Predicting Dow Jones Movement with Twitter

Esther Hsu (estherh@stanford.edu)
Sam Shiu (bwshiu@stanford.edu)
Dan Torczynski (dtor1@stanford.edu)
CS229 Final Project, Autumn 2011, Stanford University

***Abstract*—The use of machine learning in the realm of finance is becoming much more prevalent as algorithmic trading catches on. Similarly, online social networking data becomes more valuable with new research in mining. The goal of this project is to take the next steps in these directions. Using different methods to interpret Twitter content, we hope to predict movement in the Dow Jones Industrial Average. More specifically, we find the word counts of our corpus of tweets using different word lists, calculate high-level features, and use machine learning techniques to see whether a correlation with the stock market is likely to exist.**

I. Introduction

Predicting the seemingly unpredictable stock market has always been a subject of study. Factors like current events and human emotion have been shown to play a role in the behavior of the stock market; however, the difficulty in evaluating these has prevented significantly accurate prediction. With the rise of online social media, more data than ever is available for analyzing the current state of the population. Our aim in this study is to make use of Twitter content to find trends or characteristics that could correlate with the movement of the Dow Jones. Each of our methods in extracting features begins with counting the number of times certain words appear in our corpus of tweets in the days leading up to the date for which we are trying to predict, and developing higher-level features from these counts that we postulate could help predict movement in the stock market.

II. Prior Work

Our project began with a 2010 study by Bollen, Mao and Zeng[1]. Their work was based on sentiment analysis of Twitter content to predict movement in the DJIA. Using a word list generated from the Google Profile of Mood States, they were able to describe each day as 6 different moods as features: Calm, Alert, Sure, Vital, Kind and Happy. They found that Calm had high correlation with the stock market and their predictions achieved 86.7% accuracy. Although the mood of the nation is almost certainly an indicator of economic behavior, we felt that limiting sentiment analysis to GPOMS did not allow for full use of the data available. We hope to explore more techniques to see what other ways Twitter content is predictive of the DJIA.

III. Dataset and Resources

Our corpus of tweets comes from Stanford SNAP's dataset collection by Jure Leskovec [2]. It consists of approximately 476 million tweets (although we did not use the entire set) from June 11, 2009 to December 31, 2009. We divided the data into separate days, and used approximately 1-2 million tweets per day (except for July 15, 30, 31, and October 31, for which there was no data).

We used several different tools to implement our methods:

- SVM: SVM-Light, Liblinear
- Neural networks: FANN, nnet R Library
- Recursive partitioning trees: rpart R Library

IV. General Method

To generate features for each day based on its tweets, we would keep track of the number of times certain words appeared that day. In some of our methods, we used these numbers directly as features (sometimes normalized and sometimes not), while in others, we calculated higher-level features from them. Therefore, the first step for each of method was to create a lexicon of words of which we would keep count. The second was to decide on how to interpret these counts.

Obviously, in trying to predict the DJIA, we would in real life only have information from previous days. Therefore, the features we assigned to day i were the word counts (or the features derived from them) from days $i-1$, $i-2$ and $i-3$. We included the DJIA from the previous days as features for each model as well. Generally, then, the features for day i for each of our models would follow the format:

$$x^{(i)} = [d_{i-1}, d_{i-2}, d_{i-3}, F_{i-1}, F_{i-2}, F_{i-3}]$$

where d_i is the DJIA of day i and F_i is the vector of word counts (or features derived from them).

Our prediction labels were based on the DJIA close of each day. Specifically, we wanted to predict whether the close of day t would move up or down from the close of day $t-1$. We used different ways of interpreting $d_t - d_{t-1}$, where d_t is the DJIA of day t, as labeling schemes:

- Binary: +1 if $d_t - d_{t-1} \geq 0$; -1 if $d_t - d_{t-1} \leq 0$
- Upper Band: +1 if $d_t - d_{t-1} \geq b$; -1 if $d_t - d_{t-1} \leq b$
- Lower Band: +1 if $d_t - d_{t-1} \geq -b$; -1 if $d_t - d_{t-1} \leq -b$

The purpose of the upper and lower bands was to create a "band" of width $b * 2$ (illustrated in figure 1) where $b > 0$, so that the predictor would to be able to detect large movement in the DJIA, as opposed to small deviations, as this would be important to know in actual trading.

Our features and labels were generally plugged into a linear-kernel SVM, and sometimes a neural net or recursive partitioning tree if felt that the SVM results were inconclusive. Initially,

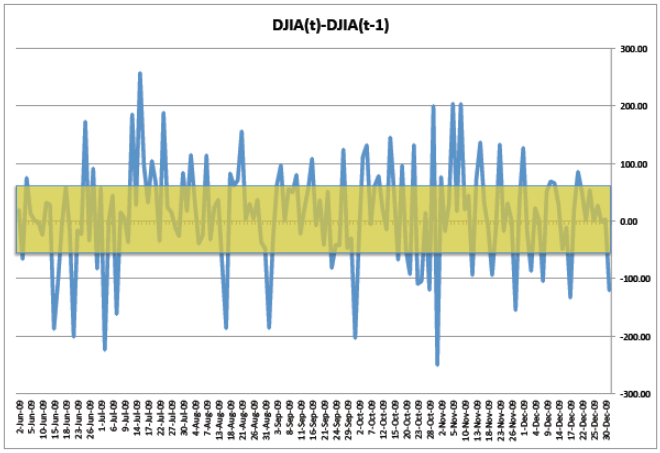

Fig. 1: "Band" with $b = 50$. Points inside the band represent small changes in the DJIA, while those outside the band represent more significant movement.

we attempted training on the first 4 months (June-October) and testing on the last two (November-December). However, we found that since November and December had mainly upward movement, our results were difficult to interpret. We then changed to training on the first 3 and testing on the last 3, since the October labels had more variance to them.

We also took a different approach to reflect the dynamical nature of the stock market. Our data was divided into M time periods of (almost) equal length: $T_1, T_2, ...T_M$. Then, for testing on T_i, we only used T_{i-1} for training. We varied M but usually found that time periods of approximately a month worked well. We felt that this helped to model the ever-changing tendencies of the stock market.

V. Baseline

To act as a baseline, we put together as many low-level features as possible for prediction. We were mainly looking to achieve slightly more than 50% accuracy, as an indicator that there could be some set of higher-level features to be calculated that would perform even better.

A. Features

In light of the results of the previous 2010 study, we wanted to include words to reflect sentiment analysis. However, we also wanted to expand the lexicon to find any other possible connection between Twitter words and the DJIA. Accordingly, we combined two lists:

1. **Alex Davies**: Specifically created for sentiment analysis on Twitter, this list consists of approximately 7400 words that with high probability was associated with the moods "happy" or "sad". Note that although this focus on positive/negative mood is different from POMS sentiment analysis, we felt the fact that it included very common words was important in making the most use of our data.
2. **Fiction**: A list of approximately 2000 of the most commonly used words in fiction, obtained from Wikipedia.

This resulted in a lexicon of approximately 8000 words, or about 24000 features per day. The normalized feature corresponding to day i and word j was calculated as:

$$f_j^{(i)} = \frac{\#\text{ of times word } j \text{ appeared on day } i}{\sum_k \#\text{ of times word } k \text{ appeared on day } i}$$

B. Results

When testing and training on different time periods, we found that using the baseline features in a linear or polynomial SVM and different labeling schemes would result in entirely +1 predictions. Although this yielded a decent accuracy of 69% (the end of 2009 consisted of mostly upward movement for the DJIA), it was clear that always predicting upward movement would be as inaccurate as random guessing. Our interpretation after seeing these results was that the baseline data was too convoluted to separate, and therefore the model would choose whichever label was more common. We also found that although decision trees did not have this problem, their results had very close to 50% accuracy. Thus, our next step was feature selection.

VI. Highly correlation words

We hoped to improve from the baseline approach by cutting down on the number of features and perhaps remove noise. Our approach was to choose a subset of the words that we felt would be most predictive of the DJIA.

A. Method

To reduce the number of features, we decided to keep only the features corresponding to the words that were most correlated with DJIA movement. Correlation for word j was calculated as the covariance of vector f_j, where $f_j^{(i)}$ is the ith entry, and vector D, where the ith entry is the DJIA for day i. After calculating the correlation coefficient for each word for the days in the training set (covariance matrix illustrated in figure 2), we kept any word with correlation $\|\rho_j\| \geq 0.85$. This resulted in a subset of 168 words.

B. Results

Our results for the correlation model ended up being almost identical to that of the baseline. For each training set, SVM and neural networks predicted almost entirely +1 on the test set, and decision trees were almost completely random. So, our next approach was to develop fewer, higher-level features so that our data was less convoluted.

VII. POMS Sentiment Analysis

A. Model

The previous paper by Bollen et al. claimed that the sentiments of the USA as measured by Twitter had predictive correlation with the DJIA. In attempt to confirm or reject this claim, a similar approach was attempted. From the original 65 seed words of the POMS [4], a new list of 350+ words was constructed using a Thesaurus implemented by Dekang

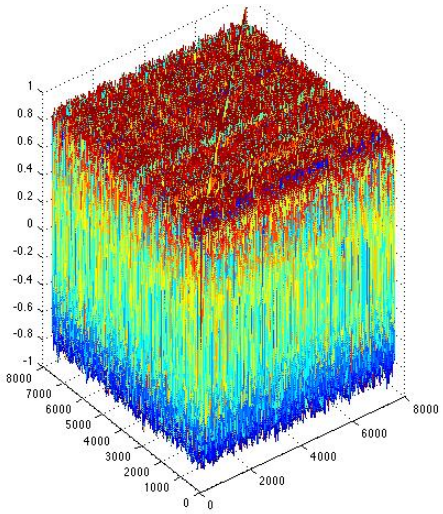

Fig. 2: Illustrated covariance matrix of baseline word list and DJIA movement. The words with the highest correlation were chosen as features for our second method.

Lin [5]. Like the original 65 words, each of these 350 words mapped back to one of six sentiments. The original paper said that the sentiment calm was the correlated feature, without explaining how they arrived at calm considering it was not one of the original six moods of the POMS. Consequently, our approach did not attempt to construct or interpret calm from our data, but instead used the six POMS sentiments in their original form. The six moods were generated from the Twitter corpus, as can be seen below in figure 3, and used as features for the SVM.

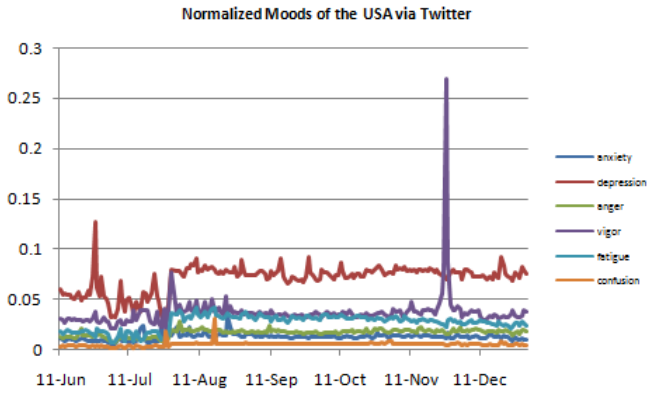

Fig. 3: Twitter moods generated from 350+ POMS word list. The six moods were normalized by the total number of tweets of that day to isolate the mood change as a percentage of the total populations feelings.

B. Results

From figure 4, it appears that not only does the SVM not learn with more training examples, it also does not ever have accuracy above 60%, compared to the 86.7% claimed by the paper. This could be for several reasons. First is that the original paper used proximity correlation to build their expanded sentiment word list, not a Thesaurus. Second is that we never identified a calm feature, and the original paper could have constructed a more informative feature from a combination of moods or another source altogether. Finally, we did not have as large a training set. Their set consisted of months February-November, whereas we only had months July-November. Consequently, our finding neither confirms nor rejects the original claim. It does, however, suggest that their claim might be slightly over-exaggerated because while not made explicit, it is likely one of our six features closely matched their calm. In addition, if calm did have a correlation, even though we used a smaller training set, we would have expected to see a higher than 60% predictive accuracy.

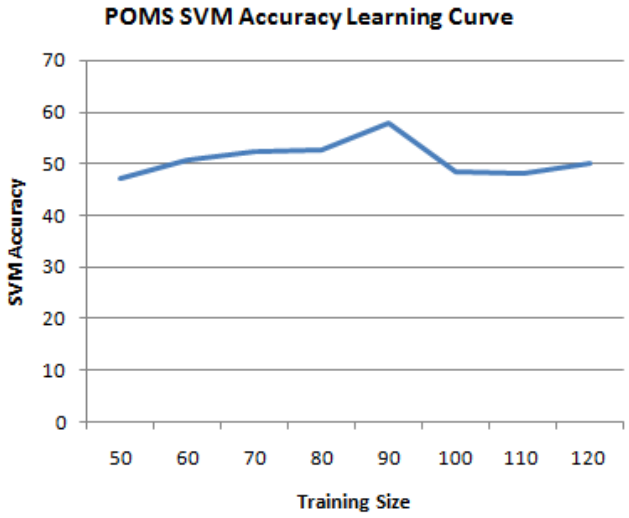

Fig. 4: Learning curve for SVM using Twitter POMS Features.

VIII. SVD Grouping

Our last method uses a few, high-level features, but still makes use of all the word counts we've calculated. More specifically, it uses singular value decomposition to find the largest components in the word covariance matrix. Instead of having concrete groupings, like specific moods in sentiment analysis, the motivation behind this was to find the most significant groupings of words. We expect that these groupings would correspond to sentiment and mood, but also to various trends reflected on Twitter.

A. Features

In addition to the baseline lexicon, we also added a word list developed from Google ngrams [6]. From a corpus of over 1.5 TB of 5 word ngrams, 15 GB were chosen at random and used to construct a proximity list of about 4200 extra words. This proximity list consisted of words which were frequently found in conjunction with the 65 seed words from the POMS as measured by the Google ngrams. Words with counts below a threshold were discarded as well as common but meaningless articles and prepositions.

Firstly, we took the SVD of the word covariance matrix for month-long time periods in our training set. Since covariance matrices are positive semi-definite, this is equivalent to eigenvalue decomposition. We found that these were quite similar for each time period; we decided to use the decomposition for the time period corresponding to June 15-July 15. The groupings were then formed from the first 7 eigenvectors (meaning, those eigenvectors corresponding to the largest eigenvalues):

$$u^{(1)}, u^{(2)}, ...u^{(7)}$$

Each eigenvector corresponded to two groups, based on the indexes of the positive and negative elements of the eigenvector. We kept the most significant indexes, ignoring elements in each u where $-0.025 \leq u_i \leq 0.025$ to simplify and eliminate noise. Since the first eigenvector had only positive elements, we were then left with 13 sets of indexes:

$$w_+^{(1)}, w_+^{(2)}, w_-^{(2)}...w_+^{(7)}, w_-^{(7)}$$

Each w had approximately 50-100 non-zero indexes, which corresponded to the words in that group. Feature i for day j was then calculated as the number of times a word in group i appeared on day j. We found that the groups had varying degrees of correlation, as shown in figure 6.

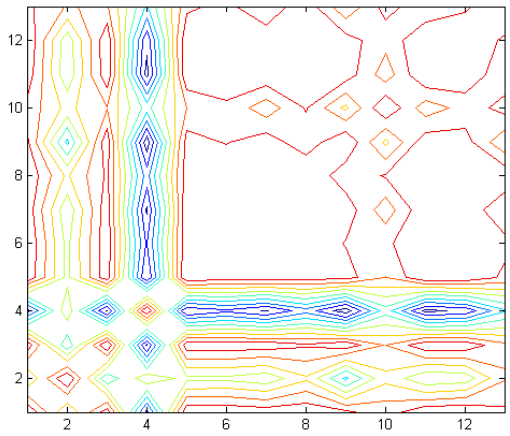

Fig. 5: Illustrated covariance matrix of words, grouped using SVD. We postulate that these groups correspond to moods or trends in Twitter.

B. Results

Our results using SVD groupings worked quite well. Using a linear L1-regularized L2-loss support vector classification from Liblinear, we trained and tested on time periods $T_1, T_2, ...T_M$ of varying M. Accuracy for band-labeling with $b = 10$ was slightly above 70%, with higher accuracy as M became larger and time periods became smaller. Although this goes against the intuition that more training data is better, we believe that it is indicative of the sporadic nature of the stock market; the most recent data is the most important.

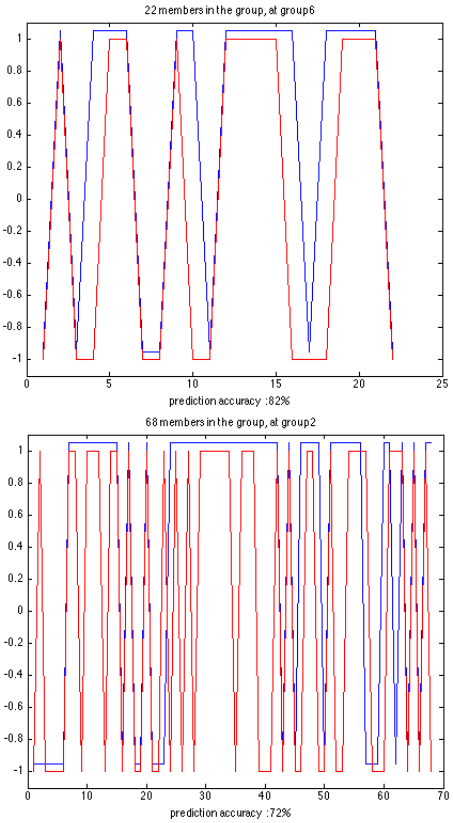

Fig. 6: Results of SVM grouping; blue are the actual labels and red is the predictions. Top: $M = 6$, where each time period had 22 days. 82% accuracy was achieved. The test period corresponded to some time in October. Bottom: $M = 2$, where each time period had 68 days. 72% accuracy was achieved. The test period corresponded to September-December.

C. Portfolio Application

Since SVD grouping yielded such high-accuracy results, we decided to apply our predictions to an actual portfolio simulation to see if we could make a profit. Our trading account was set up to employ a long trading strategy only, ignoring short strategies for simplification. We also assumed our account to be 50% margin-able. Since our results were binary, we used the up band labeling scheme with $b = 10$, with +1 indicating a long signal.

The sell decision was more mechanical. When a trade was committed, we let the market decide for us by simply looking at the drawdown. The drawdown is a price reversal from its highest point in a given period. Thus, a 5% drawdown means a stock price has moved up to its highest point in that period and then drop 5% below the highest point. If the price pulled back beyond 5% of the highest price achieved, we automatically sold that trade.

Our trading model only committed 15% of the portfolio for each trading decision. Our result indicates that different drawdown percentages results in different holding periods for each trade. A 5% drawdown provides a holding period of about 15 to 3 days in our testing period. The simulated profit that resulted was about 6% in 70 days. A 10% drawdown provides

a holding period of about 30 to 12 days, and the simulated profit was about 11%.

IX. Further Research

Throughout our study, we felt that our analysis was slightly wanting in the fact that we had a very limited data set to train and test on. Although the data itself was huge and pre-processing was arduous, it resulted in only about 160 days. It was difficult, then, to state accuracy percentages when each of our groupings was only about a month's worth of days. Ideally, to confirm our results, we would have liked to train and test on years worth of data. While the highest month accuracy of 82% for a grouping of 6 is promising, it is important to note that the average accuracy for this grouping of 6 is only 62%. This suggests that we have isolated an eigenvector that resonates well with one particular group at one particular group sizing, and does moderately well with the other groups at that sizing. In addition, it is unclear whether this group sizing is optimal and will hold up for additional months after December. With our limited data set of only 6 months, it is hard to determine a robust group sizing, and we leave this hypothesis for further testing in subsequent research with larger data sets.

As the SVD results returned the highest rates of accuracy, we believe this method of feature extraction from Twitter to be the most promising. Our study was only able to touch on this technique, and so we hope to gain more insight to improve on these features. For example, since the features were merely unnormalized, unweighted sums of word counts, we speculate weighing these numbers by their eigenvalues and eigenvector elements could help.

We also thought that perhaps our method of data collection could be improved upon. For example, instead of simply using word counts, there are methods available that extract the most significant words from a body of writing. This would remove any noise that common or irrelevant words might cause.

X. Conclusion

In the end, we were able to achieve significant levels of accuracy with SVD grouping. This was perhaps due to the fact that this model used a few high-level features, as opposed to the more "brain-dead" approach of using a huge amount of low-level ones. It was clear that our baseline approach was much too convoluted to make predictions that made sense. The SVD grouping made use of a large amount of data, but had few features that were especially modeled to capture the principal components of Twitter and DJIA correlation. Our POMS approach that was based on the 2010 study was unable to capture any useful features, which was perhaps because our lexicon was build on heuristics instead of around the dataset.

In conclusion, we feel that the SVD grouping method has potential to give real results. Ideally, we would have liked a larger dataset for testing to confirm this; however, it is a topic worth exploring and a step towards learning about the connection between online social media and economics.

XI. Acknowledgments

We would like to thank Mihai Surdeanu for all the advice and help he provided through multiple meetings and e-mails, and of course Professor Andrew Ng and the rest of the Autumn 2011 CS229 staff for their dedication to making Machine Learning a fun and rewarding experience.

References

[1] J. Bollen, H. Mao, X. Zeng. Twitter mood predicts the stock market. Journal of Computational Science, Volume 2, Issue 1, March 2011, pgs 1-8.
[2] J. Leskovec. Snap.stanford.edu. Stanford Large Network Dataset Collection.
[3] A. Davies. Alexdavies.net. A word list for sentiment analysis on Twitter.
[4] M. Lorr, D. McNair, L. Droppleman. Profile of Mood States. Mult-Health Systems, Inc.
[5] D. Lin. Dekang Lins Proximity Based Thesaurus. http://webdocs.cs.ualberta.ca/ lindek/downloads.htm
[6] Google Ngrams. http://books.google.com/ngrams/datasets