Advanced Stock Price Prediction Via Ensemble Machine Learning and Sentiment Analysis

Team: Jin Yan, Zhiwei Huang, Zhan(Sam) Shi **Project Mentor TA:** Jongkook Remy Kim

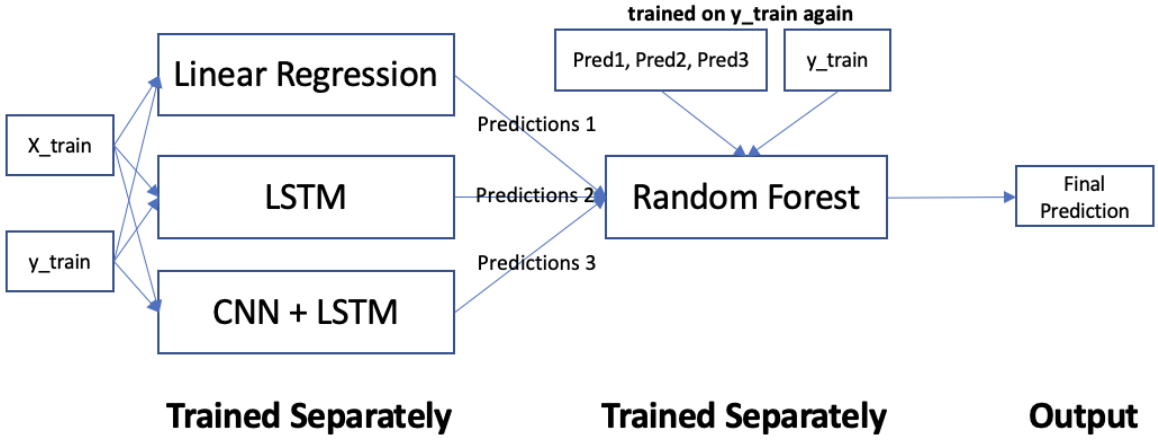
1. Abstract

The main objective of our project is to predict future stock prices and provide investors with a tool to make informed investment decisions, minimize risk exposure, and execute profitable trades. To achieve this, we implemented several models, including KNN, Linear Regression, LSTM, and Ensemble via Stacking models. We began with two simple models and then progressed to XGBoost and LSTM models, which resulted in significant improvements in our evaluation metrics. We then explored three different approaches: first, we extended the model to a CNN+LSTM model, and second, we introduced a new feature by extracting Indian news headlines and calculating sentiment scores for each trading day using FinBERT. With the new feature added, we retrained our LSTM model using a new set of inputs. This change can be visualized from

$$Input_{before} = \{S_1, S_2, \ldots S_{15}\} \text{ to } Input_{after} = \{S_1, S_2, \ldots S_{15}, Sentiment_{15}\}$$

where S's are stock prices of all 15 trading days and sentiment score of the Indian news headlines on 15th day. Finally, we combined the predictions from these models using a separate Random Forest. Stacking is a typical ensemble approach that aggregates the predictions from different models to boost the overall performance by building a new model. The schema using Random Forest can be visualized as

Stacking Pipeline Visualization



After comparing the results of each model, we concluded that the best model for one year of data is the LSTM model that incorporates sentiment scores from Indian news headlines. For five years of data, the best model is the Stacking model.

To better visualize the model performance, we made a chart to display the four metrics we used: R^2, MSE, MAE, and RMSE. Kaggle project proposed the LSTM model as the best fit. However, our LSTM + Sentiment Score model surpassed its performance by achieving a lower Mean Squared Error (MSE) and a higher R-squared value.

2. Introduction

Set up the problem:

- **Datasets**: We use the stock price data ranging from 2020 to 2021, and the Indian news headlines ranging from 2001 to 2022.
- **Inputs and Outputs**:
  In the LSTM model, each row of the input data has 15 columns, which represents a trading day. For each day, the model uses the stock prices from the previous 15 days as input for making predictions. Subsequently, we include an additional column representing the sentiment score derived from news headlines for each trading day. The output for all the models is the predicted stock price.

Implementation:

- LSTM: we utilize stock price data with a window size of 15.
- LSTM+sentiment score: we employ FinBERT, a sentiment analysis tool, to determine the sentiment score based on Indian news headlines data.
- CNN+LSTM: We implement a CNN-LSTM model, where each layer incorporates the TimeDistributed function to capture features for each temporal slice of data over time. Subsequently, the data is passed to the LSTM layers.
- ARIMA: After utilizing auto_arima, which automatically explores various parameter combinations to identify the model with the lowest information criterion value, we proceed to train the ARIMA model using the training data with an order of (0,1,0).
- XGBoost: We utilize stock price data with a window size of 1.
- Stacking: In an attempt to improve the accuracy of our predictions, we employed an ensemble method using Random Forest to combine the output of well-performing models mentioned above. This was done with the intention of creating a more robust prediction model.

Evaluation:

- Our stock price dataset spans from August 2020 to August 2021, and we utilize a 65:35 train-test split ratio. Therefore, we will train our model using 161 days of data and assess the accuracy of the stock price predictions using the remaining 87 days of data. The evaluation metrics considered include Mean Square Error and R^2 score.
- In order to overcome the problem of insufficient training data, we have extended the time range of the stock data to five years, thereby obtaining a larger number of training samples and a longer duration for testing. Since we have more data, we split our data into 80:20 ratios for the train test split.

3. How We Have Addressed Feedback From the Proposal Evaluations

During the first proposal evaluation, our mentor TA provided valuable feedback for our project. Specifically, he recommended that we include the timeframe of our datasets, incorporate an industry feature to enhance our predictions, and cite the relevant papers we reference in our work. Additionally, he advised us against using Linear Regression to capture the volatility of our model. To address his feedback, we have made several improvements to our project. Firstly, we

have specified the timeframe of our data as being from 2020-08-19 to 2021-08-17. Secondly, we have included Indian news headline data as an industry feature to provide additional context for our predictions. Thirdly, we have cited all relevant papers in our subsequent proposals. Lastly, we have removed Linear Regression from our project development, in line with our mentor TA's guidance.

4. Background

We have identified one prior work relevant to our project. The project is called Advanced Stock Pred using SVR, RFR, KNN, LSTM, and GRU. A major shortcoming of this work is that the authors used testing data as validation data, resulting in biased results. To address this issue, we will use separate validation data in our approach. Additionally, we leverage sentiment score and sentiment index as additional features to improve the accuracy of our predictions. These features were not considered in the prior work.

Advanced Stock Pred using SVR, RFR, KNN, LSTM, GRU:
https://www.kaggle.com/code/ysthehurricane/advanced-stock-pred-using-svr-rfr-knn-lstm-gru/notebook
Description: It is a sample notebook provided by the professor. We plan to build on it to start our research. We will use its EDA part to learn about the data itself and extend it to recreate the LSTM model used by the writer of this Kaggle report and explore the impact of the addition of the sentiment score and sentiment index.

5. Summary of Our Contributions

Implementation contribution:
To begin, we will perform data preprocessing in order to prepare for the implementation of our models. Next, we will implement, train, and evaluate 13 models for the two approaches listed below. Finally, we will compare the predictive accuracy of the models and determine the best model under each approach.

Evaluation contribution:
Our primary contribution to the evaluation of the implemented models is the comparison of evaluation metrics. We analyze the R-squared, MSE, MAE, and RMSE for each model and use these metrics to determine the relative accuracy of each model. Additionally, we provide plots for each model to further aid in the evaluation process.

6. Detailed Description of Contributions

6.1 Implementation Contributions

We have implemented and compared two approaches:

- Our first approach is using one year of data, from 2020-08-19 to 2021-08-18 for advanced stock prediction. This is the original dataset given by the professor. We implemented six models below:
    1. Using the KNN model
    2. Using the Linear Regression Model
    3. Using XGBoost

4. Using the LSTM model
5. Using the CNN model plus the Bidirectional LSTM model
6. Using LSTM model plus the Indian News Headlines
7. Stacking, the combination of KNN, LR, XGBoost and LSTM

- Our second approach is using five years of data, from 2016-08-19 to 2021-08-18 for advanced stock prediction. To address the issue of inadequate training data earlier, we have extended the time range of the stock data in order to acquire more training samples and a longer period for testing data. We implemented
    1. Using the KNN model
    2. Using the Linear Regression model
    3. Using XGBoost model
    4. Using the LSTM model
    5. Using the CNN model plus the Bidirectional LSTM model
    6. Stacking, the combination of KNN, LR, XGBoost and LSTM

Below is the chart showing the evaluation metrics for each model. The part above displays the evaluation metrics for one year of data, and the part below shows the evaluation metrics for five years of data. After comparing the R^2, MSE, MAE and RMSE of each model, we have come to the conclusion that the LSTM Model combining the Indian News Headlines has the best prediction accuracy for future stock prices.

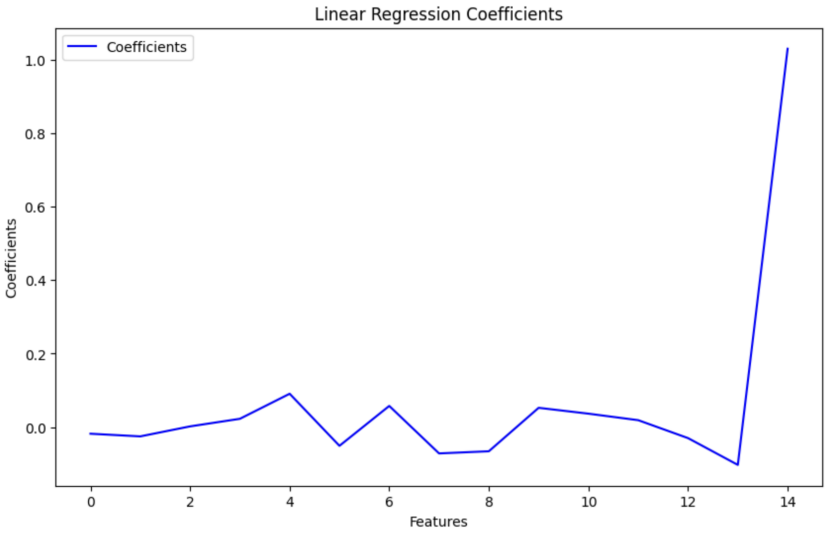
One Years Data: Aug 19th, 2020 - Aug 18th, 2021				
Models	Evaluation Metrics (Based on Test Data)			
	R^2	MSE	MAE	RMSE
KNN	0.574	3,355.356	42.393	57.925
Linear Regression	0.905	744.979	20.735	27.294
XGBoost	0.760	1,893.840	34.264	43.518
LSTM	0.799	1,580.654	31.409	39.757
CNN + Bidirectional LSTM	0.827	1,578.743	30.017	39.733
LSTM + Sentiment Score	**0.837**	**1,235.134**	**26.868**	**35.144**
Stacking (KNN+LR+XGBoost+LSTM)	0.751	1,962.396	35.465	44.299
Five Years Data: Aug 19th, 2016 - Aug 18th, 2021				
Models	Evaluation Metrics (Based on Test Data)			
	R^2	MSE	MAE	RMSE
KNN	-28.678	278,263.047	519.929	527.506
Linear Regression	0.855	1,357.188	28.295	36.840
XGBoost	-1.072	19,430.919	113.580	139.395
LSTM	0.634	3,435.474	50.450	58.613
CNN + Bidirectional LSTM	0.226	7,257.219	68.880	85.189
Stacking (LR+LSTM+CNN&LSTM)	**0.696**	**2,854.019**	**43.946**	**53.423**

6.2 Evaluation Contribution & Insights

Research Objective:
We aim to explore patterns in the stock prices of Reliance Industries Limited and build a stronger prediction model than all the existing projects on Kaggle.
We wish to create novel approaches to solve the research objective via the approaches of stacking ML models and incorporating sentiment scores calculated from Indian News headlines. Though we are taking a creative attempt, we still implemented a wide variety of different models to systematically compare and select the best one as our final model.
Using the Reliance stock prices as well as the Indian news headlines, we established KNN as our baseline model and evaluated its performance within both a one-year and five-year range. We observed that KNN provided superior predictions compared to random guessing, and all the models we implemented outperformed the baseline in terms of the four chosen evaluation metrics: R^2, MSE, MAE, and RMSE.

For the one-year stock prices of Reliance, LSTM plus sentiment scores calculated from Indian news headlines (LSTM + Sentiment Score) is the best model with an R^2 of 0.837 MSE of 1235.14. In the chart, the LSTM model is the existing best model so far on Kaggle, and our LSTM + Sentiment Score model outperformed it by a reduced MSE and higher R^2.
We managed to obtain a longer range of stock price data for Reliance. For this five-year stock prices dataset of Reliance, the Ensemble meta model via Stacking Linear Regression, LSTM, and CNN + LSTM is the best one exceeding all the other models we attempted. Through training a separate Random Forest on the predictions from all three models, we generate the best predictions for the stock on this extended dataset.

When selecting the top models, LSTM + Sentiment Score and Stacking (LR+LSTM+CNN&LSTM), we don't just rely on the four previously mentioned metrics. We also analyze the model's prediction schema. This is crucial because we discarded Linear Regression when we examined its predictions and discovered that it only forecasts the next day's stock price based on the previous day's price. By examining the coefficient plot below, we noticed that Linear Regression relies heavily on the 15th day's stock price. This renders the model ineffective in real-world scenarios where such a shortcut would lead to a lower MSE score.



In the end, we address our initial research objective perfectly. Given only a year of data as the existing Kaggle projects, with the limitation of the amount of training data, we use the approach of LSTM + Sentiment Scores which perform better at stocking prediction. Given an extended dataset range for five years, we managed to build an even stronger model using Stacking three separate trained models.

7. Compute/Other Resources Used

In our data analysis project, we utilized several resources including Colab, Kaggle Dataset, FinBERT, and Yahoo Finance. Colab, a cloud-based development environment, provided us with free access to resources such as GPUs, CPUs, and TPUs. We leveraged Kaggle Dataset, a platform that hosts a wide variety of datasets, to download and access financial data easily. We also employed FinBERT, a pre-trained sentiment analysis model for financial data, to analyze the sentiment of financial news articles. Additionally, we extracted stock prices from Yahoo Finance, which we used to train and evaluate our machine-learning models.

8. Conclusions

We had initially planned to implement the ARIMA model along with the LSTM model, and then build a Linear Regression Model on top of these two models. However, in the end, we decided not to pursue this approach. After using auto_arima for hyperparameter tuning, we got the order(p=0,d=1,q=0). This ARIMA model is also known as a random walk model. With p=0, the model assumes that the future values of the series are not dependent on its past values. With q=0, the model assumes that there is no systematic influence of past forecast errors on the current value. Therefore, we discarded the ARIMA model and explored more on LSTM.
Stock price predictions can have significant consequences for investors and the economy as a whole. It is crucial to ensure that prediction models are accurate and reliable, as inaccurate predictions can lead to financial losses or misleading investment decisions. However, the availability of accurate stock price predictions can potentially be misused for market manipulation or insider trading. Developers, users, and regulators should be vigilant to prevent any unethical or illegal activities that exploit prediction models for personal gain at the expense of others.

9. Roles of team members

Zhan(Sam) Shi: Sam proposed the idea of incorporating XGBoost into our project and took the lead in its implementation. Additionally, he suggested utilizing sentiment analysis to enhance the accuracy of our prediction. He also took on the task of performance evaluation. **Sam made that fabulous evaluation chart!**
Zhiwei Huang: Zhiwei played a vital role in data preprocessing and feature engineering. She applied various techniques to clean and transform the raw stock price data into a suitable format for machine learning algorithms. She has diligently structured the paragraphs and polished the wording, resulting in a visually appealing and well-crafted document.

Jin Yan: Jin was responsible for implementing and optimizing the neural network models. She designed and fine-tuned various architectures, including convolutional neural networks (CNNs) and long short-term memory (LSTM) models.

Link to the Github Project: https://github.com/Jin1179/519-FinalProj

(Exempted from page limit) Other Prior Work / References (apart from Sec 3) that are cited in the text:

1. T. Sidogi, R. Mbuvha and T. Marwala, "Stock Price Prediction Using Sentiment Analysis," 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 2021, pp. 46-51, doi: 10.1109/SMC52423.2021.9659283.
2. Hochreiter, S., & Schmidhuber, J. (n.d.). *LONG SHORT-TERM MEMORY*. https://doi.org/https://www.bioinf.jku.at/publications/older/2604.pdf
3. McMahon, Adrian. "Reliance Future Prediction of Close Price (LSTM)." Kaggle, 2021. Available at: [https://www.kaggle.com/code/adrianmcmahon/reliance-future-prediction-of-close-price-lstm/log]
4. Hariharan, Ramesh. "Reliance Stock Prediction using Linear Regression." Kaggle, 2023. Available at: [https://www.kaggle.com/code/rameshhariharan/reliance-stock-prediction-using-linear-regression]
5. Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. *ArXiv*. /abs/1908.10063
6. Fukushima, T. (1980). A comparative study of adaptive filters. IEEE Transactions on Acoustics, Speech, and Signal Processing, 28(4), 339-344. doi: 10.1109/TASSP.1980.1163420

Advanced Stock Prediction via Stacking LSTM and ARIMA

Team: Jin Yan, Zhiwei Huang, Zhan(Sam) Shi **Project Mentor TA:** Jongkook Remy Kim

1) Introduction

Set up the problem:

- The datasets we need for our model are:
    1. The stock price data ranging from 2020 to 2021
    2. News headlines ranging from 2003 to 2021collected via https://www.kaggle.com/datasets/therohk/million-headlines?resource=download
    3. Indian news headlines ranging from 2001 to 2022 collected via https://www.kaggle.com/datasets/therohk/india-headlines-news-dataset
    4. Stock Market Index Data India ranging from 1990 to 2022 collected via https://www.kaggle.com/datasets/debashis74017/stock-market-index-data-india-1990-2022?select=NIFTY+50_Data.csv
- The inputs and outputs of our eventual machine-learning model:
    - First Model: The input is stock price data. The output is a robust prediction for the stock price in 2022.
    - Second Model: The inputs are sentiment score and stock price data. The output is a robust prediction for the stock price in 2022.
    - Third Model: The inputs are sentiment index and stock price data. The output is a more reliable and accurate prediction of the stock price in 2022.
    - Finally, the inputs will be the three predictions and the output will be a final prediction of the stock price in 2022 (the best approach of the three).

Implementation:

- First Model: We will implement an LSTM model based on the stock price data and an ARIMA model based on the same dataset. Then using stacking to train a linear regression model on the predictions made by the LSTM model and the ARIMA model.
- Second Model: We will calculate the sentiment score using the news headlines data. Then we will implement an LSTM model based on the stock price data and the sentiment score. Next, using stacking to train a linear regression model on the predictions made by the LSTM model and the ARIMA model.
- Third Model: We will first calculate the “sentiment Index” using the Indian news headlines data and the Indian stock market index. Then we will implement an LSTM model based on the stock price data and the sentiment index. Next, using stacking to train a linear regression model on the predictions made by the LSTM model and the ARIMA model.
- Finally, output our final best model, chosen by the evaluation metrics discussed below.

Evaluation:

Our stock price dataset spans from August 2020 to August 2021, and we utilize a 65:35 train-test split ratio. Therefore, we will train our model using 161 days of data and assess the accuracy of the stock price predictions using the remaining 88 days of data. The evaluation metrics considered include Root Mean Square Error, Mean Square Error, Explained Variance Score, R^2 score, Mean Gamma Deviance Regression Loss, Mean Poisson Deviance Regression Loss, and Mean Absolute Percentage Error. In addition, we will leverage the auto_arima tool to automatically detect the optimal order for an ARIMA model.

2) How We Have Addressed Feedback From the Proposal Evaluations

TA feedback is positive but suggests addressing several key points. First, consider using stationary tests. Second, carefully select and train data, considering timeframe, intervals, and potentially adding an industry feature. Note that linear regression captures price trends, not volatility, while classification may offer better performance measurement. Finally, cite relevant papers, such as LSTM and ARIMA, and clarify their combination in the next milestone.

We have developed solutions to address each issue highlighted by our TA. Firstly, since financial data often lacks stationarity and exhibits high volatility, we will consider testing the effectiveness of the GARCH model as recommended by our TA. Secondly, our dataset spans from August 2020 to August 2021, which falls within the period of the COVID-19 pandemic. Although it is worth considering the pandemic's impact, the Indian stock market index indicates that the economic downturn caused by COVID-19 should have ended in August 2020. Therefore, we believe it is reasonable to dismiss concerns about the pandemic's effects on the data. Thirdly, we will not utilize linear regression for direct prediction. Instead, we will use linear regression to stack LSTM and ARIMA models. We will train a separate linear regression model on the predictions generated by LSTM and ARIMA and merge the two results into a single prediction.

3) Prior Work We are Closely Building From

A. Advanced Stock Pred using SVR, RFR, KNN, LSTM, GRU:
    a. https://www.kaggle.com/code/ysthehurricane/advanced-stock-pred-using-svr-rfr-knn-lstm-gru/notebook
    b. Description: It is a sample notebook provided by the professor. We plan to build on it to start our research. We will use its EDA part to learn about the data itself and extend it to recreate the LSTM model used by the writer of this Kaggle report and explore the impact of the addition of the sentiment score and sentiment index.
B. Stock market forecasting/ARIMA:
    a. https://www.kaggle.com/code/nageshsingh/stock-market-forecasting-arima/notebook
    b. Description: This is the sample notebook we will study to build an ARIMA model for our dataset to capture the trend, seasonality, and cycles within the dataset.
C. Stock Sentiment Analysis using News Headlines:
    a. https://www.kaggle.com/code/rohit0906/stock-sentiment-analysis-using-news-headlines
    b. Description: Based on the same methodology used in the Kaggle project, we intend to apply a similar approach to the Indian News Headline dataset. We aim to make predictions on the sentiment index, which will determine the direction of the market index for the following day. To assess its effectiveness, we will incorporate this index into our existing dataset and evaluate its performance.

4) Contributions

The key question we are trying to answer is how can we predict future stock close prices. We are thinking about integrating both historical stock prices with daily news headlines. The baseline for our approach is the Long Short Term Memory model, which was first proposed by Hochreiter and Schmidhuber in 1997. The LSTM model is a modified version of the RNN method.

The datasets we used are

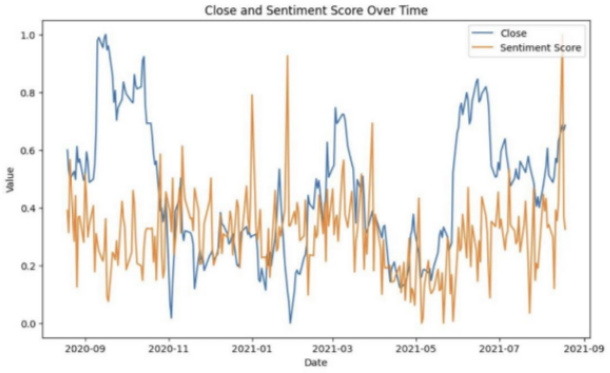
1) Reliance Industries Limited Stocks 2020-21
2) India Headlines from India news dataset

We try several ways to do sentiment score calculation

1) Bag of Words
   This is the technique used to determine the sentiment of a given text by counting the occurrence of certain words in the text. However, this appraovet does not take into account the context of the words.

2) DistilBERT base uncased finetuned SST-2 model

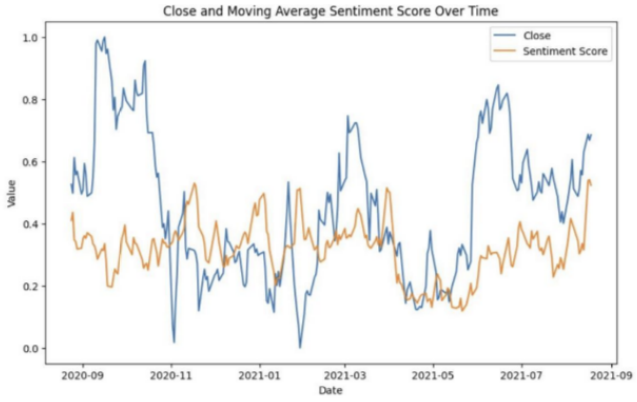
   This is a transformer based deep learning model. To obtain the sentiment score, the model is tuned on a labeled dataset of sentiment analysis and then use the fine-tuned model to predict the sentiment of new data.



   After gaining this sentiment score, we plot these two data. From the plot, we see that the sentiment score oscillates very frequently. Therefore, we want to find a way to smooth this data.

3) Moving Average

   In order to smooth the oscillating data, we apply a moving average filter by averaging the values of close points. We choose `window_size = 5` and calculate the average of the values within the window



See the plot above with moving average sentiment score. It is worth noting that the overall trend of close stock price and news headline sentiment score after March 2021 seem corresponds.

Then, we build the model based on the previous model.

1) Add sentiment scores based on new headline for each trading day
2) Scale the stock close price to range (0,1)
3) Train our new LSTM model with the sentiment score included
4) Use `RandomizedSearchCV` to tune hyperparameters
5) The model we have is

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 16, 128)           66560

 dropout (Dropout)           (None, 16, 128)           0

 lstm_1 (LSTM)               (None, 16, 128)           131584

 dropout_1 (Dropout)         (None, 16, 128)           0

 lstm_2 (LSTM)               (None, 128)               131584

 dropout_2 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 1)                 129

=================================================================
Total params: 329,857
Trainable params: 329,857
Non-trainable params: 0
_________________________________________________________________
```

Result:

```
Train data MSE:  1345.5295873946397
Test data MSE:  1955.8632327405471
Train data R2 score: 0.8976590516703389
Test data R2 score: 0.7423319013387304
```

Our current model has lower performance compared to the model that incorporates sentiment score. However, despite its suboptimal performance, our attempt provides valuable insights. For example, we observe a strong correlation between the sentiment score of news headlines and the stock price after March 2021 in the second plot. To validate this correlation, we plan to gather more recent data. Additionally, we aim to collect news articles specific to this stock for improved model construction.

5) Risk Mitigation Plan

1. **How will you build a minimum viable project within the remaining time?**
   To develop the most effective models within our given timeframe, we will identify the core features necessary and focus our efforts on these essential features due to limited time. We will establish a timeline and break down development process for the three models into smaller tasks, with deadlines for each. Prioritizing the first two models as outlined above, we will ensure they are well-constructed before proceeding with the third model.

2. **What if you find that you need too much computing?**
   Based on the size of our dataset, which consists of only one year of data, we do not anticipate encountering any issues that could impede the project's progress. Furthermore, we have run the LSTM model on the dataset, and it only takes 2-3 minutes, which is an acceptable timeframe for processing.

3. **Will you try your algorithm on different, simpler data, such as a “toy” synthetic dataset you generated?**
   We do not believe testing our algorithm on simpler data would be highly beneficial since financial data is highly volatile and our model may not perform well on any other stock. However, it is worth testing whether our model effectively predicts stock prices for the same company during other periods.