G-Money: Earnings calls Predict Stock Performance

Gierad Laput
SI
glaput@umich.edu

Gaurav Paruthi
SI
gparuthi@umich.edu

Gaurav Singhal
SI
gsinghal@umich.edu

Abstract

1 Introduction

This study's makes two contributions to the literature each in a different field. First, of interest to machine learning researchers is how to extract useful information from text. There has been significant progress in this arena, but specialized contexts such as finance present a significant challenge due to their highly esoteric language (Das & Chen 2007). Our study specifically aims to extract useful information from financial text data that can be used to predict a stock's *cumulative abnormal return* (CAR). Herein is the second contribution of our study to the literature. Given a buy-sell window the abnormal return is the difference between a stock's actual return over that time versus its expected return as given by the Capital Asset Pricing Model (CAPM). This is a fundamentally more challenging problem than simply predicting stock movement, for abnormal returns have little to no theoretical basis (Fama, 1998) **[]**. However, abnormal returns are usually triggered by *events*, defined as occurrences of information that has not already been priced by the market. This can include mergers, dividend announcements, company earning announcements, etc.

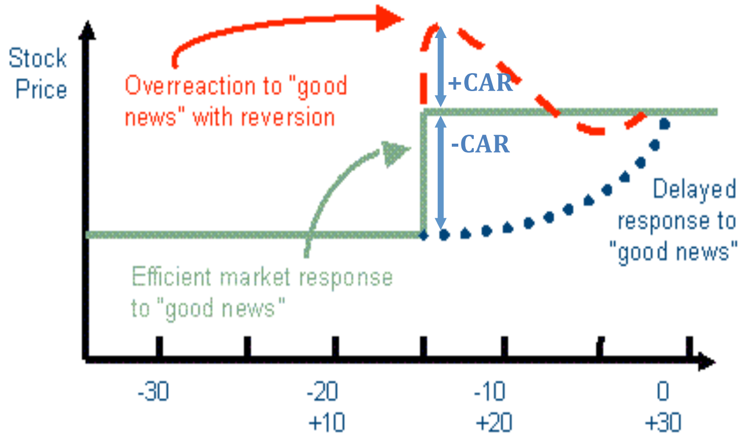
Stock Price
Overreaction to "good news" with reversion
+CAR
-CAR
Delayed response to "good news"
Efficient market response to "good news"
-30
-20
+10
-10
+20
0
+30

Figure 1: Cumulative Abnormal Return (CAR) and the Efficient Market Hypothesis

As an example, suppose a stock previously trading at a price/earnings ratio of 20 (i.e. costs $20 and earns $1 in profit per share) suddenly announces earnings of $1.25 per share. CAPM would predict a new price of $25, however the market could under-react or over-react to the news as shown in the above figure. Wall Street practitioners devote considerable resources to discovering such opportunities for such abnormal returns, investing heavily in stock analysts who research firms in order to generate superior returns. External studies have validated their claims (Bradshaw, 2012). However, even the best stock analysts can predict such abnormal returns with a frequency of XX% (CITATION).

Researchers have documented (Frankel et al. 2006) that quarterly conference calls are crucial to increasing the quality of analyst forecasts; hence analysts must be updating their beliefs using soft information obtained during these calls. At such calls the C-level managers of the company gather to announce earnings and important news and stock analysts engage in a back and forth dialog with the managers. Unfortunately, not only is analyst research proprietary and expensive, out of the reach of normal investors, but also the market reacts incredibly quickly to news events. The arbitrage opportunity window may have already closed by the time an investor can react to a particular event. This provides motivation for using machine learning to algorithmically analyze quarterly earnings conference call transcripts. By being able to predict a positive (negative) CAR the investor can thus know the probability of market over-reaction (under-reaction) and can short (buy) a stock accordingly, and institutions can even use such an algorithm as an input to their existing algorithmic trading agents.

The rest of this paper is organized as follows: in Section 2, we introduce the related work on this problem. We describe our data our proposed methods and the related motivations behind each technique in Section 4. In Section 5 we go over the experiment designs and their immediate results, further analyzing the results in Section 6. Lastly, Section 7 concludes this paper.

2 Related Work

Researchers have also shown that breaking news articles can be used as salient features for predicting stock market values. For example, Gidófalvi [5] used text from news articles as the main features for a naive Bayes classifier, which allowed him to predict the general movement of stock index movement for individual companies. Instead of predicting stock movement for an extended period, Gidófalvi limits his prediction within a predetermined "window of influence" (i.e., 20 minutes after a news article is published) to preemptively counter the effects of the "efficient market," thereby increasing the applicability of the predictive power of his system. Similarly, Schumaker and Chen also used news articles for estimating discrete stock price based on financial news articles [11] using a specially trained support-vector machine (SVM). And just recently, researchers have also shown that sentiments from Twitter can be a good metric for predicting stock market index movement [2]. All of these works show that news articles and general chatter, which are somewhat indicative of the reactions and generic generic sentiments of market investors, have an influence on stock price index, and that this phenomena can be modeled using machine learning.

Finally, our project extends Schoenfeld's work [10] on predicting stock market movements using signals from an analyst's tone during a conference call. Schoenfeld identified the overall sentiment of analysts by mining publicly available transcripts (text data) of quarterly earnings call, and he used this information to make a reasonable prediction of company stock price movement.

4 Data & Methodology

Because Schoenfeld was very interested in the work that we are doing, he facilitated the acquisition of our data. Using the financial information service ThomsonReuters, we were able to attain 47906 U.S. public company quarterly earnings conference call transcripts and 1285474 individual analyst statements, with each analyst statement assignable to a particular conference call. The conference calls cover 2136 companies over a period of eight years from January 2001 to April 2008. Then we were also able to attain the necessary CAR values for 1654 of the 2136 companies from the Center for Research in Security Prices' database. In total we have 33,346 quarterly earning conference calls from 1654 companies. In addition we were able to code the statements of every call to the title of its speaker—either CEO, CFO, COO, "Other Manager," or Analyst.

We pre-processed our text data to treat specific patterns as special tokens. For example, we detected for the presence of "dollar figures" such as $1,000,000 or $25K and converted matching patterns into generic "MONEY" token. We also eliminated special characters and numeric values to avoid unnecessarily complicating our vocabulary. Word pairs that have "negation terms" (i.e., not, never, no, nothing, etc.) are concatenated into a single term. For example, the phrases "not good" and "not bad", are converted terms into the respective tokens "n__good" and "n__bad." The associated dictionaries described in the next section were also updated to include their n__*antonym* pairs.

The efficient market hypothesis (EMH), which asserts that financial markets are "informationally efficient," allows us to treat each conference call as an independent event. Although at the micro-scale the market is not efficient and in fact is the very basis for predicting CAR, each conference call is spaced sufficiently far apart from each other that the market would have priced in all prior knowledge. Thus new "additions" of information can be considered independent events. CAR is normalized as a percentage of the expected return in order to compare across all the samples in our dataset. By "de-panelizing" the data we are able to use a variety of off-the-shelf machine learning techniques and approaches. As anecdotal evidence of the EMH at the macro-level, 49.8% of our samples had negative CARs (50.2% positive), whereas EMH would predict exactly 50%.

4.1 Feature Definition

Extracting “useful information” from unstructured text is a vexingly ambiguous and difficult problem. Limits in time and domain expertise prohibited using more robust mining approaches that rely upon lexico-syntactic pattern recognition wherein classes and hierarchies of certain domain-specific knowledge are pre-programmed. Furthermore the lack of theory as to why CAR can be negative or positive makes this an incredibly difficult approach. Our approach instead, builds upon the work of Das & Chen (2007) [CITATION] who use a number of sentiment classifiers to predict stock movement from stories on financial blogs. We not only use several sentiment classifiers, but also including topic modeling, and vector-distance approaches using n-grams. In effect we use one set of machine-learning algorithms to generate features then use another set to classify. The details of each are described below:

- **Naïve Classifier**: This algorithm is based on a word count of positive and negative connotations words. We utilize a specifically designed financial dictionary for positive and negative sentiment from Loughran & McDonald (2011) [CITATION]. The sentiment measure is defined as the ratio of the word count of positive sentiment words to the word count of negative sentiment words:

$Sentiment_i = \frac{|positive\ words|}{|negative\ words|_i}$ for conference call *i*. Some words may have unique importance per classification that simply are not capturable by the prior method. For example, the word “buy” can be a very strong indication of bullish sentiment. Using the training corpus we compute a measure of discriminating ability for each word in the lexicon and then replace the simple word count used by the naïve classifier algorithm with a weighted word count. The discriminant formula for each word is $F(w) = \frac{\sum_{i \neq k}(\mu_i - \mu_k)^2/|I|}{\left[\sum_i \sum_j (m_{i,j} - \mu_i)^2\right]/\sum_i n_i}$ where $i \in I$ is the set of classes (buy, sell) , n_i is the set of messages in each category, μ_i is the average number of times a word (w) appears in category *i*, and the number of times a word appears in a conference call transcript (j) is denoted $m_{i,j}$. We also use four other specialized financial dictionaries that connote words for uncertainty, litigiousness (per legal matters), superfluity (indicating frill), and constraint (indicating evasion).

- **NLTK**: The Natural Language Toolkit (NTLK) Classifier is also a bag-of-words approach, but utilizes the naïve-bayes algorithm, analogous to SPAM email classification. However, for this we could not use Loughran & McDonald’s financial sentiment dictionary as it lacked the requisite log-probabilities. Instead we use an existing movie-review corpus with reviews categorized into *positive* and *negative* categories, and a number of trainable classifiers.

- **Word Counts**: Harvard’s IV4 dictionary [CITATION] contains nearly 12000 words broken down into 182 different categories including four semantic dimensions—over/understatement, positive/negative, strong/weak, active/passive—and numerous other specialized categories. For example there exists a “litigious” category specifically for words that indicate legal matters. Similar to the naïve approach, a score is assigned to each conference call along each of the 182 different categories based on the number of words it contained that matched the particular dictionary

- **Latent Dirichlet Allocation (LDA):** In LDA each document maybe viewed as a mixture of various topics. Given a fixed number of topics, LDA uses collapses Gibbs Sampling to produce the optimal set of words associated to each topic. Then per each document it assigns a weight measure to each topic indicating what mix of topics compose the corpus. Thus these become a new set of features for the document, and the LDA corpus can then be used to pare down the dictionary for vector-distance classification.

We used the Mallet package [CITATION] for training the LDA topic model. Using the training set, we train an LDA model with K = 20 topics. We use 500 iterations and optimize hyperparameters for every 20 iterations. The parameter of smoothing over topics is $\alpha = 50.0$, smoothing parameter over words is $\beta = 0.01$.

- **Vector Distance Classifier:** Each conference call transcript is converted into a sparse vector indicating what elements in the word-set (dictionary) comprise the message. We pare down the dictionary using the output from LDA. Then each new message is classified by comparison to a cluster of pre-trained vectors (T_j) in this space using the familiar cosine similarity measure. The conference call is assigned the class of the message in training set it is most similar to where for each conference call (m) we calculate its cosine similarity with every pre-trained vector (T_j) in our set: $\cos(\theta_{m,j}) = \frac{m \cdot T_j}{|m||\cdot T_j|} \forall j$. We vectorize each message not only using unigrams but also bi-grams and tri-grams, creating dictionaries accordingly.

Prior to running any of the aforementioned algorithms we iterated through several approached for negation and lemmatization. Negation contends with statements that are invert propositions, i.e. “not good” can be considered equivalent to “bad.” Lemmatization “stems” awards in order to improve performance, i.e. “experienced” becomes “experience.” These are both discussed in detail later. Last but not least we also had the following bits of financial data as features: expected earnings, actual earnings, and prior quarter earnings. The latter two variables were normalized with respect to the expected earnings, and an indicator variable was included for if earnings were positive or negative.

4.2 An Ensemble Approach

Similar to our shot-gun approach per producing features, we use an ensemble approach per classification. We designed our approach knowing full well that a single classifier will never give us a very strong result due to the difficulty of our problem and the weakness of text as an information source. Hence we sought ways to extract marginal gains. Using multiple classifiers improves the signal-to-noise ratio and also allows us to implement meta methods such as boosting and bagging, which are discussed later.

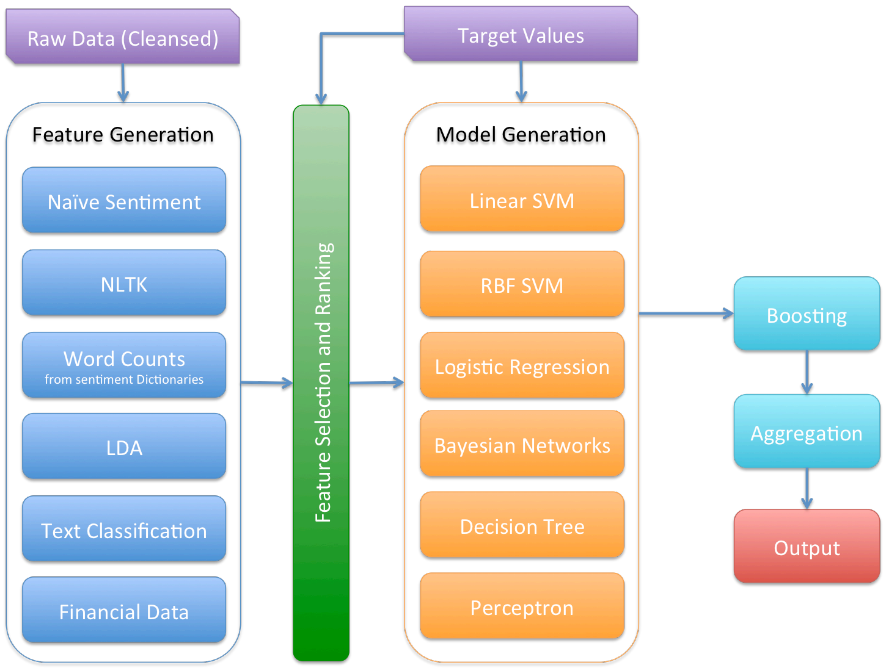

Figure 2: Machine Learning Pipeline

Linear SVMs can be considered weak classifiers; hence we use an assortment of such classifiers along with ADABoost to produce a single classifier. This is combined with other strong classifiers such RBF-kernel SVM, Bayesian networks, and logistic regression, that have all been shown to be very powerful for text classification (CITATION). We combine the multiple strong classifiers via aggregation.

5 Experiments & Results

Generally we followed a process that mirrors the pipeline depicted in the above figure. We first narrowed down our feature set, then refined and selected amongst different classifiers before performing various meta techniques in an attempt to achieve marginal gains.

5.1 Feature Extraction

We use Information Gain Attribute Evaluation [cite] as the metric for feature extraction. In general terms, the expected *information gain* (IG) is the change in information entropy (H) from a prior state (S) to a state that takes some information as given (a), i.e. $IG(a)=H(S)-H(S|a)$. This is operationalized as the Kullback-Leiber divergence which measures the expected number of extra bits to encode the probability distributions for S versus $S|a$. We chose this evaluation metric because of its speed and superiority to paired t-tests. T-tests require a strong assumption that the dataset be normally distributed; in our case, we cannot fully justify that our dataset follows a normal distribution. A rank-sum test for significance might have been another good option.

WEKA is a well-known framework for machine learning with the ability to provide a ranked-list of features using the Information Gain metric. Our top 20 features are shown in the below table.

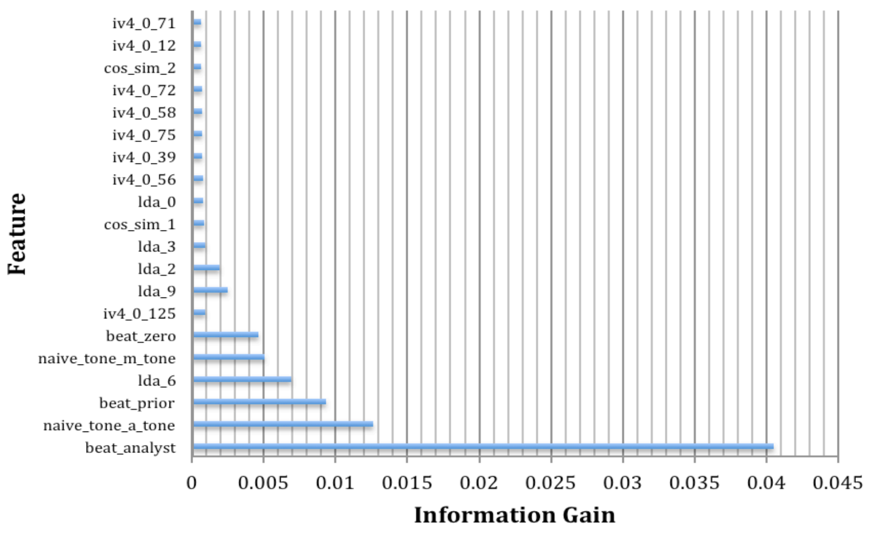

Figure 3: Feature Extraction Results

As one can see many of the IV4 features are not included. Although the IV4 dictionary was quite large, divided over 182 categories, each category contained no more than several hundred words, and sometimes as few as 25. Thus the relative per category infrequency of these words in the conference call corpus ascribes little to no value to the features mapping to small IV4 dictionaries. Additionally, the NLTK features produced no value whatsoever, this was somewhat expected given that it was trained on an entirely different corpus; however, what is unexpected is that the only relevant naïve feature is positive/negative valence to tone. Other features in this class such as litigious, uncertainty, superfluity, and constraint produced no gain.

5.2 Classification

After selecting the optimal features we first refined our text classification models over a possible 180 different permutations as given in the below table. We trained each permutation using an SVM with text-only features.

Text Classification Refinement	
Variable	**Values**
Corpus	Analyst, Manager, Analyst + Manager
Lemmatize	Original, Lemmatized
Vocabulary	LDA Vocabulary, Original Vocabulary
N-grams	unigram, bigram, trigram
% Training Data	25%, 50%, 75%, 85%, 90%

Table 1: Text Training Permutations

We discovered that lemmatization is too aggressive and we achieve better performance by not stemming words. Additionally the vector-distance algorithm produces best results using trigrams and only using analyst statements. This makes sense as trigrams allow the algorithm to consider "collocations" of words that appear close to each other, and manager statements are often very “fluffy” and over-optimistic, thus adding a bias to cosine distance.

Given the optimal permutation for maximizing the impact of our text features, we trained a gamut of classifiers using the features given by the information gain. The results are shown in Figure 4 below. To reduce variability, we performed 10 fold cross validation using different portions of the data and averaged the final results. The results shown reflect the cross validation accuracy.

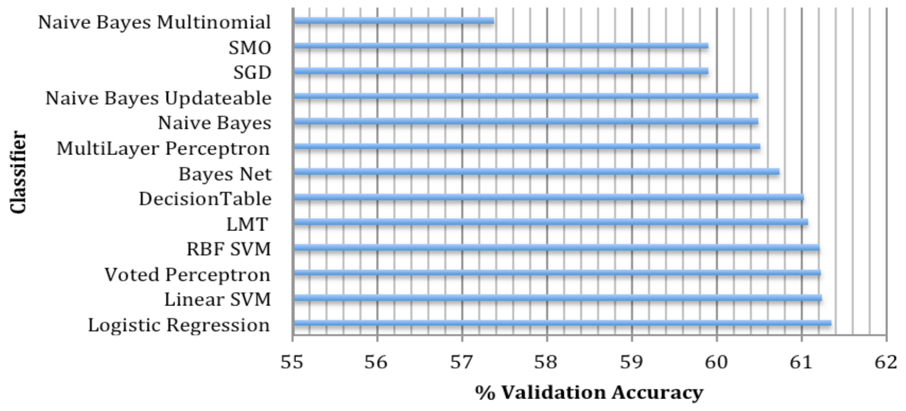

Figure 4: Classifier Performance

Logistic Regression is the optimal classifier as given by this experiment. **WHY?** To produce our *baseline* result we use logistic regression on only the financial data, excluding all text features, for the aim of our endeavor is to improve upon this result using text data. This produces a baseline test accuracy of 60.3%.

Finally the ranking from Information Gain was subsequently used to sequentially train the best classifier over the top N features, iterating N from 3 to 15. At each iteration we tally the classification result (using a validation and testing dataset). We found that features beyond our top twelve features actually decrease performance due to overfitting. Figure 5 shows the results of this experiment

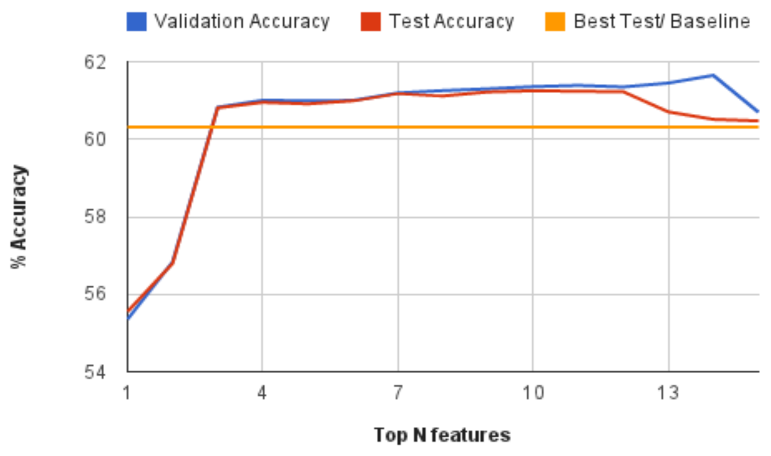

Figure 5: Number of features vs. Model Accuracy

5.2 Meta

Since we use many training features and classification algorithms, we experiment with the ensemble-meta classification algorithms in the WEKA toolkit.

Bagging, an ensemble meta-algorithm designed to reduces variance and avoid overfitting in classification, is a special case of model averaging. In our experiments bagging did not show any improvements when used with our best classifier (logistic).

Boosting: We applied the AdaBoost boosting algorithm to improve weak learners in our training set. Boosting on our best classifier was not found to be useful, In future work, we would like to perform AdaBoost with features like the IV4 dictionary, which are week classifiers but were removed from our training.

Voting and Stacking also didn't give us any significant improvements.

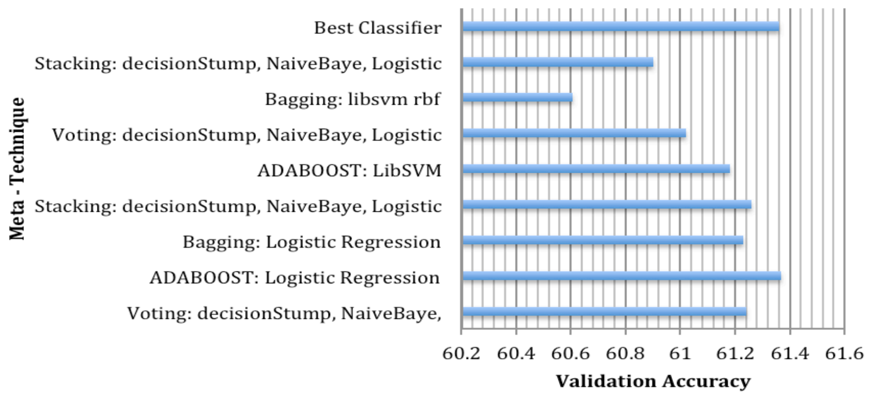

Figure 6:

6 Discussion & Analysis

We consistently achieve a 1% increase in performance using our text features as compared to our baseline. This is actually not a bad result in this domain. Even the best analysts can reasonably predict only

One over-arching theme that explains much of our result is the "curse of dimensionality" haunted our dataset. For us to achieve our best results we first had to find the optimal number of features, i.e. reduce the dimensionality of our problem to its max-min. In our case increasing dimensions beyond this point only decreases the linear separability of the data. This is the reason why discriminant-based classifiers (Logistic Regression, SVM, Voted Perceptron) not only perform the best, but also all perform approximately the same. Given the optimal dimensions they all essentially found the same discriminant between classes. Logistic Regression gives a slightly better result most likely due to the fact it uses a log-loss function and thus fits "less-tightly" than SVM, which utilizes a hinge-loss function. Conversely, the other algorithms we employed such as decision trees and Bayesian nets produce only marginal (<0.2%) increases in accuracy with text features. This also makes sense given the special orientation of our data. Most of the features are continuous nominal values that are easily represented using Euclidian geometry.

Also, we have validated our biggest assumption that analyst conference call transcripts can be used to predict CAR. Without a shadow of a doubt we can say that conference calls are indeed.

Dirichlet parameter	Keywords								
0.62479	good	great	guys	**quarter**	morning	kind	question	give	bit
0.83077	ve	kind	time	lot	sort	guess	question	don	back
0.30879	business	growth	revenue	terms	margin	year	margins	bit	acquisition
0.61062	year	guidance	growth	question	kind	bit	guess	give	rate
0.05497	questions	revenue	talk	question	wondering	cable	business	market	advertising

Figure 7: Topic keywords and their Dirichlet parameter (proportional to the frequency of the topic)

It is also a somewhat surprising finding that our LDA topics make "sense." The word choices the algorithm produced are sensible representations of topics that are commonly talked in the calls. When these topics are classified along with the sentiment they indicate the analysts sentiment with respect to the topic.

6.1 Bla

7 Conclusion

Bla

6 References

Bradshaw, M. 'Analysts' Forecasts: What Do We Know after Decades of Work?' Working paper (2012).

Eugene F. Fama, Market efficiency, long-term returns, and behavioral finance, *Journal of Financial Economics*, Volume 49, Issue 3, 1 September 1998, Pages 283-306, ISSN 0304-405X, 10.1016/S0304-405X(98)00026-9.

Frankel, R., Kothari, S.P., and Weber, J. 'Determinants of the informativeness of analyst research.' *Journal of Accounting and Economics* 41 (2006): 29-54.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System*. New York: TELOS/Springer-Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hiippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.

Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.