Central finite differences is a methodology for exploring parameter values for multinomial functions by changing more than one parameter value at a time in the direction of greatest decrease of regression. In our experiments, we implement this as a gradient descent approach to arrive at the least regression between predicted IPO 30-day performance and the actual performance.

5 Experiments

In order to test the efficacy of our large scale data mining, TF-IDF classification, dimensionality reduction, and machine learning approach to predicting IPO 30-day performances, we collected over $100,000$ articles about $6,516$ stocks. Out of those stocks, 317 IPOs' performances were learned. In total, the data was over 6 gigabytes in size before the processing performed in Sec. 3. After processing, the total corpus was 700 megabytes and each feature vector contained $108,876$ unique words, though the vectors were very sparse. For each stock, around $1,000$ words had TF-IDF values indicating significance.

We chose 100 hyperdimensions and merged any buckets that had at most 1 bit difference, which resulted in 67 buckets. Several buckets were quite small, and buckets that contained IPOs but no stocks were merged with the closest bucket that had stocks. Small buckets with only stocks were not merged, and thus did not play a role in the learning. We performed 10-fold cross validation on the 317 IPOs in the dataset and averaged the resulting regression of each test set. We ran the α, ϕ-gradient descent on each of the 4 candidate models discussed in Sec. 4.

In every case, α converged to 1, indicating that there is no uniform meta-performance for the set of IPOs. These results are not included.

Model	Avg. Regression	Std. Error
Linear	5.60693	0.30827
Exp. Decay	5.60703	0.30829
Concave	5.60829	0.30842
Sigmoid	5.60646	0.30823
Increasing	11.22136	0.56112

Table 1: Average regression and standard error for every model

Results were not particularly illuminating between the various monotonically decreasing models. However, they all dramatically outperformed the monotonically decreasing function, indicating that stocks that are closer in dimensionality to the IPO do influence it's 30-day price more significantly than those farther away. Table 1 annotates the respective performance of each model. Please note that the standard error of each model overlaps, so there is no significant difference between each model. The regression listed is over all stocks. Translated to a per-stock value, each monotonically decreasing function mispredicted the IPO's performance by only half a point.

6 Discussion and Future Work

While much is to be desired of the rather lackluster results of the approach, there remains quite a bit of room to improve. Observe Fig. 6, which shows an example distribution of neighbors to an IPO (which can be thought of as the point of origin). While LSH does a good job of reducing the dimensionality by bucketing nearby TF-IDF vectors, neighbors still tend to clump at particular distances from the IPO. This can be due to the shape of the multinomial distribution, or that very small tightly knit clusters are ending up in buckets with other tightly knit clusters. This is problematic, as the approach relies on the assumption that there are a sufficient number of neighbors to make a well-informed inference.

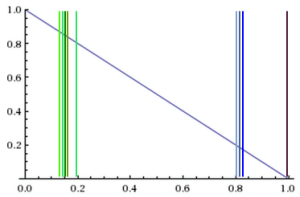

Figure 5: Example visualization of neighbor distribution using company Avinger's bucket.

The LSH bucketing appears to be performing well, though any reasoning is conjecture considering the large amount of neighbors (100+) each IPO has. For example, the company used in generating Fig. 6, Avinger, is an IPO that specializes in novel surgical materials. Just prior to going public, Avinger was awarded a patent for a sturdier and longer surgical catheter. Metabolix is a company with a much longer history. This company specializes in bioplastics. Interestingly, Metabolix provided the material for Avinger's new catheter, though that information was not present in the articles about either company. Metabolix was very close to Avinger in order 2 Minkowski distance, only 0.12 out of 1.0. However, Houston American Energy Corp, an oil mining company, also appeared in the same bucket, but was the maximum distance from Avinger in its bucket.

In order to tackle the dimensionality issue, another approach could be to utilize SVD in Phase 1 instead of LSH. A distributed KNN could be used in place of LSH in order to guarantee some number of existing stocks are used in the machine learning stage. Additionally, since the use of α and ϕ brought no reduction in regression, it should be removed to reduce complexity of the model, and should be replaced by a single mixed effect model with parameters for curvature and elevation. This would allow the true function shape to surface without the assumptions of shape made in this work.

7 Conclusion

In this work, we propose an approach for extracting and leveraging a large corpus of information about existing companies in order to predict the performance of new IPOs that do not have a large amount of empirical data. Experts at various financial firms, such as Bloomberg, utilize meta-information about markets or popular sentiment about domains in order to infer about companies. In order to explore these hidden indicators with unlabeled data in an unsupervised fashion, we create TF-IDF vectors out of all articles about a particular stock, pulled from the Yahoo! Finance News RSS feed. In order to create neighborhoods of lower dimensionality, we perform locality-sensitive hashing using 100 features and bucket similar stocks, ensuring that each IPO has at least a few neighbors to compare against. Then, using several monotonically decreasing functions and normalizing the output values, we see how well propagating all neighbors' 30-day performances predicts each IPO's 30-day performance.

The results did not indicate which monotonically decreasing function was the best representation. This was likely due to the neighbors each being clustered into one or two groups, making their relative weights rather meaningless. However, these monotonically decreasing functions all significantly outperformed a linearly increasing function. This result is promising, and indicates that a more subtly tunable function might capture the relationship more accurately. The bucketing itself seems justified, and interesting relationships are represented by individual buckets, such as Avinger's use of the bioplastics produced by its neighbor, Metabolix. Other dimensionality reduction techniques could be used instead of LSH to tackle the issue of small close-knit neighborhoods in each bucket.

References

[1] J. E. Berg, G. R. Neumann, and T. A. Rietz. Searching for google's value: Using prediction markets to forecast market capitalization prior to an initial public offering. *Management Science*, 55(3):348–361, 2009.

[2] J. E. Berg, G. R. Neumann, and T. A. Rietz. Searching for google's value: Using prediction markets to forecast market capitalization prior to an initial public offering. *Management Science*, 55(3):348–361, 2009.

[3] A. Brav, C. Geczy, and P. A. Gompers. Is the abnormal return following equity issuances anomalous? *Journal of Financial Economics*, 56(2):209 – 249, 2000.

[4] E. C. Notes on discovering trading strategies. 1999.

[5] H. Gunduz and Z. Cataltepe. Prediction of istanbul stock exchange (ise) direction based on news articles. *The Third International Conference on Digital Information Processing and Communications*, 2013.

[6] T. Hellstrom and K. Holmstrom. Predicting the stock market. *Technical Report Series IMATOM- 1997-07*, 1998.

[7] L. Huang, D. Yan, N. Taft, and M. I. Jordan. Spectral clustering with perturbed data. In *Advances in Neural Information Processing Systems*, pages 705–712, 2009.

[8] C. M. C. Lee, A. Shleifer, and R. H. Thaler. Investor sentiment and the closed-end fund puzzle. *The Journal of Finance*, 46(1):75–109, 1991.

[9] J. K.-S. Liew and G. Z. Wang. Twitter sentiment and ipo performance: A cross-sectional examination. *Available at SSRN: http://ssrn.com/abstract=2567295 or http://dx.doi.org/10.2139/ssrn.2567295*, 2015.

[10] A. Ljungqvist, V. Nanda, and R. Singh. Hot markets, investor sentiment, and ipo pricing*. *The Journal of Business*, 79(4):1667–1702, 2006.

[11] A. Ljungqvist, V. Nanda, and R. Singh. Hot markets, investor sentiment, and ipo pricing. *The Journal of Business*, 79(4):pp. 1667–1702, 2006.

[12] M. M. Forecasting intraday stock price trends with text mining techniques. *Conference on System Sciences*, 2004.

[13] C. D. Manning, P. Raghavan, and H. Schtze. Scoring, term weighting, and the vector space model. In *Introduction to Information Retrieval*, pages 100–123. Cambridge University Press, 2008. Cambridge Books Online.

[14] M. Mitchell and J. Mulherin. The impact of public information on the stock market. *The Journal of Finance*, 1994.

[15] A. Nikfarjam, E. Emadzadeh, and S. Muthaiyah. Text mining approaches for stock market prediction. In *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on*, volume 4, pages 256–260, Feb 2010.

[16] A. Rajaraman and J. D. Ullman. Data mining. In *Mining of Massive Datasets*, pages 1–17. Cambridge University Press, 2011. Cambridge Books Online.

[17] J. R. Ritter. The long-run performance of initial public offerings. *Journal of Finance*, 46(1):3–27, 1991.

[18] S. Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of documentation*, 60(5):503–520, 2004.

[19] S. K. Samanta and A. H. Zadeh. Co-movements of oil, gold, the us dollar, and stocks. *Modern Economy*, 3(01):111, 2012.

[20] R. P. Schumaker and H. Chen. A quantitative stock prediction system based on financial news. *Information Processing & Management*, 45(5):571 – 583, 2009.

[21] B. Wuthrich, D. Permunetilleke, S. Leung, V. Cho, J. Zhang, and W. Lam. Daily prediction of major stock indices from textual www data. *KDD*, 1998.

[22] M. Zekic. Neural network applications in stock market predictions-a methodology analysis. In *proceedings of the 9th International Conference on Information and Intelligent Systems*, volume 98, pages 255–263, 1998.

[23] J. Zeng, S. Zhang, C. Wu, and J. Xie. Predictive model for internet public opinion. In *Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007. Fourth International Conference on*, volume 3, pages 7–11. IEEE, 2007.