Empirical Evaluation of Machine Learning Algorithms for Stock Market Prediction using News Headlines

Lavanya Sharma L9SHARMA@UWATERLOO.CA
MDSAI
Student ID:20865950
University of Waterloo

Abstract

In recent years, many machine learning algorithms for the prediction of stock price trends have been proposed. In this research, the prediction performance of two traditional machine learning models (Logistic Regression and SVM), three tree-based machine learning models (Random Forest, XGBoost, and Gradient Boosting), and one deep learning model(LSTM) is compared. The combined dataset from the news dataset consisting of top 25 news headlines from Reddit World-News Channel from 2008-2016 and Stock data from Dow Jones Industrial Average (DJIA) for the corresponding period from Yahoo Finance is used for evaluation. The experimental results demonstrate that the random forest machine learning algorithm performs slightly better than other machine learning algorithms when compared based on accuracy alone. When based on execution time and accuracy both, logistic regression also performs well on the dataset. Performance of deep learning algorithm (LSTM) is least among all the methods used for comparative analysis. Also, deep learning algorithm needs much more time for execution as compared to other methods.

Keywords: Stock market, Machine learning models, News data

1. Introduction

1.1 What is the problem?

Stock market prediction is an important problem in finance and economics. It is an attempt to forecast the future value of an individual stock for a particular sector or the market. Stock prices generally follow historical patterns and prediction of future stock prices can be made by fundamental analysis of previous trends. Unpredictable factors such as political situations and natural calamities make stock prediction a very challenging problem.

In the real world, stocking and trading involves much more than just monitoring the stock price. Traders use a vast amount of market information such as news and reports to trade stocks. Nowadays, most of the market information can be easily obtained online. News websites provide a huge amount of unstructured data and can have potentially strong influences on stock markets and currencies. The goal of this project is to empirically analyze several machine learning algorithms for stock market prediction by using online news data.

1.2 Why is it an important problem?

Stock price prediction is a very challenging problem due to the number of unforeseen events that cause the stock price to change. It is a highly researched and debated topic, and its successful ventures can prove highly profitable. An accurate model will guide investors for buying and selling for profitable transactions. Unfortunately, stock prices are considered to be very dynamic and are susceptible to quick changes. Developing a method that successfully predicts changes in stocks requires great patience, experimentation, and insights. Multiple factors can influence stock market prediction when using machine learning algorithms. Hence, doing a comparative analysis of different machine learning techniques and selecting the best method for predicting accurately can be very beneficial for stock price prediction.

2. Related work

Zhai et al. (2007) combined news headlines with technical indicators in order to do stock prediction. The main idea in the paper is that stock prediction needs information from various sources like mass media. In their proposed methodology, they use multiple support vector machine models to predict the stock prices.

Another related work Nabipour et al. (2020) discusses machine learning and deep learning algorithms for predicting stock trends. In their study they use ten years of historical data and several techniques for predicting stock trend. Their study suggests that there was a significant improvement in the performance of models while using binary data as opposed to continuous data. Chen and Hao (2017) also investigated prediction of stock market with various machine learning methods. In their study, they proposed a basic hybridized framework of the feature weighted support vector machine as well as feature weighted K-nearest neighbor to effectively predict stock market indices. Lot of popularity is also seen for ensemble techniques for stock price prediction. For example, Ballings et al. (2015) benchmarked ensemble methods (Random Forest, AdaBoost and Kernel Factory) against single classifier models (Neural Networks, Logistic Regression, Support Vector Machines and K-Nearest Neighbor) and suggested that more novel studies in the domain of stock price direction should include ensembles. Also, Deepak et al. (2017) applied SVM on stock exchange data and produced daily prediction accuracy based on different stocks which produced great results.

3. Machine learning models

In this research, I will be concentrating on comparing prediction performance of two traditional machine learning models (Logistic Regression and SVM), three tree-based machine learning models (Random Forest, XGBoost and Gradient Boosting) and one deep learning model(LSTM) to predict stock market movement using both news data and stock trend data. Here, I have considered stock prediction as a classification problem where closing value is defined as 0 if DJIA has decreased from previous day's close, and is defined as 1 if closing value of DJIA has increased or stayed same as previous day's close. The goal is to study and assess which algorithm will be most suitable for stock prediction when using daily news headlines.

3.1 Logistic Regression

Logistic regression is a classification algorithm used to solve binary classification problems. The logistic regression classifier uses the weighted combination of the input features and passes them through a sigmoid function. Sigmoid function transforms any real number input, to a number between 0 and 1. Mathematically, the logistic regression hypothesis function can be defined as follows:

$$h_0(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}} \tag{1}$$

The objective is to minimize the cost function to achieve an optimal probability. The cost function is calculated as shown below.

$$Cost(h_0(x), y) = \begin{cases} log(h_0(x)), & \text{if } y = 1. \\ -log(1 - h_0(x)), & \text{otherwise.} \end{cases} \tag{2}$$

3.2 Support Vector Machines

Support vector machine (SVM) is another model for binary classification problem and is available in various kernel functions. The objective of an SVM model is to estimate a hyperplane (or decision boundary) on the basis of feature set to classify data points.The dimension of hyperplane varies according to the number of features. As there could be multiple possibilities for a hyperplane to exist in an N-dimensional space, the task is to identify the plane that separates the data points of two classes with maximum margin. A mathematical representation of the cost function for the SVM model is defined and is shown below.

$$J(\theta) = \frac{1}{2} \sum_{j=1}^{n} \theta_j^2, \tag{3}$$

such that

$$\theta^T x^{(i)} \geq 1, y^{(i)} = 1, \tag{4}$$

$$\theta^T x^{(i)} \leq -1, y^{(i)} = 0 \tag{5}$$

The function above uses a linear kernel. Kernels are usually used to fit data points that cannot be easily separable or data points that are multidimensional. In our case, we have used Gaussian SVM model and basic linear SVM model for evaluation.

3.3 Random Forest

Random forest (RF) is an advanced form of decision trees (DT) which is also a supervised learning model. RF consists of large number of decision trees working individually to predict an outcome of a class where the final prediction is based on a class that received majority votes. The error rate is low in random forest as compared to other models, due to low correlation among trees.

3.4 Gradient Boosting

Gradient Boosting is a technique which combines the predictions from multiple decision trees to generate the final predictions. In Gradient Boosting the nodes in every decision tree take a different subset of features for selecting the best split. Additionally, each new tree takes into account the errors or mistakes made by the previous trees. So, every successive decision tree is built on the errors of the previous trees.

3.5 XGBoost

Extreme Gradient Boosting or XGBoost is another popular boosting algorithm. In fact, XGBoost is simply an improvised version of the Gradient Boosting algorithm. The trees in XGBoost are built sequentially, trying to correct the errors of the previous trees. But there are certain features that make XGBoost slightly better than Gradient Boosting. One of the most important points is that XGBoost implements parallel preprocessing (at the node level) which makes it faster than Gradient Boosting. XGBoost also includes a variety of regularization techniques that reduce overfitting and improves overall performance. Additionally, XGBoost model can handle the missing values on its own. During the training process, the model learns whether missing values should be in the right or left node.

3.6 LSTM

LSTMs are deep neural networks which are special case of recurrent neural networks. They are capable of handling long term dependencies. LSTMs also incorporate input and forget gates which allows the units to keep an internal state. This enables LSTMs to pass information from cells early in the chain to later cells.

4. Data

The dataset is obtained from Kaggle. The name of the dataset is *Daily News for Stock Market Prediction*. The dataset consists of three files.

- RedditNews.csv: The dataset consists of historical news headlines from Reddit WorldNews Channel (/r/worldnews). The news headlines are ranked by reddit users' votes. The top 25 news headlines are only considered for a single date. Headlines are collected from date 2008-06-06 to 2016-07-01. The file consists of two columns. The first column is the 'date', and second column is the 'news headlines'. There are 25 headlines for each date.

- DJIA_table.csv: The dataset consists of Stock data from Dow Jones Industrial Average (DJIA) from date 2008-08-08 to 2016-07-01. The dataset is directly downloaded from Yahoo Finance.

- CombinedNewsDJIA.csv: The news data and stock market DJIA data are combined to form a single dataset that contains the date on which the news got published, label (i.e. performance indicator of DJIA) and Top1 to Top25 headlines corresponding to the date. The dataset contains of 1989 records. Label is defined as 0 if closing value of DJIA has decreased from previous day's close, and is defined as 1 if closing value of DJIA has increased or stayed same as previous day's close. The data is balanced with a data split with 925 records labeled as 0 and 1064 records labeled as 1.

Model	**Model Parameters**
Logistic Regression	Default parameters
SVM Gaussian	C=1 regularization parameter, gamma=0.1 corresponding to amount of influence of a single training sample
SVM Linear	C=1 regularization parameter
Random Forest	n_estimators=200 i.e. no. of trees in forest, criterion=entropy
XGBoost	Default parameters
Gradient Boosting	Default parameters
LSTM	Single Embedding layer of dimension(500,128,100) and LSTM(196) with SpatialDropout1D=0.2, batch size = 10, epochs=10

Table 1: Model Parameters used for training machine learning algorithms

5. Experiment

5.1 Data Preprocessing

The combined news and stock market data is imported for evaluation of machine learning algorithm. This data has some extra information (e.g. quotes, single quotes, upper case letter and articles) which is not useful for text processing. We can pre-process the data by removing them from the dataset. Then the data is combined by all sentences or headlines in a row (i.e. Top1 to Top25 news). Finally, CountVectorizer function from Scikit-learn is used to convert the processed text data to vector of token counts.

5.2 Model Parameters and Training

I have experimented with two main parameters of CountVectorizer function for comparative analyses of algorithms. The two parameters used are ngram_range:tuple(min_n,max_n) and max_features. The experiment is carried out using 5 different N-grams (i.e. 1-gram,2-gram,3-gram,4-gram,5-gram) and 3 different max_features(500,1000,3000). A collection of character N-grams makes the representation resilient against misspellings and derivations as opposed to single unigram. Therefore, the effect of various N-grams on machine learning algorithms is studied. Also, training the whole vocabulary present in the text is not feasible and is not effective for training. Max_features parameter helps in extracting the most frequently used text for transformation while avoiding redundant text features. The transformed text data is used for training the machine learning models. Table 1 presents model parameters and training parameters used while training the models.

6. Evaluation metrics

Accuracy: Accuracy is often the most used metric representing the percentage of correctly predicted observations, either true or false. To calculate the accuracy of a model performance, the following equation can be used:

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

In most cases, high accuracy value represents a good model. In addition to accuracy other evaluation metric F1-score is also used.

F1-score: F1-Score represents the trade-off between precision and recall. Precision score represents the ratio of true positives to all events predicted as true.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

Recall represents the total number of positive classifications out of true class.

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

F1-Score calculates the harmonic mean between each of the two. Thus, it takes both the false positive and the false negative observations into account. F1-Score can be calculated using the following formula:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{9}$$

7. Empirical Analysis

7.1 Performance

Figure 1 summarizes the maximum and minimum accuracy attained by each algorithm. On an average the maximum accuracy of models is 80%. We observe Gradient Boosting and LSTM have lower maximum accuracy as compared to others. Also, on an average minimum accuracy is around 58% for all the models.

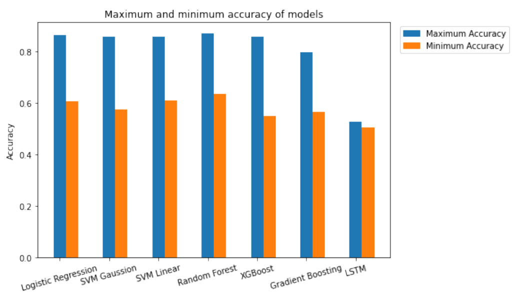

Figure 1: Maximum and minimum accuracy attained by machine learning models during the experiment

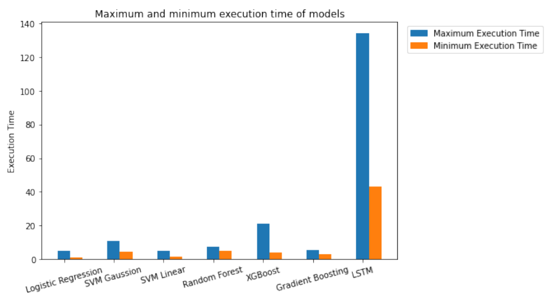

Figure 2: Maximum and minimum execution time attained by machine learning models during the experiment

Table 2 summarizes the accuracy and F1-Score achieved by each algorithm on the dataset for 5 different N-grams and 3 different max_features. It is observed that the maximum accuracy and F1-Score is achieved by random forest algorithm which is 87% and 87.5% respectively when $N\text{-}gram = (3, 3)$ and $max\_features = 500$. Logistic Regression also attains a similar accuracy (86.5%) and F1-Score (86.8%) when $N\text{-}gram = (3, 3)$ and $max\_features = 3000$. Other machine learning algorithms like Linear SVM, Gaussian SVM

			F1 Score(%)					**Test Set Accuracy(%)**					**Execution Time(Sec)**				
			NGrams					*NGrams*					*NGrams*				
			(1,1)	(2,2)	(3,3)	(4,4)	(5,5)	(1,1)	(2,2)	(3,3)	(4,4)	(5,5)	(1,1)	(2,2)	(3,3)	(4,4)	(5,5)
Logistic Regression	*Max Features*	500	65.6	70.5	72.0	71.3	69.2	64.7	69.0	70.4	67.5	60.6	1.29	2.8	4.3	4.4	5.1
		1000	74.7	77.3	76.8	73.2	74.6	74.3	77.0	75.9	71.2	68.8	1.3	2.9	4.3	5.0	4.7
		3000	82.4	84.1	86.8	83.2	80.4	82.0	84.1	**86.5**	82.5	77.5	1.3	3.1	4.0	4.6	**5.2**
SVM Gaussian	*Max Features*	500	87.3	87.6	81.7	63.5	68.6	85.2	**85.7**	82.0	69.8	57.4	6.3	4.4	4.5	4.9	4.6
		1000	87.3	87.3	84.7	74.1	71.6	85.2	85.2	84.7	74.3	62.4	8.7	5.1	4.7	4.6	4.9
		3000	87.3	87.3	85.5	81.0	79.2	85.2	85.2	84.7	81.7	76.2	**10.8**	6.2	5.2	4.8	5.0
SVM Linear	*Max Features*	500	63.6	67.2	71.1	65.9	68.9	64.0	67.2	70.9	69.3	61.1	1.6	3.2	**5.1**	4.5	4.5
		1000	71.8	76.0	73.9	74.1	71.8	72.2	75.9	73.8	74.1	66.9	1.9	2.8	4.0	4.7	4.5
		3000	82.1	85.0	85.9	81.2	79.8	81.7	84.9	**85.7**	81.0	77.5	1.9	3.5	4.0	4.6	4.8
Random Forest	*Max Features*	500	85.9	86.0	87.5	82.8	71.0	84.6	85.2	**87.0**	80.9	63.5	5.0	5.3	5.5	6.7	5.6
		1000	84.9	87.1	85.6	81.6	76.3	83.6	86.5	85.7	81.0	72.0	5.3	5.1	5.8	6.7	6.6
		3000	85.7	84.9	87.3	86.5	82.0	84.7	84.1	86.8	85.4	79.6	5.2	5.2	6.2	6.6	**7.3**
XGBoost	*Max Features*	500	84.7	85.2	76.9	69.6	67.6	84.4	84.4	75.4	63.5	55.0	3.8	5.6	6.6	7.8	7.7
		1000	85.9	85.3	76.3	72.8	67.6	**85.7**	85.2	74.9	66.7	53.0	6.3	8.1	9.3	10.4	9.9
		3000	83.1	84.5	79.9	72.8	67.6	82.8	84.1	79.1	66.7	55.0	18.3	19.3	20.3	**21.1**	20.8
Gradient Boosting	*Max Features*	500	79.3	75.8	73.6	71.0	69.5	76.7	71.4	68.0	61.4	56.6	2.8	3.3	5.1	4.6	**5.4**
		1000	76.3	77.2	74.5	72.4	70.7	74.1	73.5	69.0	63.5	59.5	3.2	3.5	4.5	4.6	4.8
		3000	79.5	81.6	77.3	75.2	75.3	78.0	**79.6**	73.3	68.5	67.7	4.1	3.9	5.2	4.7	5.0
LSTM	*Max Features*	500	65.6	68.0	65.9	67.4	67.3	51.0	**53.0**	52.0	51.0	50.0	47.7	42.9	43.7	43.4	44.5
		1000	67.4	66.9	66.8	67.1	67.3	51.0	51.0	52.6	51.0	51.0	69.9	71.9	73.4	73.8	73.5
		3000	66.7	67.4	67.0	67.0	67.3	51.1	51.0	51.0	51.0	51.0	130.4	131.6	133.2	133.7	**134.0**

Table 2: The table consists of F1-Score, Accuracy and Execution Time for different machine learning models.The maximum accuracy and execution time are highlighted in blue while minimum accuracy and execution time are highlighted in light red.

and XGBoost attain maximum accuracy of 85.7%. While Gradient Boosting achieved maximum accuracy and F1-Score of only 79.6% and 81.6% respectively when $N\text{-}gram = (2,2)$ and $max\_features = 3000$. Interestingly, the worst performing algorithm was LSTM which achieved an maximum accuracy of only 53%. The reason behind bad performance of LSTM could be overfitting, since the dataset is too small for training a deep learning model.

We can also observe that the most ideal N-gram for the presented models for current dataset seems to be (3,3) and (2,2), followed by unigrams. Also, it is observed that when N-gram=(5,5) and max_features=500, the models perform the worst and give out minimum accuracy for that model.

7.2 Execution Time

Table 2 and Figure 2 also represent the maximum and minimum execution time for various machine learning algorithms. We observe that Logistic Regression takes the least time (1.29 seconds) as compared to other machine learning models. It can also be observed in general except for few exceptions that execution time increases as N-grams increases. Logistic Regression takes around 5.2 seconds when $max\_features = 3000$ and $N\text{-}grams = (5,5)$. It can also be observed that SVM Linear also takes very less time to execute (1.6 seconds) when $max\_features = 500$ and $N\text{-}grams = (1,1)$. In general it can be observed that machine learning models take less execution time when $max\_features = 500$ and $N\text{-}grams = (1,1)$. The reason for this could be that less features and smaller N-gram takes less time to be processed by machine learning algorithms as compared to more features and higher N-gram values. It can also be seen that LSTM being a deep learning model is taking the most amount of time. When $max\_features = 3000$ and $N\text{-}grams = (5, 5)$ the maximum execution time is around 134.0 seconds.

8. Conclusion

The goal of the study was to empirically evaluate different machine learning algorithms on stock market data combined with daily news. For the analyses two traditional machine learning algorithms (Logistic Regression and SVM), three tree-based machine learning models (Random Forest, XGBoost and Gradient Boosting) and one deep learning model (LSTM) were used. The models were analyzed based on accuracy, F1-Score and execution time.

Both Logistic Regression and Random Forest have performed best on the dataset as compared to other machine learning models based on accuracy and F1-Score. Though Random Forest performs slightly better as compared to Logistic regression based on accuracy and F1-Score, it can be observed that Logistic regression performs much better as compared to Random Forest when analyzed based on execution time. SVM Linear, SVM Gaussian and XGBoost also have good accuracy and F1-Score with moderate execution times.

It can be concluded that traditional machine learning algorithms and tree-based machine learning algorithms perform well on the given stock market data which is combined with news data.

9. Future Work

Stock market produces a time series data, in which there can be many hidden and unknown indicators which influence the trend of the stock price. In the present work I have used only news headlines as one such indicator. But there can be many unknown indicators which can help in improving stock market prediction accuracies. Selecting proper indicators for stock market can be a future research problem. Also, unsupervised learning methods where model can extract vital information from many indicators which might influence stock market can be one of the possible directions for future work. Another direction could be to assess deep learning models on a much bigger dataset using domain knowledge and several market indicators.

References

M. Ballings, D. Van den Poel, N. Hespeels, and R. Gryp. Evaluating multiple classifiers for stock price direction prediction. *Expert Syst. Appl.*, 42(20):7046–7056, 2015.

Y. Chen and Y. Hao. A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction. *Expert Syst. Appl.*, 80(C):340–355, 2017.

R. Deepak, S. Uday, and D. Malath. Machine learning approach in stock market prediction. *International Journal of Pure and Applied Mathematics*, 115(8):71–77, 2017.

M. Nabipour, P. Nayyeri, H. Jabani, S. S., and A. Mosavi. Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. *IEEE Access*, 8(3):150199–150212, 2020.

Y. Zhai, A. Hsu, and S. K. Halgamuge. Combining news and technical indicators in daily stock price trends prediction. *n International symposium on neural networks*, page 1087–1096, 2007.