Stock Market Prediction using NLP

Dr. Thangarajah Akilan
Faculty of Computer Science
Lakehead University
Thunder Bay, Canada
takilan@lakeheadu.ca

Jugal Shah
Department of Computer Science
Lakehead University
Thunder Bay, Canada
jshah5@lakeheadu.ca

Pankti Joshi
Department of Computer Science
Lakehead University
Thunder Bay, Canada
pjoshi@lakeheadu.ca

Tejas Wadiwala
Department of Computer Science
Lakehead University
Thunder Bay, Canada
twadiwal@lakeheadu.ca

Vikas Trikha
Department of Computer Science
Lakehead University
Thunder Bay, Canada
vtrikha@lakeheadu.ca

***Abstract*—Sentiment analysis is an analytical approach of deriving and labeling the emotions for a given structured, semi-structured, or unstructured data. This paper attempts to present a comprehensive sentiment analysis on the news headlines dataset to predict the rise and fall of a stock price. The stock market having an unpredictable nature, presents numerous challenges in performing accurate analysis of news headlines or stock tweets. Several methods have been investigated for stock analysis up until now; this paper contains a brief survey of existing researches on stock market analysis. Additionally, it focuses on the use of natural language processing (NLP) based text classification methods like sentiment analysis to estimate whether the stock price would increase or decrease. NLP algorithms like bag of words (BoW), term frequency-inverse document frequency (TF-IDF), assists in extracting features from news headlines by converting text data to mathematical representation. Different machine learning and deep learning models like support vector machine (SVM) and convolution neural network (CNN) respectively can be trained on this vector representation of textural headlines for stock sentiment analysis. This study aims to predict whether the stock price would increase or decrease, given the top 25 headlines for a single day. A long short term memory (LSTM) model is proposed to perform the prediction. The model is trained and tested using the Daily News for Stock Market Prediction dataset as a corpus. The paper additionally provides a comparison of the different models builds on the same dataset. All the models are evaluated on their binary cross-entropy loss, accuracy, F1 score, precision, and recall.**

***Index Terms*—Convolution neural network, lemmatization, long short term memory, sentiment analysis, stock prediction, support vector machine, term frequency-inverse document frequency**

I. Introduction

Today technology has connected everyone with everything-articles and reviews are being posted online more frequently than ever. This large amount of data present in natural human language can be used as an opportunity to understand how articles reported on the news, or people's online reviews of a product can influence the stock market. Since the stock markets are highly dynamic, companies develop various statistical reports and employ multiple data analysts to analyze the ongoing situation of the stock market. Therefore, this project would be inclusive of all the challenges faced by analysts and hold a thousand capacities to predict the stock market behavior. The critical hurdle encountered by an investor is the scope of interpretation of stocks due to the massive volume of disorganized data available on the internet and narrowing it down to define a non-linear relationship among existing information. General Stock Market outlines confront way too much inconsistency in nearly every stock price, which makes it very random to analyze the behavior of shares and risks to investing in the same. The fluctuations in the stock market can also be referred to as volatility, which turns out to be an essential factor in mitigating the risk of investing in any stock [1]. To deal with the existing challenges, machine learning comes in to play and attempts to provide an assertive solution by using various NLP techniques. An overabundance of research has been performed in the scope of deep learning to produce baseline architecture and conduct effective ways of data mining to cleanse and organize data. Various text pre-processing techniques such as tokenization, stemming, and lemmatization has been used to classify the data and make data ready to be fed to machine learning models. Furthermore, different classifiers such as random forest and SVM can be used to classify the stock market sentiments. Also, CNN's can be implemented to predict the movement of the stock market. Performance metric parameters like accuracy, F1 score, precision, and recall can be used to compare the results attained from different models and can be used to find the best performing model.

II. Literature Review

There has been recently some effort to mine social media data for stock market analysis using the public sentiment. LI Bing et al., 2014, have performed sentiment analysis on twitter data (textual tweet); they have done it by extracting ambiguous tweet data through NLP techniques to define public sentiment [2]. The authors then have discovered a pattern between public opinion and real stock price movements using one of the data mining techniques. The stock market prediction has always been a fancy issue for the past few decades but has only been able to predict with reduced accuracy. Also, it has been widely accepted by economic specialists that there is a potential connection between the company's stock price and the published information about it [3]. LI Bing et al. has classified the tweets in one of the five categories - Positive+, Positive, Neutral, Negative, Negative-. This classification is done by considering each tweet structure as a combination of several words and phrases. Then a chi-squared test is used for association rule mining, which is used to identify the relating patterns between public sentiments and stock market prices. In their experiment, 30 representative listed companies from different industries as the target were selected, and they were able to retrieve almost 15 million records of the Twitter data, which may mention these companies, like their products. The objects are then classified into sentiment categories to predict the hourly rising and falling of the stock. The algorithm that is proposed by LI Bing et al. has been found to have overall accuracy better than SVM, C4.5, and Naive Bayes. Also, if there is a need to increase the overall efficiency, more data can be collected through Facebook.

Tabari et al., 2018, focused on causality between twitter tweets and stock market returns. Tabari et al. pulled the twitter dataset of three months using the twitter API. A list of 100 common stock symbols was utilized to filter only stock related tweets from the collected dataset. Any tweet that comprises at least one symbol from the list was selected for further processing. A total of 20013 tweets were obtained post applying the filter and were submitted to Amazon Mechanical Turk for labeling. At Amazon Mechanical Turk, four workers assigned a number -2 and 2 for every entry. An average of this input was used to attach a sentiment as a label for the tweet. Table I below shows the marking mechanism.

TABLE I
SUMMARY OF TWEETS LABELED BY AMAZON MECHANICAL TURK

Range	Label assigned to tweets
[-2, -0.5]	Negative
[-0.5, 0.5]	Neutral
[0.5, 2]	Positive

Next, to build a classification model, the data was pre-processed by deleting the punctuations, tokenizing the spans, and deleting the stop words. Post pre-processing new feature is created by calculating the count of positive words and negative words in each tweet using the predefined list of positive and negative words from past research. The newly created feature, in association with the TF-IDF vectors, is used as a feature set for the SVM based classification model. The model gave an accuracy of 79.9%. This stock related tweets classified by the Amazon Mechanical Turk and the SVM model are then compared against the stock market return of Apple, and Facebook using the statistical method Granger Causality to represent meaningful causality between stock return and tweets [4].

Researchers have recently found that news are one of the most influential sources that affect the stock market and are necessary for achieving more accurate predictions. Azadeh Nikfarjam et. al., 2010, have categorized stock market prediction systems in different dimensions: input data - it is the data which will be used for the prediction, prediction goal - it is the possible market prediction goal, it can be the future stock price or the volatility of the prices or market trend, and prediction horizon - is the period in which the prediction would be valid. The author has given a generalized overview of the news based market prediction in two main phases: training and operational phase. In the training phase, a set of training data is prepared, i.e., pre-classified train data, which includes news and market information such as market prices. This is fed into the classifier, where one of the predefined classes will be assigned to incoming news. The authors have also stated the comparison between classification systems that have been developed by different authors, which includes feature selection technique, feature representation(weighting), including elements, and the news source(where they have collected the data) [5].

III. Dataset

In this study, Daily News for Stock Market Prediction dataset is used to perform sentiment analysis on news headlines and predict if the stock price would increase or decrease [6]. The dimension of the dataset is 1989 $\times$ 27. Table II below, shows the head of the dataset. Each entry in the dataset represents the top 25 headlines for a single day and a label indicating whether the headlines resulted in stock price rise or fall. As the end goal is to anticipate the tendency of stock of a specific company, the news headlines that lead stock price to decline are labeled as 0, while the headlines that impacted the stock price to increase are labeled as 1. Pandas library is utilized for reading this dataset. The dataset does not suffer from class imbalance, as indicated by Fig. 1.

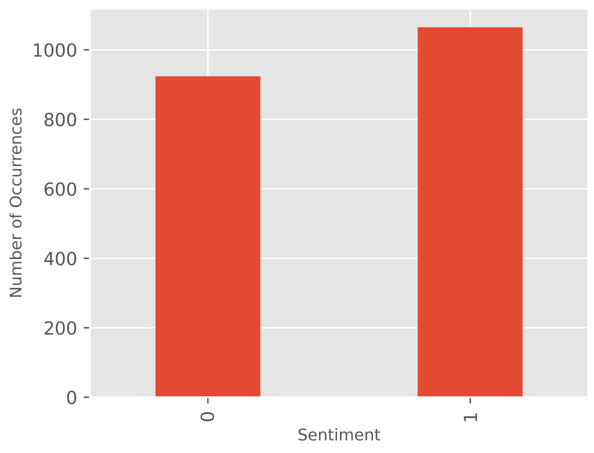

Fig. 1. Sentiment Label Count.

TABLE II
DATASET HEAD

Date	Label	Top1	Top2	....	Top25
2008-08-08	0	b"Georgia downs two Russian warplanes as countries move to brink of war"	b'BREAKING: Musharraf to be impeached.'		b"No Help for Mexico′s Kidnapping Surge"
2008-08-11	1	b'Why wont America and Nato help us? If they wont help us now, why did we help them in Iraq?'	b'Bush puts foot down on Georgian conflict'		b"So this is what itś come to: trading sex for food."
2008-08-12	0	b'Remember that adorable 9-year-old who sang at the opening ceremonies? That was fake, too.'	b"Russia ends Georgia operation"		b"BBC NEWS — Asia-Pacific — Extinction by man not climate"
2008-08-13	0	b' U.S. refuses Israel weapons to attack Iran: report'	b"When the president ordered to attack Tskhinvali [the capital of South Ossetia], we knew then we were doomed. How come he didnt́ realize that?"		b'2006: Nobel laureate Aleksander Solzhenitsyn accuses U.S., NATO of encircling Russia'
2008-08-14	1	b'All the experts admit that we should legalise drugs '	b'War in South Osetia - 89 pictures made by a Russian soldier.'		b'Philippines : Peace Advocate say Muslims need assurance Christians not out to convert them'

IV. Data Preprocessing

The primary step in the preprocessing is to check for null values, and then partition the data set into the training and testing tuples. Data is partitioned using the train and test split function from the sklearn library. With the split functionality, the data is divided in the ratio of 70:30 for training and testing data, respectively. The random state is set to 2003 so that during each execution, the train and test data is not modified. After data splitting, below-preprocessing steps are applied independently to the train and test data.

A. Headlines Merging

Considering that all 25 headlines influence the stock price, in this step, all 25 headlines in a row are combined, and a string is generated. The string is then added as an additional entry to that row. This step is executed on all the entries in the dataset; it helps in improving the efficiency of preprocessing and model training.

B. Data Cleaning

Data cleaning is the first and foremost step in data preprocessing. In this step, the combined headlines generated in the last step is converted to lower case and is tokenized using the NLTK word tokenize library. Post tokenizing, the stop words and punctuations are removed from the headlines. The stop words removal is done according to the english stopwords list defined in the NLTK corpus library. The actual code snippet for data cleaning is given in the Appendix A.

C. Data Lemmatization

In the process of data lemmatization, the words present in the headline are converted to its root word. Data lemmatization is performed using the WordNetLemmatizer package of the NLTK library. Since the meaning of the words in the news headlines is essential for stock analysis, in the proposed study, lemmatization is performed.

D. Data Vectorization

Data vectorization is a technique to map the words from the vocabulary to real numbers. This can be done using a variety of NLP techniques like BoW, TF-IDF, and Word2Vec. The BoW is the simplest to implement. In BoW, a matrix of documents against tokens is created, with each entry storing the token count in the document. Here a word that has the highest occurrence has higher importance. In TF-IDF, the focus is on relevant words rather than frequent words. TF-IDF of a term t in document d that belongs to the collection of documents D is represented in Equation (1) [7].

$$tf\text{-}idf(t,d,D) = tf(t,d) \times idf(t,D) \tag{1}$$

Equation (2) shows the expansion of *idf(t,D)*

$$idf(t,D) = log\frac{|D|}{1+|\{d \in D : t \in d\}|} \tag{2}$$

In above equation, the numerator represents the number of documents in the corpus, while denominator presents the number of documents that consist of the term t. In the experiment analysis, TF-IDF with ngram set to (2,2) is preferred over BoW for data vectorization as it leads to improved model performance.

For the proposed LSTM model, data vectorization is performed using the Tokenizer class of Keras library. Tokenizer vectorizes the input by converting the headlines into a sequence of integers. The tokenizer word vocabulary is restricted to 20000. Additionally, padding is performed on all sequences of integers so that sequences are of equal length before passing as input to the model.The stock labels feed during training are also converted to categorical to support binary classification.

E. Data reshaping

Data reshaping is performed with CNN based model, mentioned under experimental analysis. Post data cleaning and vectorization, training, and testing data is converted into a NumPy array. This is done using the NumPy library. The

NumPy array of the news headlines are then reshaped to include the third dimension, as a requirement for working with Conv1D library in Keras.

V. Proposed Model

The paper proposes an LSTM model to predict if the stock price would increase or decrease given the top 25 headlines for a single day. Fig. 2 below shows the complete architecture of the proposed model. The model is build using the Keras library. The preprocessed training and testing data are first

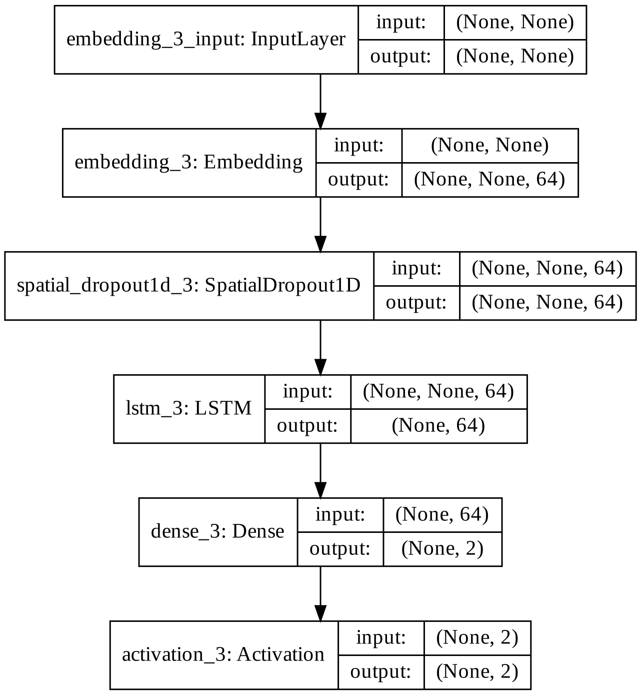

Fig. 2. Proposed LSTM Model Architecture.

converted into batches of fixed size and then passed to the model. Additionally, to check if the model overfits after a certain number of epochs, the training data is divided into train and validation sets. For this study, twenty percent of the training data is utilized for validation. The batches of input data are first forwarded to the embedding layer. The embedding layer expects the input words to be integer encoded. This is achieved, by vectorizing the input using the Tokenizer class, as a part of the preprocessing step. Additionally, a dropout rate of 0.5 is applied to the neurons, indicating deactivation of 50 percent of neurons to avoid overfitting. The embedded output is then passed to the LSTM layer. At the LSTM layer dropout of 0.5 is applied to the input, and a recurrent dropout of 0.5 is applied to the recurrent state. The LSTM layer outputs a vector of dimensionality 64, which is directly passed to the dense layer. The sigmoid activation function is utilized on the dense layer output to predict mutually exclusive stock labels.

A. Parameters Configurations

During the research, different permutations and combination of the hyperparameters, model parameters, and model layers were tried and tested. Table III below shows the key configuration that contributes to the performance of the proposed model. Additional models build during the research are covered under experimental analysis.

TABLE III
Proposed Model Parameters Configurations

Parameter	Configurations
Tokenizer Max Features	20000
Padding Length	200
Batch size	32
Epoch	3
Optimizer	Adam
Activation function	Sigmoid
Loss function	Binary Cross-Entropy
Learning Rate	0.001

B. Model Implementation

Python language is used to define the model structure. Each layer of the proposed model is imported from the Keras library and stacked using the Keras sequential model. The learning process is configured using the compile method. Once compiled, the model training is performed using the fit method. The fit method takes as input the training data, the label data for learning, the number of epochs, the batch size, and the validation split. Verbose is activated during training to archive the epoch results. The actual code snippet is mentioned in Appendix B.

C. Model Evaluation

The proposed model is evaluated using binary cross-entropy loss, accuracy, precision, recall, and F1 score. Table IV below shows the score of performance metrics for the proposed model. Keras backend library is used to calculate binary cross-entropy and accuracy; however, the recall precision and F1 score are calculated using the user-defined equations. Below are the equations of the performance metrics [8].

TABLE IV
Proposed Model Performance Metrics

Parameter	Score
Loss	0.69
Accuracy	0.56
F1 Score	0.59
Precision	0.56
Recall	0.64

$$Recall = \frac{truepositives}{truepositives + falsenegatives} \tag{3}$$

$$Precision = \frac{truepositives}{truepositives + falsepositives} \tag{4}$$

$$F1Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

In the Equation (6) *TP* is True Positive, *TN* is True Negative, *FP* is False Positive and *FN* is False Negative.

VI. Experimental Analysis

In the experimentation phase of this study, various models were trained, and test results were examined. Table V below shows the comparison between all the models built in this research. The sections below present a brief overview of the implementation details of the experimental models.

A. Random Forest

In this experiment, data pre-processing steps, similar to that of the proposed model, were executed first. The TF-IDF vector generated post data preprocessing are fed to the Random Forest classifier along with the target label for model training. For this analysis, the RandomForestClassifier class from the sklearn library was employed. For the classifier, the n_estimators variable used to control the number of trees is set to 200, and the criterion is set to entropy to apply information gain. The classifier performance is slightly lower than the proposed model. However, the F1 score is impacted due to low precision.

B. SVM

As SVM classifiers are considered suitable for binary classification, and the dataset size is less than ten thousand, in this study, we are using the support vector classification (SVC) method of the SVM algorithm for building the classifier. For the classifier, the kernel is set to linear, and the regularization of 1.0 is applied. The SVM classifier performance is almost equal to the random forest classifier with a reasonable F1 score, but it does not perform better than the proposed model.

C. CNN

The Conv1D model is built using Keras library. The model consists of two pairs of Conv1D, MaxPooling1D, Dropout layer with the rectified linear unit (ReLU) applied. A flattening layer and dense layer with sigmoid activation follow the pair. Stochastic gradient descent (SGD) is applied as the optimizer. The CNN model accuracy is equal to the proposed model. However, the model F1 score is too high due to the high recall score, indicating that the recall dominates the accuracy and is not dependable for prediction.

VII. Contribution

Table VI below reflects the contribution of every team member along with the approaches adopted by each member to find a key solution to the existing challenges. Narrowing it down to report writing each member provided the experimental analysis of their respective models and equal amount of contribution was made to the rest of document writing.

TABLE V
Experimental analysis using different Models

Model	Results
Random Forest	Accuracy: 0.52 F1: 0.44 Precision: 0.47 Recall: 0.52
SVM	Accuracy: 0.51 F1: 0.51 Precision: 0.51 Recall: 0.51
CNN	Accuracy: 0.55 F1: 0.70 Precision: 0.55 Recall: 0.97
LSTM	Accuracy: 0.56 F1: 0.59 Precision: 0.56 Recall: 0.64

TABLE VI
Team Contribution

Member	Contribution	Model Implementation
Jugal Shah	25%	LSTM Model
Pankti Joshi	25%	CNN Model
Tejas Wadiwala	25%	SVM Classifier
Vikas Trikha	25%	Random Forest Classifier

VIII. Conclusion

In this paper, a comprehensive literature review has been presented on sentiment analysis for stock market prediction, and a model has been proposed to mitigate the existent challenge using the LSTM, which uses news headlines to classify the current stock market position. The proposed LSTM model outperforms other models by achieving an accuracy of 0.56 and an F1 score of 0.59. Various preprocessing methods have been exercised for cleansing and extracting meaningful features from the dataset. The model is then evaluated using various performance parameters such as accuracy and F1 score. The study experiments with different machine learning and NLP techniques providing experimental analysis for different classifiers. Furthermore, the predefined Word2Vec model gives us another area to explore to improve model performance. In addition to that, the data plays an essential role in the performance of the model; thereby, different datasets can be explored to check the impact of the existing model and to widen the future scope of the model's performance.

Appendix

A. Data Cleaning

```
#utility function to clean the dataset
def datacleaning(remove_stopwords,useStemming,
    useLemma,removePuncs,newdata):
  cleanReview=[]
  for x in range(0,len(newdata.values)):
    tmpReview=[]
    for w in nltk.word_tokenize(newdata.values[x]):
        newWord = str(w).lower() #Set newWork to be
    the updated word
```

```
            if remove_stopwords and (w in stopwords_en):
        #if the word is a stopword & we want to remove
        stopwords
                continue #skip the word and don t had
        it to the normalized review
            if removePuncs and (w in punctuations):#if
        the word is a punc. & we want to remove
        punctuations
                continue #skip the word and don t had
        it to the normalized review
            if useStemming: #if useStemming is set to
        True
                #Keep one stemmer commented out
                #newWord = porter.stem(newWord) #User
        porter stemmer
                newWord = lancaster.stem(newWord) #Use
        Lancaster stemmer
            if useLemma:
                newWord = wordnet_lemmatizer.lemmatize(
        newWord)
            tmpReview.append(newWord) #Add normalized
        word to the tmp review
        cleanReview.append(' '.join(tmpReview))
    return cleanReview
```

Listing 1. Data Cleaning.

B. LSTM Model

```
#LSTM Model
model = Sequential()
model.add(Embedding(max_features, 64))
model.add(SpatialDropout1D(0.5))
model.add(LSTM(64, dropout=0.5, recurrent_dropout
    =0.5))
model.add(Dense(2))
model.add(Activation('sigmoid'))

#Compile the model
model.compile(loss='binary_crossentropy',optimizer='
    adam',metrics=['accuracy',f1_m,precision_m,
    recall_m])

#Train the model
modelhistory=model.fit(X_train_pad, Y_train_pad,
    batch_size=32, epochs=3,validation_split=0.2)

#Test the model
model.evaluate(X_test_pad,Y_test_pad,batch_size=32)
```

Listing 2. LSTM Model.

References

[1] J. Santhappan and P. Chokkalingam, "An Intelligent Market Capitalization Predictive System Using Deep Learning," 2018 International Conference on Advanced Computation and Telecommunication (ICACAT), Bhopal, India, 2018, pp. 1-9.doi: 10.1109/ICACAT.2018.8933727

[2] L. Bing, K. C. C. Chan and C. Ou, "Public Sentiment Analysis in Twitter Data for Prediction of a Company's Stock Price Movements," 2014 IEEE 11th International Conference on e-Business Engineering, Guangzhou, 2014, pp. 232-239. doi: 10.1109/ICEBE.2014.47

[3] T. Leung, H. Daouk and A. Chen. Forecasting stock indices: a comparison of classification and level estimation models. International Journal of Forecasting, 16, 173190. 2000.

[4] Tabari Narges, Piyusha Biswas, Bhanu Praneeth, Armin Seyeditabari, Mirsad Hadzikadic, and Wlodek Zadrozny. "Causality Analysis of Twitter Sentiments and Stock Market Returns." In Proceedings of the First Workshop on Economics and Natural Language Processing, pp. 11-19. 2018.

[5] A. Nikfarjam, E. Emadzadeh and S. Muthaiyah, "Text mining approaches for stock market prediction," 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), Singapore, 2010, pp. 256-260. doi: 10.1109/ICCAE.2010.5451705

[6] Aaron7sun. "Daily News for Stock Market Prediction". kaggle.com. https://www.kaggle.com/aaron7sun/stocknews (Accessed March 19, 2020).

[7] Ethen Liu. "TF-IDF, Term Frequency-Inverse Document Frequency". github.io. https://ethen8181.github.io/machine-learning/clustering_old/tf_idf/tf_idf.html (Accessed March 21, 2020).

[8] Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures. exsilio.com. https://blog.exsilio.com/all/accuracyprecision-recall-f1-score-interpretation-of-performance-measures/ (Accessed March 20, 2020).