Predicting Stock Price Movement Using Social Media Analysis

Abstract

In this project, we aim to predict stock prices by using machine learning techniques on data from StockTwits, a social media platform for investors. We demonstrate the results, and compare the prediction error, of several classification and regression techniques using aggregated StockTwits messages as a source of input.

I. Introduction

Social media platforms such as StockTwits can provide a wealth of information on real-world patterns and behaviors. We offer an analysis on a specific application of social media, pertaining to finance: using aggregated StockTwits message data to make statistically significant price predictions. Our underlying assumption is that there exists a correlation between market price action and the metrics that we extract from this aggregate sentiment, indicating that it can provide a meaningful, actionable slice of real market conditions and psychology.

Existing works looking into this topic have found correlations between bullish sentiment on Twitter and short-term price anomalies of stocks [1], as well as message volume peaks and abnormal price action [2]. However, a critical issue commonly found in previous research is statistical inaccuracy and over-fitting, including the selection of equities for which the results are demonstrably favorable. Given that firms increasingly use data mining, machine learning and other automated statistical techniques in various stages of trading, including decision-making and execution [3], further rigorous development of this kind of large-scale social media analysis has the potential to provide or augment an additional source of investment alpha.

II. Data

We performed analysis on the component stocks of the Dow Jones Industrial Average. Data was collected for the period December 2013 to December 2016, totaling 756 trading days. Two main datasets were used:

i. StockTwits Data

StockTwits data was collected and downloaded in raw JSON format, totaling over 540,000 messages. Significant pre-processing was necessary to generate "bag-of-words" feature vectors, including:

- removing stop-words and company names
- removing posts mentioning/tagging multiple stocks (i.e. "$AAPL $FB $GOOG")
- aggregation of posts by date

Sentiment polarity was also extracted from user-generated "bullish"/"bearish" tags, for which a rolling mean of the ratio was calculated.

ii. Price Data

Daily split-adjusted stock price data was collected via the Yahoo Finance API. We focus only on the closing price data for the purposes

of this project. The main piece of data that we extract from this is the forward 3-day return, calculated as a percentage change for the future price movement three days ahead of today's trading day's price; the formula for this is:

$$return = \frac{p_{t+3} - p_t}{p_t} \tag{1}$$

where p_{t+3} is the price at time period $t + 3$ (in this case, 3 days ahead). This data was used as the prediction target to model the short-term correlation with social media activity. A shorter period was not selected, in order to smooth out the effects of daily market noise. Although we initially sought to use binary classification only, regression models tended to outperform classification with respect to accuracy and performance.

iii. Training

70% of the data was used for feature selection, of which a third was reserved exclusively for feature selection and cross validation. The remaining 30% was used as the test set; this was the StockTwits and price data from January 2016 to December 2016, a time period separated off from that of the training set, to avoid any inclusion of look-ahead bias.

iv. Features

Two feature sets were used for the different methods: a pure bag of words model, and a model using a combination of word frequency features and sentiment metrics.

Pure Bag of Words. After pre-processing, the frequencies of all 6,839 words occurring at least 25 times in the data are calculated using 1) the tf-idf metric, a statistical metric for word importance in a document that takes the product of term frequency (TF) and inverse document frequency (IDF), and 2) Laplace smoothing. These metrics are then used as features in our multinomial model.

$$\text{TF}(t) = \frac{\text{\# of times } t \text{ occurs in the document}}{\text{total \# of terms in the document}}$$

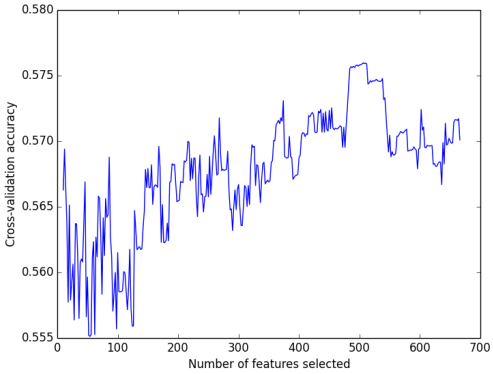

Figure 1: *Cross-validation accuracy in recursive feature elimination.*

buy	earnings	new	today
eps	good	price	long
short	pt	support	bearish

Table 1: *Sample of words selected by the mixed feature model. NB: "pt" shorthand for "price target"*

$$\text{IDF}(t) = \log \frac{\text{total \# of documents}}{\text{\# of docs with the term } t \text{ in it}}$$

Mixed Feature Model. The number of word frequency features was reduced to 1000 first by using uni-variate chi-squared score ranking, and then finally to 506 by using recursive feature elimination with 5-fold cross validation. Other features from the social media data, including message volume change, and polarity metrics were later included in this model:

- Message volume, 1-day change (percentage)
- Message volume vs. 10-day average message volume (percentage)
- Message polarity, calculated as the difference of bullish to bearish messages, divided by the total number of sentiment-tagged messages

III. Methods

We consider several methods for analyzing our data, with the primary goal of obtaining a prediction success rate above 50% (and ideally,

above 60% to be considered significant). Since this particular problem is classification-based in nature (returning a 'bullish' or 'bearish' signal for the desired time period, we will first test out the efficacy of a classification model. All models were tested using scikit-learn.

i. Naive Bayes

We use the multinomial model for this generative learning algorithm, which assumes feature independence and seeks to maximize:

$$p(y) = \prod_{i=1}^{n} p(x_i|y)$$

For the input, Naive Bayes was trained using just a pure bag of words model, and assigned the probabilities to either a positive $y = 1$ or negative $y - 0$ class, from which a long/short prediction can be generated.

ii. Support Vector Regression

We use this good "off-the-shelf" learning algorithm for regression, which, similar to how SVC finds a decision boundary that maximizes the margin, aims to minimize the ϵ-insensitive loss function created by Vapnik [4]:

$$\begin{cases} 0 & |y - f(x,\omega)| \leq \epsilon \\ |y - f(x,\omega)| - \epsilon & \text{else} \end{cases}$$

This model is trained on the mixed feature model. For our regression cases we simply map the predicted y-values beyond a reasonable threshold (0.5%) to valid positive and negative signals.

iii. K-Nearest Neighbors Regression

We tested a non-parametric method, k-nearest neighbors clustering on the same features and prediction testing schema as SVR, using the Euclidean distance metric. We hypothesized that this method could potentially outperform the parametric models, which may overfit on all of the noise in our financial data. The parameter k was optimized with 5-fold cross validation.

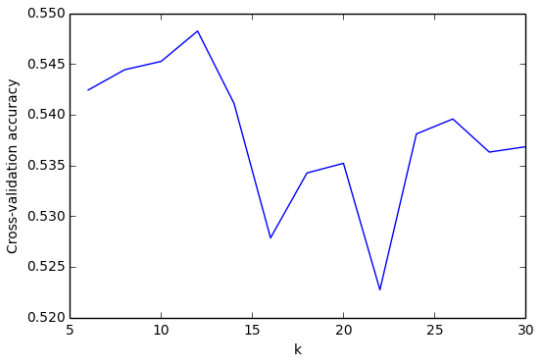

Figure 2: *Cross-validation accuracy in recursive feature elimination on k-NN regression.*

IV. Results

The test accuracy for each of the learning models is shown below:

Table 2: *Test accuracy, compared*

Ticker	Test Accuracy
Naive Bayes	0.5099
SVR	0.5682
k-NN Regression	0.5448
Average	0.5410

Table 3: *Best and worst stocks, Naive Bayes example*

Ticker	Test Accuracy	Trades
UTX	60.0%	80
JNJ	59.3%	86
INTC	57.9%	126
MRK	43.6%	78
BA	42.2%	166
MMM	37.5%	16

For practical trading purposes, however, the more important metric is profitability. The model predictions for the positive and negative classes were used to generate long/short signals on the test data, and a simulated portfolio allocated 33% equity to daily equal-weighted long/short positions each held for 3 days. The resulting profit-and-loss (PnL) curves are

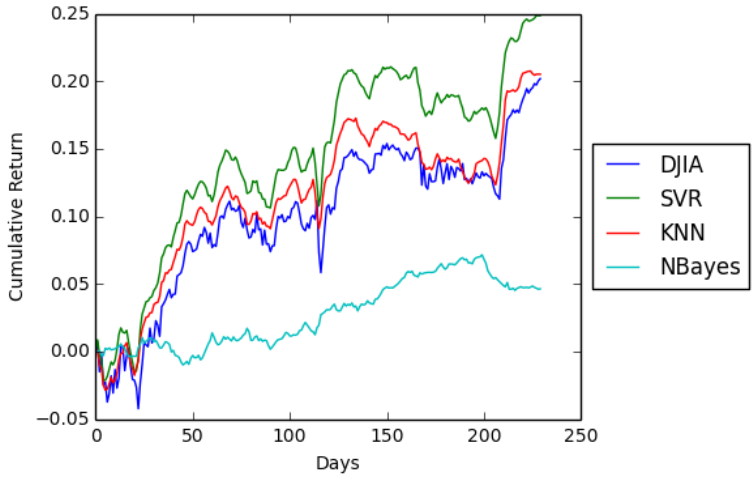

Figure 3: *Comparative PnL curve. Dow index movement in blue.*

graphed on the next page in comparison to the DJIA index performance for the same time period.

This graph shows the results on the test period, from Jan 2016 to Dec 2016. Although Naive Bayes is consistently profitable, it significantly underperforms the other methods and the broader market. SVR returns a total of 25% within this 11-month period, after simulated commission costs.

Overall, the regression models proved to be more accurate and more actionable as trading signals compared to binary classification. One plausible reason for this result is that binary classification (especially the generative model used in Naive Bayes) will attempt to fit to the noise inherent in stock market price movement, and lumps small, statistically insignificant upward movements indiscriminately with large ones. The test error rates were below what this project initially aimed to achieve; however, the signal's positive performance indicates that the selected features are in fact meaningful, and capture some insight into short-term market movements. Notably, the SVR and KNN models are still very correlated to the market, although the SVR model does deviate from the markets in certain time-frames. Despite the broad selection of stocks used in this analysis compared to previous works, the training set and test set are biased towards positive classes for this time period (the 2013-2016 period was overall a positive time for markets). They are also all large-cap, blue chip companies that largely move in tandem with the broader market as a whole. Thus, further research could also analyze a wider of basket of stocks, ideally not highly correlated to broader markets, to verify and improve on our findings. Future work on this topic could also involve testing recurrent neural networks on the data, which would be particularly suitable for time series prediction problems like this one.