STOCK PRICE PREDICTION USING MACHINE LEARNING

Project report submitted to
Visvesvaraya National Institute of Technology, Nagpur
in partialfulfillment of the requirements for the award of
the degree

Bachelor of Technology
in
"Computer Science and Engineering"
By

Shreyas Deshmane
(BT18CSE016)

Sachin Gautam
(BT18CSE122)

Hrishikesh More
(BT18CSE092)

Pankaj Sharma
(BT18CSE086)

Ansh Rathod
(BT18CSE132)

under the guidance of

Dr R.B. Keskar

Department of Computer Science Engineering
Visvesvaraya National Institute of Technology
Nagpur 440 010 (India)2021-2022

Department of Computer Science and Engineering
Visvesvaraya National Institute of Technology, Nagpur

Declaration

We, Shreyas Deshmane, Sachin Gautam, Hrishikesh More, Pankaj Sharma and Ansh Rathod hereby declare that this project work titled “Stock Price Prediction Model” iscarried out by us in the Department of Computer Science and Engineering of Visvesvaraya National Institute of Technology, Nagpur. The work is original and has not been submitted earlier whole or in part for the award of any degree at this or any other Institution.

Date:

Enrollment No	Name	Signature
BT18CSE016	Shreyas Deshmane	
BT18CSE122	Sachin Gautam	
BT18CSE092	Hrishikesh More	
BT18CSE086	Pankaj Sharma	
BT18CSE132	Ansh Rathod	

Certificate

This to certify that the project titled "**STOCK PRICE PREDICTION USING MACHINE LEARNING**", submitted by **Shreyas Deshmane, Sachin Gautam, Hrishikesh More, Pankaj Sharma** and **Ansh Rathod** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering**, VNIT Nagpur. The work is comprehensive, complete and fit for final evaluation.

Dr. R.BKeskar
Associate Professor,
Computer Science and Engineering,
VNIT,
Nagpur

HoD,
Department of Computer Science and Engineering
VNIT, Nagpur
Date:

ACKNOWLEDGEMENT

We extend our sincere gratitude to our project guide, Dr. Ravindra Keskar for affording us the opportunity to work on this project and guiding us painstakingly for the duration of this work. His advice, time and wealth of experience have proven invaluable to us in the course of this project.

Also, we are sincerely thankful to the entire faculty of the Department of Computer Science and Engineering, VNIT Nagpur, for their support.

Finally, we wish to thank our colleagues for their constant encouragement and motivation for the project.

ABSTRACT

In this project we have attempted to implement some Machine Learning algorithms to predict the stock value for intraday trading. Machine Learning is effectively implemented in prediction of stock prices. The main objective is to predict the stock prices to get more informed about the stock market and to make successful and profitable trade. We propose a stock price prediction model which integrates mathematical functions, machine learning algorithms, and candlestick chart patterns for accurately predicting the stock value and to get profitable trades.

There are two types of trading- Intraday trading and Swing trading. If we buy and sell a stock on the same day, it is called 'Intraday Trading' often known as 'Day Trading'. If we buy and keep a stock for more than one day, it is known as Swing Trading. In this project we are going to focus on Intraday Trading only and we will be using previous stock prices for predicting the stock price in future. We have chosen a forty-five minutes time frame which means we are going to use previous forty-five minutes data for prediction from the time the user made the request. Using the previous stock price data and machine learning algorithms like CNN, the system will predict the future stock price and based on that price the system will give an output whether the person should buy the stock or not.

INDEX

CHAPTER 1: INTRODUCTION

1.1 Introduction:

The financial market in India is growing rapidly and is expected to emerge as one of the leaders in the international arena very soon. The history of the share market of India dates back to 1875, the name of the first share trading association in India was “Native Share and Stock Brokers Association” which later came to be known as Bombay Stock Exchange(BSE).

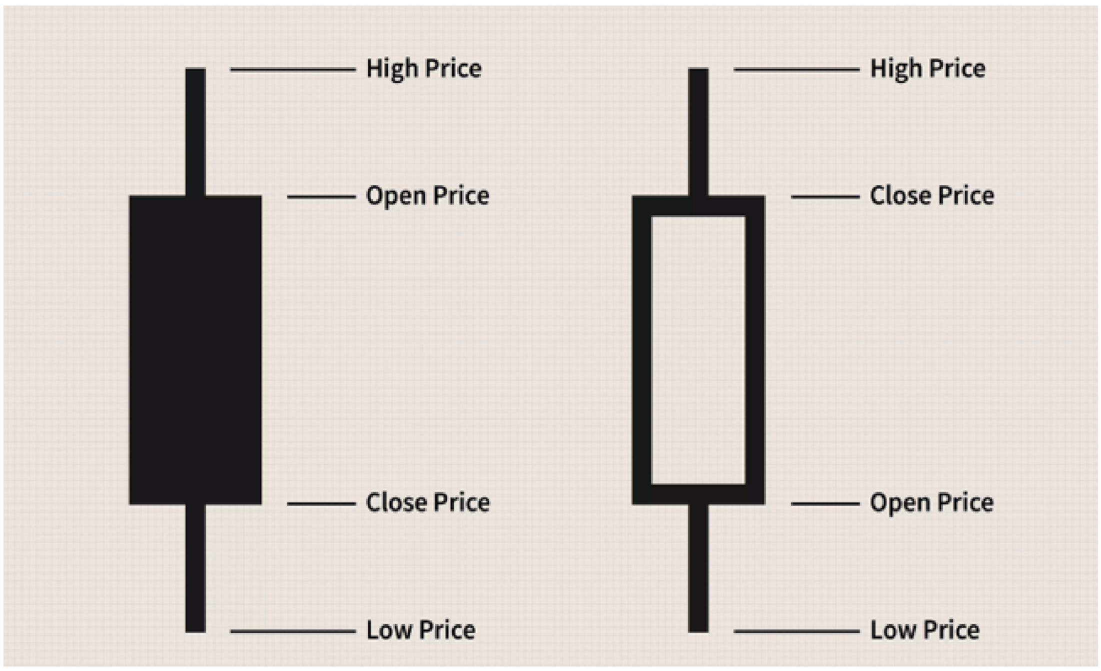
The Indian share market is divided into two segments:

- Primary market-This is the market where new securities are issued to the public.
- Secondary market-This market consists of trading in the shares of the listed companies.

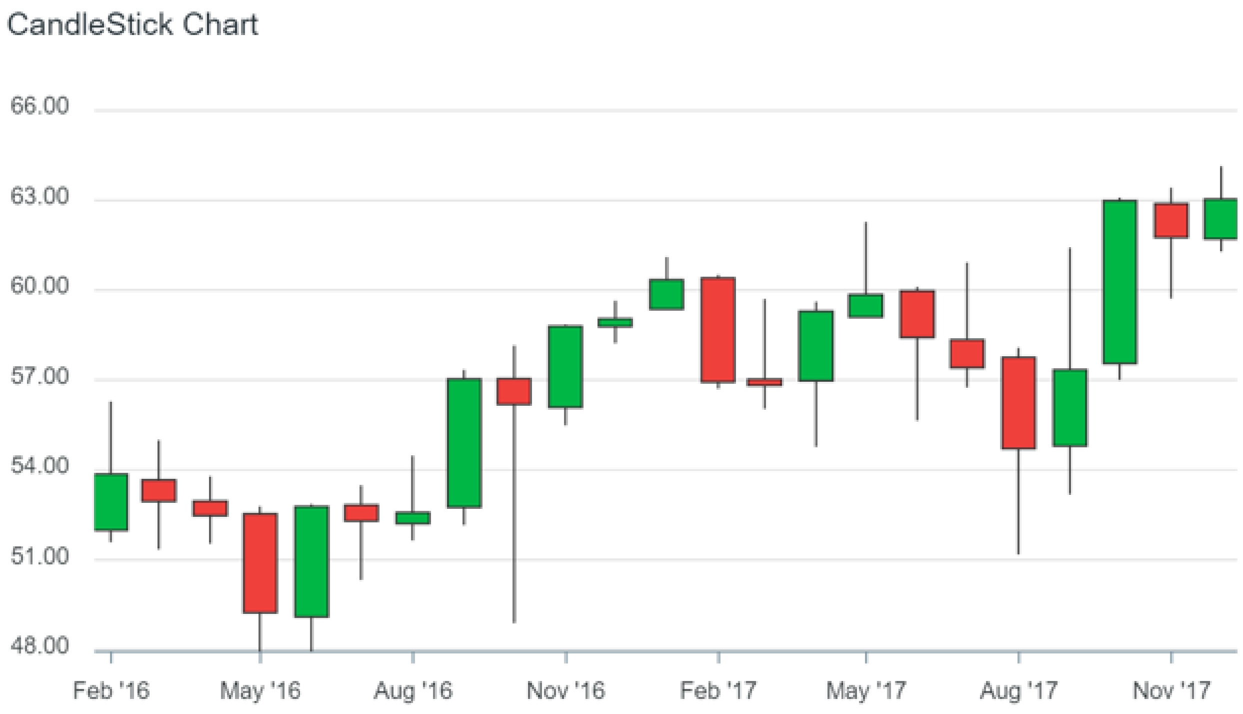
As this sector is growing rapidly there is a new market created which is tools that will help investors, traders in their decision making. At present they use a set of patterns to identify the trend of the market. Being fairly accurate, though the catch is they are difficult to spot. This brings us to our project in which we will use a different set of machine learning algorithms such as Convolutional Neural Network, Time Series, Reinforcement Learning, Linear Regression to identify such patterns.

1.2 Candlestick Patterns:

1.2.1 Candlesticks- A daily candlestick chart illustrates the market's open, high, low, and closing prices for the day, similar to a bar chart. The "true body" of the candlestick is a large section of the candlestick. The price range between the open and closing of trade on that particular day is represented by this physical body.

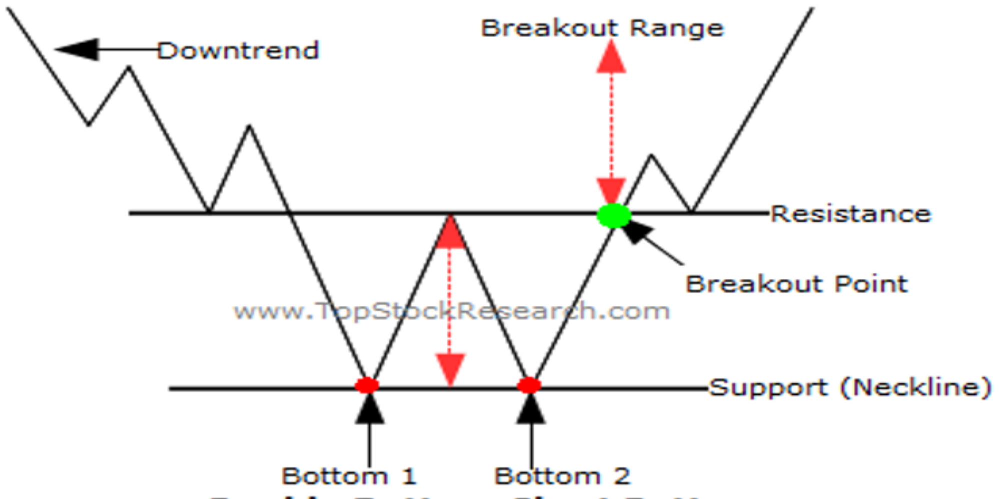


1.2.2 Candlestick Chart- Candlestick charts are used to forecast price movement based on historical trends. Candlesticks depict this emotion by using different colors to graphically express the size of price movements. Traders use candlesticks to make trading decisions based on patterns that appear on a regular basis and help foretell the price's short-term direction.



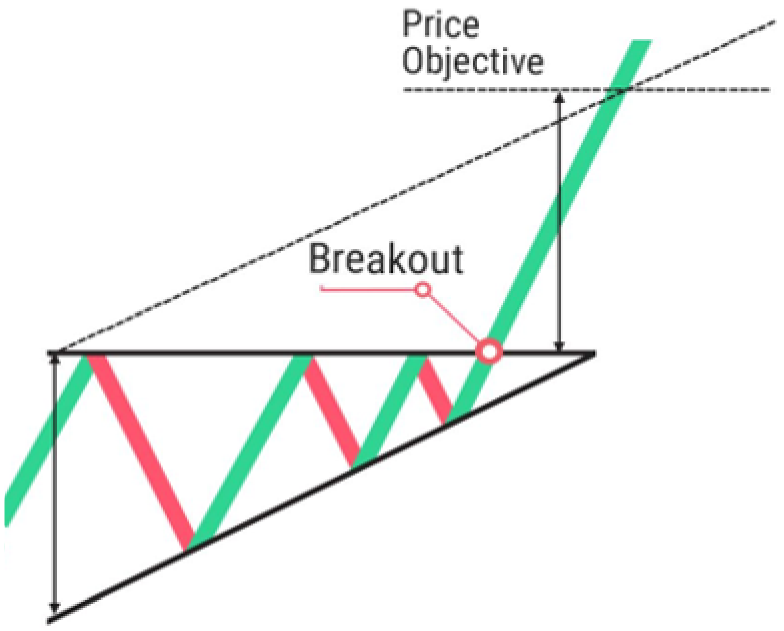
1.2.3 Candlestick Patterns- There are a total of above 35 candlestick chart patterns from which we have shortlisted 2 patterns to be used in our project i.e Bullish Double Bottom and Ascending Triangle.

1.2.3.1 Bullish Double Bottom- A bullish reversal chart pattern developed after a downtrend is known as a double bottom. Before initiating a position, traders should always validate the reversal using double top and double bottom chart patterns together with additional indications such as volume.



Double Bottom Chart Pattern

1.2.3.2 Ascending Triangle- An ascending triangle is usually regarded as a continuation pattern, which means it is relevant whether it appears in an uptrend or decline. Traders usually purchase or sell the asset aggressively once the triangle has broken out, depending on which way the price has broken out.



1.2.4 Data Collection- In this project we started with collecting data manually. We first created a dataset with over 200 candlestick chart patterns to compare with the users input chart.
This is how we planned to execute-
Step 1- Adding lots of Analyzed charts in training set
Step 2- Taking input from users for target chart
Step 3- Comparing target chart with all charts available in training sets
Step 4- Finding most optimal amongst them
Step 5- Desired output to the user

1.3 Machine Learning Algorithms:

We have used 4 main algorithms to compare and give the most optimum solution, they are –

- Convolutional Neural Network
- Linear Regression
- Time Series
- Reinforcement Learning

The main reason why we choose these four algorithms is because we found them most relevant for our topic. This is due to the salient features of these algorithms which will be later discussed in the *Design* and *Implementation* modules.

1.3.1 Convolutional Neural Network- A convolutional neural network (CNN/ConvNet) is a type of deep neural network used to evaluate visual images in deep learning. When we think about neural networks, we usually think of matrix multiplications, but this isn't the case with ConvNet. It employs a method known as Convolution.

How does it work?

CNN makes use of spatial correlations found in the input data. Some input neurons are connected by each concurrent layer of the neural network. A local receptive field is the name given to this area. Hidden neurons are the center of the local receptive field.

1.3.2 Linear Regression- Linear Regression is a supervised Machine Learning model that identifies the best fit linear line between the independent and dependent variables, i.e. it discovers the linear connection between the two variables.

How is it helpful in prediction?

One of the most often used predictive modeling approaches is linear regression. The formula is $Y = a + bX + e$, where an is the intercept, b is the slope of the line, and e is the error term. Based on a given predictor variable, this equation may be used to predict the value of a target variable (s).

1.3.3 Time Series- A time series is a set of data that follows the progression of a sample through time. A time series, in example, allows you to examine what causes impact specific variables from one period to the next. Time series analysis is important for determining how an asset, security, or economic variable change over time.

How to analyze Time Series data-

Time series data may be analyzed using statistical techniques in two ways: to make conclusions about how one or more factors impact a variable of interest across time, or to anticipate future trends. The arrow of time helps an analyst to make more credible causal statements than cross-sectional data, which is effectively one slice of a time series.

1.3.4 Reinforcement Learning- Reinforcement Learning is a Machine Learning technique that focuses on how software agents should behave in a given environment. Reinforcement Learning is a deep learning technique that allows you to optimize a fraction of your total reward.

What is the purpose of this algorithm?

The goal of reinforcement learning is for the agent to develop an optimal, or nearly optimal, policy that maximizes the "reward function" or other user-provided reinforcement signal that builds up from the immediate rewards. This appears to be akin to animal psychology processes. Pain and hunger, for example, are encoded into biological brains as negative reinforcements, whereas pleasure and food intake are seen as positive reinforcements. Animals can learn to participate in activities that maximize these rewards under certain situations. This indicates that animals have the ability to learn by doing things again and over again.

1.4 Problem Statement-

We are trying to come up with a product to help the user to find the best trading opportunity based on technical analysis in the intra-day market. We have had a look at some of the machine learning algorithms which we will be testing and training in the upcoming modules.

The report is organized as follows. In chapter 2, we present a literature survey about stock market prediction. In chapter 3, we present design of our system. In chapter 4, the implementation of our system is presented and discussed. Chapter 5 concludes the report and future work is discussed.

CHAPTER 2: LITERATURE SURVEY

The stock market is basically an aggregation of various buyers and sellers of stock. A stock in general represents ownership claims on business by a particular individual or a group of people. The attempt to determine the future value of the stock market is known as a stock market prediction. The prediction is expected to be robust, accurate and efficient. The ability to precisely predict the price movement of stocks is the key to profitability in trading.

2.1 Stock Market Prediction

Stock Market prediction is defined as trying to determine the stock value (price) after a certain period of time and offers an idea to the people to know and predict the market. Predicting how the market will move in future is quite a difficult task. The stock market movement is usually decided by the sentiments of millions of investors (if they want to buy or sell the stock). If more people want to buy the stock, then the stock price rises and if more people are interested in selling the stock, then the stock value decreases[1].

But besides the sentiments of investors, there are several other factors that affect the value of stock and events that are related to investors and stock. These events can be political events like statements given by a political leader, business takeovers, recentnews, scam. It can also be the release of the financial reports about the profit and losses of a respective company. It can also be an international event like sudden movement in currency and commodities value. All these events affect the company's earnings which in turn affects the sentiments of investors. It is out of scope for any investor to predict consistently and correctly the movement of any stock. All these factors make stock market prediction very difficult[1]. But still, we can predict the movement to some extent. And for that we need to have a lot of data and using the data and some machine learning algorithms we can accomplish the task of predicting the stock market.

There are many methods for predicting stock market like:

- Machine Learning Algorithms
- Technical Analysis
- Fundamental Analysis
- Historical Data Analysis
- Support Vector Machines

Out of all these methods, we are going to use **Machine Learning Algorithms** for prediction[2].

2.2 Stock Market Prediction Using Machine Learning Algorithms

There are several Machine Learning algorithms which we can use for prediction such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Reinforcement Learning, Time Series, Linear Regression etc[4].

2.2.1 Convolutional Neural Network (CNN):

CNN is widely used in the field of image recognition due to its ability to recognize patterns. CNN is composed of multiple neurons connected by a hierarchical structure, and the weights and bias between layers can be trained. If the characteristics of the stock market at a specific time point are regarded as a feature graph, CNN has the potential to extract the characteristics of the stock market at the corresponding period from these feature graphs. Therefore, CNN can be used to build a timing-selection model and can ultimately be used to complete the construction of the timing-selection strategy[6].

2.2.2 Reinforcement Learning:

Reinforcement learning is one of the deep learning methods that focuses on how to act according to the current situation to maximize profits. In reinforcement learning, there are two elements: state and action. If the state is regarded as the attribute and the action is the label, it is easy to know that both supervised learning and reinforcement learning try to find a map and to infer the label/action from the known attribute/state[6].

Therefore, reinforcement learning learns the best timing trading action according to the market response. It can view the contextual information (price, news, public opinion) of the transaction as an environment of reinforcement learning. Profits or losses could be thought of as the reward for learning, trading actions could be thought of as actions, and factors could be thought of as states or observations to realize the prediction of the stock trends.

2.2.3 Time Series:

Time-series forecasting models are the models that are capable to predict future values based on previously observed values. Time-series forecasting is widely used for non-stationary data.

A popular and widely used statistical method for time series forecasting is the ARIMA (Autoregressive Integrated Moving Average) model. It is one of the most popular models to predict linear time series data. This model has been used extensively in the field of finance and economics as it is known to be robust, efficient, and has a strong potential for short-term share market prediction.

2.2.4 Linear Regression:

Linear Regression is used to find the relationship between a dependent variable and an independent variable. In case of stock market prediction, the independent variable is time and the dependent variable is price of the stock as it changes with time. We have to get the historic data of the stock which includes the date, time and the price movement of the stock at a particular time.

2.3 Steps for Prediction of Stock Market

Every Machine Learning Algorithm is going to work in a different manner but there are some common steps that all of the algorithms are going to have in common[1].

2.3.1 Dataset Collection:

Data Collection is a very basic and initial step towards a project. We are going to need a dataset for every algorithm. The dataset would consist of the historical data for every stock which includes the open price, closeprice, date and time.

2.3.2 Pre-Processing:

Data preprocessing is a part of data mining, which involves transforming raw data into a more coherent format. Raw data is usually inconsistent or incomplete and usually contains many errors. The data pre-processing involves checking for missing values, looking for categorical values, splitting the data-set into training and test set.

2.3.3 Training the Machine:

Training the machine is similar to feeding the data to the algorithm to touch up the test data. The training sets are used to tune and fit the models. The test sets are untouched, as a model should not be judged based on unseen data. The idea behind the training of the model is that we use some initial values with the dataset and then optimize the parameters which we want to in the model.

2.3.4 Testing the Machine:

Testing is referred to as the process where the performance of a fully trained model is evaluated on a testing set. Testing identifies explicitly which part of the code fails and provides a relatively coherent coverage measure. It helps us in two ways: Quality assurance: whether the software works according to requirements. Identify defects and flaws during development and in production.

This was a brief about the Literature Survey of the stock market prediction that we have done. The next chapter discusses about the design and training of the model.

CHAPTER 3: DESIGN

In this chapter, we will be discussing the code flow and design. This section Includes everything from the API usage, Candle chart preparation from our API data,our training and testing Models in various algorithms that we have used.

3.1 CNN (Convolutional Neural Network):

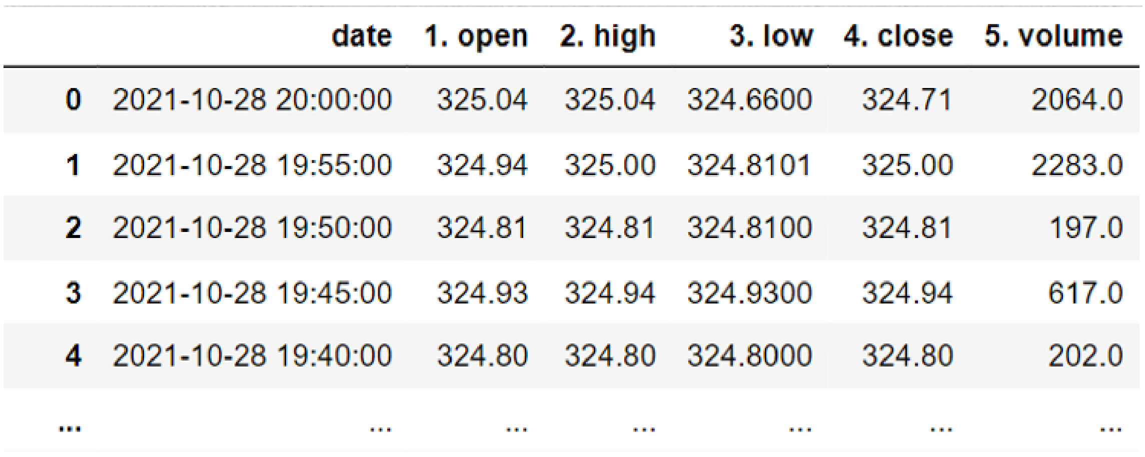
In this section we'll discuss how we have used Convolutional neural networks to compare different chart patterns that we have created using API (Alpha Vantage).

3.1.1 Dataset

We have used a manually created dataset for which we gathered around more than Two hundred Ascending triangle and Bullish Double Bottom chart patterns from various stocks.

3.1.2 API (Alpha Vantage)

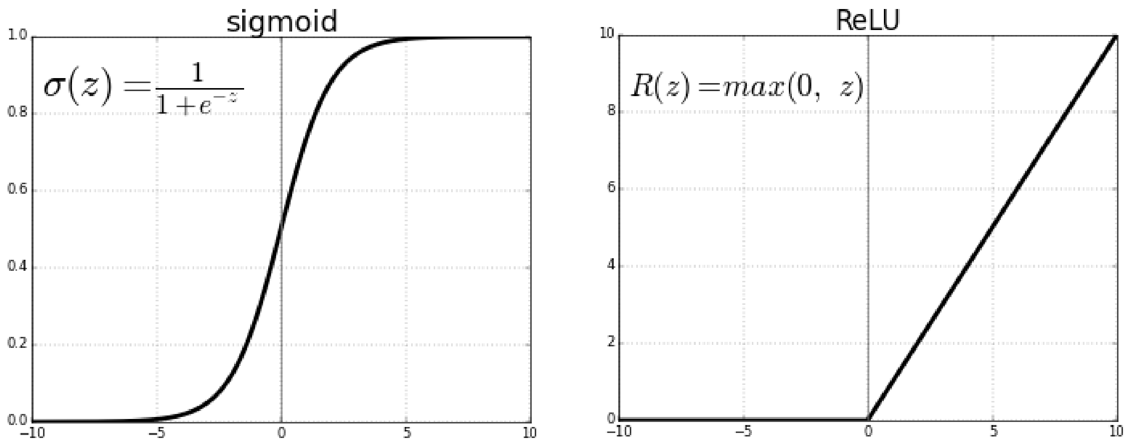
API provides us with the Candlestick Chart of Desired Stock. It gathers data of real-time stocks. It Helps us to have a chart of any time limit, we have a chart of 45minute time span in which each candle will be of 5 min.



	date	1. open	2. high	3. low	4. close	5. volume
0	2021-10-28 20:00:00	325.04	325.04	324.6600	324.71	2064.0
1	2021-10-28 19:55:00	324.94	325.00	324.8101	325.00	2283.0
2	2021-10-28 19:50:00	324.81	324.81	324.8100	324.81	197.0
3	2021-10-28 19:45:00	324.93	324.94	324.9300	324.94	617.0
4	2021-10-28 19:40:00	324.80	324.80	324.8000	324.80	202.0
...	...	...	...	...	...	...

3.1.3 Model Training

Model is implemented with Keras Sequential Model with ReLU (Rectified Linear Unit) and Sigmoid Activation Function with Three Convolutional 2D layers and 2 Dense layers and trained with 30 epochs taking three steps per epoch which also provides validation dataset training accuracy around 75%.

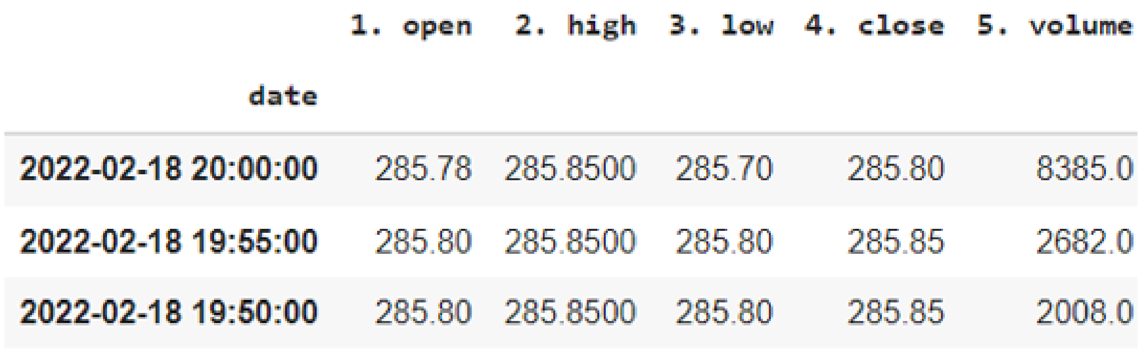


3.2 Reinforcement Learning:

In this section we'll discuss how we have used Reinforcement Learning to predict stock Price using the real-time data that we have gathered through API (Alpha vantage).

3.2.1 Dataset

For Reinforcement Learning we have used a data frame generated by Alpha vantage with attributes **data, 1. open, 2. high, 3. low, 4. close, 5. volume**



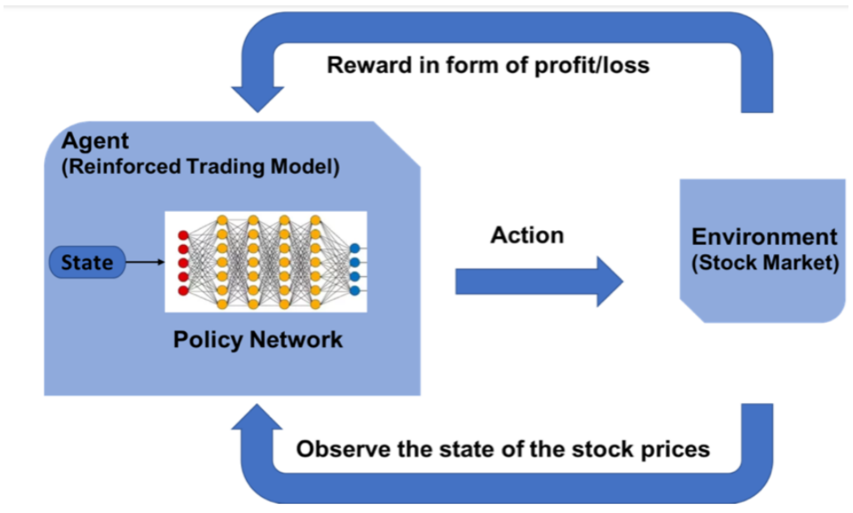
	1. open	2. high	3. low	4. close	5. volume
date					
2022-02-18 20:00:00	285.78	285.8500	285.70	285.80	8385.0
2022-02-18 19:55:00	285.80	285.8500	285.80	285.85	2682.0
2022-02-18 19:50:00	285.80	285.8500	285.80	285.85	2008.0

3.2.2 Q-Learning

Q-learning is a model-free reinforcement learning algorithm to learn the quality of actions telling an agent what action to take under what circumstances. Q-learning finds an optimal policy in the sense of maximizing the expected value of the total reward over any successive steps, starting from the current state.

3.2.3 Reinforcement Learning Environment

In this environment Section we have created an agent which makes all the decisions inside the environment. Actions like Buy/sell/hold are managed by the agent and then it gets rewards in the form of profit/loss by the environment.


Reward in form of profit/loss
Agent
(Reinforced Trading Model)
State
Policy Network
Action
Environment
(Stock Market)
Observe the state of the stock prices

The feedback i.e. reward is given to the Agent for further processing

3.2.4 Model Training

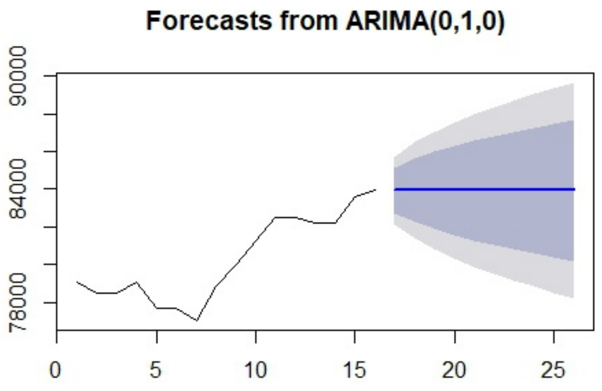
The data that we have used is real time data from API in a 5min time frame window and created our Environment and then visualizing it in Matplotlib graphs. Model is trained as an RL agent using a Gymenvironment.

3.3 Time Series:

In this section we'll discuss models that we've used while implementing the time series for real time stock market data.

3.3.1 ARIMA

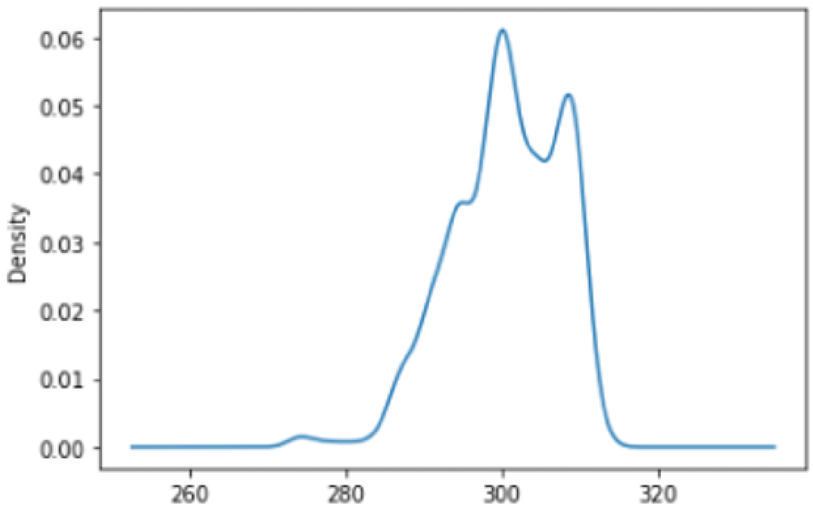
The ARIMA model has been widely utilized in banking and economics since it is recognized to be reliable, efficient, and capable of predicting short-term share market movements.Before working with non-stationary data, the Autoregressive Integrated Moving Average (ARIMA) Model converts it to stationary data. One of the most widely used models for predicting linear time series data is this one.



By using the closing price of the stocks and time frame for intraday we have plotted various graphs using matplotlib to generate respective outcomes.

3.3.2 Test for Stationarity:

Tests for checking the stationarity of the time series algorithm have checked the distribution of the dataset using matplotlib graphs. Also performing dickey fuller tests to get more precise statistics which also yielded test statistics, No. of lags used, No. of observation used which presents a clearer picture to the observer.



3.3.3 Separating the trend and the seasonality from a time series:

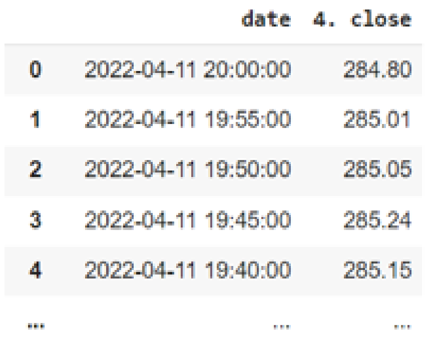
Using seasonal decomposition from statsmodel.tsa.seasonal we have decomposed the time series which is plotted through Matplotlib.

3.4 Linear Regression:

During prediction we use some variables as dependent variables and few considered as independent variables. In situations when there is one dependent and one independent variable, we prefer to use linear regression methodologies. Regression can be single variable or multi variable, it depends upon situations named as single variable or multi variable regression.

3.4.1 Subset with relevant features

We have used the 5 min Time Frame closing price Close as the value to predict, so we can discard the other features. ‘Close' column has a numeric data type. The 'Date' is the index column and contains datetime values



	date	4. close
0	2022-04-11 20:00:00	284.80
1	2022-04-11 19:55:00	285.01
2	2022-04-11 19:50:00	285.05
3	2022-04-11 19:45:00	285.24
4	2022-04-11 19:40:00	285.15
...	...	...

3.4.2 Explore the Data

When we had a look at the price movement over time by simply plotting the *Closing price* vs *Time*, we observed that the price continuously increases over time and we can also estimate that trend could be linear.

3.4.3 Formulation:

Our data contains only one **independent variable (X)** which represents the *date* and the **dependent variable (Y)** we are trying to predict is the *Stock Price*. To fit a line to the data points, which then represents an estimated relationship between X and Y, we can use a **Simple Linear Regression**.

The best fit line can be described with $Y = \beta_0 + \beta_1 X$ were,

- Y is the predicted value of the dependent variable
- β_0 is the y-intercept
- β_1 is the slope
- X is the value of the independent variable

The goal is to find such coefficients β0 and β1 that the **Sum of Squared Errors**, which represents the difference between each point in the dataset with its corresponding predicted value outputted by the model, is minimal.

3.4.4Model:

We have used linear regression model from Sklearn.Model_selection which is fed by real time data from API (alpha vantage)

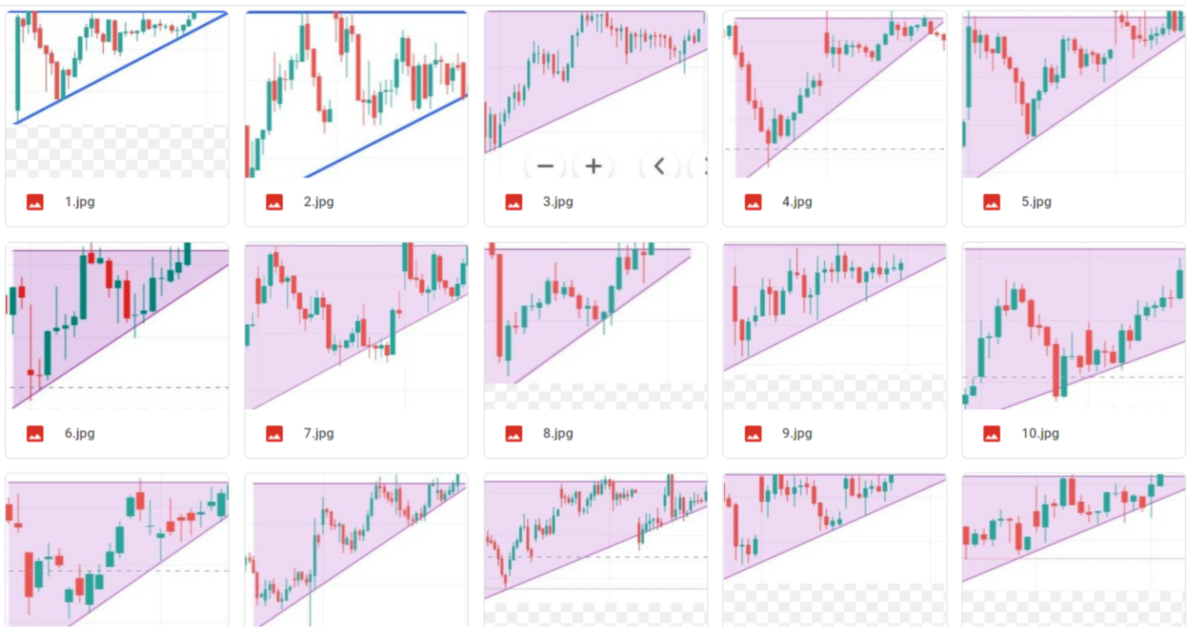
In this module we presented design of our model. In the next chapter, we will look into implementation.

CHAPTER 4: IMPLEMENTATION

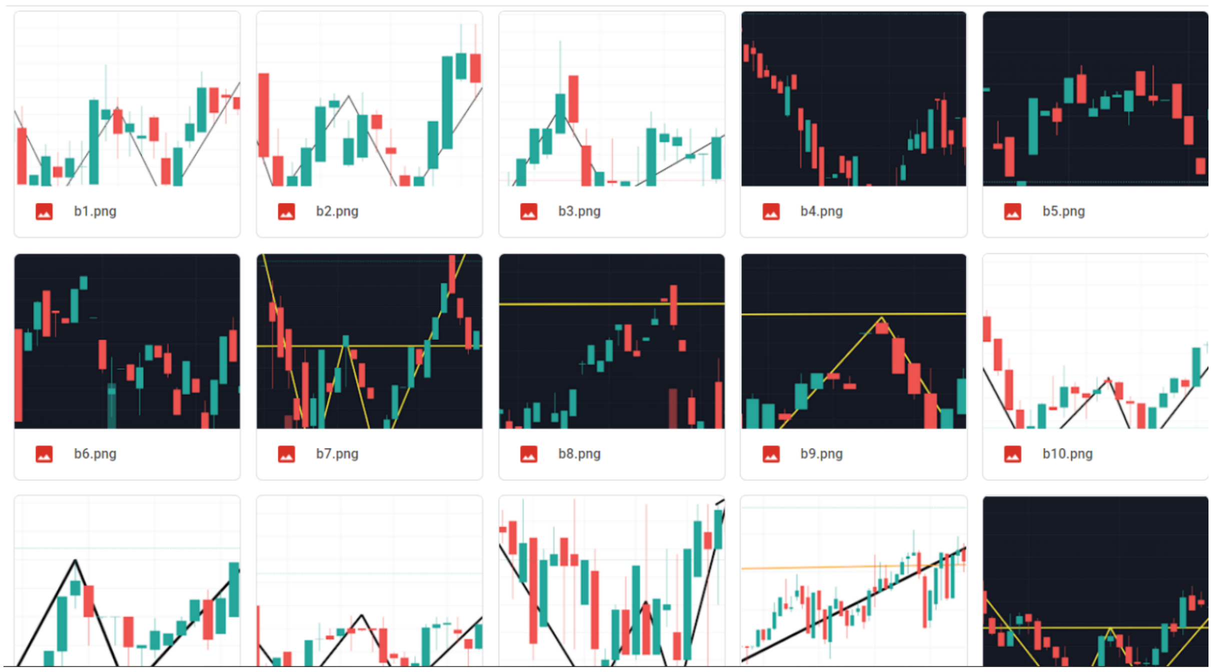
This section discusses the general process of utilizing machine learning to collect literature about SMP. The phrase "stock market prediction using machine learning" was first associated with a number of search engines, digital libraries, and databases, including "google scholar," "research gate," "ACM digital library," "IEEE Explore," and "Scopus," among others. Various phrases including "stock market prediction methods," "effect of feelings on stock market prediction," and "machine learning-based strategy for stock market prediction" were keyed throughout the literature search. As a result, some of the most important articles in the subject of stock market forecasting have been published.

Datasets :

An ascending triangle is a chart pattern used in technical analysis. It is created by price moves that allow for a horizontal line to be drawn along the swing highs, and a rising trendline to be drawn along the swing lows. The two lines form a triangle. Traders often watch for breakouts from triangle patterns.



A double top pattern is a technical analysis charting pattern that describes a change in trend and momentum reversal from prior leading price action. It describes the drop of a stock or index, a rebound, another drops to the same or similar level as the original drop, and finally another rebound

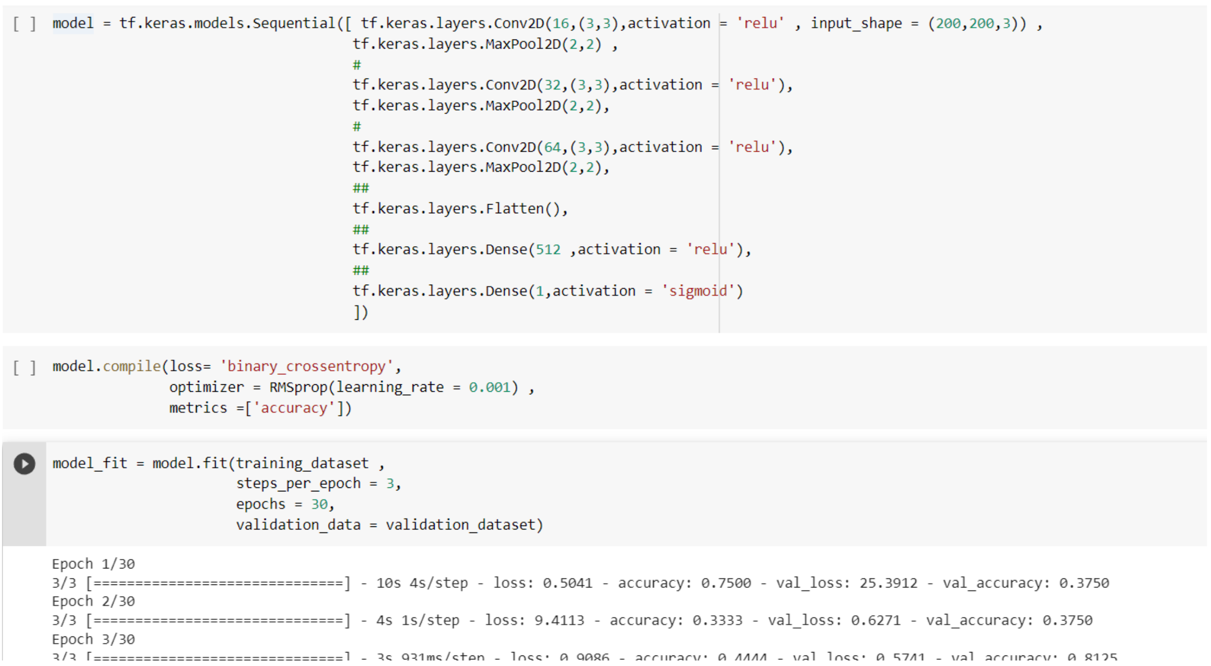


4.1 Convolutional Neural Network (CNN)

The training of a simple Convolutional Neural Network (CNN) to classify candle graph images. Because this uses the Keras API, creating and training the model.

4.1.1 Model training:

CNN takes input, assigns importance to various objects in image and differentiate between them.
The model implementation was done using the Keras and model training with 30 Epochs.



```
[ ] model = tf.keras.models.Sequential([ tf.keras.layers.Conv2D(16,(3,3),activation = 'relu' , input_shape = (200,200,3)) ,
                                        tf.keras.layers.MaxPool2D(2,2) ,
                                        #
                                        tf.keras.layers.Conv2D(32,(3,3),activation = 'relu'),
                                        tf.keras.layers.MaxPool2D(2,2),
                                        #
                                        tf.keras.layers.Conv2D(64,(3,3),activation = 'relu'),
                                        tf.keras.layers.MaxPool2D(2,2),
                                        ##
                                        tf.keras.layers.Flatten(),
                                        ##
                                        tf.keras.layers.Dense(512 ,activation = 'relu'),
                                        ##
                                        tf.keras.layers.Dense(1,activation = 'sigmoid')
                                        ])
```

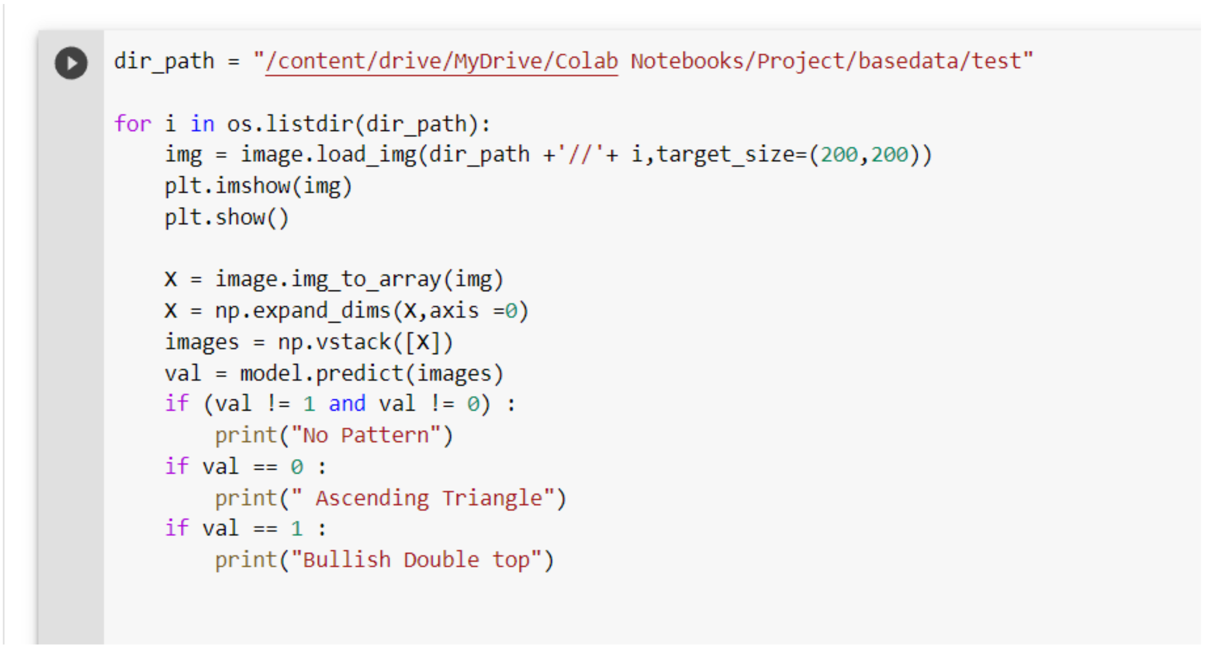
```
[ ] model.compile(loss= 'binary_crossentropy',
              optimizer = RMSprop(learning_rate = 0.001) ,
              metrics =['accuracy'])
```

```
model_fit = model.fit(training_dataset ,
                      steps_per_epoch = 3,
                      epochs = 30,
                      validation_data = validation_dataset)

Epoch 1/30
3/3 [==============================] - 10s 4s/step - loss: 0.5041 - accuracy: 0.7500 - val_loss: 25.3912 - val_accuracy: 0.3750
Epoch 2/30
3/3 [==============================] - 4s 1s/step - loss: 9.4113 - accuracy: 0.3333 - val_loss: 0.6271 - val_accuracy: 0.3750
Epoch 3/30
3/3 [==============================] - 3s 931ms/step - loss: 0.9086 - accuracy: 0.4444 - val_loss: 0.5741 - val_accuracy: 0.8125
```

4.1.2Model Testing:

Here, we are taking the Test images at "test" folder and testing all using the CNN Algorithm. Final obtained result,



```
dir_path = "/content/drive/MyDrive/Colab Notebooks/Project/basedata/test"

for i in os.listdir(dir_path):
    img = image.load_img(dir_path +'//'+ i,target_size=(200,200))
    plt.imshow(img)
    plt.show()

    X = image.img_to_array(img)
    X = np.expand_dims(X,axis =0)
    images = np.vstack([X])
    val = model.predict(images)
    if (val != 1 and val != 0) :
        print("No Pattern")
    if val == 0 :
        print(" Ascending Triangle")
    if val == 1 :
        print("Bullish Double top")
```

4.1.3 Final Results obtained:

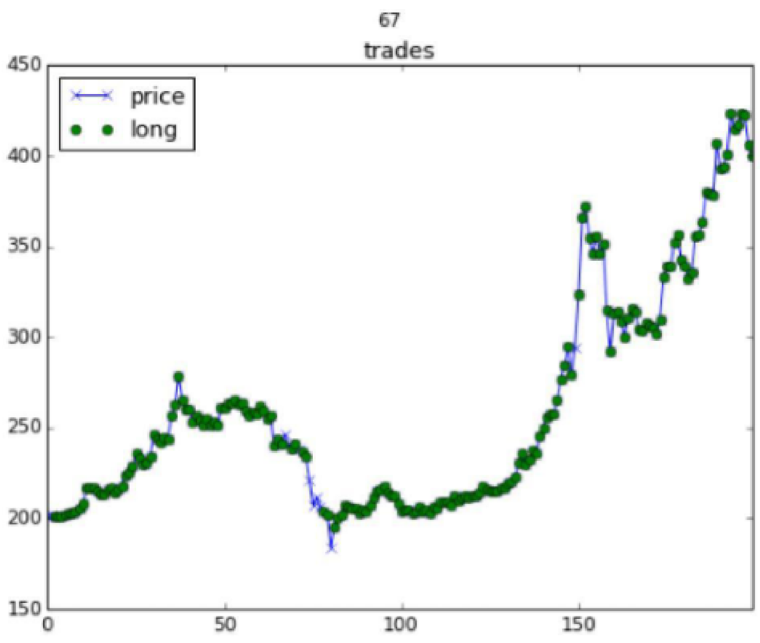
Two types of result:

1.convolved feature is reduced in dimension
2.convolved feature increases or remains same in dimension

~Predicted results with 75%+ accuracy

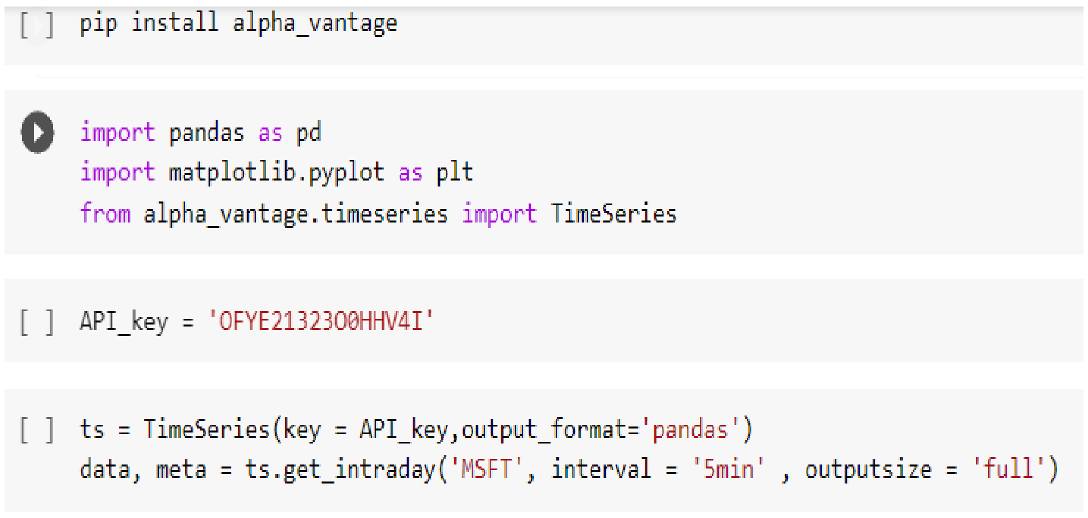
4.2 Reinforcement Learning

The concept of reinforcement learning can be applied to the stock price prediction for a specific stock as it uses the same fundamentals of requiring lesser historical data, working in an agent-based system to predict higher returns based on the current environment.



4.2.1 Obtaining Data (5 mins Time Frame):

Here we are giving the algorithm live data gathered through API –Vantage.

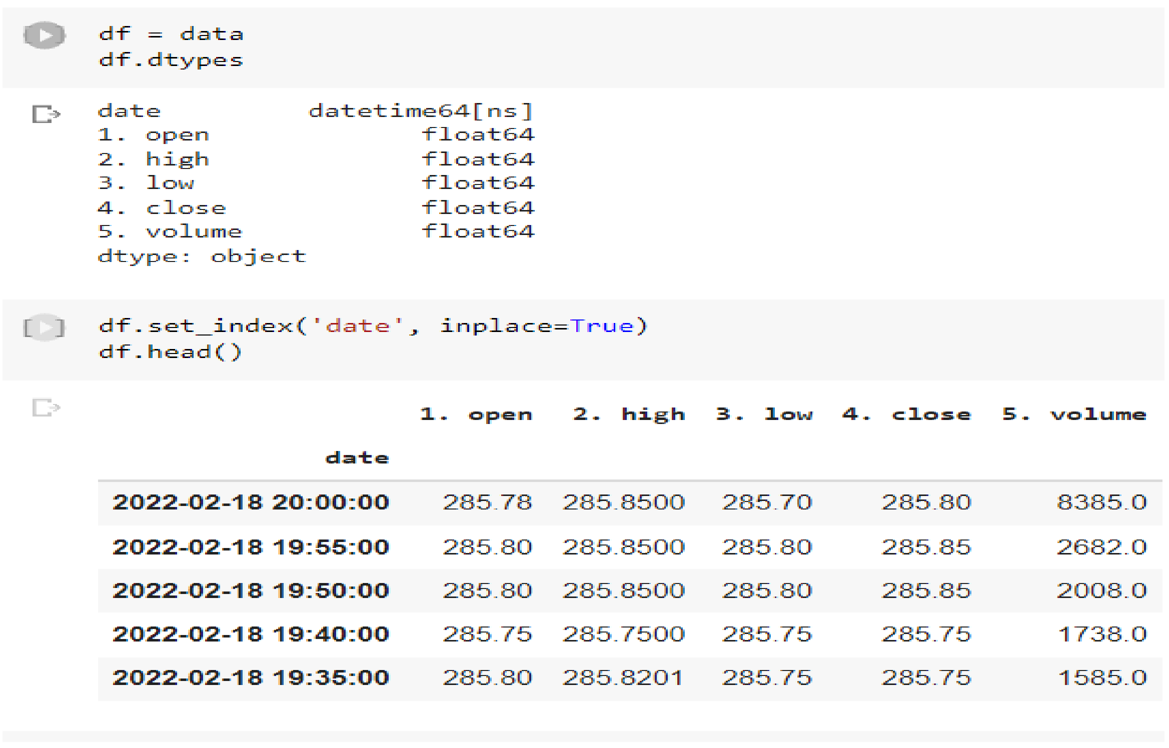


```
[ ] pip install alpha_vantage
```

```
import pandas as pd
import matplotlib.pyplot as plt
from alpha_vantage.timeseries import TimeSeries
```

```
[ ] API_key = 'OFYE2132300HHV4I'
```

```
[ ] ts = TimeSeries(key = API_key,output_format='pandas')
    data, meta = ts.get_intraday('MSFT', interval = '5min' , outputsize = 'full')
```



```
df = data
df.dtypes
```

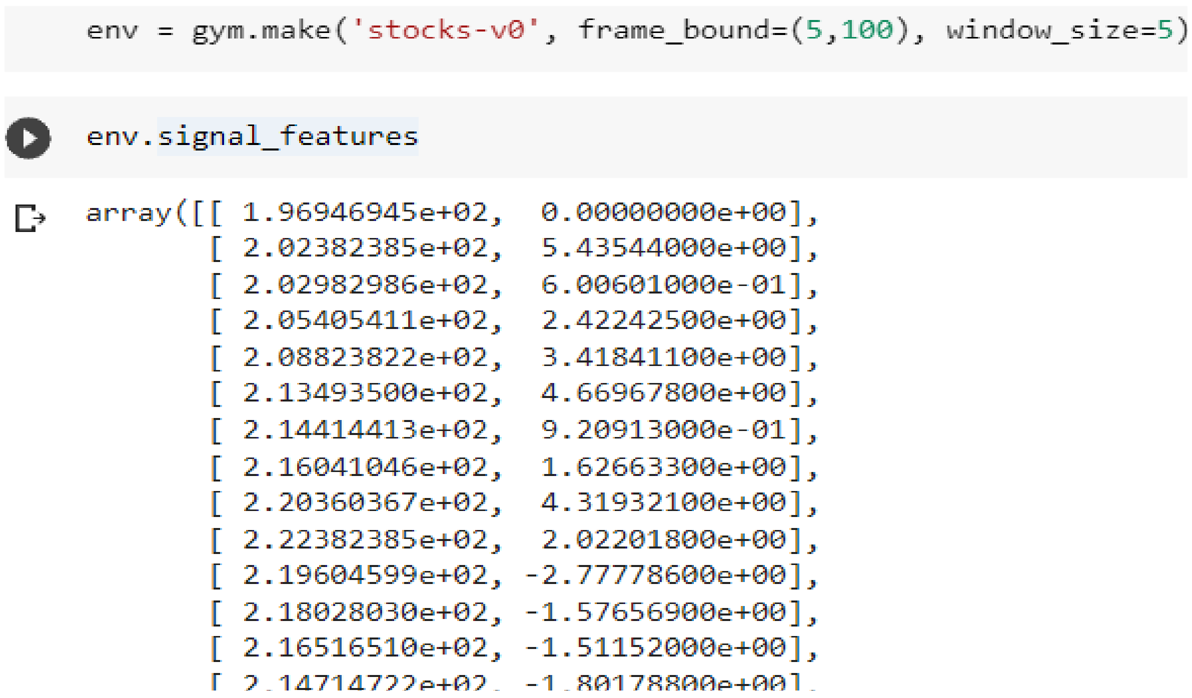
```
date            datetime64[ns]
1. open                float64
2. high                float64
3. low                 float64
4. close               float64
5. volume              float64
dtype: object
```

```
df.set_index('date', inplace=True)
df.head()
```

```
                     1. open   2. high  3. low  4. close  5. volume
date
2022-02-18 20:00:00   285.78  285.8500  285.70    285.80     8385.0
2022-02-18 19:55:00   285.80  285.8500  285.80    285.85     2682.0
2022-02-18 19:50:00   285.80  285.8500  285.80    285.85     2008.0
2022-02-18 19:40:00   285.75  285.7500  285.75    285.75     1738.0
2022-02-18 19:35:00   285.80  285.8201  285.75    285.75     1585.0
```

4.2.2 Building the test environment after creating own environment:

Ideally, the data that you use should mimic the frequency that you want to trade. For example, if you want the RL agent to trade daily data, use the daily data to train the agent and not hourly data. (we’re using 5 min Time Frame)



```
env = gym.make('stocks-v0', frame_bound=(5,100), window_size=5)
```

```
env.signal_features
```

```
array([[ 1.96946945e+02,  0.00000000e+00],
       [ 2.02382385e+02,  5.43544000e+00],
       [ 2.02982986e+02,  6.00601000e-01],
       [ 2.05405411e+02,  2.42242500e+00],
       [ 2.08823822e+02,  3.41841100e+00],
       [ 2.13493500e+02,  4.66967800e+00],
       [ 2.14414413e+02,  9.20913000e-01],
       [ 2.16041046e+02,  1.62663300e+00],
       [ 2.20360367e+02,  4.31932100e+00],
       [ 2.22382385e+02,  2.02201800e+00],
       [ 2.19604599e+02, -2.77778600e+00],
       [ 2.18028030e+02, -1.57656900e+00],
       [ 2.16516510e+02, -1.51152000e+00],
       [ 2.14714722e+02, -1.80178800e+00],
```

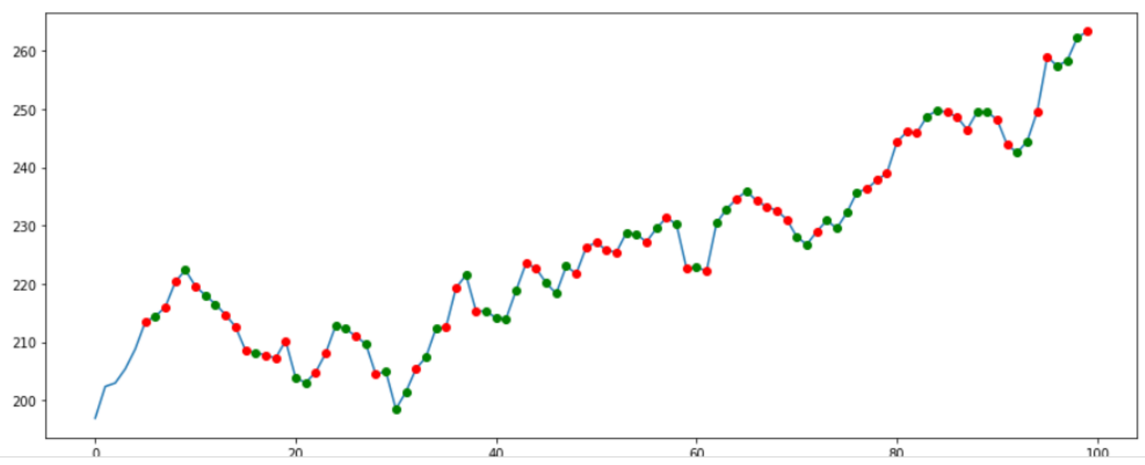
If we look at the actions, we can take using environment.action_space, we'll notice that we only have two actions we can take. We can only Short or Long. In other algorithms, you can Hold. Visualizing the environment using matplotlib.

```
[ ]  state = env.reset()
     while True:
         action = env.action_space.sample()
         n_state, reward, done, info = env.step(action)
         if done:
             print("info", info)
             break

     plt.figure(figsize=(15,6))
     plt.cla()
     env.render_all()
     plt.show()

     info {'total_reward': 11.491531999999978, 'total_profit': 0.7273767463315592, 'position': 0}
```

Total Reward: 11.491532 ~ Total Profit: 0.727377



4.2.3 Training an RL agent to trade using the Gym environment:

We begin by wrapping our environment inside the dummy vectorized environment wrapper, DummyVecEnv. Creating an env_build function. We are taking that function and putting it inside the Dummy RecEnv. Finally, we save the result inside the env variable so that when we start building our training model. We'll now use the env variable.

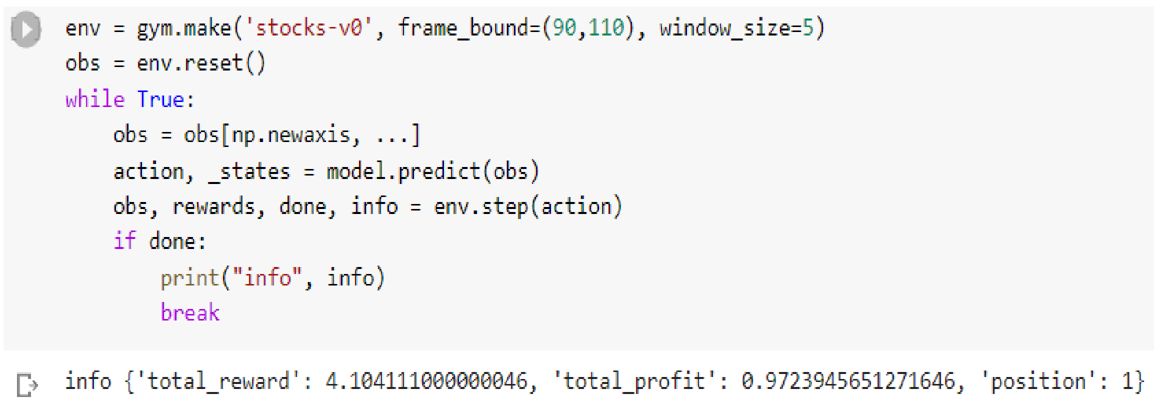
```
env_maker = lambda: gym.make('stocks-v0',  frame_bound=(5,100), window_size=5)
env = DummyVecEnv([env_maker])
```

```
[ ]  model = A2C('MlpLstmPolicy', env, verbose=1)
     model.learn(total_timesteps=10000)
```

```
---------------------------------
| explained_variance | -0.909   |
| fps                | 21       |
| nupdates           | 1        |
| policy_entropy     | 0.693    |
| total_timesteps    | 5        |
| value_loss         | 7.41     |
---------------------------------
---------------------------------
| explained_variance | -23.4    |
| fps                | 381      |
```

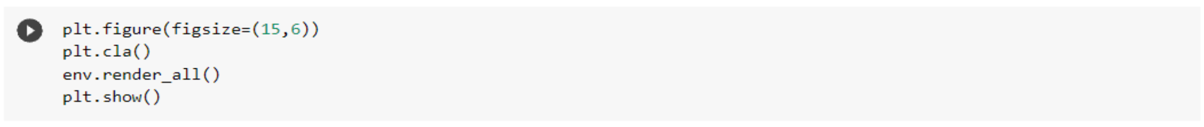
4.2.4 Output Obtained:

As we can see, our agent RL bought and sold stocks at random. Our profit margin appears to be greater than 1, so we can determine that our bot has made us profit from the trades it has made. But these were random steps, now let’s properly train our model to get better trades. The model isn’t perfect. It has made some long and short trades. Some good and bad trades . With a few tweaks, this model can be trained to trade with stocks, forex, equities, and securities.

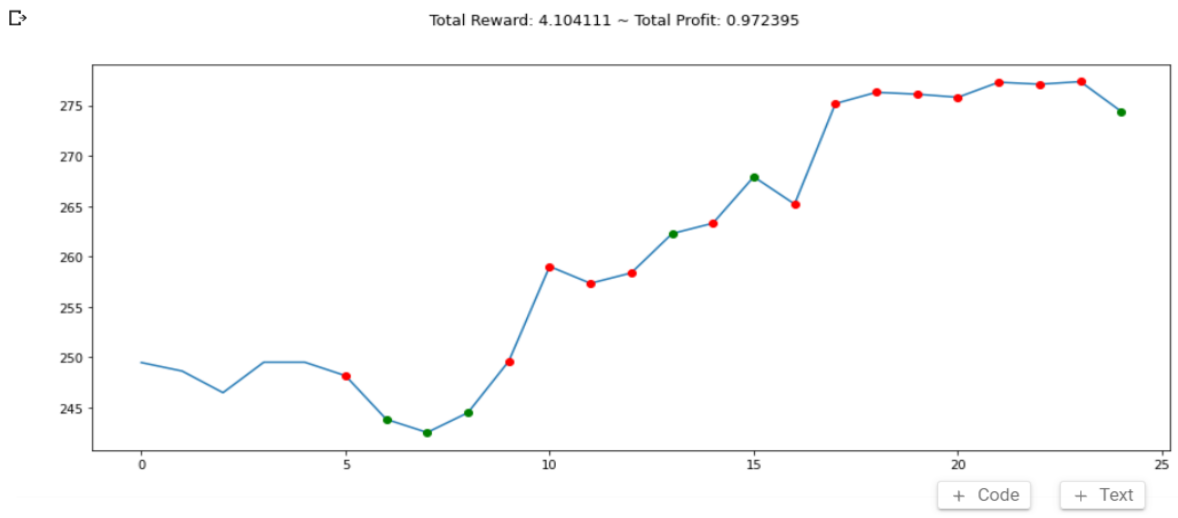


```
env = gym.make('stocks-v0', frame_bound=(90,110), window_size=5)
obs = env.reset()
while True:
    obs = obs[np.newaxis, ...]
    action, _states = model.predict(obs)
    obs, rewards, done, info = env.step(action)
    if done:
        print("info", info)
        break
```

```
info {'total_reward': 4.104111000000046, 'total_profit': 0.9723945651271646, 'position': 1}
```



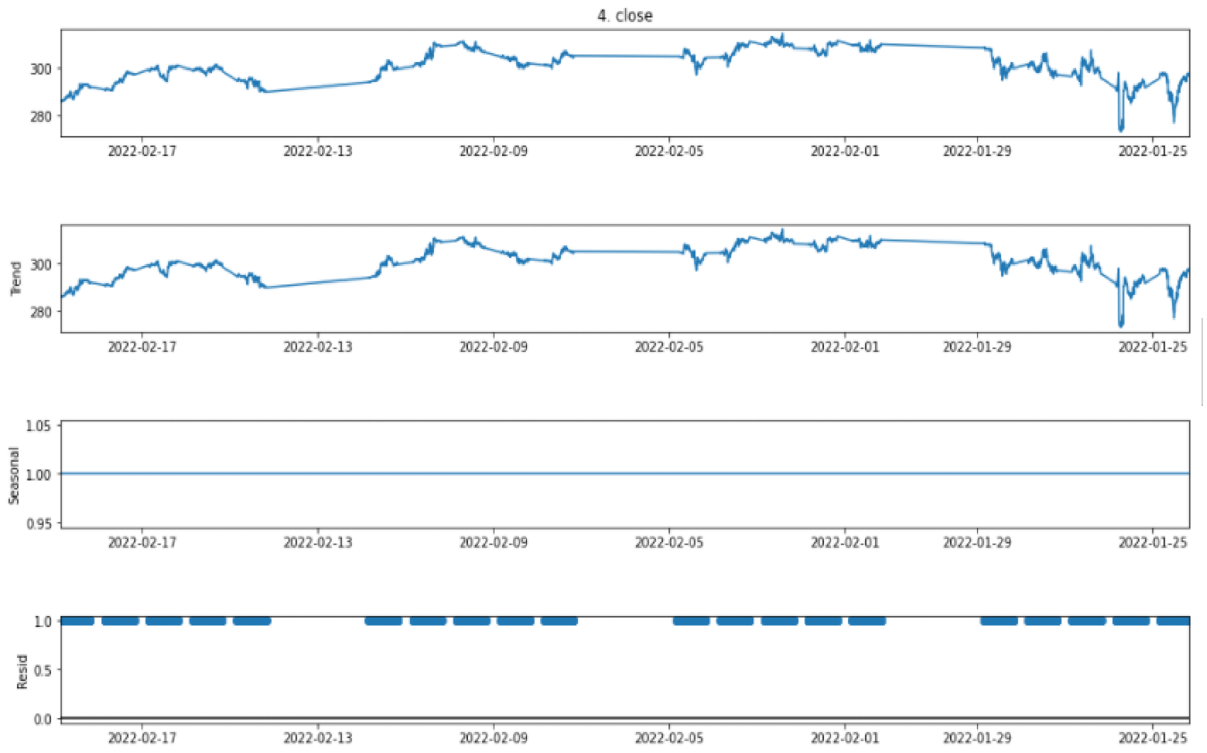
```
plt.figure(figsize=(15,6))
plt.cla()
env.render_all()
plt.show()
```



4.3 TIME SERIES

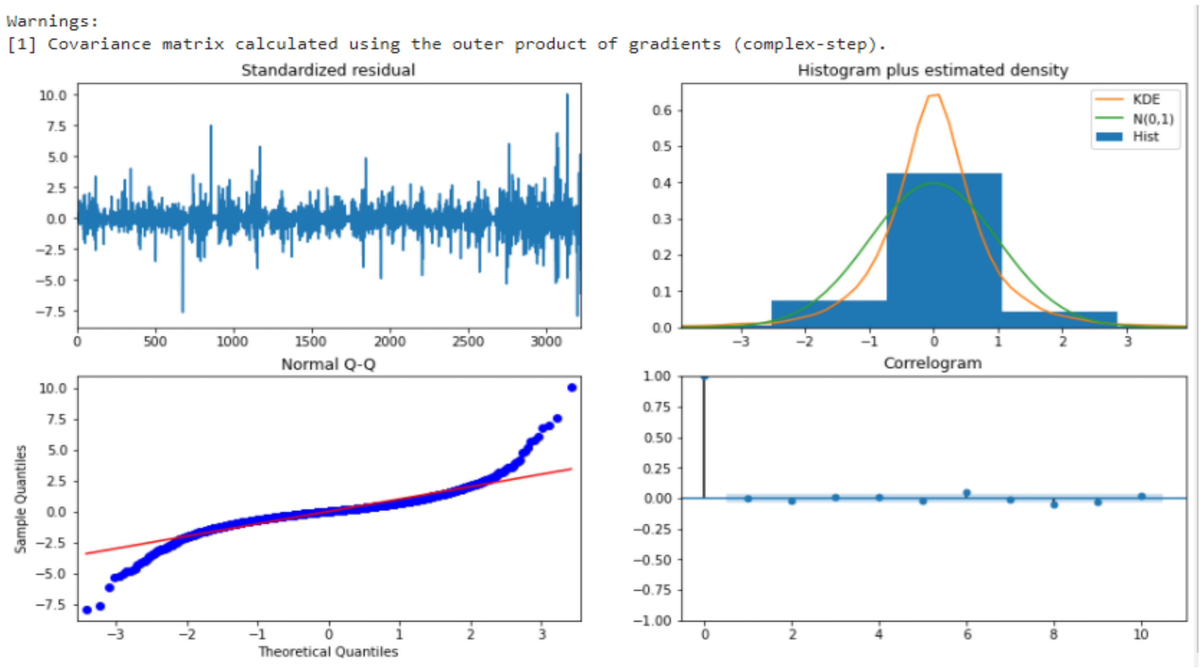
A time series is a data set that tracks a sample over time. In particular, a time series allows one to see what factors influence certain variables from period to period. Before working with non-stationary data, the Autoregressive Integrated Moving Average (ARIMA) Model converts it to stationary data. One of the most widely used models for predicting linear time series data is this one. The ARIMA model has been widely utilized in banking and economics since it is recognized to be reliable, efficient, and capable of predicting short-term share market movements.

4.3.1 Separating trend and the seasonality from the time series:



4.3.2 Visual summary of ARIMA model:

Separating the trend and the seasonality from a time series we can decompose the series



```
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

4.3.3 Final trained ARIMA model:

```
import statsmodels.api as smapi

model = smapi.tsa.arima.ARIMA(train_data, order=(0,1,1))

result = model.fit()
print(result.summary())
```

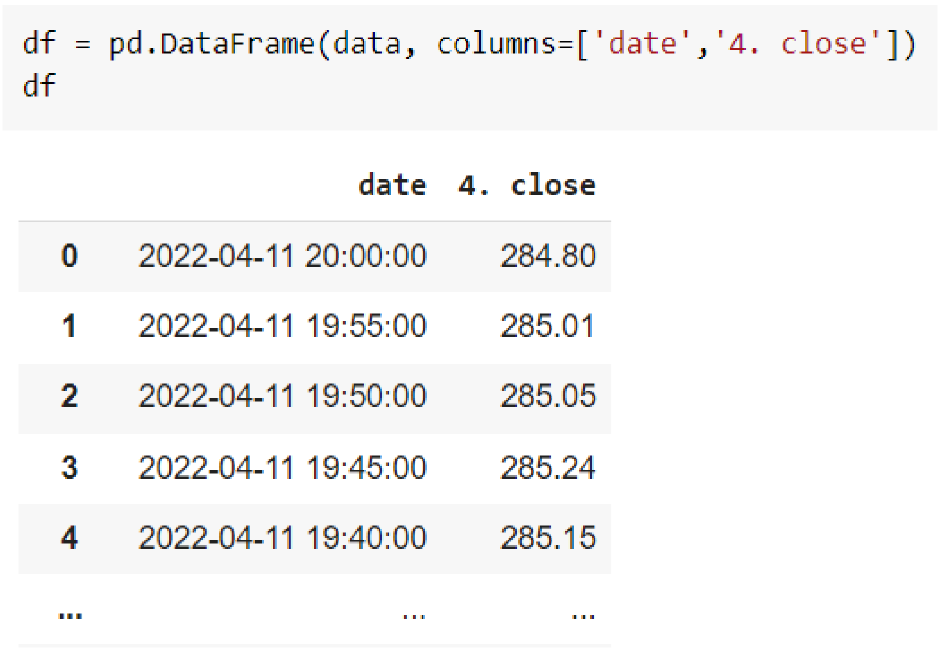
```
                               SARIMAX Results
==============================================================================
Dep. Variable:               4. close   No. Observations:                 3222
Model:                 ARIMA(0, 1, 1)   Log Likelihood               15768.718
Date:                Mon, 21 Feb 2022   AIC                         -31533.436
Time:                        15:38:55   BIC                         -31521.282
Sample:                             0   HQIC                        -31529.080
                               - 3222
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ma.L1         -0.0884      0.010     -9.065      0.000      -0.108      -0.069
sigma2      3.275e-06   3.09e-08    105.815      0.000    3.21e-06    3.34e-06
===================================================================================
Ljung-Box (L1) (Q):                   0.00   Jarque-Bera (JB):             19111.30
Prob(Q):                              0.99   Prob(JB):                         0.00
Heteroskedasticity (H):               2.34   Skew:                             0.24
Prob(H) (two-sided):                  0.00   Kurtosis:                        14.92
===================================================================================
```

4.4 LINEAR REGRESSION

Linear regression is used for predictions with data that has numeric target variables. During prediction we use some variables as dependent variables and few considered as independent variables.

Select Subset with relevant features

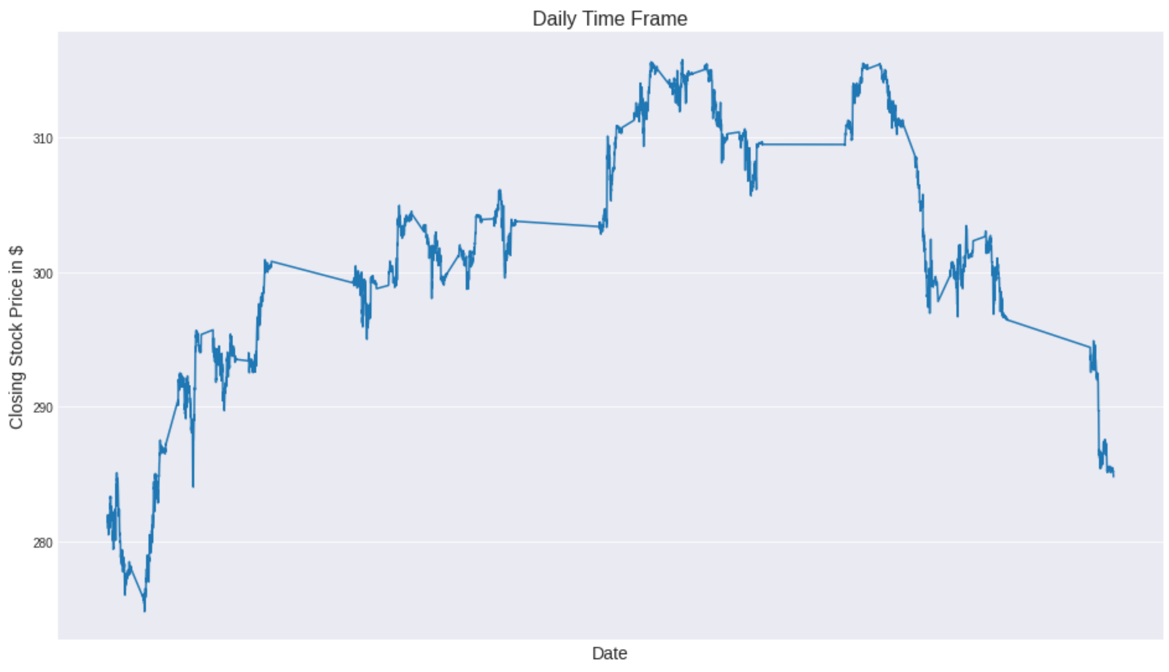
We use the 5 min Time Frame closing price Close as the value to predict, so we can discard the other features. ‘Close' column has numeric data type, The 'Date' is the index column and contains datetime values.



```
df = pd.DataFrame(data, columns=['date','4. close'])
df
```

	date	4. close
0	2022-04-11 20:00:00	284.80
1	2022-04-11 19:55:00	285.01
2	2022-04-11 19:50:00	285.05
3	2022-04-11 19:45:00	285.24
4	2022-04-11 19:40:00	285.15
...	...	...

When we take a look at the price movement over time by simply plotting the Closing price vs Time, we can already see, that the price continuously increases over time and we can also estimate that trend could be linear



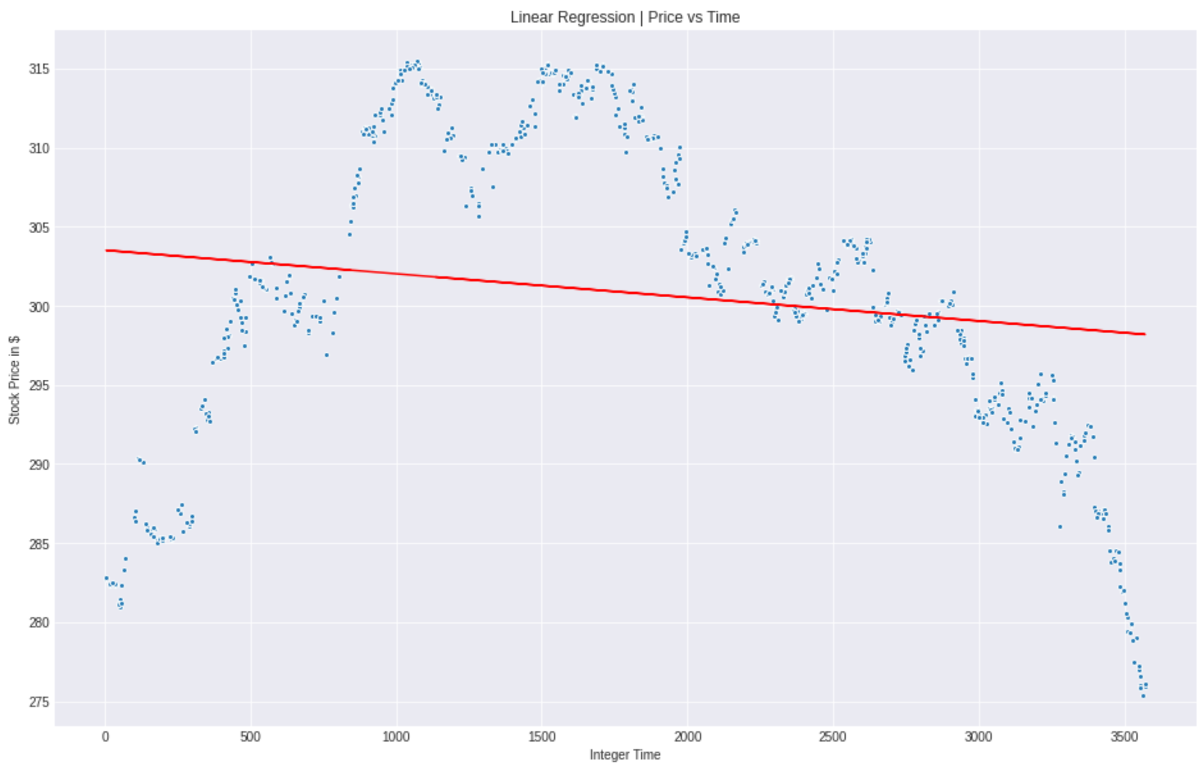
4.4.1 Training Linear Regression Model:

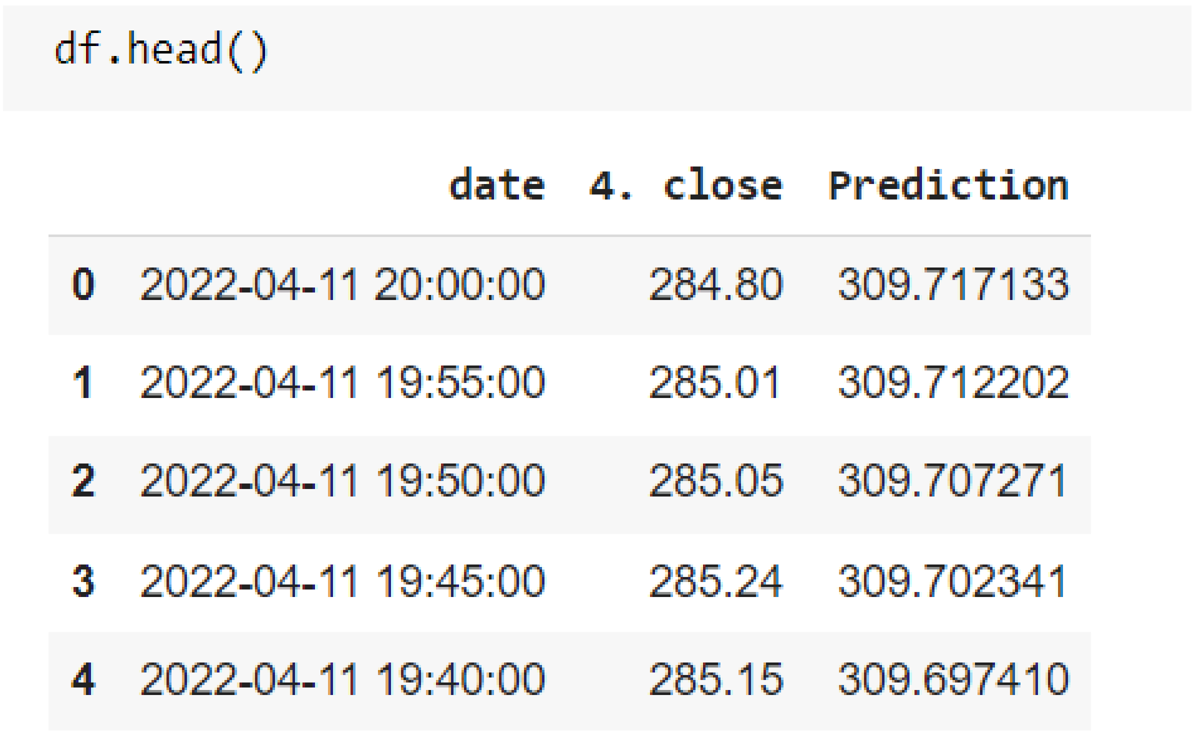
```
plt.figure(1, figsize=(16,10))
plt.title('Linear Regression | Price vs Time')
plt.plot(X_test, model.predict(X_test), color='r', label='Predicted Price')
plt.scatter(X_test, y_test, edgecolor='w', label='Actual Price')

plt.xlabel('Integer Time')
plt.ylabel('Stock Price in $')

plt.show()
```

4.4.2 Stock price V/S Integer time graph:





```
df.head()
```

```
                 date  4. close  Prediction
0 2022-04-11 20:00:00    284.80  309.717133
1 2022-04-11 19:55:00    285.01  309.712202
2 2022-04-11 19:50:00    285.05  309.707271
3 2022-04-11 19:45:00    285.24  309.702341
4 2022-04-11 19:40:00    285.15  309.697410
```

So, this was all about the implementation of the four *Machine Learning Algorithms*. Now let's compare and conclude all the four algorithms and have a look at what can be done in the future to improve the performance of the model in the next chapter.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 CONCLUSION:

The patterns used for testing are Ascending Triangle and Bullish Double Bottom. The API used was Alpha Vintage. After comparing all the four Algorithms and collecting the respective data, we came to a Conclusion that CNN was the best among these four for the following method.

The Model we used was a Keras2d model. The comparison was done with the Dataset which consisted of more than 100+ images. The highest accuracy obtained by CNN is about 77%.

.

Name of Algorithm	**Accuracy Obtained**
CNN	77%
Time Series	54%
Reinforcement Learning	47%
Linear Regression	20%

The Model we used was a Keras 2D model. The comparison was done with the Dataset which consisted of more than 100+ images. The highest accuracy obtained was by CNN of about 77%.

5.2 FUTURE WORK

We currently worked only on two patterns in our project; we have proposed a way to find the best possible intraday trade on the basis of Chart Patterns. We have used four Machine Learning Algorithms for the same. The following are :

1. CNN
2. TIME SERIES
3. REINFORCEMENT LEARNING
4. LINEAR REGRESSION

The patterns used for testing are Ascending Triangle and Bullish Double Bottom. To get more efficient trades which was the main goal of the whole project we can Implement various different patterns.

By adding different Patterns, we can improve the accuracy in many cases. We can add a clear nature indication after adding more than 8 patterns. There should be 4 from the Bullish side and 4 from the Bearish side. After implementing these 8 patterns we can add an indicator which can predict the overall nature of the upcoming trend. Going further we can also get the actual information about which pattern is present.

This model can also be further used for doing long term, midterm, short term trades. It can also be done for a larger time frame to obtain results. Based on these results, investment strategies can be planned accordingly.

This was the conclusion and the future work we are planning to add on this project. Now let’s have a look at the references taken for this project.

CHAPTER 6: REFERENCES

[1] Chavan, P. S. & Patil, S. T. 2013. Parameters for stock market prediction. International Journal of Computer Technology and Applications, 4(2), 337.

[2] Art Paspanthong , Nick Tantivasadakarn , Will Vithayapalert. Spring 2019. Machine learning in intraday stock trading. Stanford University.

[3] Kranthi Sai Reddy Vanukuru. 2018.Stock market prediction using machine learning. Srineedhi Institute of Science & Technology.

[4] M Umer Ghani, M Awais and Muhammad Muzammulla. 2019. Stock market prediction using machine learning (ML) algorithms. Department of Software Engineering, Government College University, Faisalabad.

[5] K. Hiba Sadia, Aditya Sharma, Adarrsh Paul, SarmisthaPadhi, Saurav Sanyal. Stock market prediction using machine learning algorithms. International Journal of Engineering and Advanced Technology (IJEAT). ISSN: 2249 – 8958, Volume-8 Issue-4, April 2019

[6] Hu, Z.; Zhao, Y.; Khushi, M. A survey of Forex and Stock price prediction using deep learning. Appl. Syst. Innov. 2021, 4, 9.