Stock Market Prediction through Twitter

Luke Plewa, *Student, Cal Poly San Luis Obispo*

Abstract—Twitter is an excellent source of public sentiment. By identifying tweets based on keywords such as “rising” or “risky” coupled with company names and products, knowledge can be derived from Twitter regarding the stock market. Previously, stock market trading has been done through expert systems and machine learning. This paper discusses specific implementations, as well as a new neural network implementation based on Twitter data. These learners look at current tweets to generate knowledge about current or upcoming trends. This research also unveils existing implementations that intend to help people trade in the stock market from a knowledge usability standpoint, as well as implementations that try to trade on their own.

Keywords—Twitter, Stock Market, Boosting, Machine Learning, Knowledge Based Systems, Expert Systems, Fuzzy Logic, Trading

1 Introduction

THIS paper is intended to serve the general public by demonstrating current topics in the integration of stock market prediction, knowledge based systems, and machine learning. This paper proposes that boosting on Twitter data can be used to predict changes in the stock market.

June 3, 2014

2 Background Information

2.1 Web Crawling

Recently, traders have taken an interest in big data. [1] This interest is inspired by using web resources such as news articles and social media to predict rises and falls in stock. [3] The idea is to track public sentiment so that traders are not surprised when stocks rise and fall. There are companies use this data to give an edge to their customers by being better informed. [3]

2.2 Twitter

Twitter is an excellent source of public sentiment. [11] The 140 character window seems like a strange place to collect data, but it is readily available, up to date, and always streaming. The Twitter Developer API allows 480 individual requests within a 15 minute window for free. For the purposes of small projects, this is suitable. For larger projects, Twitter has struck deals to sell their data to companies. [5] This includes stock traders, who have taken a recent interest in the social media site’s data. [1]

2.3 Expert Systems

Expert Systems are commonly used for stock market predictions. [8] [13] Knowledge is obtained from stock market analysts to create rule based systems. These systems use their knowledge and these rules to predict the rise and fall of the stock market.

2.4 Fuzzy Logic

Fuzzy logic is based on the mapping of values to fuzzy sets. [8] Fig. 1 demonstrates an example of these fuzzy logic sets. The input value here is a person’s age. Based on their age, a person can be categorized into four fuzzy logic sets: kid, young, middle age, and old. There is an overlap between these fuzzy logic sets, because a person can belong to multiple age sets. For example, a person of age ten has a 0.5 value for being considered a kid and a 0.9 value for being considered young. These values are determined by some

- *L. Plewa is with the Department of Computer Science, Cal Poly San Luis Obispo, San Luis Obispo, CA, 93405.*
E-mail: lplewa@calpoly.edu

Manuscript received May 12, 2014.

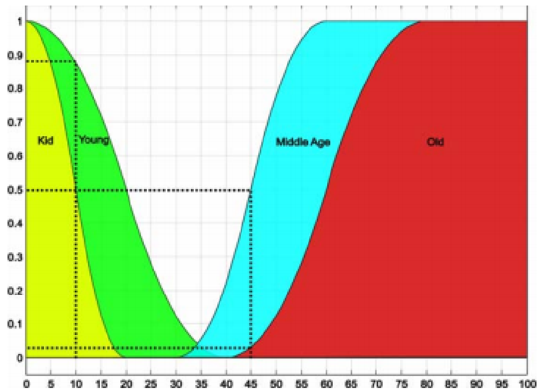

Fig. 1. Fuzzy Logic Sets Based on Age [8]

underlying algorithm represented by Fig. 1.

Fuzzy logic is a three step system. The first step is fuzzification, where the input values are mapped to truth values and given degrees of membership into fuzzy logic sets. In the previous example, a ten year old has a 0.9 value membership into the young set. The next step is rule evaluation, where the truth values are turned into fuzzy outputs. The final step is defuzzification, where the fuzzy outputs are mapped to discrete values.

2.5 Boosting

Boosting is based on the idea that many weak learners can be combined to produce a strong learner. [6] These weak learners differentiate by changing weights placed upon input variables and the contribution of the weak learner to the strong learner. In doing so, the strong learner can apply to varied, non-linear data and improve the accuracy of the weak learner alone. The best part about applying boosting to a learning algorithm is that it does not care what the learning algorithm is. The weak learner could be a decision tree, neural network, or any kind of learner. Boosting is also not prone to overfitting of data sets, due to the weighted nature of the combined algorithm. [10]

3 Expert Systems Related Work

3.1 Fuzzy Logic Expert System

Merloti demonstrates an approach that uses fuzzy logic to build an expert system. [8] Fuzzy logic focuses on reasoning that is approximate. In the domain of the stock market, an approximate approach seems appropriate. A small amount of uncertainty will not affect the decision to invest or not to invest in a company. The amount a stock rises or falls is as important as the binary prediction that the value of stock will either rise or fall. This helps traders know whether they should buy or sell, also a binary trait, but in terms of how much stock they should buy or sell. If the expert system predicts a large rise in stock value, then the trader will know to buy more stock before that happens.

Fuzzy logic sets fit in nicely with this approach of determining whether to buy or sell an amount of stock. This system takes in two values: the value of the stock and a stock indicator called MAD (moving average divergence). [8] Stock prices have fuzzy logic sets of low, medium, and high. MAD values have crisp sets of negative, zero, and positive. The fuzzy expert system uses these inputs to return a fuzzy logic set of how to trade. The fuzzy logic sets are buy many, buy few, do not trade, sell many, and sell few.

EXPERIMENT 1

	Low	Medium	High
P	BM	BF	SF
Z	BF	DT	SF
N	BF	SF	SM

EXPERIMENT 2

	Low	Medium	High
P	BM	DT	SM
Z	BM	DT	SM
N	BM	DT	SM

EXPERIMENT 3

	Low	Medium	High
P	BM	BF	DT
Z	BF	DT	SF
N	DT	SF	SM

Fig. 2. Fuzzy Logic Tables [8]

The fuzzy logic expert system starts with the fuzzification of the input values. Because stock values and the MAD indicator can have a large range of values, they are first normalized. This normalized value is then used in rule evaluation as input for a set of rules that determine fuzzy set values. Since there are three sets for two input values, there are nine total rules based on the inclusion of exclusion of these sets. Fig. 2 shows three different examples of these nine rules. These rules create values for each of the five sets. The defuzzification algorithm selects the appropriate trading decision based on these set values.

Fig. 3 displays the results of these three experiments. Of the three rule sets, the second experiment shows the best results. This is like due to the buy many sets being applied when the stock is at a low value, and the sell many sets being applied when the stock is at a high value. That is a standard approach when trading and has proven itself in this expert system. [8]

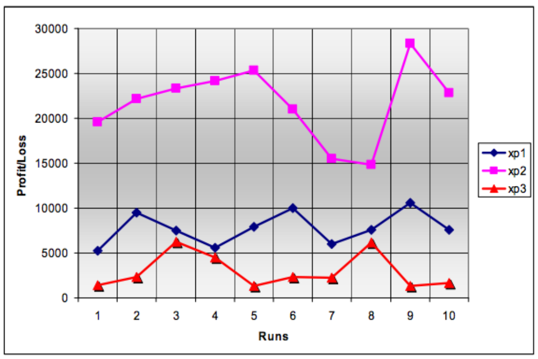

Fig. 3. Fuzzy Logic Results [8]

3.2 Rule Base Expert System

Yamaguchi demonstrates an approach that focuses on rule base refinement. [13] Like all expert systems, Yamaguchi's approach uses a rule base that contributes to an inference engine. This rule base is broken down into two sets of rules: object-level rules and meta-level rules. The object-level rules apply to the stock market data and stock market prediction. The object-level rules within this expert system are evaluated by meta-level rules. If the object-level rules predict correctly, the meta-level rules will keep them in the system. If they predict poorly, the meta-level rules will created a refined rule candidate to replace it. This is used to ensure that the rule base is accurate.

Fig. 4 presents the rule refinement subsystem that exists in Yamaguchi's solution. ES stands for Expert System. This is the whole system. IE stands for Inference Engine, the system that uses the rule base to make predictions. WM stands for Working Memory, the area that holds the stock market data. RB stands for Rule Base,

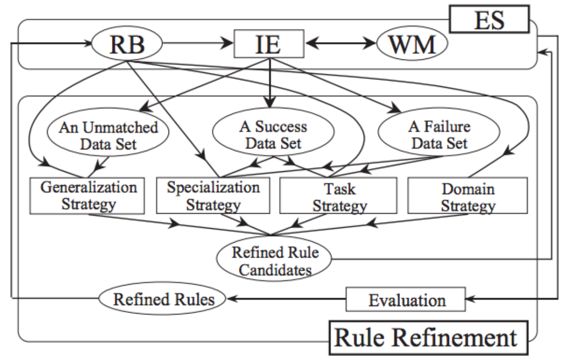

Fig. 4. Rule Refinement Subsystem [13]

the set of object-level and meta-level rules that contribute to the prediction and perfection of the system. This figure shows how the rule base uses meta-level rules to adjust its object-level rules. Failed, unmatched, and successful data sets are all grouped to create a refined set of rules, where are re-evaluated before going back into the Rule Base. This is done through four strategies (the meta-level rules): generalization strategy, specialization strategy, task strategy, and domain strategy. [13]

4 Boosting Related Work

The boosting approach to stock market prediction is featured in Creamer and Freund's research article. [4] They combine Logitboost with Alternating Decision Trees to take a variety of inputs and produce a market value score. Logitboost is largely based on Freund's Adaboost algorithm with some modifications. [6]

4.1 Adaboost

Adaboost, short for Adaptive Boost, is a boosting algorithm proporsed by Freund and Schapire. [6] This boosting approach applies the same weak learner to the same data set multiple times. Between each application, a change in weights to the variables is applied or a change in the data set. For example, certains items in the data set may be included multiple times or not at all in an attempt to vary the data set.

$$
\begin{aligned}
&F_0(x) \equiv 0\\
&\text{for } t = 1 \dots T\\
&\qquad w_i^t = e^{-y_i F_{t-1}(x_i)}\\
&\qquad \text{Get } h_t \text{ from } \textit{weak learner}\\
&\qquad \alpha_t = \frac{1}{2} \ln \left(\frac{\sum_{i:h_t(x_i)=1, y_i=1} w_i^t}{\sum_{i:h_t(x_i)=1, y_i=-1} w_i^t} \right)\\
&\qquad F_{t+1} = F_t + \alpha_t h_t
\end{aligned}
$$

Fig. 5. Adaboost Algorithm [4]

In Fig. 5, the Adaboost algorithm is shown. 't' represents the index of iterations the weak learner is applied. 'w' represents the weight applied. 'h' represents the hypothesis of the weak learner, its stock market prediction. 'α' represents the learning rate, or how much the weak learner contributes to the strong learner each iteration. 'F' represents the value of the strong learner.

4.2 Logitboost

Logitboost is an adaptation of Adaboost. It follows the same principles, but the algorithm for deciding the weights of the input variables is different. In Fig. 6, this is shown by the equation that produces value 'w'.

$$
\begin{aligned}
&F_0(x) \equiv 0\\
&\text{for } t = 1 \dots T\\
&\qquad w_i^t = \frac{1}{1+e^{-y_i F_{t-1}(x_i)}}\\
&\qquad \text{Get } h_t \text{ from } \textit{weak learner}\\
&\qquad \alpha_t = \frac{1}{2} \ln \left(\frac{\sum_{i:h_t(x_i)=1, y_i=1} w_i^t}{\sum_{i:h_t(x_i)=1, y_i=-1} w_i^t} \right)\\
&\qquad F_{t+1} = F_t + \alpha_t h_t
\end{aligned}
$$

Fig. 6. Logitboost Algorithm [4]

4.3 Alternating Decision Tree

Creamer uses an Alternating Decision Tree (ADT) combined with the Logitboost algorithm to accurately predict the stock market. [4] The ADT is the weak learner that Logitboost applies to. There are two types of nodes in an ADT, represented in Fig. 7. The red rectangles represent the splitter nodes. These are the nodes that make a binary decision based on the input value, and control the flow of the tree by choosing left or right. Splitter nodes operate on and produce prediction nodes, represented by the blue ovals. Prediction nodes contain the values that are input and produced by the rules of the splitter nodes. The leaf prediction nodes are the final value of the ADT. By summing all of the leaf nodes, the ADT produces a hypothesis (the prediction of the stock market trend).

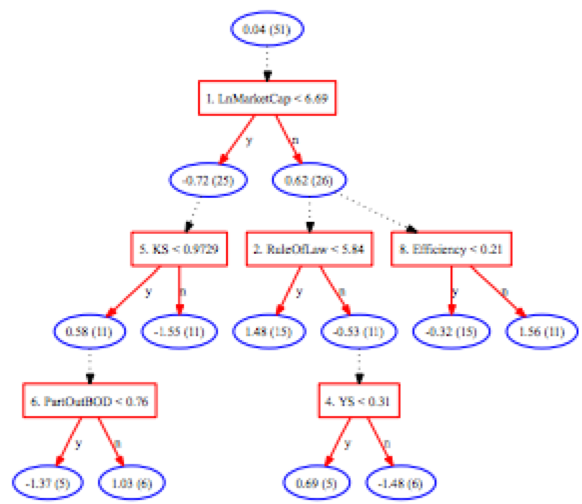

Fig. 7. Alternating Decision Tree [4]

4.4 Boosting Conclusion

By combining ADTs and Logitboost, Creamer demonstrates an accurate way to predict the stock market. [4] Logitboost contributes to both the decision rules within the tree and the combination of the rules through a weighted vote. Boosting has proven to improve the accuracy and avoid overfitting for most learners, and decision trees are often the beneficiary. [?]

5 Twitter Mood Related Work

In 2011, Dr. Johan Bollen created a solution to predicting the stock market by using Twitter. [2] This approach collected the overall mood of Twitter in a snapshot, and created a trend of moods over time. This mood trend is compared to the trends of the stock market (specifically the Dow Jones Industrial Average). These moods were collected with two applications: OpinionFinder and Google-Profile of Mood States. While six mood indicators were tracked (Calm, Alert, Sure, Vital, Kind, and Happy), it was found that the level of Calm in Twitter correlated best with stock market trends. The results showed an 80 percent correlation between calmness and the stock market.

Fig. 8 displays the change in Twitter mood over time. The selected time period is interesting, because it contains both the 2008 presidential election and Thanksgiving. The election creates a drastic change in mood before and after the results. Thanksgiving provides a small bump to most moods, but a drastic increase in happiness. Happiness was hypothesised to be the best indicator, but proved to be less useful compared to calmness. [2]

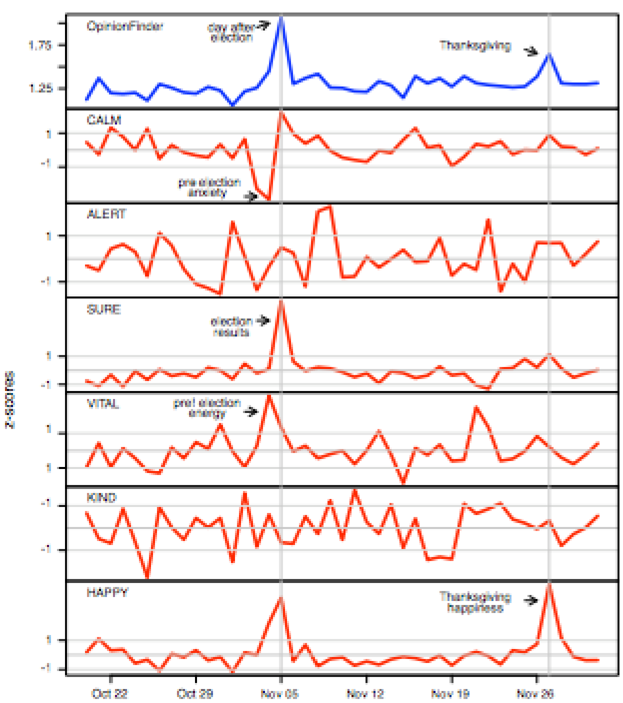

Fig. 8. Twitter mood trends [2]

Fig. 9 compares Twitter calmness and the Dow Jones Industrial Average. The combined chart is the best demonstration of the 80 percent correlation between Twitter mood and the stock market trends. Calmness trends three days in advance of the stock market, so the current level of calm actually applies to the stock market values three days afterwards. [2]

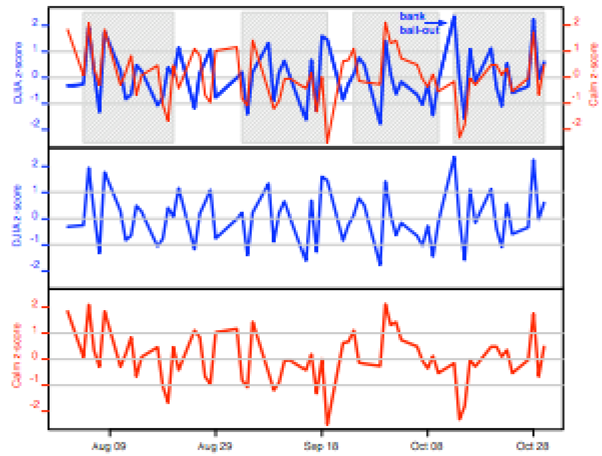

Fig. 9. Twitter mood vs. Dow Jones Index [2]

6 Stock Market Prediction Through Twitter

The previously cited research based their input values on common stock market values such as market cap, stock market average price, call rates, exchange rates, and money supply. [4] [13] Twitter Mood broke this trend by basing all of its data on Twitter. This paper proposes the idea that machine learning can be applied to variables found within Twitter's 140 character tweets. Possible variables include overall sentiment (represented by a score, generated by the occurence of positive or negative tweets) and frequency of tweets (based on the idea that any publicity is good publicity). The following sections will cover the stock market prediction through Twitter solution that is separate from Bollen's Twitter Mood predictor.

7 Components

The stock market prediction through Twitter solution is comprised of three components. There is a Twitter Engine, Neural Network, and Booster. The Twitter Engine feeds the Neural Network features, and the Booster is comprised of many Neural Networks. There is also a market score component, but its features are standard and not worth going into detail for. It is used for training data.

7.1 Twitter Engine

The first task this solution needs to complete is to collect as many tweets as possible from Twitter. The collected tweets are all related to a given company, using the search features of the Twitter API. These tweets are then parsed for keywords that would help indicate the success of the given company. In comparison to Twitter Mood, these parsers are very simple. [2]

7.2 Neural Network

The neural network used in this solution uses three layers of nodes (neurons). The first layer is the input layer, and its number of nodes is equivalent to the features found within the Twitter Engine. The second layer is the hidden layer, and the number of nodes within it are flexible. The final layer is the output layer, and it has a single node. A single node is necessary as this network acts as a binary classifier. It only cares about whether the stock will rise or fall (0 or 1). An output of 0.5 shows that the stock will not change in price.

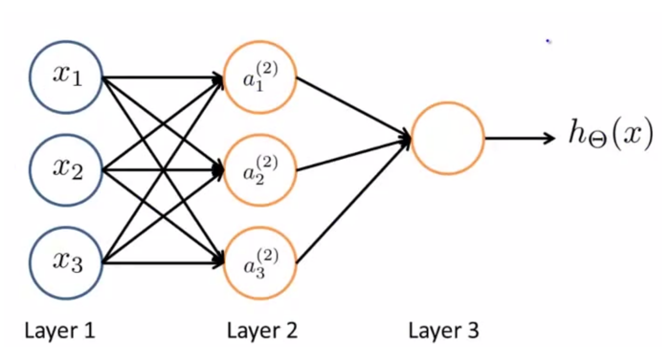

Fig. 10. Neural Network [9]

Fig. 10 represents a neural network. In this figure, layer 1 is the input layer, layer 2 is the hidden layer, and layer 3 is the output layer.

The input layer has 3 neurons in this example, and these represent the input features for the neural network. In this solution, there is an input layer node for each tweet term and a node for the total number of tweets. The 'x' value represents the amount of times that feature was found. In this solution, most of the input layers have a value of zero when testing, because no tweets were found to contain that feature.

The hidden layer is flexible, and adding more nodes to this layer increases the accuracy and complexity of the neural network. It has been proven that only one hidden layer is necessary if there are a sufficient number of hidden layer nodes. [10] Oftentimes more than one hidden layer is used, but this solution is kept to one in order to reduce complexity. 'α' represents the activation value for the hidden layer node. It is computed as the sum of all the input nodes, which have different weights when added to that hidden layer node. [10] These weights are represented by the black lines that connect the input and the hidden layer. This means that the number of weights is the product of the number of input and hidden layer nodes.

The output layer only has one node in this solution, because it is focused on a binary classification. The output node has a similar activation function. It takes sum of all the values stored by all the hidden layer node activations multiplied by their respective weights corresponding to the output layer node. This value is on a 0 to 1 scale, where values at 0 predicting a stock declining in value and values at 1 predicting a stock rising in value.

7.3 Neural Network Training

The process of producing an output value, the hypothesis, of a neural network is called forward propagation. [9] This is the algorithm that goes node by node and produces values through the activations functions. This step is dependant on weights which must be trained to correct values. The process of training a neural network and updating its weights is called back propagation. [9] Starting from the output layer, the algorithm goes node by node and updates the weights to better fit the expected value of the neural network (based on stock market data). If the neural network produces an output value close to the expected value, then the weights will not change much.

7.4 Booster

Boosting is used to improve the accuracy of the neural network. This is proven by the research done by Schwenk and Bengio. [10] When obtaining data from Twitter, different search terms provide a different set of tweets. This fits the boosting mindset well, as it hopes to train the same classifier on different sets of data.

For this solution, different classifiers varied in both the stocks they trained on as well as the tweets. The weights that these classifiers contributed to the strong learner increased or decreased based on a cost function. If the weak learners predicted correctly, their weighted contribution to the strong learner increased. The opposite is true if the weak learners predicted incorrectly.

8 Experimental Results

8.1 Overall Results

The binary classifier determines whether a given stock rises or falls in price from 0 to 1. By this schema, a value of 0.5 would indicate a negligible change in price. This sliding scale is important to consider when regarding the results of this experiment.

Fig. 11 demonstrates some of the successful classifications made by the Twitter solution. These classifications were made with 13 weak learners (neural network binary classifiers). The stock value changes were recorded from May 27, 2014 to June 3, 2014. The Twitter data used was collected during this time frame. There were just under 2,000 tweets collected for the training data, and over 100 tweets collected for the test data. Twitter search terms included the company names as well as the names of their prominent products (i.e. Apple's iPad or Hasbro's Transformers).

8.2 Results Trends

In Fig. 12 there are a couple outliers that do not show excellent prediction. For instance, Broadcom has the highest stock value increase, yet it does not have the highest classification

Company	Classification	Stock Value Change
Broadcom	0.75	4.13
Apple	0.96	0.96
Google	0.58	0.81
Adobe	0.58	0
Time Warner Cable	0.6	-0.05
E*Trade	0.25	-0.13
Hasbro	0.63	-0.14
Tesla	0.58	-0.17
Facebook	0.19	-0.46
Gamestop	0.24	-1.29
Wynn Resorts	0.24	-5.1

Fig. 11. Overall Results

(0.75 points compared to Apple's 0.96 points). Another instance is Google's low 0.58 classification despite gaining 0.81 points. This means that the solution is by no means foolproof. Regardless, Fig. 12 does show a trend in the correct direction which shows promise for this solution.

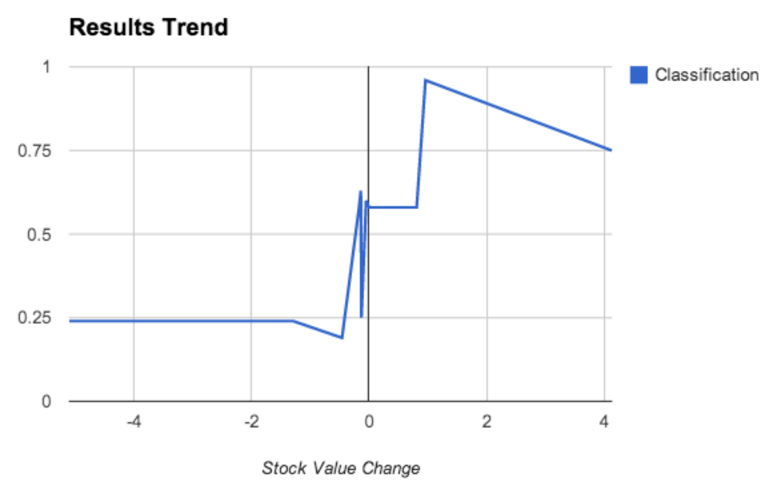

Fig. 12. Results Trend

8.3 Effect of Boosting

The effects of boosting can be seen in Fig. 13. The three stocks shown are Apple, Tesla, and Adobe. These three stocks were chosen because they have unique stock price changes. Apple rises in stock by 0.96 points, Adobe does not change in value, and Tesla goes down by 0.17 points. In all three cases, the addition of classifiers (up to 13) increases the accuracy of the classifiers. The classifiers used a randomized set of the same training data used for the results in Fig. 11.

It is interesting to note the change in classification for Apple's stock. It is expected for the trend to go upwards the whole way, but there is a dip when 4 and 8 classifiers are used. This may be due to the randomness of the selected features in such a small selection. Regardless, boosting is shown to work as it eventually predicts a higher value.

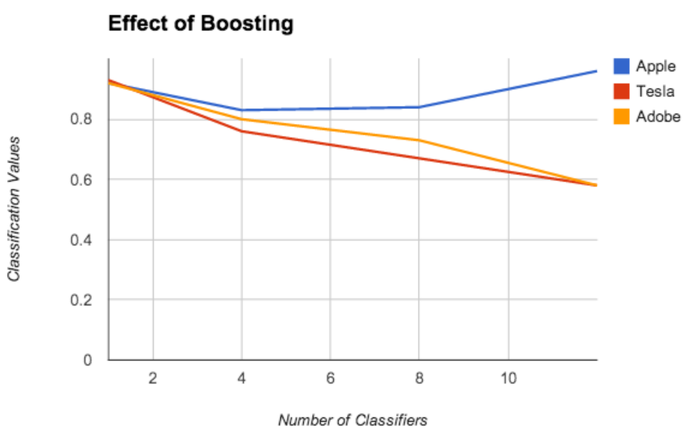

Fig. 13. Effects of Boosting

9 Future Work

9.1 Natural Language Processing

Twitter Mood uses advanced Natural Language Processing (NLP) algorithms to determine the mood of tweets [2]. This solution draws its features from the contents of tweets, but is unable to determine moods. A mood score could be added as a feature to the neural network presented in this solution. This added feature will hopefully improve the accuracy of the solution.

9.2 Higher Granularity

Due to the limitations of the features (see NLP), high granularity of prediction seemed impossible for this project. The scope of this solution was limited to a binary classifier (increasing or decreasing value of stock). With a better feature set, it would be reasonable to predict how much the stock prices would change. One solution would be to have a normalized function that would, in reverse, change the 0-1 scale into a real stock market price value. Another solution would be to have a couple sets in the classifier so that a given set of features can be classified into either high increase, low increase, no change, low decrease, high decrease sets. This would increase the size of the output layer to multiple nodes, but the hypothesis would be the best classification of those output nodes. A binary classification would provide enough information to trade upon, but producing an exact value would be more exciting and challenging.

10 Comparison of Solutions

Twitter Mood was able to predict with 80 percent accuracy compared to the Dow Jones Industrial Average. In comparison, the Twitter solution is only able to predict accurately for stocks that are heavily communicated on Twitter. The amount of publicly traded companies that are represented on Twitter represent less than this, and so it would seem Twitter Mood would be a better predictor. This is a severe limitation of the Twitter solution, and would greatly benefit from integrating Twitter Mood. This would mean that the mood could be used when no meaningful tweets were available.

The expert systems use reliable data to create their stock market predictions. This seems like a more practical approach compared to the Twitter solution. Twitter is prone to false data. [1] [2] This would require a safety check to limit the trades made, because there is nothing stopping a bot from create many accounts and tweeting about fake stock information. It would seem reasonable to include both actual stock market data as well as Twitter data.

11 Conclusion

Stock market prediction has proven to be a pursuable problem to tackle. Many approaches have shown a high degree of success. [2] [13] [8] The Twitter solution has shown success with predicting the stock market in using While tweets may not be an extensive enough data set, the combination of tweet data and Twitter mood may prove to be successful for high granularity in predicting the stock market.

ACKNOWLEDGMENTS

The author would like to thank Dr. Franz J. Kurfess for teaching Knowledge Usability (CSC 581) at Cal Poly San Luis Obispo. The progress made within this paper would not be possible without his infectious passion towards Artificial Intelligence. Also thanks to the Computer Science Department at Cal Poly San Luis Obispo, whose facilities and administration made this possible.

REFERENCES

[1] S. Alcorn, *Twitter Can Predict the Stock Market*, Fast Company, 2013.

[2] J. Bollen, H. Mao, X. Zeng, *Twitter mood predicts the stock market*, Indiana State University, 2011.

[3] G. Bowley, *Computers that Trade on the News*, The New York Times, 2010.

[4] G. Creamer and Y. Freund, *Automated Trading with Boosting and Expert Weighting*, Rutgers Univserity, 2004.

[5] E. Downskin, *Twitters Data Business Proves Lucrative*, Wall Street Journal, 2013.

[6] Y. Freund and R. Schapire, *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*, AT&T Labs, 1996.

[7] J. Lin and D. Ryaboy, *The Twitter Experience*, Twitter, Inc, 2012.

[8] P. Merloti, *A Fuzzy Expert System as a Stock Trading Advisor*, San Diego State University, 2005.

[9] A. Ng, *Machine Learning, Coursera*, Stanford, 2014.

[10] H. Schwenk and Y. Bengio, *Boosting Neural Networks*, University of Montreal, 2000.

[11] V. Shah, *Machine Learning Techniques for Stock Prediction*, NYU, 2007.

[12] S. Shen and H. Jiang, *Stock Market Forecasting Using Machine Learning Algorithms*, Stanford University, 2012.

[13] T. Yamaguchi and Y. Tachibana, *A Rule Base Renement System in a Stock Market Technical Analysis*, Shizuoka University, 1993.

Luke Plewa Luke Plewa is a student at Cal Poly pursuing a joint BSMS degree for Computer Science. His other works include progress made in computer architecture, OpenGL video games, Android applications, and machine learning.