# Stock Market Trends Prediction after Earning Release

Ran An - 06117810 (anran), Chen Qian - 06116065(cqian23), Wenjie Zheng- 05999176 (zhwj814)

*Abstract*— **This project involves discovering how the company's stock performs in after-hours on the release day of its quarterly financial report and the day after. Project aims to use a variety of machine learning models to make predictions regarding the stock price movements based on the factors that reflect and affect investor reactions, namely, media coverage of financial news, companys' quarterly financial report and the difference between market performance and consensus expectation. The goal is to help finance companies to build their trading strategies, and furthermore, identify the key factors for company valuation.**

## I. INTRODUCTION

It is said that accurately predicting a chaotic system like stock market is basically impossible, however, it is still a significant progress if we can lower the risk of loss and raise the probability of success with machine learning algorithm of proper model and feature selection. According to the efficient-market hypothesis, newly revealed information about a company's prospects is almost immediately reflected in the current stock price, along with all available information and rational expectations.Therefore, we can make prediction on stock market trends by selecting corresponding input features.

Federal regulation of public traded companies requires them to release quarterly and annual earnings statements to present their financial status to investors, reporting financial statistics such as net income, earnings per share, total asset, total equity and total revenue. By analyzing earnings report and compare against company's previous financial performances, it is possible to evaluate the financial health of the company and determine the corresponding investment strategies. Therefore, in short term, stock prices can be significantly affected by company?s quarterly performance as well as market surprise due to consensus forecast. In addition, general public shareholders' trading strategies can also be influenced by analysis and articles from mainstream media. As non-professionals tend to take advice from analysts instead of reading from opaque reports.

In sum, the input to our algorithm is divided into two sets, one is a length-five vector which contains digital data from earning report and market surprise (net income, earnings per share, total asset, total equity, total revenue, EPS surprise), and another is sentiment analysis of relevant articles which is represented by the percentage of word categories. We then processed two sets of input features with SVM, Naive Bayes, Locally Weighted Linear Regression and Boosting algorithms to output the prediction of stock markets trends with two labels: the price change between next-day opening price and release-day close price, and the price change between next-day closing price and release-day close price. More specifically, the output label is marked as 1 or -1, indicating an increase or a decrease.

Our project tends to explore the impact of financial report factors, such as net income, market surprise of earning per share (EPS), and the sentiment analysis of relevant news articles from mainstream professional media on short-term stock market trends during the after-market time period and the next trading date.

## II. RELATED WORK

Stock price prediction has long been a popular topic. In the past 10 years, many works have been done on application of machine learning techniques for stock price prediction. SVM has been proved a powerful algorithm by researches, including Subhabrata Choudhury's [1] and Jigar Patel's work [2]. In addition to SVM, genetic algorithms [3] [4] , Decision Tree [5], [6], and some other regression models [7] [8] have been widely utilized in this topic. Moreover, in recent years, many researches have been done on applying Neural Networks to predict the stock price, including [9], [10]. Some of these studies tend to be quite successful, such as [10] used Maple as measurement, and achieved 98% accuracy.

The idea of using text information to predict stock price has also been widely studied. Hagenau proposed a model based on financial news using context-capturing features in 2013 [11], and Shynkevich used different categories of articles to do predictions [12]. Other works include the AZFintext system proposed by Schumaker [13], and etc. In these works, the most widely studied issue is how to extract valuable information related to stock price prediction. Briefly, bag of words [13] and noun phrases [11] are most commonly applied technique, but generally cannot produce satisfactory result.

## III. DATASET AND FEATURES

In terms of earnings report, we selected four features with the advice of professionals, which are total asset, total equity, total revenue and net income. The reason is that these values reflects the In addition, through the talk with portfolio manager, we were told that the market surprise due to difference between market real performance and consensus expectation actually affect the stock market trends to a large extent. Therefore, we looked at https://www.last10k.com to obtain quarterly earnings report for the companies.

As aforementioned introduction shows, we exploited two types of data targeted on 12 selected companies in our model, one from financial earnings reports, and the other from analytic articles. It is unfair to assume that the stock market is invariant during a long period, but instead it is

acceptable to assume the stock market is somewhat stable within several years. Guided by this idea, we only focused on data from 2011. In total, we collected 237 quarterly financial report for each company, roughly 20 for each company, and 305 articles, roughly 25 for each company.

For data obtained from financial report, we computed the following: $surprise = \frac{earn - expectedearn}{expectedearn}$ , concatenating the percentage of change of total asset, total equity, total revenue and net income compared to the previous quarter. Thus, it has feature $f \in R^5$.

For data obtained from analytic articles, we first computed the sentiment of each article and represent the sentiment in a $R^5$ vector as shown in Section III. The five dimensions of the vector separately represent the strength of (very negative, negative, neutral, positive, very positive), with the summation of 5 dimensions equal to 1. Notice the fact that the summation is 1, we could reduce the feature dimension by 1. In practice, we simply get rid of the value representing the degree of neutral since intuitively, the degree of neutral does not impact the movement of the stock price. To decrease the impact of scaling, we modify the new features to render the summation of 4 dimensions still be 1 by performing:

$$f(i) = \frac{f(i)}{\sum_{k=1}^{4} f(k)} \tag{1}$$

For both types of data, we perform supervised learning with two different supervisory signals: the after-market movement of stock price and the next-day movement of stock price. The after-market movement of stock price is defined as the percentage of increase of the stock price when market opens the next day compared to the stock price when the market closes at the financial report release day. On the other hand, the next-day movement is defined as the percentage of increase of the stock price when market closes compared to market opens the next day to the financial report release day. Hence, we have continuous labels for the training set. Then, for classification model, such as SVM and boosting, we need to discretize the continuous label into 0 and 1 following the simple policy.

## IV. Data Pre-processing

### A. Financial data Relativization

As we target different technology companies in different business scale, the absolute financial value from reports of each company can not be interpreted by other companies therefore it is meaningless to combine data from each company together. In order to solve the data merging issue, we discussed with colleagues in financial department and investment companies and found that we can solve this and make our data more representative by looking at the growth rate of each financial features compared to the same value from last quarter. We decide to relativizate all financial data by calculating the growth rate of each financial feature we collect from reports, including the EPS vs market expectation. Thus all values are in the range from -100% to 100% and we are able to combine all data to form a large training set.

### B. Sentiment Analysis Result

We interacted with Stanford NLP toolkit to perform sentiment analysis of financial articles. Comparing the analysis results with our own reading we found that Negative attitude dominants the distributions (most of them are over 90%) while we think the articles are more towards neutral or positive. We researched more and found that this toolkit evaluated article sentiment in the unit of sentences and the algorithm is too general to apply to financial articles. So we modified our script to analyze sentiment in the unit of word and the result tended to be much better than before and the sentiment feature distribution tended to follow Gaussian distribution.

### C. Normalization

The sentiment analysis feature include five categories: Very negative, Negative, Neutral, Positive and Very positive. We record the total word count of an article and count how many words are in each category to calculate the weight of each category for an article. This approach allows us to obtain the normalized sentiment features vector to feed into our models. As all distribution sums to 1 so therefore we can reduce the amount of category from 5 to 4 (Any one of them can be represented by other four) therefore 4 features will be retained (Netural is eliminated) and they are re-normalized before feeding into models.

## V. Methods and Models

We have prepared two sets of features: One uses relativized financial data and another one comes from our NLP process. And two label sets: one is the stock market trend during aftermarket period and another one is the stock market trend during next trading date. The main goal of our project aims to explore the performance of each popular machine learning algorithm we have learned on different labels. The algorithms/models we choose the evaluate are: Locally Weighted Linear Regression (LWLR), Naive Bayes (NB) and Support Vector Machine (SVM).

### A. Locally Weighted Linear Regression

Locally Weighted Linear Regression(LWLR) model is derived from linear regression model, introducing weights among training data and draw a non-linear line as the decision boundary. The parameter $\tau$ is used to control the bandwidth of the learning algorithm. Compare the origin linear regression, LWLR aims to Fit $\theta$ to minimize:

$$\sum_i w_i (y_i - \theta^T x_i)^2 \tag{2}$$

$w_i$ is a non-negative weight. Intuitively, if $w_i$ is large, then $(y_i - \theta^T x_i)^2$ should be small. On the other hand, if $w_i$ is small, even if $(y_i - \theta^T x_i)^2$ is large, the item $w_i(y_i - \theta^T x_i)^2$ is small. The normal equation for LWLR is described as

$$\theta = (X^T W X)^{-1}(X^T W Y) \tag{3}$$

Where $X \in R^{m*n}$ is the sample matrix, and $W \in R^{m*n}$ is the weight matrix. In our model, Gaussian smoother is used, i.e.,

$$w_i = e^{\frac{-(x_i - x)^2}{2\Gamma^2}} \tag{4}$$

The parameter $\Gamma$ controls how quickly the weight of a training example falls off with distance of its $x_i$ from the query point $x$.

### B. Gaussian Naive Bayes

Naive Bayesian model is a generative model based on Bayesian equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \tag{5}$$

which is intent to maximize the posterior likelihood.

In our studied situation, where features take continuous value, we resort to Gaussian Naive Bayesian method instead of the popular multinomial Naive Bayesian Model. The Gaussian Naive Bayesian model is based on Eq. 6, which is the expression of posterior probability:

$$P(x = v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp \frac{-(v - \mu_c)^2}{2\sigma_c^2}. \tag{6}$$

The target is to find the class (or label) that maximizes the posterior probability:

$$y = argmax_k(p(C_k) \prod_{i=1}^{n} p(x_i|C_k)) \tag{7}$$

In order to fit the model, we first pre-process the data by segmenting them into different classes and for each class we compute its mean value and variance. Let $\mu_c$ be the mean and $\sigma_c$ be the variance of the values x associated with class c. Let the collected value be $v$ then the probability distribution of v given class c, $P(x = v|c)$ can be represented as a Gaussian distribution parameterized by $\mu_c$ and $\sigma_c$ and are estimated using maximum likelihood.

The open source scikit python library allows us to easily implement this model, and so do the following two models.

### C. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning model with associated learning algorithms that analyze data used for classification and regression analysis. When apply SVM to our application, take the sentiment feature as an example, each training feature set can be viewed as a p-dimensional vector (in our case p = 5), and we want to know whether we can separate such points with a (p - 1)-dimensional hyper-plane. SVM is designed to find the hyper-plane that separate data with maximum margin between two classes. Since the data does not follow a linear distribution, kernel trick is exploited to map the original feature space to a higher dimensional space in order to find the separating plane. RBF (Gaussian) kernel was selected, whose expression is shown in the below equation:

$$K(x, x') = \exp(\frac{-||x - x'||^2}{2\Gamma^2}) \tag{8}$$

$\Gamma$ represents the bandwidth. The target of SVM model is to minimize:

$$\varepsilon(f) = \frac{1}{n} \sum_{i=1}^{n} [l(y_k, f(x_k)] \tag{9}$$

where $l(y, z) = \max(0, 1 - yz)$ is called *hinge loss*, and the decision function $f^*(x)$ is:

$$f^*(x) = 1, f(x) \geq 0.5 \tag{10}$$

$$f^*(x) = 0, otherwise \tag{11}$$

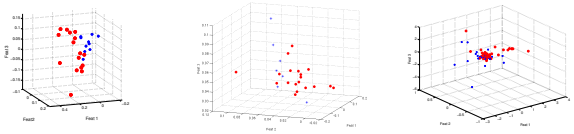
The SVM model with appropriate Kernel function will converge to the simplest function that correctly classifies the data with a high-dimensional hyper-plane. We also apply L2-regularized L1-loss in our SVM regression. Geometrically speak, the SVM minimize the empirical risk by finding the maximum margin hyper-plane in the mapped space from kernel trick. It turns out that SVM performs quite well in our application for both feature sets and label sets. We also perform 5-fold cross-validation to find the proper bandwidth parameter $\Gamma$ that minimize the test error for model accuracy evaluation.

## VI. Results and Discussion

### A. Model evaluation and data visualization

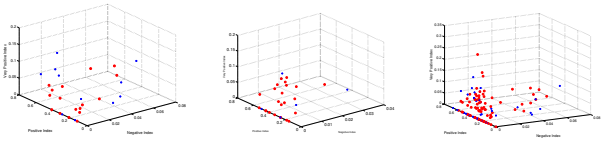
As stated in Section IV, we built regression models (LWLR) and classification models (Boosting, Gaussian Naive Bayesian, SVM) on our sample set. Accuracy is used to evaluate our models. Different from traditional works, which generally have access to a great amount of relevant data, dataset is relatively small in our targeted situation because financial report is released quarterly. For companies like Twitter or Facebook, even if we accumulate all of financial report they have ever made public, we could only get about 20 samples in total, which is far less than traditional works. From the view of machine learning, a small training set usually leads to high variance, i.e., the generalization from training set to testing set is not reliable even if the accuracy is not that bad. Thus, we need to come up with a different way to evaluate our model from directly evaluating the prediction accuracy on each company.

One simple idea is to ignore the difference among companies. Specifically, we train our model by data from all companies, and apply the fitted model to predict the price movement of each company. However, this method is only correct when the relationship between stock price movement and our selected features is almost the same for all targeted companies, which is not necessarily the case. To check the assumption, we visualize the data. For both financial report dataset and analytic article dataset, we arbitrarily selected out 3 features. For financial report dataset, the change of total asset, total equity and total avenue, i.e., the 2nd, 3rd and 4th dimension of the feature vector are selected. For analytic articles dataset, the degree of very negative, negative and positive, i.e., the 1st, 2nd and 4th dimension of the feature vector are selected. The visualized data is shown in Fig. 1 and Fig. 2.



(a) Apple (b) Google (c) Mixed

Fig. 1. Financial report data visualization



(a) Apple (b) Google (c) Mixed

Fig. 2. Analytical article data visualization

Fig. 1 shows the visualization result of financial report dataset and Fig. **??** is for analytical article dataset. In every subfigure, sample marked by red represents the positive samples, the price of which goes upwards, while samples marked by blue represent negative samples, the price of which goes downwards. For the first two figures in the first row, which are separately the data of Apple and Google, the positive samples are distinguishable from the negative samples, while there exist no apparent separating plane in the third figure , where we mix data from different companies. Thus, we claim that when we evaluate the performance of models on financial report dataset, we cannot ignore the difference among companies and must work on each company individually. Differently, in Fig. 2, no matter we focus on either individual company (first two figure ) or several companies as a whole (the 3rd figure), the data distribution is similar. Thus, we claim that when we evaluate the performance of models on analytical articles dataset, we can ignore the difference among companies.

To evaluate the performance of our models on financial report dataset, we train our model on each company, and then average the accuracy. Specifically, we use the below value to evaluate the model:

$$average \quad accuracy = \frac{\sum_{i=1}^{12} n_i * accuracy(i)}{\sum_{i=1}^{12} n_i}, \qquad (12)$$

where $n_i$ is the size of testing set of the $i$th company, $accuracy(i)$ represents the accuracy on the $i$th company.

The choice of training and testing set follows the classic idea: we randomly pick 20% of samples as the testing set, and the remaining as the training set.

### B. Experimental results

The experimental result is shown in table I.

TABLE I

EXPERIMENTAL RESULT

| predict by report | SVM | Naive Bayesian | LWLR | Boosting |
|---|---|---|---|---|
| after-market | 69.80% | 66% | 64.51% | 66% |
| next-day | 64.20% | 60.30% | 68.12% | 58.30% |
| predict by article | | | | |
| after-market | 57.64% | 52.17% | 58.96% | 53.96% |
| next-day | 54.91% | 50.94% | 64.38% | 52.17% |

On the financial report dataset, we achieved the highest accuracy (69.8%) of after-market price movement prediction through SVM with RBF kernel, and the highest accuracy (68.12%) of next-day price movement using locally weight regression. Generally, we find the accuracy of predicting the after-market price movement is much higher than next-day prediction. Since the market would get impacted immediately after the financial report is released, it is quite natural that the after-market price movement relies more on the financial report than the next-day, which accounts for the difference between accuracy.

On the analytical articles dataset, we achieved the highest accuracy (58.96%) of after-market price movement prediction through locally weight regression, and the highest accuracy (64.38%) of next-day price movement using locally weight regression. Similar to the financial report dataset, the accuracy of after-market prediction is higher than next-day prediction, which is due to the same reason as discussed in the above paragraph.

Generally, locally weight regression model produces the best result on both dataset and both time period. In addition, comparing the performance of models on financial report dataset and analytical articles dataset yields that predicting stock market movement by financial report data is much more successful than by analytical articles, with the difference of highest accuracy reach 4 per cent.

In addition to the average performance, which is the mean or expected value, we also care about the variance, which represents the stability of our model. To evaluate this aspect, we train and test our model for 20 times, randomly choosing training and testing set at each time. The result is shown in Fig. 3.

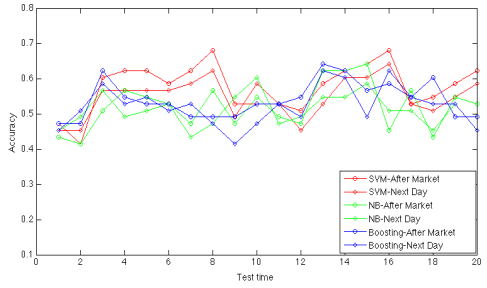



Fig. 3. The prediction accuracy on analytical article dataset

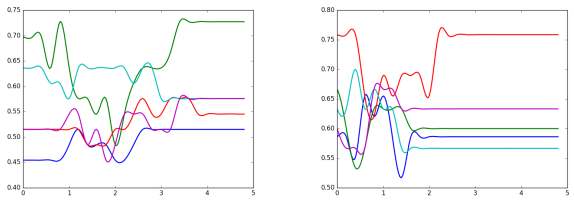
### C. Discussion

In this section, we study the performance SVM and LWLR model, and then try to interpret why they produce better result than the other two models.

*1) SVM discussion:* Recalling the form of RBF kernel:

$$K(x, x') = \exp(\frac{-||x - x'||^2}{2\Gamma^2}), \tag{13}$$

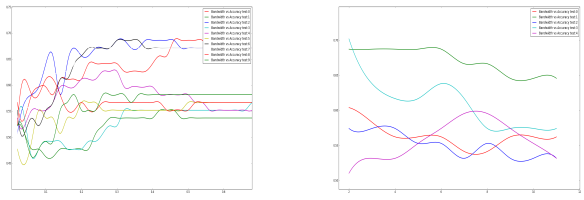
the performance of SVM with RBF kernel depends on the value of $\Gamma$. Representatively, we only work on after-market price movement prediction. To find a best $\Gamma$, 5-fold cross validation is used to test the prediction accuracy of model with different $\Gamma$. The cross-validation result is shown in Fig. 4(b) and Fig. 4(a), which are separately the performance on financial report dataset and analytical articles dataset. The x-axis is $\log_{30}\Gamma$, and the y-axis is the prediction accuracy. Revealed by the figure, the performance of SVM is correlated with the value of $\Gamma$, and generally reach the peek point when $\log_{30}\Gamma = 0.8$ and $\Gamma = 2.6$. Thus, for testing, $\Gamma$ is separately set as $30^{1.2}$ and $30^{2.6}$. In SVM model, $\Gamma$ represents the correlation between testing samples and training samples, and in traditional works, where testing samples nearly follow the same distribution with the training samples, $\Gamma$ is usually a small value. A large $\Gamma$ means the correlation between features and label is loose. Thus, we claim that predicting stock price movement via analytical articles with our selected feature is not successful.



(a) SVM on financial report dataset (b) SVM on analytical article dataset

Fig. 4. SVM prediction accuracy with different value of bandwidth $\Gamma$

*2) LWLR discussion:* LWLR is somehow similar to SVM, whose performance is also guided by a so-called bandwidth, referring to the $w_i = \exp(\frac{-||x-x_i||^2}{2\Gamma^2})$ in Eq. 4. $\Gamma$ here also represents the similarity between testing samples and training samples, or say the correlation between features and label, which is the same as SVM model. Representatively, we only work on after-market price movement prediction. 5-fold cross validation is utilized to select the best $\Gamma$, the performance on financial report dataset and analytical article dataset are separately shown in Fig. 5(a) and Fig. 5(b).



(a) LWLR on financial report dataset (b) LWLR on analytical article dataset

Fig. 5. LWLR prediction accuracy with different value of bandwidth $\Gamma$

Revealed by figures above, LWLR regression outperforms SVM. LWLR differs the most from SVM by the supervisory signal it uses. Specifically, LWLR uses the percentage of price movement as the supervisory signal while SVM only cares about whether the price goes up or down. It is unfair the assume that 50% price change is the same as 10% price change, which is done in SVM model. Thus, LWLR outperforms SVM.

The reason for LWLR and SVM outperforming other two models lies on the bandwidth parameter $\Gamma$. By adjusting the value of $\Gamma$, we control the dependency of testing samples on the training samples. However, in Naive Bayesian and Boosting model, we simply assumes that all samples follow the same distribution as each other, which is proved not the case.

## VII. Conclusion

As known to public, stock market is known as a chaotic system and it has been proved that even model built with empirical key features could still result in low accuracy. In our work, we tried to limit our scope to earning release day, and it turned out that we could build models achieving around 70% prediction accuracy. To build the model, We take financial statistics collected from company's quarterly earning report, market surprise due to consensus expectations in terms of digital data, and sentiment analysis of relevant articles from mainstream media of financial professionals as two sets of input features, and make stock market movements prediction in after-hour period and trend in the day after the release day. SVM and LWLR model outperforms other models as shown by experiments, as they control the correlation among data, which was discussed in Section VI. However, due to the limited number of company choices, we have small data size ( 300 samples) which could lead to high bias and overfitting. Stock price is not only affected by certain financial features, consensus news, but also company direction and future business guidance, which are difficult to be digitized.

## VIII. Future Improvements

The time we spent on the project is limited, so if we have more time and more computational resources, we would like to improve three parts: 1. Feature selections: Script can be written to auto collect/process data to increase the learning set size, and use feature ranking to analysis the covariance between features so that we can label and pick the top-ranking features as the model inputs; 2. Natural Language Processing analysis: The sentiment result from Stanford NLP toolkit is too general to apply for financial news analysis and we would spend more time to develop a financial-specific sentiment analysis algorithm, which might give us a better and more accurate result; 3. Model Improvement: We would spend more time on SVM algorithm and explore the most appropriate kernel to separate features in high dimension.

## References

[1] Subhabrata Choudhury, Subhajyoti Ghosh, Arnab Bhattacharya, Kiran Jude Fernandes, and Manoj Kumar Tiwari. A real time clustering and svm based price-volatility prediction for optimal trading strategy. *Neurocomputing*, 131:419–426, 2014.

[2] Jigar Patel, Sahil Shah, Priyank Thakkar, and K Kotecha. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications*, 42(1):259–268, 2015.

[3] Esmaeil Hadavandi, Hassan Shavandi, and Arash Ghanbari. Integration of genetic fuzzy systems and artificial neural networks for stock price forecasting. *Knowledge-Based Systems*, 23(8):800–808, 2010.

[4] Kyoung-jae Kim and Ingoo Han. Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with Applications*, 19(2):125–132, 2000.

[5] Jar-Long Wang and Shu-Hui Chan. Stock market trading rule discovery using two-layer bias decision tree. *Expert Systems with Applications*, 30(4):605–611, 2006.

[6] Robert K Lai, Chin-Yuan Fan, Wei-Hsiu Huang, and Pei-Chann Chang. Evolving and clustering fuzzy decision tree for financial time series data forecasting. *Expert Systems with Applications*, 36(2):3761–3773, 2009.

[7] Sheng-Hsun Hsu, JJ Po-An Hsieh, Ting-Chih Chih, and Kuei-Chu Hsu. A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression. *Expert Systems with Applications*, 36(4):7947–7951, 2009.

[8] Theodore B Trafalis and Huseyin Ince. Support vector machine for regression and applications to financial forecasting. In *IJCNN (6)*, pages 348–353, 2000.

[9] Reza Hafezi, Jamal Shahrabi, and Esmaeil Hadavandi. A bat-neural network multi-agent system (bnnmas) for stock price prediction: Case study of dax stock price. *Applied Soft Computing*, 29:196–210, 2015.

[10] Jonathan L Ticknor. A bayesian regularized artificial neural network for stock market forecasting. *Expert Systems with Applications*, 40(14):5501–5506, 2013.

[11] Michael Hagenau, Michael Liebmann, and Dirk Neumann. Automated news reading: Stock price prediction based on financial news using context-capturing features. *Decision Support Systems*, 55(3):685–697, 2013.

[12] Yauheniya Shynkevich, TM McGinnity, Sonya Coleman, and Ammar Belatreche. Stock price prediction based on stock-specific and sub-industry-specific news articles. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015.

[13] Robert P Schumaker and Hsinchun Chen. Textual analysis of stock market prediction using breaking financial news: The azfin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2):12, 2009.