Stock price prediction

Pushkal Shrivastava

ABSTRACT: In this project we explore the utility of machine learning techniques in the prediction of daily stock prices. We compare the performance of various prediction models - 1. Moving average models, 2. Recurrent Neural Networks (RNN) and 3. RNN with sentiment data. As a part of this project, we webscrape google news to obtain data for sentiment analysis. We create a Bidirectional Encoder Representations from Transformers (BERT) based classification model, Financial Sentiment Analysis using BERT (FASB), and train it on Financial Phrasebank to achieve an accuracy of close to 95%. We use FSAB to on the webscraped news to generate sentiment data for model 3. Finally, we compare the performance of all the models.

Contents

1 Introduction

The aim of this project is to explore the utility of machine learning techniques, especially neural networks, in prediction of stock prices. We create multiple models to predict the daily closing price of a stock and compare their performance. The first two models are moving average models and don't require machine learning. The third model utilizes a recurrent neural network while the final model additionally uses sentiments derived from news headlines pertaining to the stock in question (Microsoft - MSFT). In order to obtain sentiment from news headlines, we create a model for Financial Sentiment Analysis using BERT (FSAB) language model. We scrape news headlines from Google news search to generate data for sentiment analysis. We find that machine learning techniques leads to improved performance in stock price prediction.

In section 2 we describe FSAB, the BERT based model for predicting sentiment from a financial text. In section 3 we create and examine the performance of various models for stock price prediction. Finally, in section 4, we summarize the results and discuss future directions.

2 Financial Sentiment Analysis using BERT (FSAB)

2.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art pre-trained language model developed by Google. It is designed to understand and generate human-like text by capturing the contextual relationships between words. BERT has achieved remarkable success across a wide range of natural language processing (NLP) tasks, including text classification, named entity recognition, sentiment analysis, question-answering, and more.

BERT is trained on a large corpus of unlabeled text, allowing it to learn powerful representations of words and sentences. During the pre-training phase, BERT learns to predict masked words within sentences and to understand the relationships between sentences in a given context. This unsupervised pre-training enables BERT to capture a deep understanding of language and produce rich contextual embeddings.

2.2 Fine tuning BERT for financial sentiment analysis

We create a sentiment classification model using TensorFlow. We load the BERT-uncased model as a single layer in our classification model and add a classification layer. We include dropout layers to prevent overfitting. The model architecture is given in fig 1. Our model, Financial Sentiment Analysis using BERT (FSAB), contains 109,581,059 parameters.

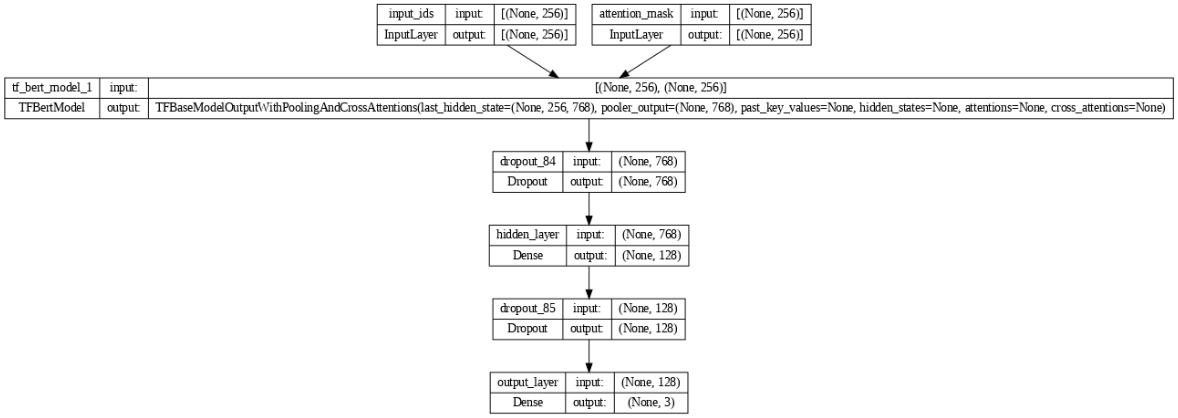
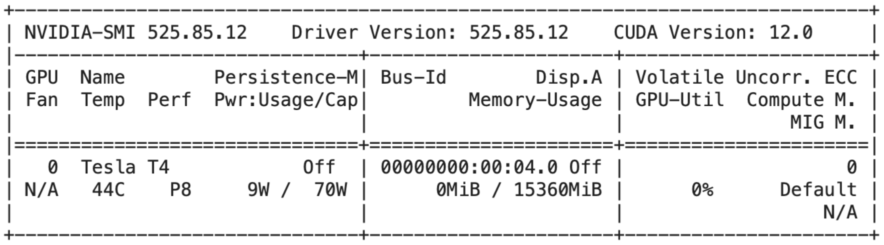

input_ids | input: | [(None, 256)]
InputLayer | output: | [(None, 256)]
attention_mask | input: | [(None, 256)]
InputLayer | output: | [(None, 256)]
tf_bert_model_1 | input: | [(None, 256), (None, 256)]
TFBertModel | output: | TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 256, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)
dropout_84 | input: | (None, 768)
Dropout | output: | (None, 768)
hidden_layer | input: | (None, 768)
Dense | output: | (None, 128)
dropout_85 | input: | (None, 128)
Dropout | output: | (None, 128)
output_layer | input: | (None, 128)
Dense | output: | (None, 3)

Figure 1: The architecture of Financial Sentiment Analysis model using BERT layer

INPUT: Our model can predict sentiment of a text of variable length subject to a constraint on the maximum length. Since our model utilizes the BERT language model, our input must be a valid BERT input. FSAB accepts two inputs - input_ids and attention_mask. To obtain the input_ids, we first tokenize a text using the BERT-uncased tokenizer. We prepend and appended the tokenized sequence with classification and separator token respectively. Using padding/truncation, we ensure that the input_ids have length 256. To help BERT distinguish between the actual sequence and padding, we pass the second input, the attention_mask.

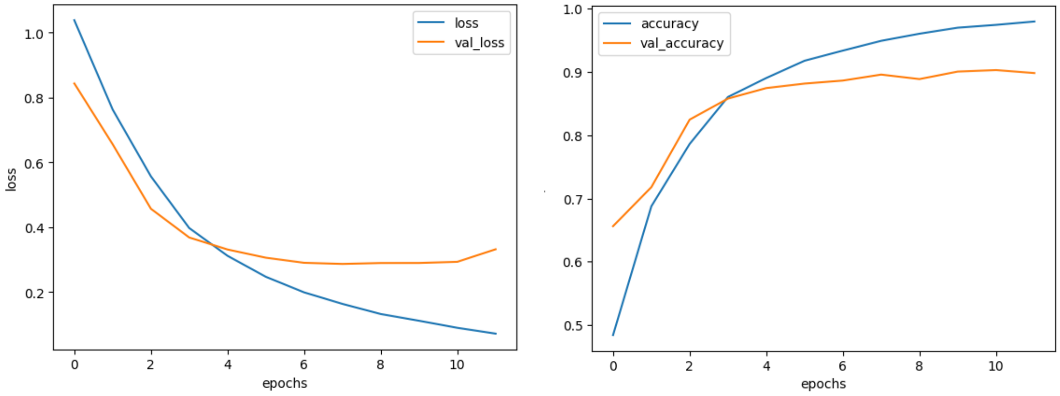
OUTPUT: The FSAB outputs a sequence of three numbers which are the probabilities that the sentiment of the input text is negative, neutral and positive.



```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.85.12    Driver Version: 525.85.12    CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   44C    P8     9W /  70W |      0MiB / 15360MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
```

Figure 2: NVIDIA System Manager Interface output

TRAINING: The model is trained on the Financial Phrasebank[1] dataset. To train our model efficiently, we used Google Colaboratory with a GPU runtime, see fig. 2. We split the dataset into train (75%), validation (12.5%) and test data (12.5%). We create TensorFlow datasets to facilitate parallelization. We used SparseCategoricalCrossEntropy as the loss function. See fig. 3 for the loss and accuracy curves.



(a) Loss curve - FSAB

(b) Accuracy curve - FSAB

Figure 3: FSAB loss and accuracy curves

2.3 Performance

The FSAB model correctly classified the input text with an accuracy close to 95%. The probability that the model classified an input of negative sentiment as positive and vice-versa is around 2.5%. See table 1 for the confusion matrix on test data. The trained model can now be used to predict the sentiment of a financial text. For example, running our model on the text ‘the profit declined by 50% compared to last year’ yields a negative sentiment probability

[1] labeled dataset of financial statements available at www.huggingface.co/datasets/financial_phrasebank

	Negative	Neutral	Positive
Negative	49	2	0
Neutral	2	248	4
Positive	4	6	107

Table 1: Confusion Matrix

of around 99.4%. On the other end of the spectrum, the text 'the profit in 2023 was 200% higher than that in 2022' yields a positive sentiment probability of 99.7%.

3 Stock price prediction models

In this section we will develop three models to predict daily stock prices using historical data. Our time series model will predict the next day closing price using past closing prices. The description of the three models are as follows.

1. Moving Averages Models - We will use past 5 data points to predict the stock price. We experiment both with simple moving averages (SMA) and exponential moving averages (EMA). In SMA the all past values are given equal weights, while in EMA, average is computed by giving larger weights to recent data points in the time series.

2. Recurrent Neural Network - In the second model, we will construct a recurrent neural network to predict the stock prices.

3. Recurrent Neural Network with sentiments - In this model, we will construct a recurrent neural network and also utilize daily sentiments to predict the stock prices. To obtain sentiments data, we will webscrape Google news for each day and compute the sentiment using FSAB described above.

3.1 Moving average models

In the simple moving average model, the stock price at next time instance is predicted to be the average of last N stock prices.

$$SMA_{N+1} = \frac{1}{N}\sum_{i=1}^{N} S_i \tag{3.1}$$

Whereas, in the case of exponential moving average the prediction is

$$EMA_{N+1} = \frac{2}{N+1}S_N + \frac{N-1}{N+1}EMA_N \tag{3.2}$$

We predict the closing price of Microsoft (MSFT) with $N = 5$. The root mean square error for the simple moving average model and the exponential moving average is 9.25 and 8.48 respectively. Fig. 4 shows the actual stock price and the predictions using moving averages.

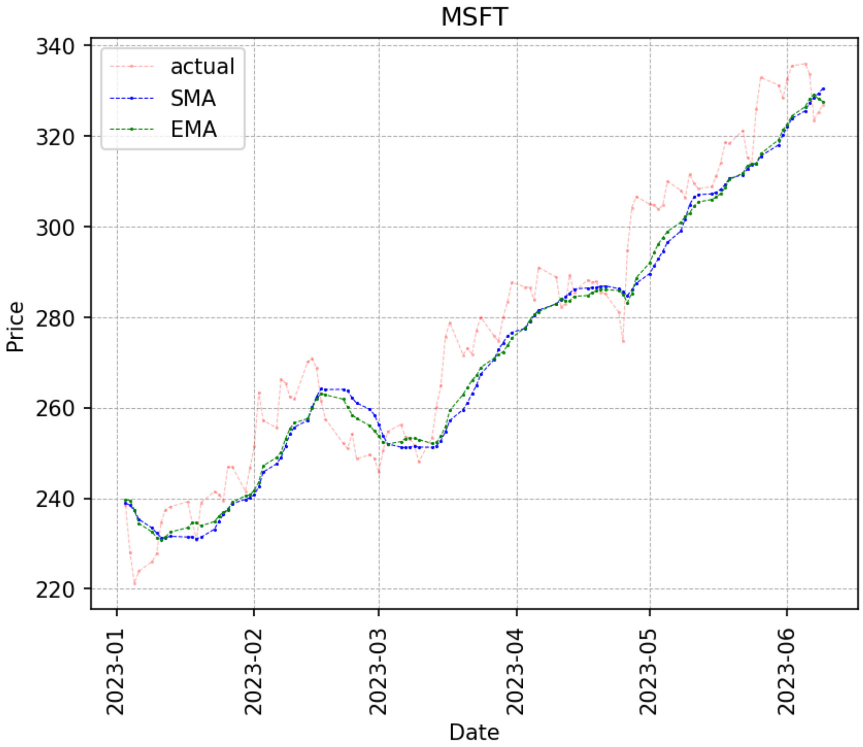


Figure 4: Stock price prediction - Moving average models

3.2 Recurrent Neural Network

A Recurrent Neural Network is often useful for modeling time series data. To model the stock price, we first assume that the stock price receives contribution from a deterministic and a stochastic term.

$$S(t) = S_0\, e^{\alpha t} + \eta(t), \tag{3.3}$$

where the first term is a deterministic drift and encodes the long time behavior, and the second term is a stochastic contribution that depends on short time market features. We estimate S_0 and α using a linear fit of $\log(S)$ as a function of t, see fig. 5. We model η using a recurrent neural network.

Model: After experimenting with the model architecture, we determined that the best performance was achieved when last 5 stock prices were used to predict the future stock price. To counter against the vanishing gradient problem, we initially experimented with an LSTM (Long Short Term Memory) model. However, since only last 5 data points were used for prediction, we found that a recurrent neural network performed better than LSTM. To prevent over fitting, we used L2-Kernel regularization, dropout and recurrent dropout. See fig. 6 for model architecture.

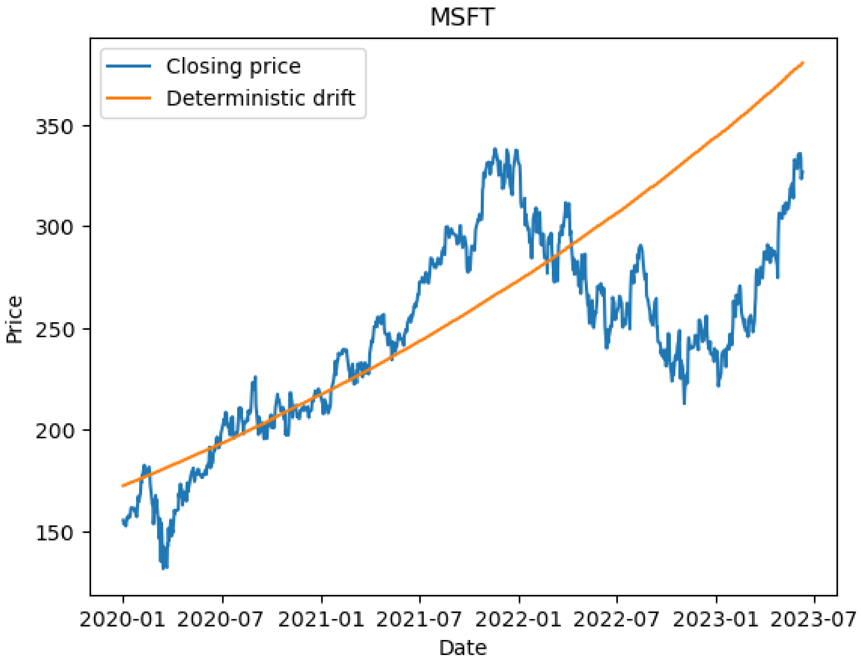


Figure 5: Long term behavior of stock price

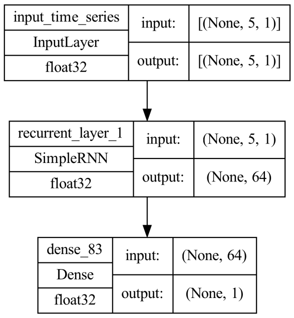


Figure 6: Architecture of model for predicting stock price using a recurrent neural network

Training and performance: The model was trained on daily closing stock price of Microsoft (MSFT) from January 2020 to December 2022 and performance was evaluated on stock prices from January to June 2023[2]. The root mean squared error on test data was close to 5.9, which is around 35% lower than moving average models. Fig. 7 shows the plot of actual stock prices and predictions of RNN model.

[2] Recall that we use RNN to predict η. For a better fitting of the model, we scaled the stock prices using StandardScaler from sklearn. To evaluate performance, we rescaled the output of our model and added the deterministic drift to obtain the final stock price.

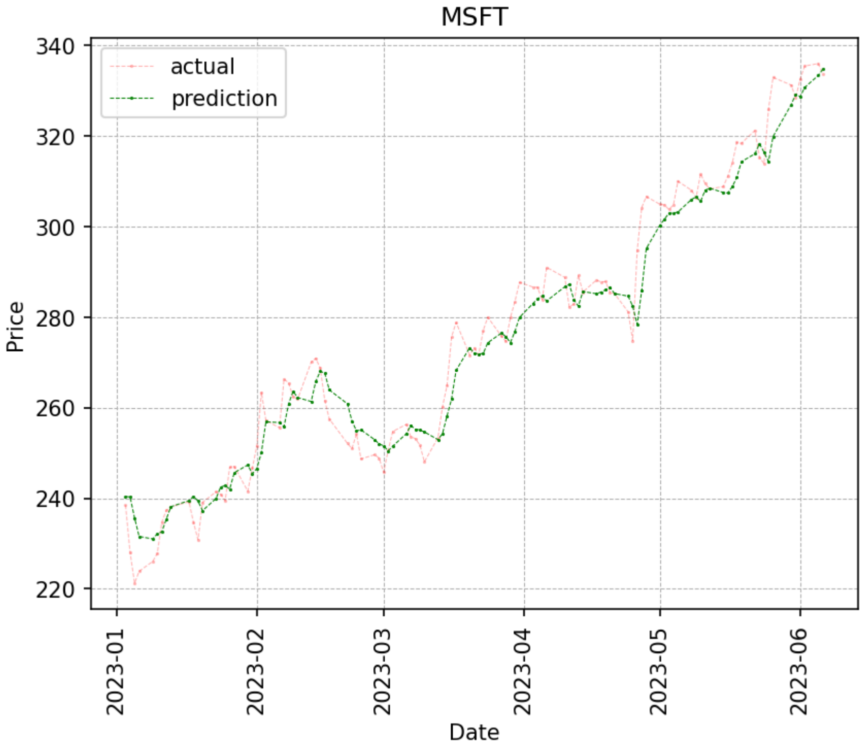


Figure 7: Stock price prediction - Recurrent Neural Network

3.3 Recurrent Neural Network with sentiment

In the final model of this project, we attempt to combine the recurrent neural network with daily sentiment data to predict the stock price. We compute the sentiment of webscraped google news headlines with search term “microsoft news”. We used the FSAB model, see sec. 2, to compute the probabilities that the sentiment is negative, neutral and positive. We then create a dataset with stock prices and sentiments. Our model takes past 5 stock prices and the sentiments as data to predict the future stock price. We now elaborate on the various steps involved.

Webscraping: We used the BeautifulSoup library to automate collection of Google news headlines. We searched Google news website for articles on “microsoft news” published on a given date. We scraped the headlines from the first page of search results, appended the headlines to a file and then varied the date. This automated process allowed us to collect top news headlines pertaining to Microsoft on each day from January 2020 to June 2023.

Sentiment analysis: We used the trained FSAB model, sec. 2, to obtain the sentiments of the headlines collected earlier. Once again, we saved the sentiment data to a file for later use.

Model: Our model takes the last 5 stock prices and average sentiment probabilities as input and outputs the prediction for next stock price. It works in two steps. First, our model uses a recurrent neural network with the last 5 stock prices as input and gives an intermediate prediction for the stock price. In the next step, the model uses this intermediate prediction and the sentient probabilities to output the final prediction for the stock price. Once again, we use various regularization techniques discussed earlier to prevent over fitting. The model architecture is shown in fig. 8.

Training and performance: As with earlier models, this model was trained on daily closing stock price of Microsoft (MSFT) from January 2020 to December 2022 and performance was evaluated on stock prices from January to June 2023. We found that there was negligible correlation between sentiment and stock prices, see fig. 9. The Pearson correlation coefficient between stock price movement and sentiment was less than 0.1. However, the correlation coefficient became -0.37, for instance, when we restricted to negative sentiment probability larger than 0.5. It appears that strong sentiment does seem to be mildly correlated with stock price movement. However, it is unclear if including sentiments would enhance the performance. In fact, we found that the root mean squared error for this model (on test data) was approximately 8.13. This model performed better than the moving average models, but worse than the simple RNN model. See fig. 10.

4 Summary and discussions

In this project we analyzed the utility of machine learning techniques in stock price prediction. We developed a machine learning model to predict the sentiment of a financial text. We also developed several models to predict the daily closing price of a particular stock. We found that our Neural Network based model improved stock price prediction by about 35%.

4.1 Financial sentiment analysis

To predict the sentiment of a financial text, we added a classification layer to the pre-trained Bidirectional Encoder Representations from Transformers (BERT) language model. We fine-tuned our model (FSAB) on the Financial Phrasebank database and evaluated the performance. We found that our model's accuracy on test data was close to 95%. See sec. 2 for further details. We later used this model to predict the sentiment of news headlines pertaining to the stock of interest.

4.2 Stock price prediction

In sec. 3, we studied the performance of 4 models to predict the stock price. The first two models were Simple Moving Average and Exponential Moving Average models. The third model implemented a Recurrent Neural Network (RNN) while the final model also incorporated sentiment data apart from RNN. To obtain sentiment data, we webscraped Google news using the BeautifulSoup library. For a comparison of the performance of these

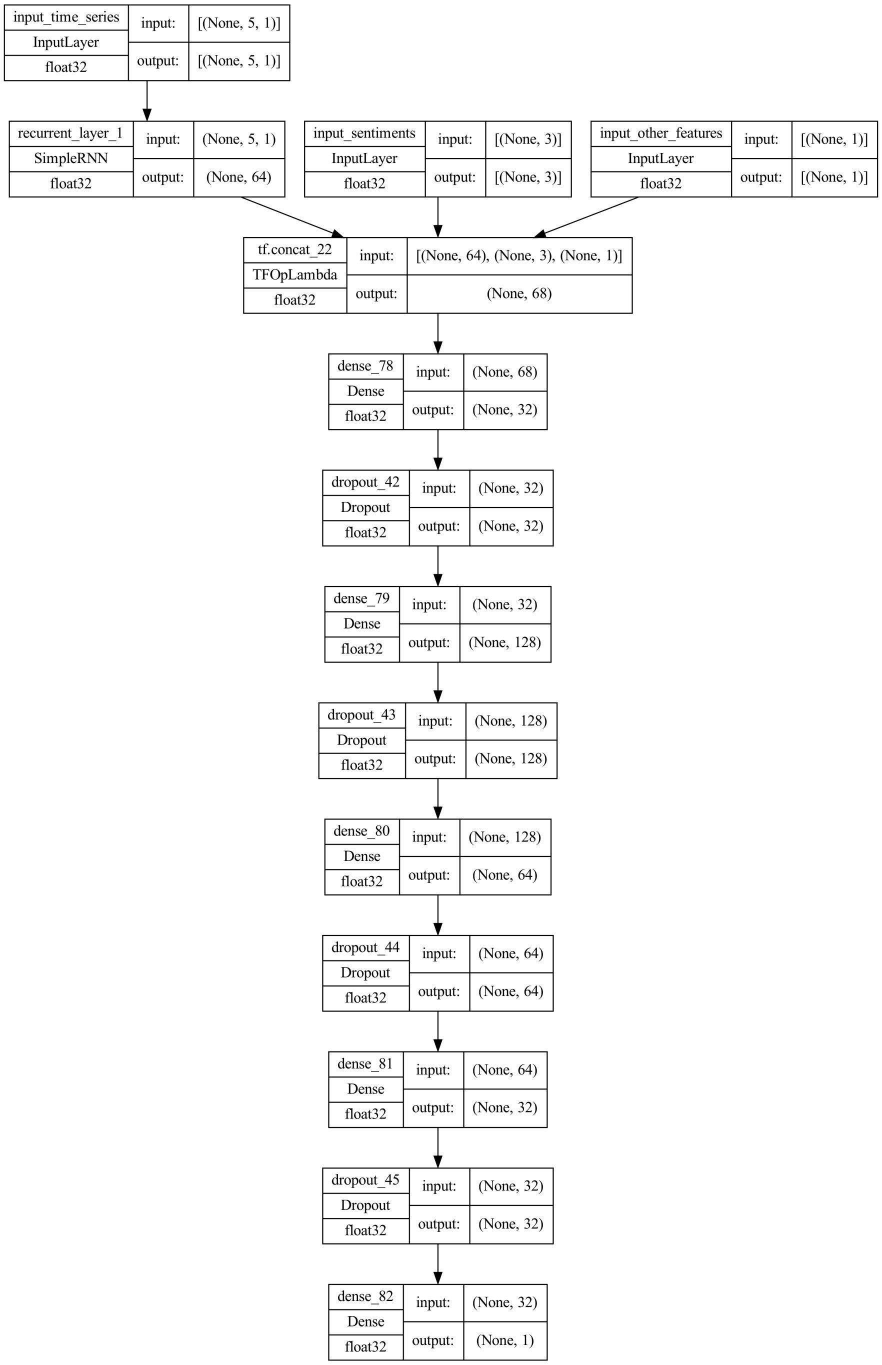


Figure 8: Architecture of the RNN with sentiment model. This model can be readily expanded to include more features. However, for the purpose of this project, we only work with closing price and sentiments.

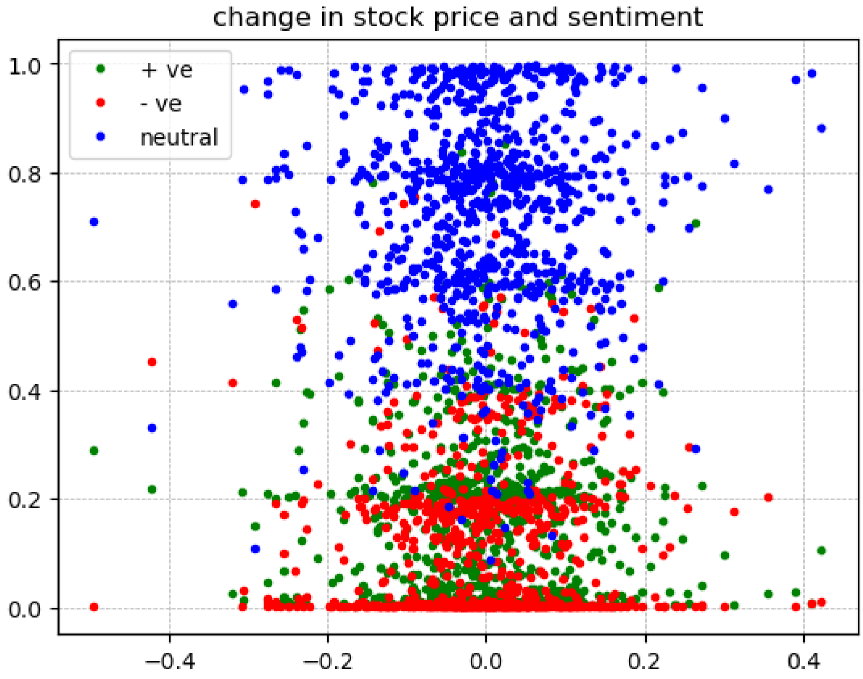


Figure 9: Sentiment probabilities and daily closing stock price movement

Model	SMA	EMA	RNN	RNN + sentiment
RMSE	9.25	8.48	**5.85**	8.13

Table 2: Comparison of stock prediction models

models, see tab 2. All of the models used last 5 data points to make predictions. At first glance, it may be surprising that the model with sentiment doesn't yield the best performance. However, this can be attributed to the quality of data collected for sentiment analysis as well as the low frequency of stock price data.

4.3 Future directions

It would be interesting to explore the role of other data, such as fundamental data and macroeconomic data, in stock price prediction since our model can be readily modified to accommodate additional features. Another direction is to study the performance of various models at a different time scale. For instance, one can try to predict stock prices at higher frequency, say every minute or even every second. It is reasonable that RNN model might yield even better results in shorter timescales. Finally, it would be interesting to explore the relation between sentiment and stock price movement at different timescales. However, this require collection of more frequenct and robust news/opinions data which is beyond the scope of this work.

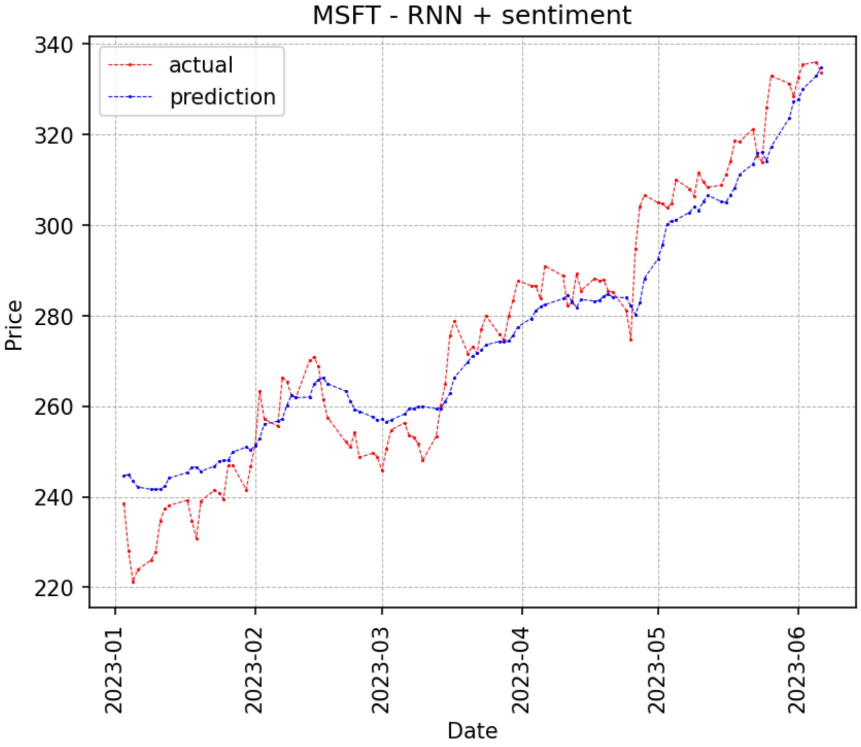


Figure 10: Stock price prediction - RNN based model with daily sentiments