DATA SCIENCE INTERNSHIP REPORT (TEAM B)

TOPIC – Intraday Stock Trading Prediction using Deep Learning

Team Members:

ISHA SINGH

AKASH KADAM

VYSHNAVI VENNAKOTA

SAMPURNA LAL

KARTIK GUPTA

INDEX

ABSTRACT

The ability to precisely predict the price movement of stocks is the key to profitability in trading. In light of the increasing availability of financial data, prediction of price movement in the financial market with deep learning has become a topic of interests for both investors and researchers.Insights about price movements from the models could help investors make more educated decisions. The most important need of any stockholder is to know the fluctuations of stock prices in the financial market. This is an important prediction as it facilitates their decision in investing or reinvest in the stock market.
This report focuses on how the predictive power of deep learning models can reap financial benefits for investors who trade based on future price prediction. The project focuses on predicting the next-minute price movement of MSFT and NIFTY 50 using Gated Recurrent Unit (GRU) model which is implemented in this project to predict the stock trends.

INTRODUCTION

In the financial market world, with great advancement of information technology, stocks represent one of the key assets that people are counting upon. Most people invest their hard earned money in financial market with good knowledge in equity market and statistics. There are different kinds of investment in the stock market such as intraday, short term, medium term and long term investments. There are people who believe in intraday and considered as risk takers. They are the key players in the financial market of the intra-day trading. Majority of people, with sound knowledge of the market, statistics and lot of 'gut' feelings, are investing their hard-earned money into company shares. And then we have people, who are termed as 'Risk Takers', who believe in the understanding of commerce, current affairs, and mathematics. They are the major players in the world of intra-day trading. One can categorize it to be one of the riskiest investments in stocks market, quite equivalent to gambling. Therefore, the ability to precisely predict the price movement of stocks is the key to profitability in trading. Many investors spend time actively trading stocks in hope of outperforming the market, colloquially referred to as a passive investment.

In this project, we aim to focus on making short term price movements prediction using the time series data of stock price. Such predictions will then be used to generate short-term trading strategies to capitalize on small price movements in highly liquid stocks. This project makes use of Gated Recurrent Unit (GRU) to predict stock trends. Here the Model is trained by the usage of actual historical data.

DATASET AND FEATURES

This project makes use of 2 datasets.

The first one contains Microsoft's (MSFT) stock data for the last 35 years (from 1986 to date) with daily interval.

The second one contains Nifty 50 stock data for 3 months (from 01/01/2021 to 31/03/2021) with an interval of 1-minute.

Original format of the dataset: CSV

A brief explanation of every column in the dataset is as follows:

- **Date**: in format: YYMMDD
- **Time:** in format: HH:MM
- **Open:** Open Price of stock i.e. The price at which stock opened
- **Close:** Close price of stock i.e. The price at which stock closed
- **High:** The highest price the stock touched
- **Low:** The lowest price the stock touched
- **Adj Close:** adjusted close price adjusted for both dividends and splits.
- **Volume:** the number of shares that changed hands during a given day

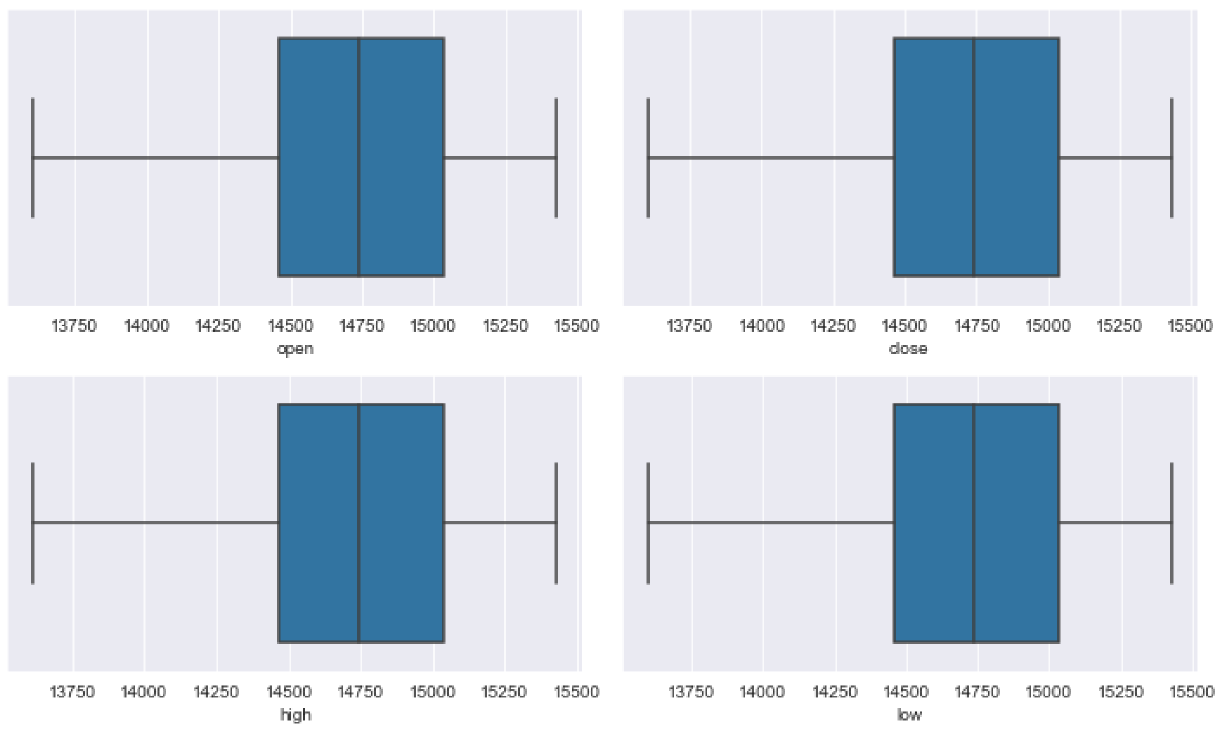
EXPLORATORY DATA ANALYSIS (EDA)

EDA refers to exploration (or explanation of the data) of the given data for the future usage. It is a good practice that we need to understand our data and gather as many insights out of the data.

EDA on DataFrame Dataset

1. **Detection of missing values:**
   No missing values were found in the dataset.

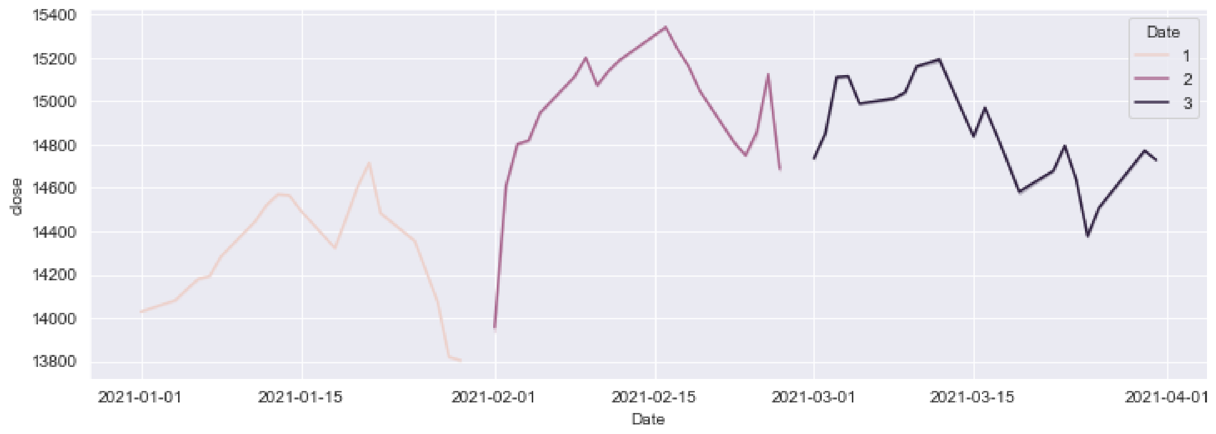
2. **Detection of outliers:**
   No outliers were found in the dataset.



As we can see from the above graph, **there are no outliers present.**

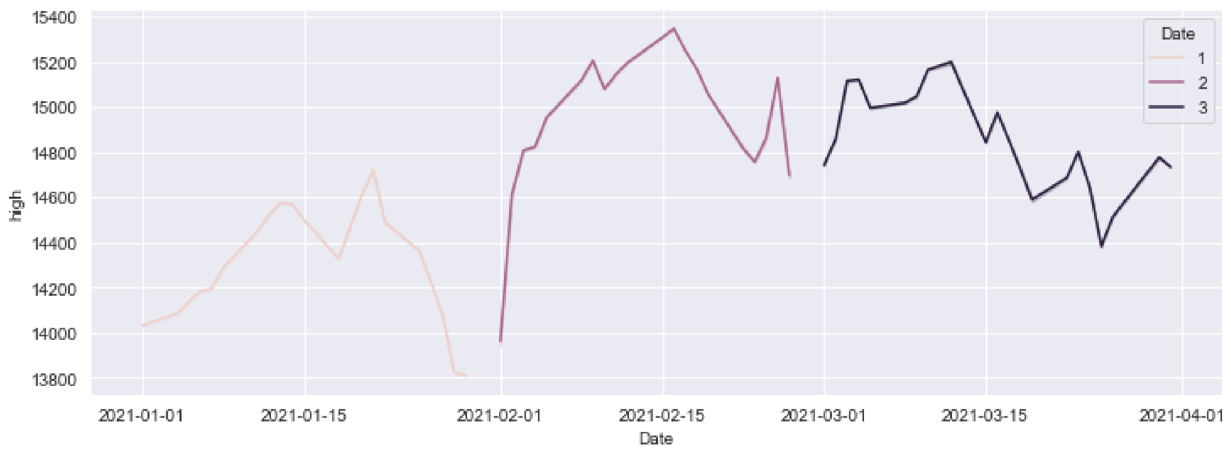
3. Line Plots:

Plot 1: Date VS Close

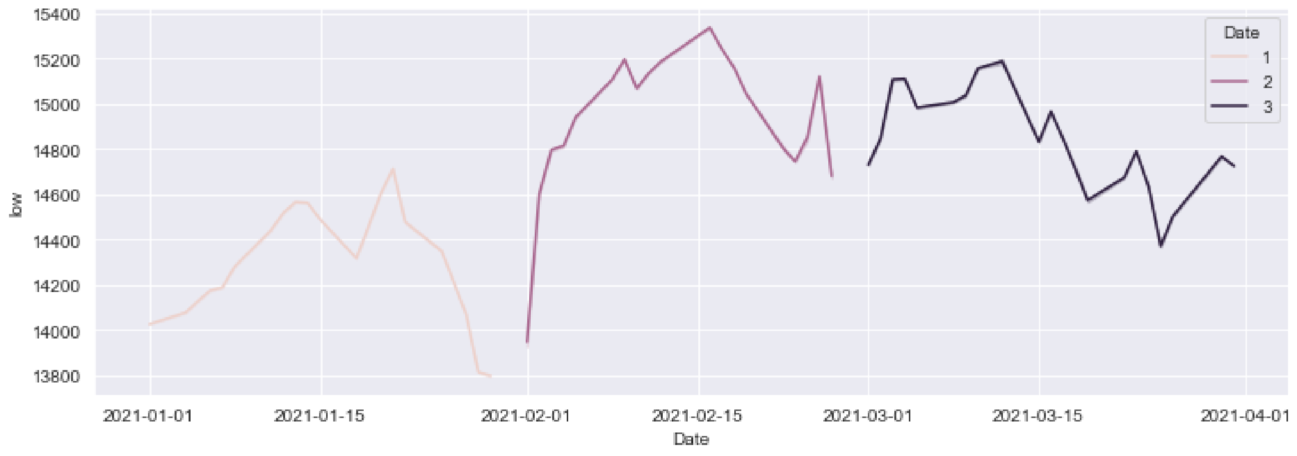


From the above graph, we can infer that the closing price reaches nearly two peaks during the month. Once near to 15th of the month and another one few days after 15th. It is also observed that there is a fall in the closing price in the end of the month. But it starts improving in the beginning of the next month.
Similar trends can be seen in the plot of high and low prices.

Plot 2: Date VS High

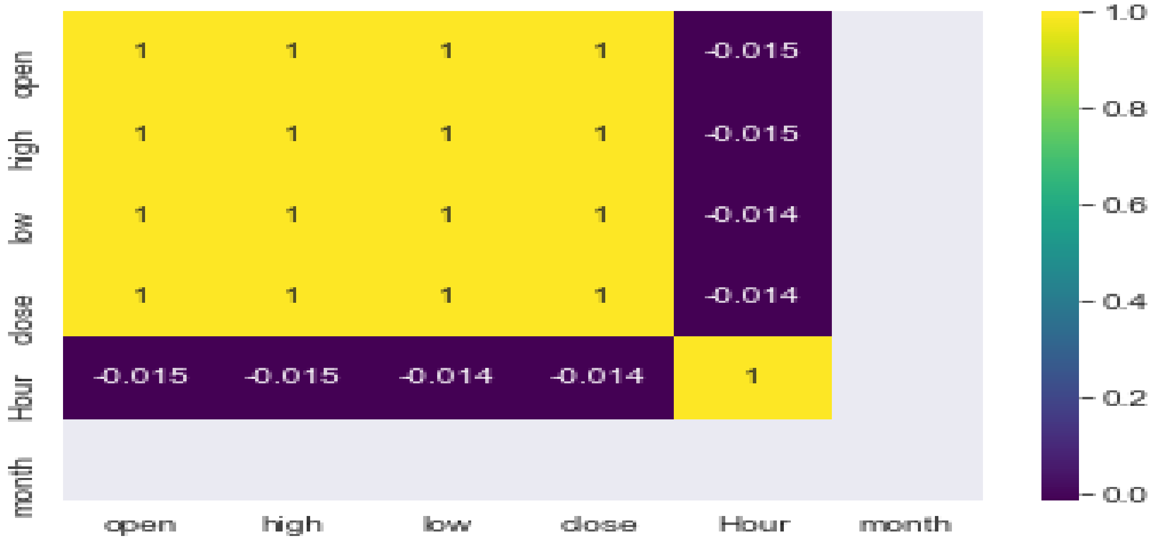


Plot 3: Date VS Low



4. Heat Map or Correlation plot:

Correlation checks for the linear dependency of any two variables among themselves. The value of correlation lies between 1 and -1.



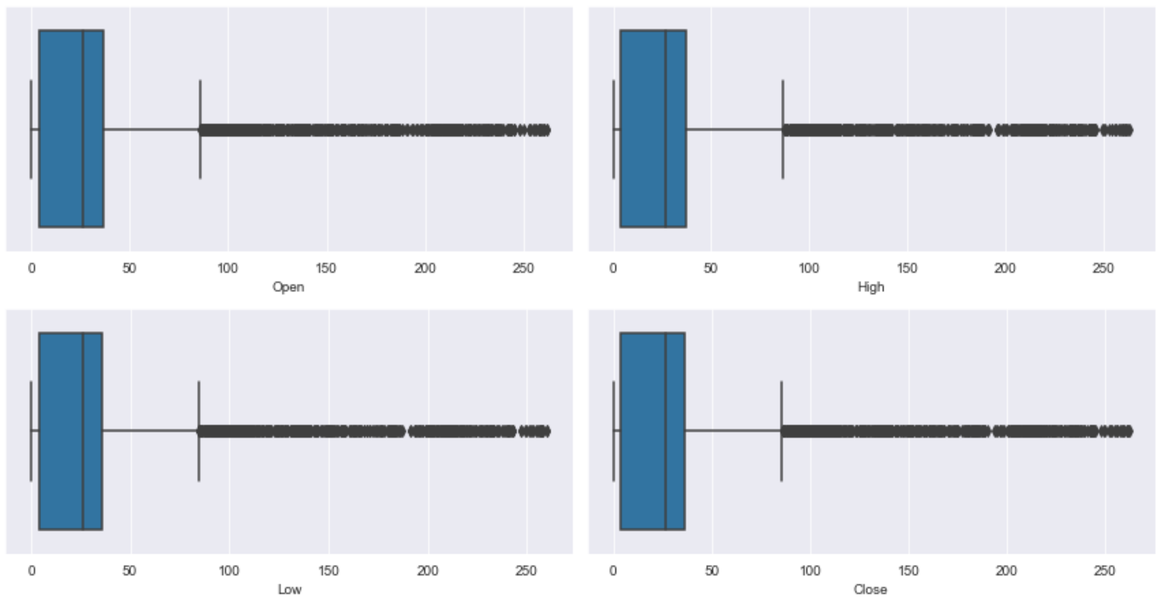
	open	high	low	close	Hour	month
open	1	1	1	1	-0.015	
high	1	1	1	1	-0.015	
low	1	1	1	1	-0.014	
close	1	1	1	1	-0.014	
Hour	-0.015	-0.015	-0.014	-0.014	1	
month						

From the heat map we can observe that Open, High, Low, Close are positively correlated to each other. The value of Open, Close, High and Low are also positively correlated to the corresponding year.

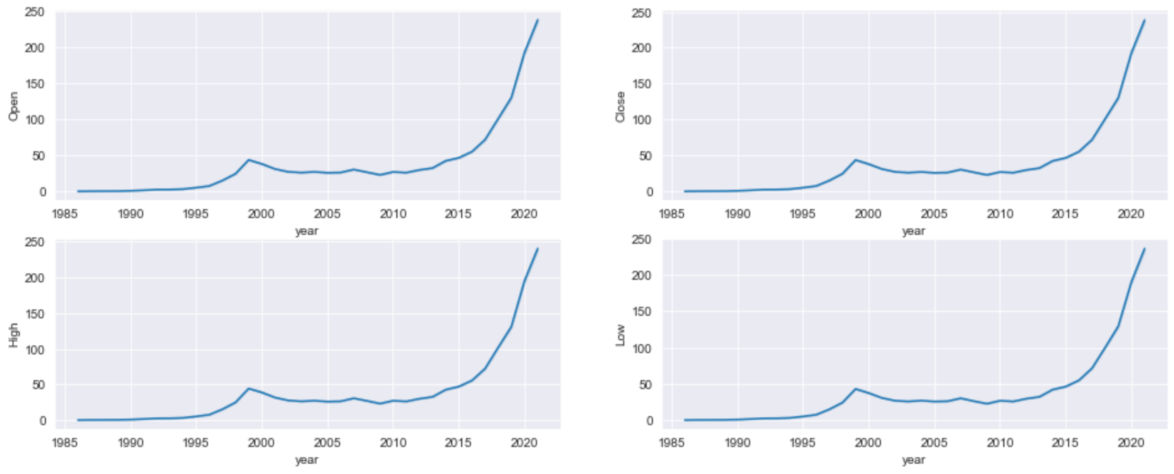
EDA on MSFT Dataset

1. **Detection of missing values:**
   No missing values were found in the dataset.

2. **Detection of outliers:**
   In this dataset, a lot of outliers were found, but the removal of these outliers will result in loss of a lot of data from the dataset. Hence, we retain the outliers.

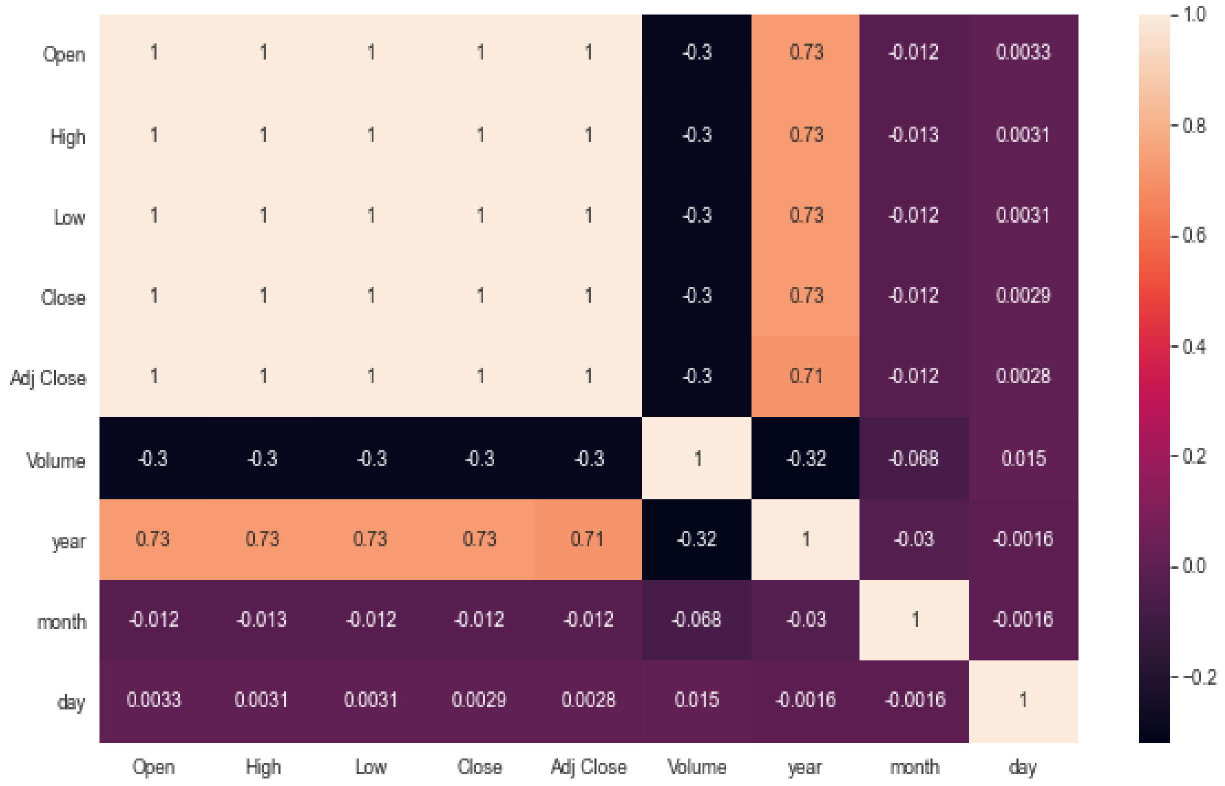


3. **Line Plot:**



We can infer from the line plot that the stock value kept on increasing as we can see the price of stock was near to zero in 1986 and gradually kept on increasing. Year 1998, 1999, and 2000 were better compared to previous year.

4. Correlation plot or Heat map:



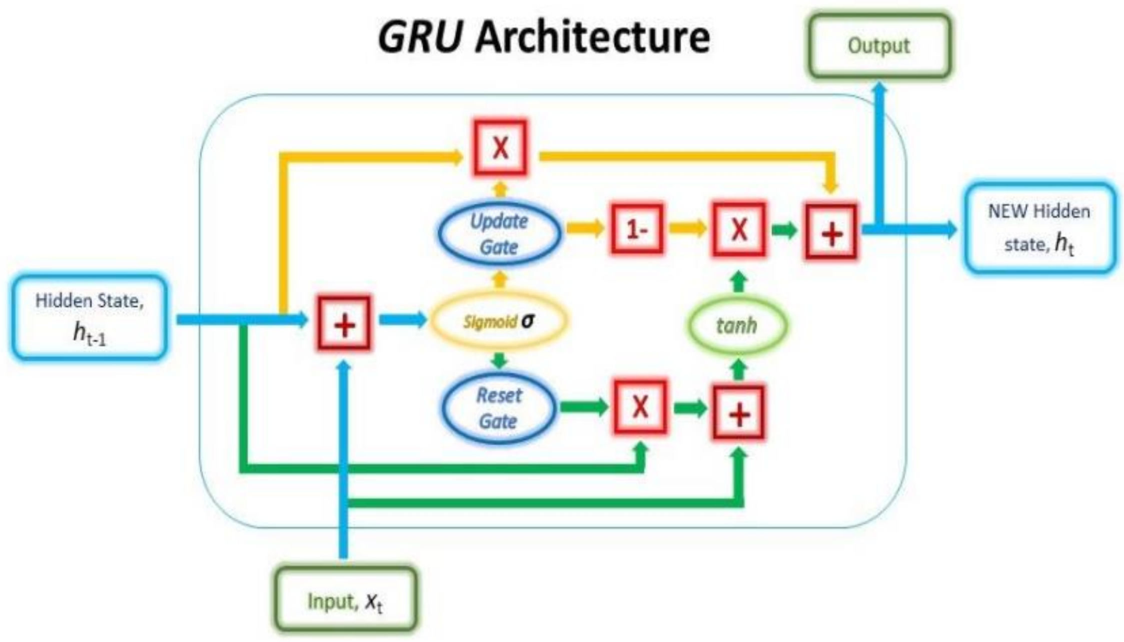
We can infer that the open, high, low and close are largely correlated to each other. The prices also relate to the year column.

MODELING

We used Gated Recurrent Unit (GRU) to build the model.

<u>Gated Recurrent Unit (GRU):</u>
A gated recurrent unit (GRU) is a gating mechanism in recurrent neural networks (RNN) similar to a long short-term memory (LSTM) unit but without an output gate. A GRU can be considered a variation of the long short-term memory (LSTM) unit. GRU performs better than LSTM when dealing with small datasets. GRU's are able to solve the vanishing gradient problem by using an update gate and a reset gate.

Below is the architecture of GRU:



<u>Parameters of GRU:</u>

- **UPDATE GATE:** The update gate controls information that flows into memory. In other words, an update gate would allow us to control how much of the new state is just a copy of the old state.

- **RESET GATE:** The reset gate controls the information that flows out of memory. For instance, a reset gate would allow us to control how much of the previous state we might still want to remember.

- **HIDDEN STATE:** Other than its internal gating mechanisms, the GRU functions just like an RNN, where sequential input data is consumed by the GRU cell at each time step along with the memory, or otherwise known as the hidden state.

Reset gate:	$r_t = \sigma\,(W^{(ir)}\bar{x}_t + W^{(hr)}h_{t-1})$
Update gate:	$z_t = \sigma\,(W^{(iz)}\bar{x}_t + W^{(hz)}h_{t-1})$
Process input:	$\tilde{h}_t = tanh\,(W^{(i\tilde{h})}\bar{x}_t + W^{(h\tilde{h})}h_{t-1})$
Hidden state update:	$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$
Output:	$y_t = h_t$

The steps involved in model building are:

1. **DATA TRANSFORMATION**: A good rule of thumb is that normalized data lead to better performance in Neural Networks. In this project, we used **MinMaxScaler** from **scikit-learn**. We define different scalers for input and output as they have different shapes. This is especially important for using the **inverse-transform** function.

2. **TRAIN TEST SPLIT:** In this project, we set the first 65% of data as train data and the remaining 35% as test data. We train the model with train data and validate its performance with test data.

3. **CREATE A 3D INPUT DATASET:** GRU takes a 3D input. So, we build a create_dataset function, to reshape input. In this project, we define **time_steps = 100**. It means that the model makes predictions based on the last 100-day data.

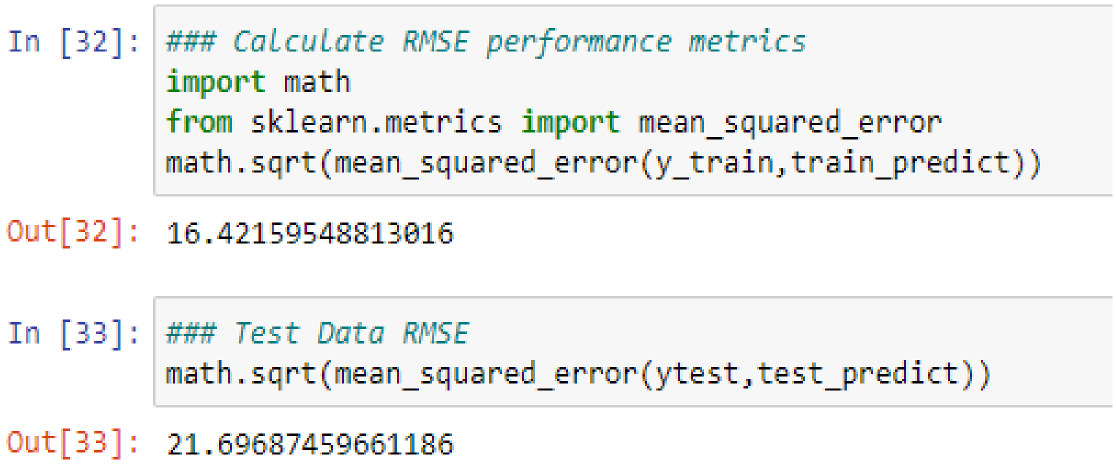
4. **GRU MODEL:** To make the GRU model robust to changes, the **Dropout** function is used. **Dropout(0.2)** randomly drops 20% of units from the network. We train the model with train data for 50 **epoch** and **batch_size** = 150.

5. **MAKING PREDICTIONS AND EVALUATING THE MODEL:** After using the predict() function, we evaluate the model using RMSE score.

MODEL EVALUATION

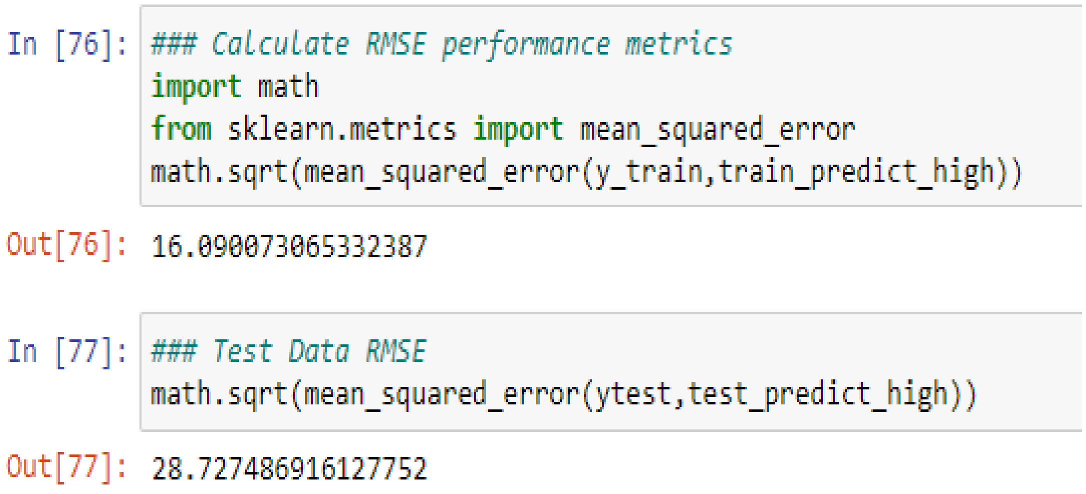
<u>MSFT Dataset:</u> The difference between the train RMSE and the test RMSE is very less. Hence, we can conclude that the model is well trained.

1. RMSE for Close:



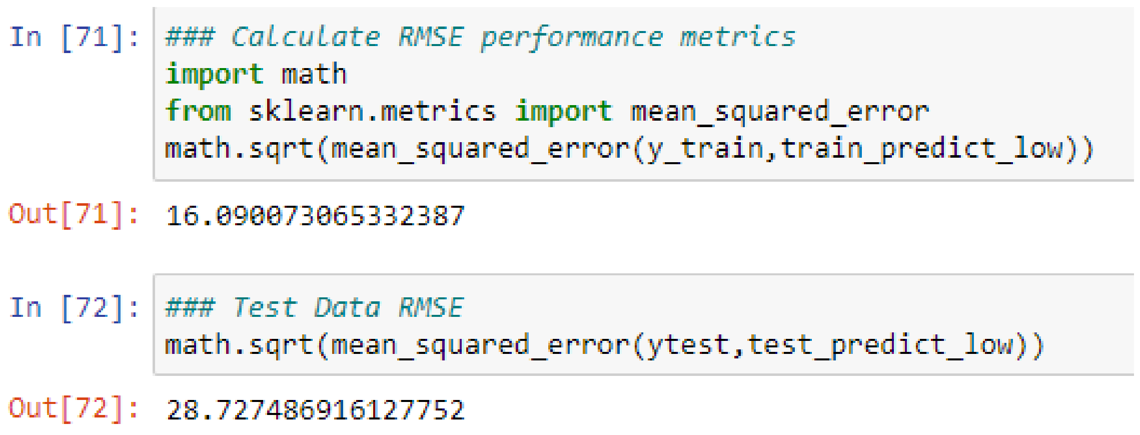
```
In [32]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict))

Out[32]: 16.42159548813016

In [33]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict))

Out[33]: 21.69687459661186
```

2. RMSE for High:



```
In [76]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict_high))

Out[76]: 16.090073065332387

In [77]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict_high))

Out[77]: 28.727486916127752
```

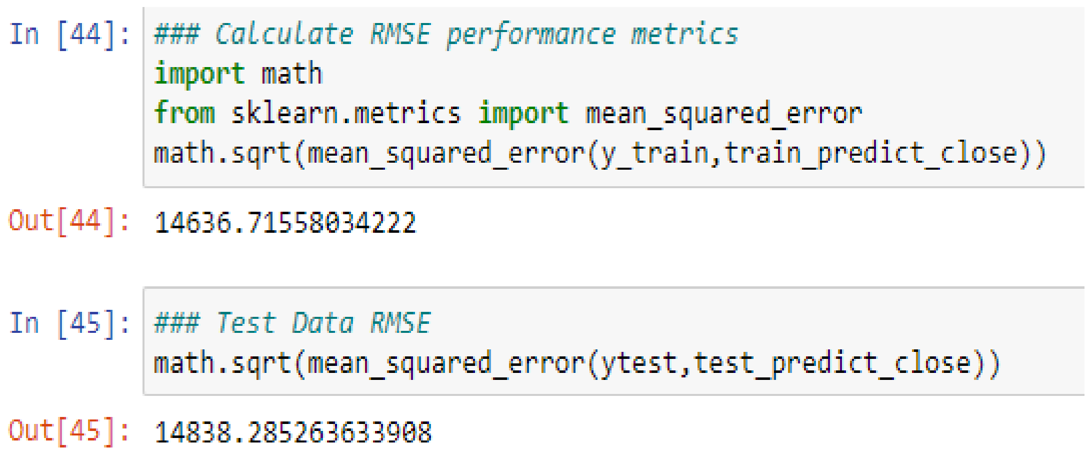
3. RMSE for Low:



```
In [71]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict_low))

Out[71]: 16.090073065332387

In [72]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict_low))

Out[72]: 28.727486916127752
```

<u>DataFrame Dataset:</u> In this dataset also, the difference between the train RMSE and the test RMSE is very less. Hence, we can conclude that the model is well trained.

1. RMSE for Close:



```
In [44]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict_close))

Out[44]: 14636.71558034222

In [45]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict_close))

Out[45]: 14838.285263633908
```

2. RMSE for High:

```
In [61]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict_high))

Out[61]: 14637.526550083294
```

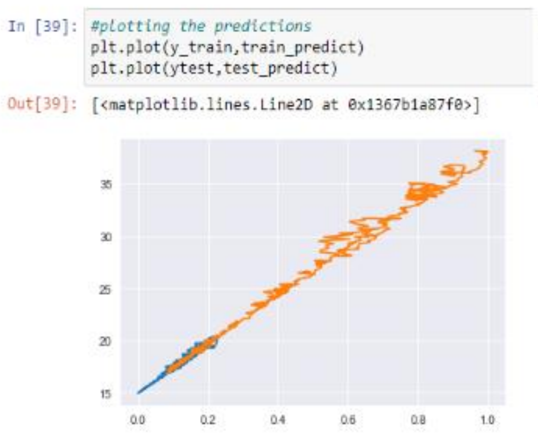
```
In [62]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict_high))

Out[62]: 14840.538578638387
```

3. RMSE for Low:

```
In [75]: ### Calculate RMSE performance metrics
         import math
         from sklearn.metrics import mean_squared_error
         math.sqrt(mean_squared_error(y_train,train_predict_low))

Out[75]: 14631.348607714352
```

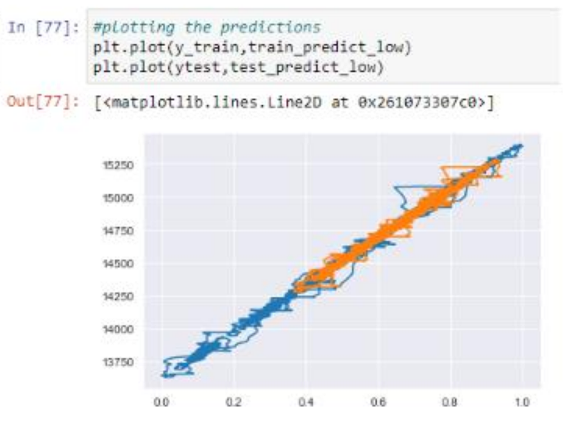
```
In [76]: ### Test Data RMSE
         math.sqrt(mean_squared_error(ytest,test_predict_low))

Out[76]: 14831.300372939906
```

PLOTTING PREDICTIONS



```
In [39]: #plotting the predictions
         plt.plot(y_train,train_predict)
         plt.plot(ytest,test_predict)

Out[39]: [<matplotlib.lines.Line2D at 0x1367b1a87f0>]
```

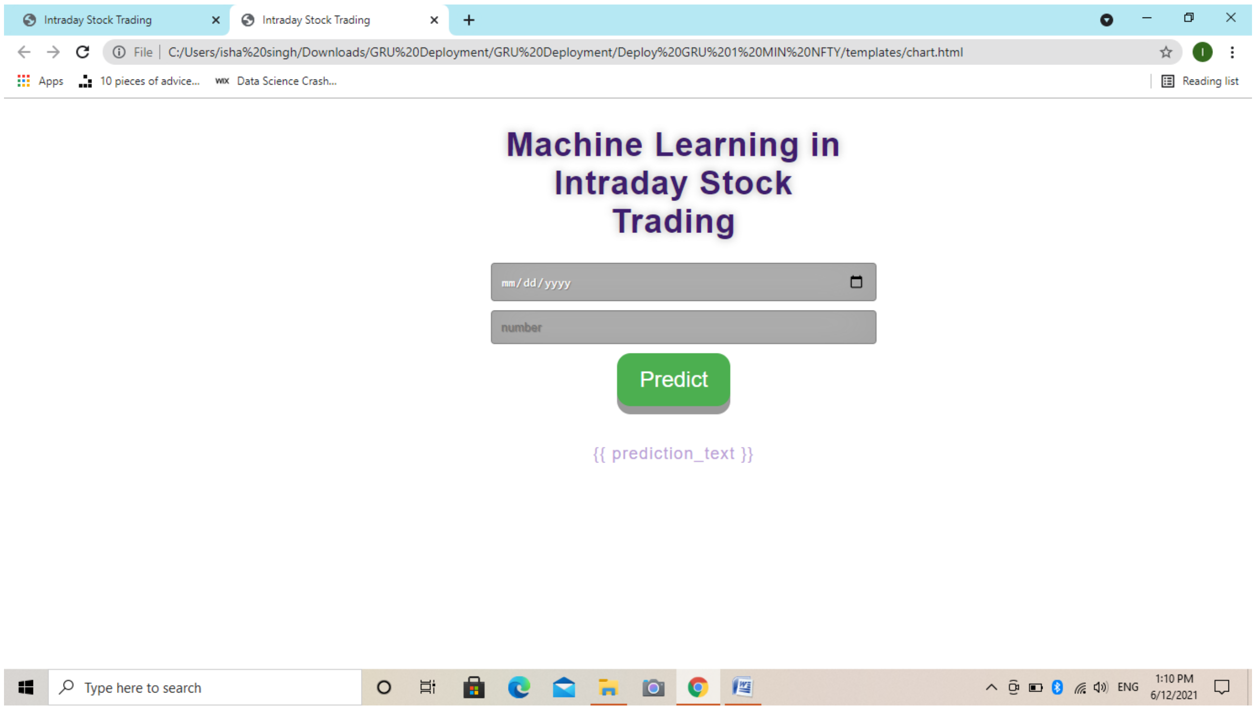
(a)MSFT



```
In [77]: #plotting the predictions
         plt.plot(y_train,train_predict_low)
         plt.plot(ytest,test_predict_low)

Out[77]: [<matplotlib.lines.Line2D at 0x261073307c0>]
```

(b)DATAFRAME

MODEL DEPLOYEMENT


Intraday Stock Trading
Intraday Stock Trading
C:/Users/isha%20singh/Downloads/GRU%20Deployment/GRU%20Deployment/Deploy%20GRU%201%20MIN%20NFTY/templates/chart.html

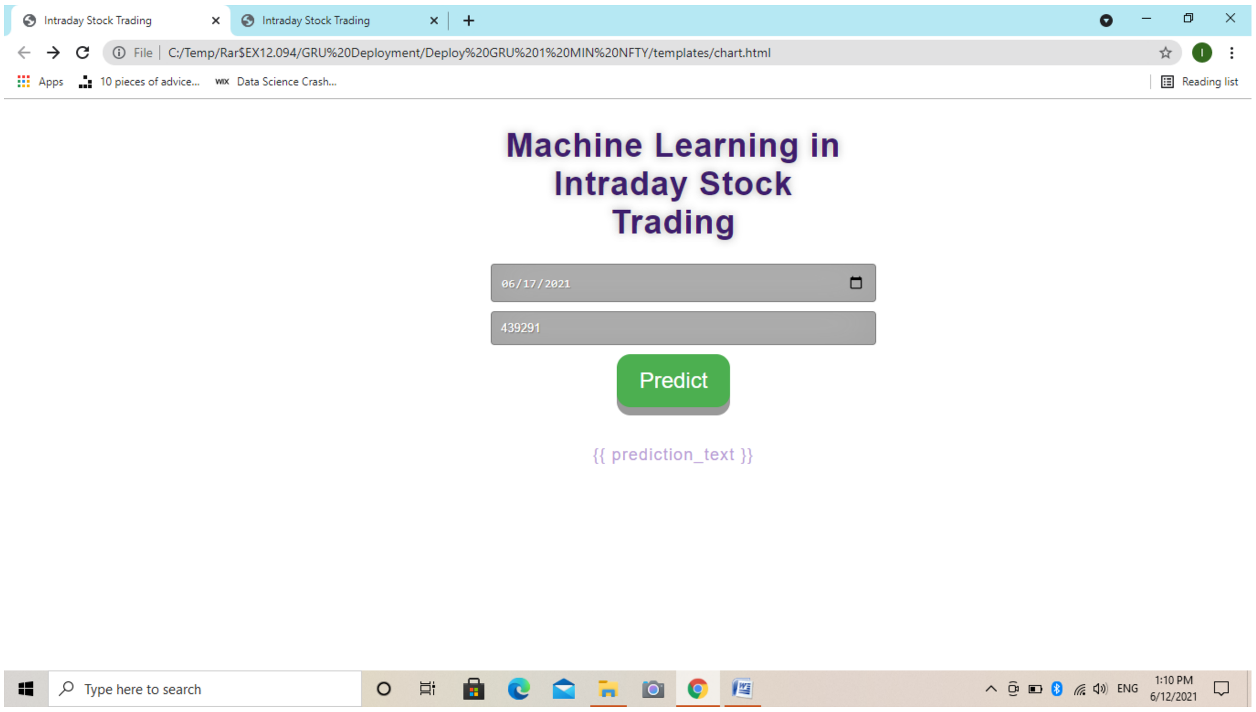
Machine Learning in Intraday Stock Trading

mm/dd/yyyy

number

Predict

{{ prediction_text }}


Intraday Stock Trading
Intraday Stock Trading
C:/Temp/Rar$EX12.094/GRU%20Deployment/Deploy%20GRU%201%20MIN%20NFTY/templates/chart.html

Machine Learning in Intraday Stock Trading

06/17/2021

439291

Predict

{{ prediction_text }}

REFERENCES

1. https://towardsdatascience.com/predictive-analysis-rnn-lstm-and-gru-to-predict-water-consumption-e6bb3c2b4b02
2. https://d2l.ai/chapter_recurrent-modern/gru.html
3. https://towardsdatascience.com/machine-learning-for-day-trading-27c08274df54
4. https://www.researchgate.net/publication/347840605_Employing_Deep_Learning_In_Intraday_Stock_Trading
5. https://deepai.org/machine-learning-glossary-and-terms/gated-recurrent-unit#:~:text=A%20GRU%20is%20a%20very%20useful%20mechanism%20for,performance%20than%20LSTM%20when%20dealing%20with%20smaller%20datasets.