STOCK PRICE PREDICTION USING REINFORCEMENT LEARNING AND GENETIC SEQUENCE ALGORITHM

Major Project Report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

Nitin 181275

UNDER THE GREAT SUPERVISION OF

Dr. Kapil Rana

Department of Computer Science & Engineering and

Information Technology

Jaypee University of Information Technology, Waknaghat,

173234,

Himachal Pradesh, INDIA

CONTENTS

DECLARATION

We hereby declare that this project has been done by me under the supervision of Dr. Kapil Rana of the Jaypee University of Information Technology. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

Dr. Kapil Rana
Assistant Professor (SG)
Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology

Submitted by:

Nitin
181275 CS 83
Computer Science & Engineering Department
Jaypee University of Information Technology

III

CERTIFICATE

This is to certify that the work which is being presented in the project report titled "STOCK PRICE PREDICTION USING REINFORCEMENT LEARNING AND GENETIC SEQUENCE ALGORITHM
" in partial fulfilment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by **Nitin (181275)** during the period from January 2022 to May 2022 under the supervision of Dr. Kapil Rana, Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

The above statement made is correct to the best of our knowledge.

Dr. Kapil Rana

Assistant Professor(SG)

Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat

IV

ACKNOWLEDGEMENT

Firstly, We express our heartiest thanks and gratefulness to Almighty God for His divine blessing that makes it possible to complete the project work successfully. We would also like to thank Jaypee University of Information Technology and the Department of Computer Science and Information Technology for giving us this opportunity.

We are grateful and wish our profound indebtedness to Supervisor Dr. Kapil Rana, Associate Professor, Department of CSE Jaypee University of Information Technology, Wakhnaghat. Deep Knowledge & keen interest of our supervisor in the field of Reinforcement Learning and Genetic Sequence Algorithm carrying out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express my heartiest gratitude to Dr. Kapil Rana, Department of CSE, for his kind help to finish our project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, We might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, We must acknowledge with due respect the constant support and patience of our parents, guardians and siblings.

Nitin

V

ABSTRACT

In this era with advent of technology people are predominantly using machine learning, deep learning and big data for predicting stock Market trends and the practice has become more prominent than ever because stock markets have fetched a lot of attention recently with the advent of new mobile applications which have made the trading process for one very easy breaking all the previous stereotypes. Currently millions of millennials are piling into the stock market. Recently even in this pandemic era market capitalization reached a record 95 trillion dollars setting an all time record [1].

In our work we have used Reinforcement learning algorithm and Genetic Sequence Algorithm for Stock Price Prediction and obtained a mean square error of as low as 0.0428.

VI

Chapter 01

INTRODUCTION

1.1 PROJECT INTRODUCTION

Forecasting stock prices has been a challenging task for business analysts, researchers and individuals for quite some time now. Predicting stock prices has proved to be an interesting and challenging field of research in recent times because of the various external factors such as social, economical, political and psychological factors proving to have such an influential margin. Most of the factors associated with the stock market are non linear and vary with time thus predicting market trends can be a tricky affair.

An investor needs to have appropriate knowledge before investing in stocks. If an individual lacks appropriate knowledge the investor might have to deal with great financial loss. There are various prediction techniques being used for appropriately predicting stocks. Mainly two methods are conventionally known for predicting stocks which do not require any computational techniques for risk analysis. The two widely known methods are Fundamental and Technical Analysis.

Fundamental Analysis: For determining the precise value of a product one needs to have an accurate and reliable financial report from the company, it's very important to know about the financial situations and competitive robustness of a company [2]. The data is needed to make an investment decision. "If intrinsic value exceeds the market evaluation it holds, invest otherwise and keep away from it as a substandard investment ”. There are other parameters like book value, ROI, earnings etc [3, 4]. All the factors should be carefully evaluated for making a viable business decision. Fundamental analysis is mostly useful in the case of predicting long term. It’s mostly useful because of it’s structured approach and capability to forecast modifications [5].

Technical Analysis: In Technical Analysis the decision is made by investors on the basis of continuously altering factors with respect to different factors. Different quantitative parameters are used generally to make predictions such as indices, daily ups and downs and stock value etc. Rules can be extracted from the data which are further used by investors to make predictions. Same chart can result in different predictions [6]. Technical analysis is useful in both long and short term analysis. It is also preferred over fundamental analysis data as system input.

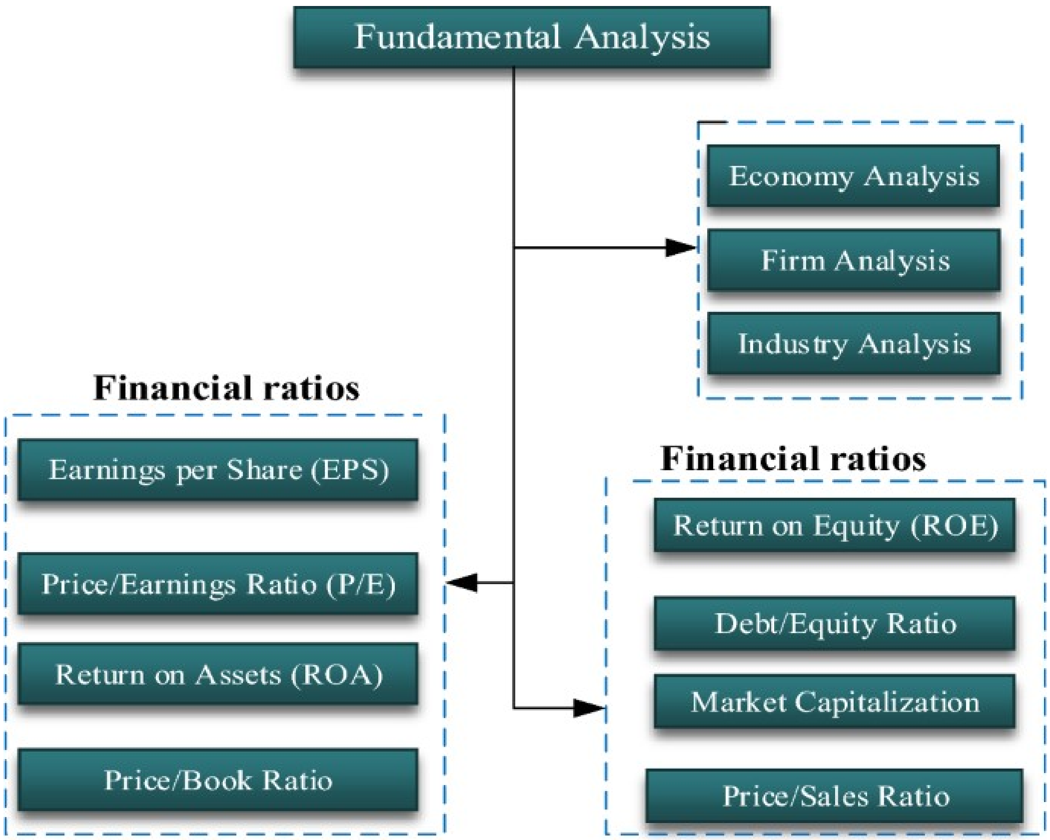


Fig 1. Analysis of Stock market

1.2 OBJECTIVES

1.2.1 GROUND WORK IN STOCK MARKET

In the stock market investors and buyers purchase and sell the shares of a company at a mutually obtained price. These prices depend on supply and demand. Basically share is a document issued by a company entitling the buyer to be the owner of the company. The buyer gets a dividend which can further be sold to earn profit. Stock Exchanges are for guaranteeing the payment thus ensuring security to the buyer. All these operations ensure the growth of the company by increasing production and employment. Stock must be listed to ensure its exchange. There are some constraints imposed on the companies who wish for getting listed. Stockbrokers are the ones having access to the stock data and are licensed to trade shares in the process charging a service fee. There are various financial instruments in which traders deal such as stocks and bonds. Stock market can be categorised into (1) Primary Market (2) Secondary Market.

1.3 MOTIVATION

India is the home to the third largest stock market in the world. Stock can basically be defined as ownership in the company. They are a partisan part of a company's ownership. For example: If a company's ownership is distributed into 100 parts and an investor buys a share of the company then that individual owns 1 % of that company. Stock exchange is an automated system which works on the basis of order. In case the number of people buying exceeds the number of people selling, the price of the stock will increase and if the number of sellers exceeds the number of buyers, the price will decrease. The best buy order is when a certain stock reaches its peak price and best sell order is when a certain stock reaches its lowest price. The most profitable buy order and best sell order are basically counterparts to each other. In 2008 the derivatives total world market was evaluated at $791 trillion which is eleven times the magnitude of the world's economy [8].

1.4 LANGUAGES USED

This project makes use of the Python programming language for its implementation . The selection of Python was based on the availability of numerous inbuilt libraries, packages and functions. For this specific project python has many libraries like yfinance , keras , stremlit etc which makes the implementation and deployment process smooth.Libraries like plotly and matplotlib are used for plotting graphs for the results obtained.

Chapter 02

Major Project SDLC

2.1 FEASIBILITY STUDY

2.1.1 FINANCIAL FEASIBILITY

Our project does not take up any major financial investment, we just needed a decent computer system to make the machine learning models. Our app can work with the most basic of the smartphones, not being a financial burden on anyone. In the future we could only need PCs with better performance so that we can train our models on various other datasets thus making our prediction a little more accurate. The funds mostly involved will be one required for investing.

2.1.2 TECHNICAL FEASIBILITY

Stock investment is a need of an hour be it an engineer, doctor or business or in fact any working professional. From industrialist to common man everyone can or needs to invest in stocks because of the upcoming mobile applications which have made the process of trading very easy. Currently the primary investment tools like gold, bank FDs not keeping on mark with market inflation rates the stock investment becomes a much viable option. An app with a very basic User Interface can be used by anyone and can predict stock markets. We have used state of the art machine learning models which have comparatively boosted up the accuracy giving the most accurate predictions.

2.2 REQUIREMENTS

2.2.1 FUNCTIONAL REQUIREMENTS ON MAJOR PROJECT

The functional requirements are the specific demands the system should offer to the end-user so that users can accomplish their tasks.

The functional requirements of this project are mentioned below.

- The project provides an analysis of the stocks for the past years.
- The yfinance package takes the information from yahoo finance and presents it in csv format.
- The reinforcement learning algorithm and genetic algorithms should provide the predictions.
- The basic functionalities of the web app like slider bar for the number of years and plotting the result should be displayed on the localhost address.

2.2.2 NON-FUNCTIONAL REQUIREMENTS ON MAJOR PROJECT

The non-functional requirements are the non-mandatory requirements that enhance the quality of the project. Some are listed below:

i. The project provides an accuracy of mean squared error of 0.310 which is quite close to the targeted value.

ii. The project can collect all stock data and make predictions in a few seconds.

2.3 Use-Case DIAGRAM:

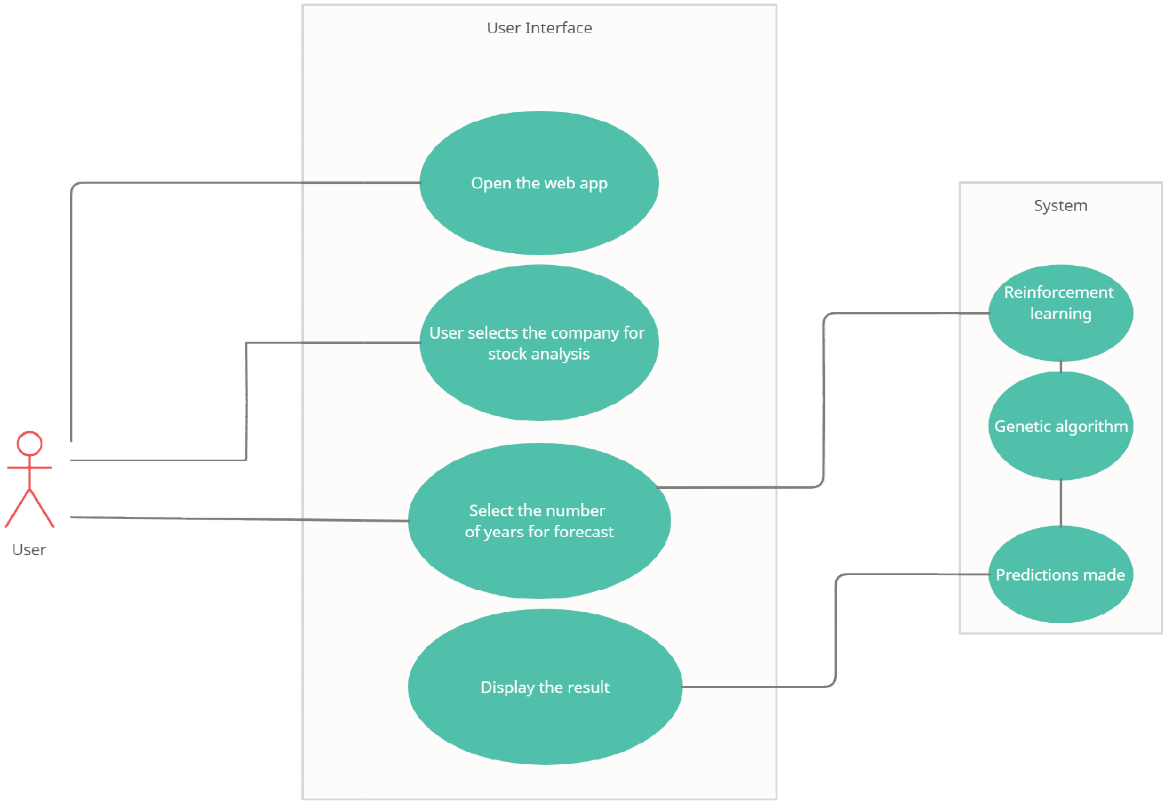


Fig 2. The Use case diagram of major project

2.4 DFD DIAGRAM:

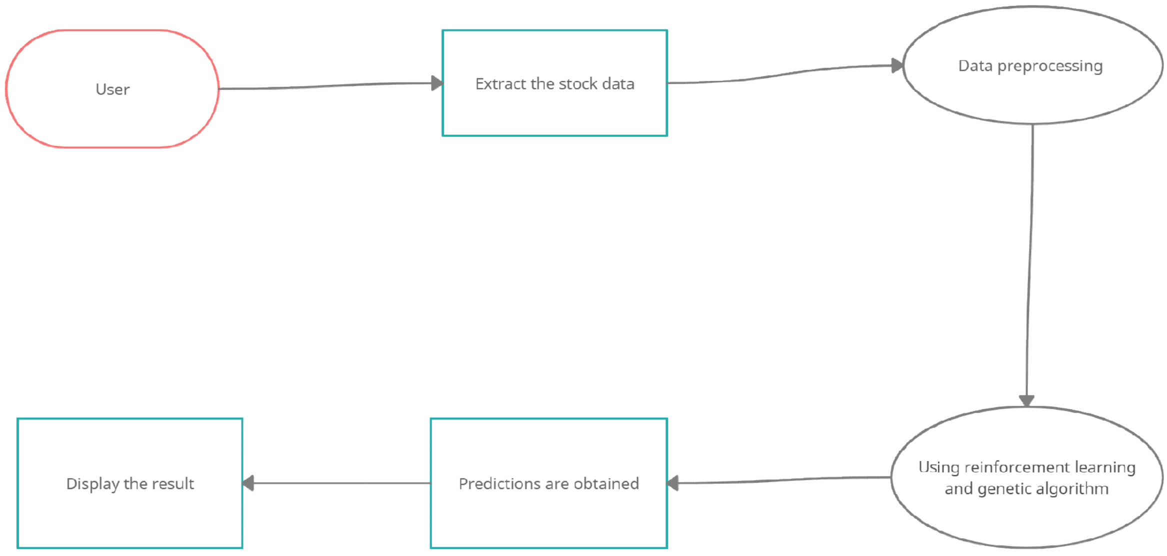


Fig 3. Data flow diagram of major project

2.5 Prediction Techniques

Below mentioned are different methods for stock prediction and their comparative analysis.

2.5.1 Holt winters

2.5.2 Artificial neural network

2.5.3 Hidden markov model

2.5.4 ARIMA model

2.5.5 Time series linear model

2.5.6 Recurrent neural Network

2.5.1 Holt winters

Holt-winters [9] provides best use case where trend and seasonal factors are involved. Holt-winters basically have two variants:

1) Additive Holt-Winters
2) Multiplicative Holt-Winters

Additive Holt-Winters is used for prediction tasks and Multiplicative Holt-Winters is used when there is no persistent seasonal variation inside the series. Holt-winters show great accuracy in prediction and outperform various models. In case of short-term forecast the method used most frequently is Holt Winters a smoothing method which includes seasonal fluctuations and trends. Following removal of seasonal trends from the dataset a below mentioned function is returned and pre-calculation is made necessary for forecasting.

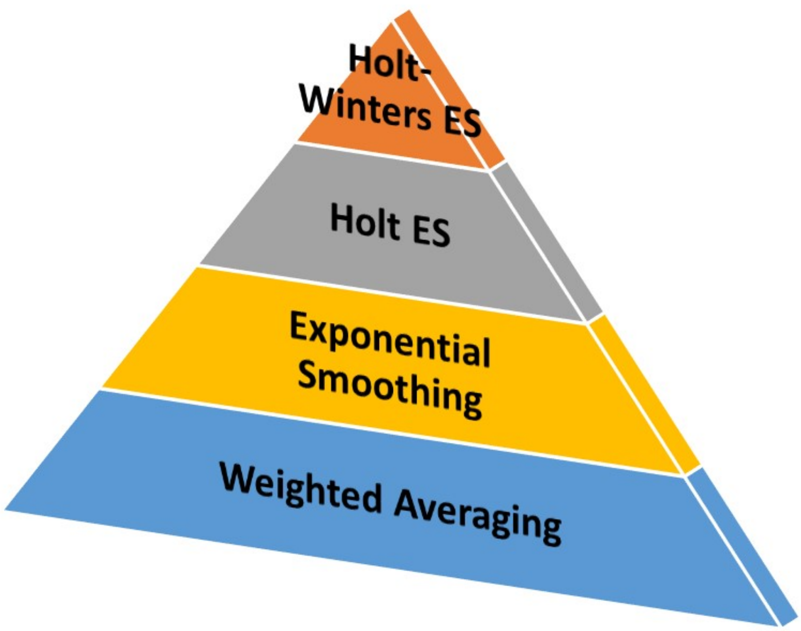


Fig 4 Holt winter hierarchy triangle

2.5.2 Artificial neural network

An Artificial neural network [10] is designed by taking inspiration from the neurological mechanism of the brain. It provides a great way to predict from large chunks of data. Using the method of backpropagation , ANN is mostly used to determine the stock prices' spikes and lows. Backpropagation uses the neural network of a multi-layer perceptron. The input layer involved is formed from a set of input nodes, various hidden layers of nodes as well as nodes in the output layers. The networks are often formed with raw data , such networks often make use of data derived from the previously discussed fundamental and technical analysis. A multilayer neural network is a network consisting of a number of hidden layers and an output layer at the end . The weights are used to propagate forward in the network and make predictions. The backpropagation algorithm is used to rectify the value of weights based on provided output so the result can be closest to the target value.

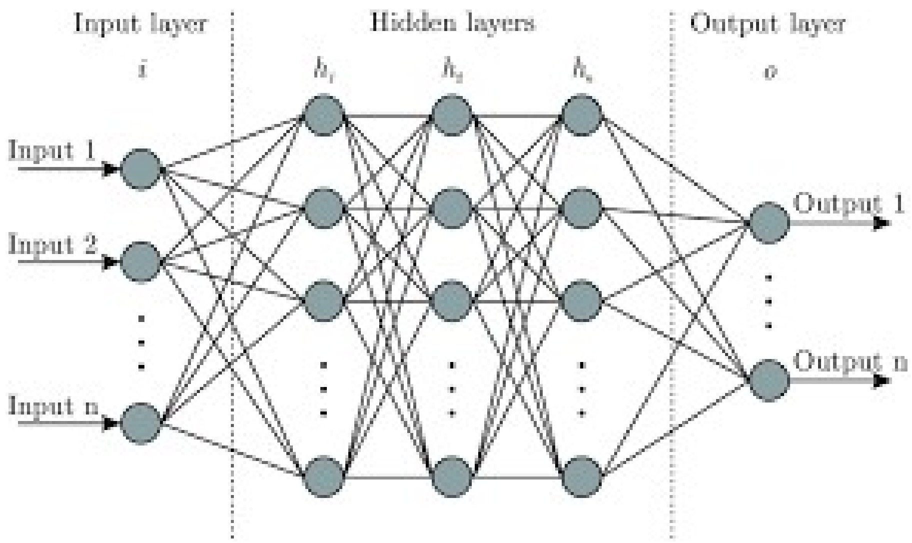


Fig 5 Structure of a neural network

2.5.3 Hidden markov model

The Hidden markov model [11] was first used for speech recognition tasks but is heavily used for stock prediction tasks currently.The simple analysis of trend of the market based on Hidden markov model because of the one day difference in closing value for a provided time duration. The model works in a way that finds hidden patterns of the state of the stocks and probability values accordingly. HMM, a stochastic method, is usually taken to be the Markov process with hidden states. The parameters in the Hidden markov model are A, B and p. The Hidden Markov Model usually provides better optimization but the problem comes while evaluating and decoding.

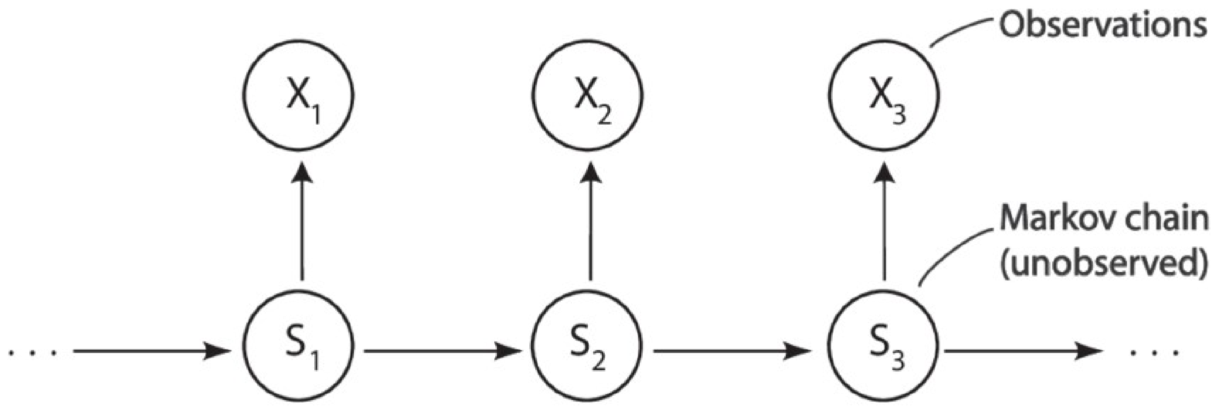


Fig 6 Hidden Markov Model

2.5.4 ARIMA Model

The ARIMA model is used for identifying, estimating and diagnosing ARIMA models using time series data. It is the most important forecasting method. ARIMA models are a very important financial forecasting method. It is generally a linear summation of previous values and errors. ARIMA model has comparatively very small standard error and sturdy forecasting technique of financial time analysis but the model is usually suitable for short term predictions.

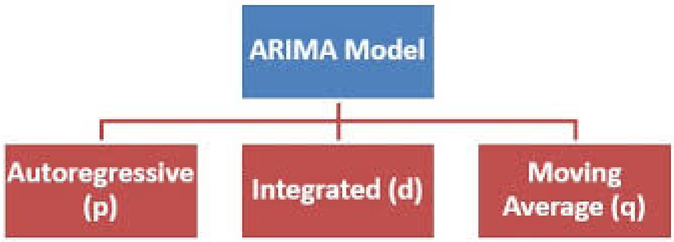


Fig 7 Arima model hierarchy

2.5.1.5 Time Series Linear Model

The linear time series model consists of the dataset which is initially corporated and the model created reflects the property of the dataset. One of the advantages of LTSM is the actual dataset is inserted in the model using both traditional and seasonal data trends. tslm() is used in R programming to implement linear models and StlStock is used to remove seasonal trends. One of the parameters h indicates the number of predicted months.

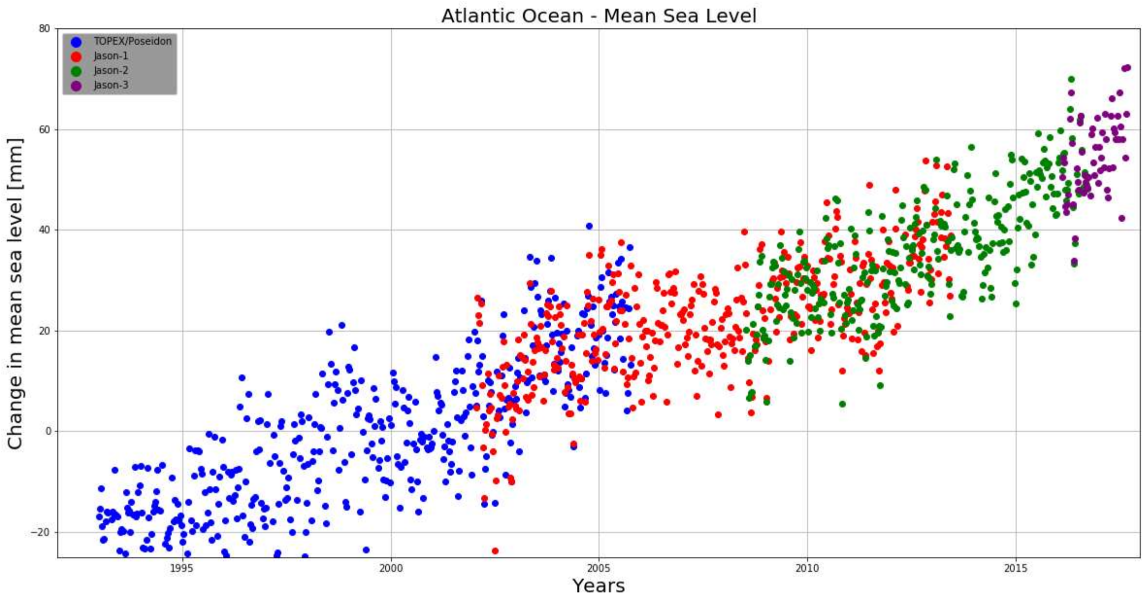


Fig 8 Time series model

2.5.6 Recurrent Neural Network

The learning in Recurrent Neural Networks [K] takes place using back propagation, but the nodes work with feedback. Thus RNN models easily predict stock prices from the data received from recent history. The main advantage it has is of it's past time points to the input layer. Although the disadvantage in Rnn is that all those words are fed via a smaller bunch of nodes.

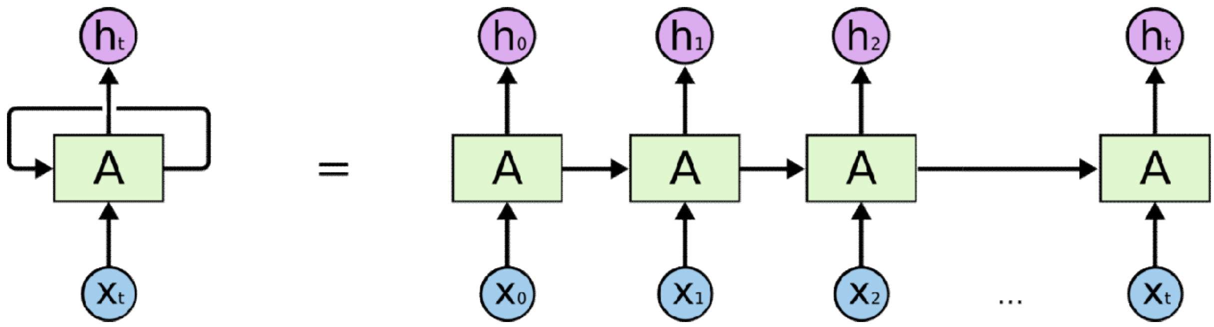


Fig 9 RNN model

2.6 Related Work

Aparna Nayak [13] in her research paper using Boosted Logistic Regression, Decision Tree and Support Vector Machine obtained an accuracy of upto 70%. Yangi Li [14] in his research paper proposed a deep learning model where he was able to increase the recall by 50% ,precision by 40% and F1-Score by 44.78%.

Waisat khan [15] in his research paper did a comparative study with and without political situation using various machine learning algorithms like Naive Bayes, Decision Tree and Multi Layer Perceptron where he observed an increase in accuracy by around 0-3% and using political situation feature observed an increase in accuracy by around 20%. Guangyu Ding [16] in his research paper used a Long Short-Term memory network and deep neural network model and achieved an accuracy of 95%.

Ya Gao [17] used LASSO, Principal Component Analysis, LSTM and GRU in prediction of stock prices and achieved an accuracy of around 68%.

Manish Agarwal [18] in his research paper built an evolutionary deep learning model for prediction of stock prices. The model was applied on HDFC, Yes bank and State Bank of India dataset and achieved an accuracy of 63.59%, 56.25% and 57.95%.

Yauheniya Shynkevich [19] used sub-industry-specific and stock-specific news articles to predict stock prices. They also noticed that using two categories of new improves prediction accuracy in comparison methods.

Kim Dongyoung[20] used Machine Learning and sentiment analysis based on SNS and news articles for predicting stock prices. They conducted a 10-fold cross validation method and achieved an accuracy of 80% and F1-Score of 0.8.

Faten Subhi Alzazah [21] used text mining techniques rather than traditional deep learning and machine learning models to predict stock prices and as a result obtained an accuracy of 74.67%. The research paper also contains a comparative analysis of previously published research papers on stock prediction using machine learning and deep learning models.

Chapter 03

Implementation of the Project

3.1 Dataset Used In OUR Project

The dataset for this project was obtained from yahoo finance using yfinance package available in python language. Stock data for Google, Apple, Microsoft and Tesla was obtained from 2018 to present date using the mentioned package. This dataset contains attributes like Date, Opening , Closing , High ,Low etc which were used for prediction purposes.

Date	Open	High	Low	Close	Adj Close	Volume
27-10-2015	31.68	32.19	31.61	31.895	31.11619	1369600
28-10-2015	31.87	31.92	31.405	31.535	30.76498	1199400
29-10-2015	31.155	31.305	31.025	31.12	30.36011	739600
30-10-2015	31.055	31.09	30.465	30.57	29.82354	1271200
02-11-2015	30.555	30.94	30.5	30.64	29.89183	1253600
03-11-2015	30.475	31.12	30.33	30.915	30.16012	1608000
04-11-2015	30.91	30.94	30.45	30.455	29.71135	1160200
05-11-2015	30.425	30.76	30.05	30.1	29.36502	1577200
06-11-2015	30.005	30.34	29.53	29.615	28.89186	1825400
09-11-2015	29.57	29.895	29.055	29.44	28.72113	1241600

Fig 10 Part of the dataset used.

3.2 DATASET PRE-PROCESSING

The data obtained using yahoo finance consisted of many attributes. Out of these attributes the Close attribute was used to make predictions. This attribute was then converted from 0 to 1 range to feed to the algorithms using minmax scaler. The training and test dataset were generated where the training size was 65 percent and the remaining was test data. This data was then split into chunks consisting of 100 entries out of which the last value was taken as target value.

3.4 ALGORITHM

3.4.1 REINFORCEMENT LEARNING

Reinforcement learning is a machine learning technique which can be used in place of stock price prediction. The learning takes place in an agent based surrounding the target being maximising record. Reinforcement learning does not require labelled data like it does in supervised way of learning. Reinforcement learning gels beautifully with primarily newer data. It uses the value function and the calculations are performed based on criteria of policy which is to be considered for action.

The Reinforcement learning is customised as markov process:

- Setting up an environment variable E and state S
- Series of actions being taken by an agent A.
- P(s,s') => P(st+1=s'|st=s,at=a) becomes the probability of transition from a state s to s'
- R(s,s') – Getting rewarded for a set of actions.

3.4.1.1 STEPS FOR PREDICTING STOCK PRICES USING REINFORCEMENT LEARNING

Reinforcement learning can have application in predicting the stock price for a specific stock making use of the concept of learning from previous year's data, while working in an agent-based environment for predicting higher price returns on the basis of live environment. In the proposed method we have applied a Q learning model for predicting the stock price of the HDFC stock.

Algorithm for creating reinforcement learning model:

- Importing all the packages
- Agent creation that makes the decision
- Defining basic functions for reading the data files, editing the input values, defining activation function etc.
- Training the agent
- Performance evaluation of the model

The below figure denotes the Markov Decision process for the Q learning model for stock price prediction.

Q learning model

It consists of a model-free reinforcement learning algorithm which is used in the environment where the agent learns what action to take under a certain situation. After every certain step there is a certain reward available which maximises the expected value beginning from the initial state.

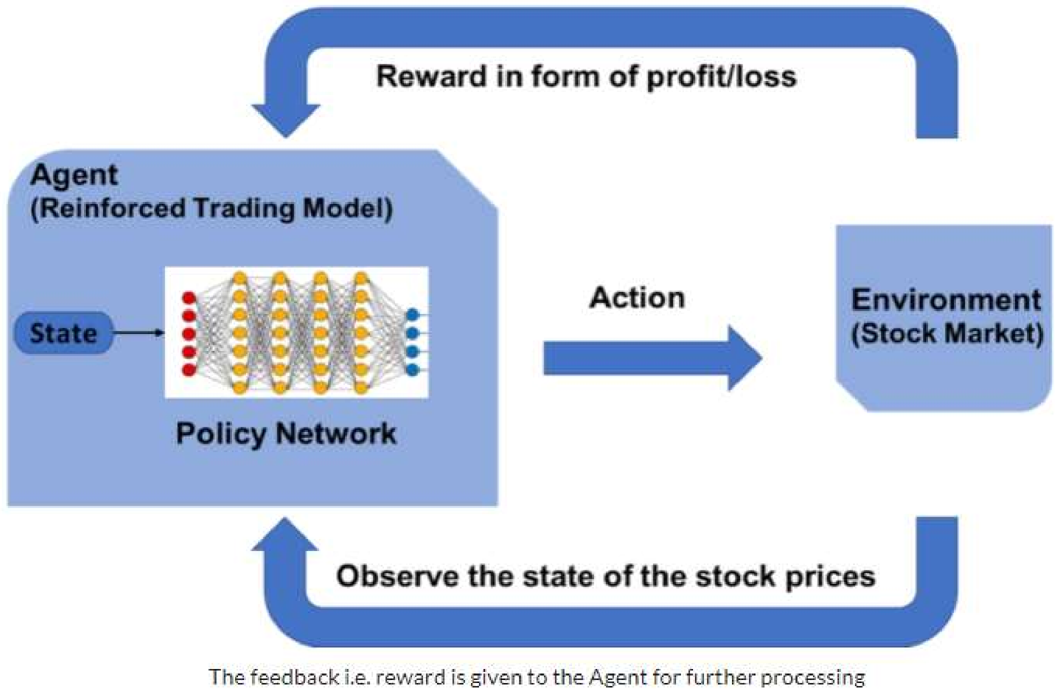


Fig 11. Reinforcement learning flow diagram

3.4.1 Genetic Algorithm

A genetic algorithm is based on a search heuristic approach inspired by Darwin's theory of natural evolution.It is based on principles like heredity ,variation and selection.

This algorithm is based on natural selection of the mates wherein the fittest individuals are selected for mating to produce offspring ,leading to passing down of superior genes. This process is repeated again and again until the target is met.

Genetic Algorithm TERMINOLOGIES

- Fitness Functions:This function helps to determine how fit the individual involved is . Based on this decision the individuals are selected to pass down offspring.
- Individuals: Individual denotes an entity which consists of certain traits based on which they are made to compete. The fittest individuals are then selected for further generations.
- Populations and Generations: A population is defined as a collection of individuals. Same individual can appear more than once inside this collection. On every iteration, the algorithm performs a series of computations on the present population to produce a new generation.
- Diversity :Diversity refers to the deviation of traits of an individual from a specific behaviour. A population has high diversity if the average variation throughout is large else it has low diversity. Diversity is important for genetic algorithms because it enables the algorithm to take more possibilities into account.
- Fitness Values : The fitness value denotes how likely the individual's traits are in sync with the target and hence determines whether the individual will be taken for mating or be filtered out during the iteration of the algorithm.

Darwinian Natural Selection

Here we describe three fundamental principles of the Darwinian natural selection process based on which our algorithm works. This is roughly identical to how natural selection occurs in the real world scenario ,when taking evolution into account.

1. ***Heredity.*** This involves passing down the dominant characteristics to the

next generation.The children get the traits if the parent survives long enough.

2. ***Variation.*** Variation is a way to include new traits in the process. This variation can help to reach the target more efficiently . Lack of variation can lead to identical offspring which are not able to reach the target hence leading to stagnation of the algorithm. New combinations of traits can never occur and nothing can evolve.
3. ***Selection.*** There should be a mechanism wherein some individuals have the opportunity to be parents and pass down their genetic information and some don't. This is typically referred to as "survival of the fittest." Based on the principle that some traits are better for survival and can reach the target in an efficient way.

ALGORITHM

- Initialise.
  Initialise a random population of N elements.
- Selection.
  Process the fitness for those N elements
  Reproduction.
  Choose two parents with probability according to fitness.
  Crossover—Mating is done to obtain a "child" by combining the DNA of these two individuals.
  Mutation—Include mutation in the offspring's DNA based on the given probability.
  Include the new offspring to the new population.

- Based on fitness score select best genes and carry the process of reproduction
- Replace the old population with the new one and return to Step 2.

Web app

A web application is used to display the results in an ordered manner . The web app was hosted on a local host and created using streamlit package in python.

Streamlit

Streamlit is an open-source package in python for building web applications for Data Science and Machine Learning. It can develop web apps and deploy them easily in an instant . Streamlit has a concurrent data flow, whenever there is some change in the code or there is some update, streamlit reruns the whole python script from top to the bottom. This takes place when the user interacts with the widgets like a select box or drop-down box or when there is some change in the source code.

3.5 Screenshots of the various stages of the Project

Importing Libraries

```
import streamlit as st
from datetime import date

import yfinance as yf
from fbprophet import Prophet
from fbprophet.plot import plot_plotly
from plotly import graph_objs as go
```

Extracting data from yfinance

```
START = "2018-01-01"
TODAY = date.today().strftime("%Y-%m-%d")

st.title('Stock Forecast App')

stocks = ('GOOG', 'AAPL', 'MSFT','TSLA')
selected_stock = st.selectbox('Select dataset for prediction', stocks)

n_years = st.slider('Years of prediction:', 1, 4)
period = n_years * 365
```

Data preprocessing

```python
# Predict forecast with Prophet.
df_train = data.reset_index()['Close']

import numpy as np

from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler(feature_range=(0,1))
df_train=scaler.fit_transform(np.array(df_train).reshape(-1,1))

##splitting dataset into train and test split
training_size=int(len(df_train)*0.65)
test_size=len(df_train)-training_size
train_data,test_data=df_train[0:training_size,:],df_train[training_size:len(df_train),:1]

import numpy
# convert an array of values into a dataset matrix
def create_dataset(dataset, time_step=1):
    dataX, dataY = [], []
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step), 0]   ###i=0, 0,1,2,3-----99   100
        dataX.append(a)
        dataY.append(dataset[i + time_step, 0])
    return numpy.array(dataX), numpy.array(dataY)

# reshape into X=t,t+1,t+2,t+3 and Y=t+4
time_step = 100
X_train, y_train = create_dataset(train_data, time_step)
X_test, ytest = create_dataset(test_data, time_step)
```

Applying Reinforcement learning

```
def formatPrice(n):
    return("-Rs." if n<0 else "Rs.")+"{0:.2f}".format(abs(n))
def getStockDataVec(key):
    vec = []
    lines = open(key+".csv","r").read().splitlines()
    for line in lines[1:]:
        #print(line)
        #print(float(line.split(",")[4]))
        vec.append(float(line.split(",")[4]))
        #print(vec)
    return vec
def sigmoid(x):
    return 1/(1+math.exp(-x))
def getState(data, t, n):
    d = t - n + 1
    block = data[d:t + 1] if d >= 0 else -d * [data[0]] + data[0:t + 1] # pad wit
    res = []
    for i in range(n - 1):
        res.append(sigmoid(block[i + 1] - block[i]))
    return np.array([res])
```

```
def __init__(self, state_size, is_eval=False, model_name=""):
    self.state_size = state_size # normalized previous days
    self.action_size = 3 # sit, buy, sell
    self.memory = deque(maxlen=1000)
    self.inventory = []
    self.model_name = model_name
    self.is_eval = is_eval
    self.gamma = 0.95
    self.epsilon = 1.0
    self.epsilon_min = 0.01
    self.epsilon_decay = 0.995
    self.model = load_model(model_name) if is_eval else self._model()
def _model(self):
    model = Sequential()
    model.add(Dense(units=64, input_dim=self.state_size, activation="relu"))
    model.add(Dense(units=32, activation="relu"))
    model.add(Dense(units=8, activation="relu"))
    model.add(Dense(self.action_size, activation="linear"))
    model.compile(loss="mse", optimizer=Adam(lr=0.001))
    return model
def act(self, state):
    if not self.is_eval and random.random()<= self.epsilon:
        return random.randrange(self.action_size)
    options = self.model.predict(state)
    return np.argmax(options[0])
def expReplay(self, batch_size):
    mini_batch = []
    l = len(self.memory)
    for i in range(l - batch_size + 1, l):
        mini_batch.append(self.memory[i])
    for state, action, reward, next_state, done in mini_batch:
        target = reward
        if not done:
            target = reward + self.gamma * np.amax(self.model.predict(next_state)[0])
        target_f = self.model.predict(state)
        target_f[0][action] = target
        self.model.fit(state, target_f, epochs=1, verbose=0)
    if self.epsilon > self.epsilon_min:
        self.epsilon *= self.epsilon_decay
```

```
stock_name = input("Enter Stock_name, Model_name")
model_name = input()
model = load_model(model_name)
window_size = model.layers[0].input.shape.as_list()[1]
agent = Agent(window_size, True, model_name)
data = getStockDataVec(stock_name)
print(data)
l = len(data) - 1
batch_size = 32
state = getState(data, 0, window_size + 1)
print(state)
total_profit = 0
agent.inventory = []
print(l)
for t in range(l):
    action = agent.act(state)
    print(action)
    # sit
    next_state = getState(data, t + 1, window_size + 1)
    reward = 0
    if action == 1: # buy
        agent.inventory.append(data[t])
        print("Buy: " + formatPrice(data[t]))
    elif action == 2 and len(agent.inventory) > 0: # sell
        bought_price = agent.inventory.pop(0)
        reward = max(data[t] - bought_price, 0)
        total_profit += data[t] - bought_price
        print("Sell: " + formatPrice(data[t]) + " | Profit: " + formatPrice(data[t] -
    done = True if t == l - 1 else False
    agent.memory.append((state, action, reward, next_state, done))
    state = next_state
    if done:
        print("--------------------------------")
        print(stock_name + " Total Profit: " + formatPrice(total_profit))
        print("--------------------------------")
        print ("Total profit is:",formatPrice(total_profit))
```

Applying Genetic algorithm

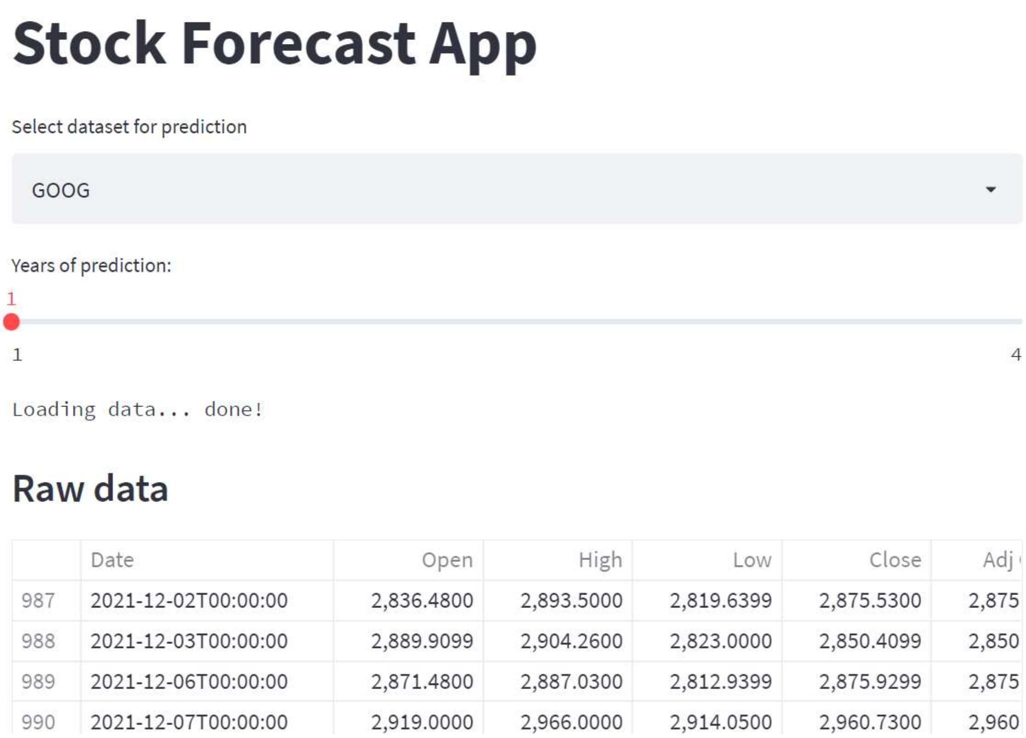
```
def fitness_func(solution, sol_idx):
    global  keras_ga, model

    predictions = pygad.kerasga.predict(model=model,
                                        solution=solution,
                                        data=X_train)

    mae = tensorflow.keras.losses.MeanAbsoluteError()
    abs_error = mae(y_train, predictions).numpy() + 0.00000001
    solution_fitness = 1.0/abs_error

    return solution_fitness

def callback_generation(ga_instance):
    print("Generation = {generation}".format(generation=ga_instance.generations_completed))
    print("Fitness    = {fitness}".format(fitness=ga_instance.best_solution()[1]))

input_layer  = tensorflow.keras.layers.Input(100)
dense_layer1 = tensorflow.keras.layers.Dense(5, activation="relu")(input_layer)
output_layer = tensorflow.keras.layers.Dense(1, activation="linear")(dense_layer1)

model = tensorflow.keras.Model(inputs=input_layer, outputs=output_layer)

keras_ga = pygad.kerasga.KerasGA(model=model,
                                 num_solutions=10)
```

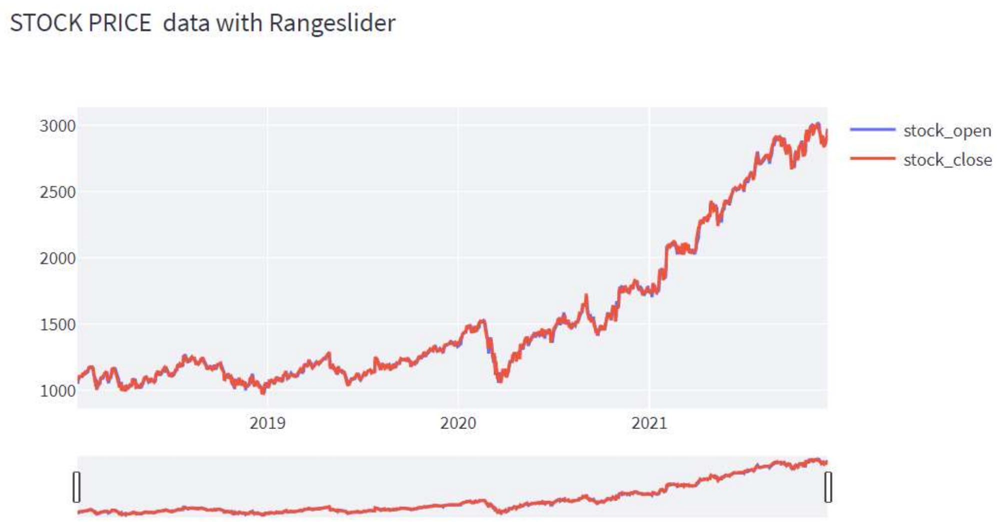
```
# Prepare the PyGAD parameters.
num_generations = 50 # Number of generations.
num_parents_mating = 5 # Number of solutions to be selected as parents in the mating pool.
initial_population = keras_ga.population_weights # Initial population of network weights

ga_instance = pygad.GA(num_generations=num_generations,
                       num_parents_mating=num_parents_mating,
                       initial_population=initial_population,
                       fitness_func=fitness_func,
                       on_generation=callback_generation)

ga_instance.run()


fig3=ga_instance.plot_fitness(title="PyGAD & Keras - Iteration vs. Fitness", linewidth=4)

# Returning the details of the best solution.
solution, solution_fitness, solution_idx = ga_instance.best_solution()
print("Fitness value of the best solution = {solution_fitness}".format(solution_fitness=solution_fitness))
print("Index of the best solution : {solution_idx}".format(solution_idx=solution_idx))

# Make prediction based on the best solution.
predictions = pygad.kerasga.predict(model=model,
                                    solution=solution,
                                    data=X_train)
```

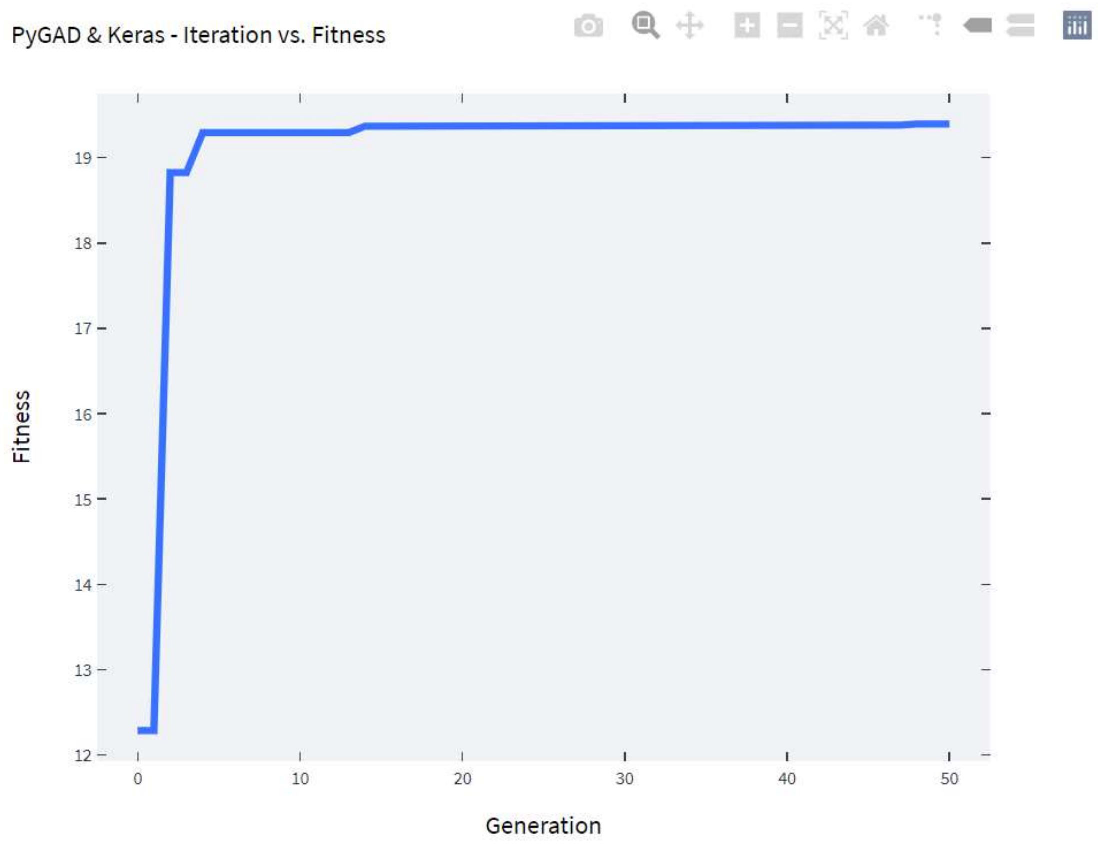
Using performance metrics and plotting it on a graph.

```python
from sklearn.metrics import mean_squared_error
mse=mean_squared_error(y_train, predictions)

st.write(f'Forecast plot for {n_years} years using GENETIC ALGORITHM')
st.plotly_chart(fig3)
st.write("The mean squared error is :",mse)
```

Web application Screenshots




STOCK PRICE data with Rangeslider
3000
2500
2000
1500
1000
2019
2020
2021
stock_open
stock_close


PyGAD & Keras - Iteration vs. Fitness
Fitness
19
18
17
16
15
14
13
12
0
10
20
30
40
50
Generation

Chapter 04

Results

4.1 DISCUSSION ON THE RESULTS ACHIEVED

For testing purposes the dataset was randomly subjected to two different train and test split ratios 70:30 and 80:20. We observed that the model showed almost the same performance when subjected to two different train and test ratios. While applying Reinforcement learning to the HDFC stock dataset we observed a mean square error of 0.310 in 80-20 train-test split and 0.308 in 70-30 train test split. Similarly we applied Genetic Algorithm on the HDFC stock dataset and obtained a mean square error of 0.0428 and 0.0517 in 70-30 and 80-20 train test split respectively.

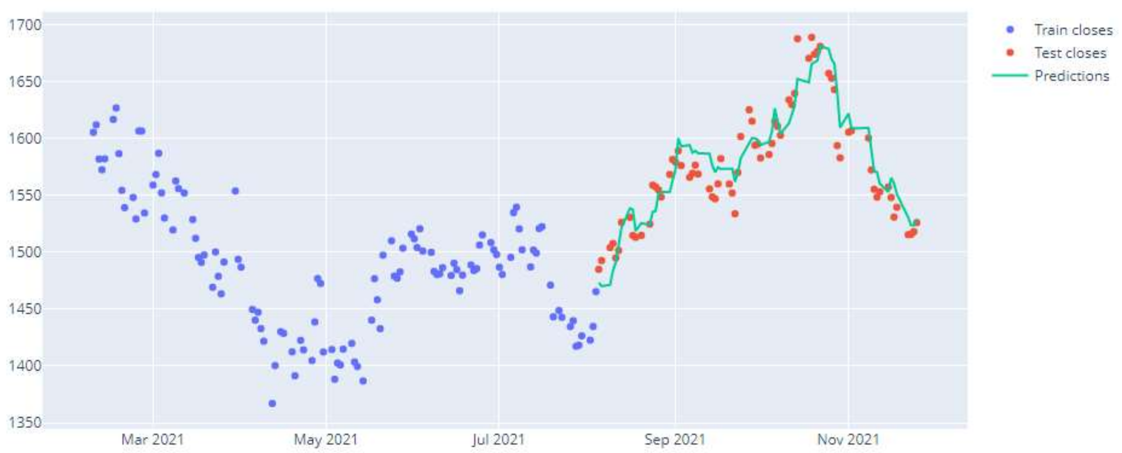


Fig 12 Prediction and test score comparison for 70-30 train test ratio of reinforcement learning on HDFC Stock

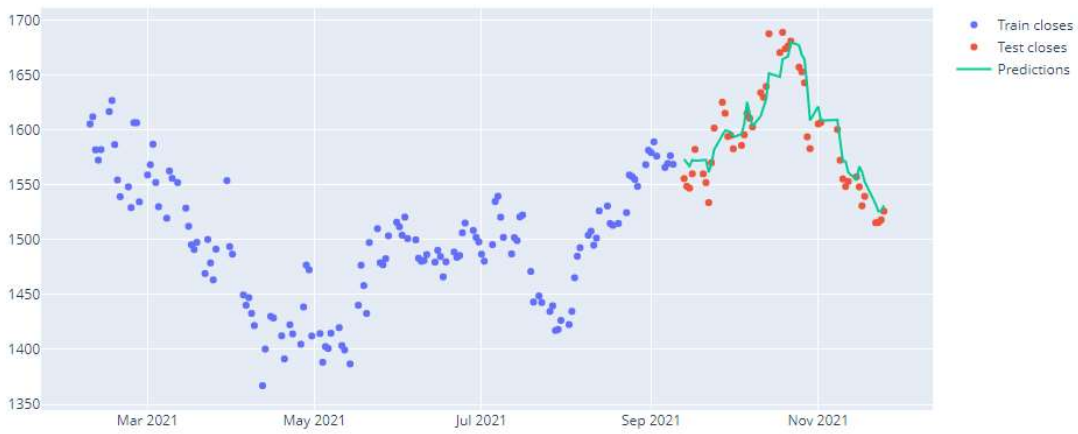


Fig 13 Prediction and test score comparison for 80-20 train test ratio of reinforcement learning on HDFC Stock

Similarly Reinforcement Learning and Genetic Algorithm were applied on different datasets like HDFC, Tesla, Google, Apple with different train and test split ratios and variable mean squared error was obtained.

We also observed that when compared to supervised and unsupervised machine learning, models like Random Forest and Logistic Regression Reinforcement learning and Genetic Algorithm proved to be accurate. We also noticed that if rigorous training is done on huge descriptive datasets the models can prove to be more accurate than Linear time series models, Artificial Neural network, Recurrent Neural Network and ARIMA time analysis.

4.2 APPLICATION OF PROJECT

Our app possesses very wide applications. It can be used by various businesses, stock brokers, students, industrialist and salaried employees as well because everyone needs to invest and since everyone do not possess the right information required for investing in stock our Machine Learning algorithms based on Reinforcement Learning and Genetic algorithm can give one appropriate idea for investing in stocks. One who already has some knowledge can use our predictions to cross check and one who does not possess any knowledge can rely on our prediction as well. We are willing to create an Interface which can be used by a person of any age group with ease.

4.3 FUTURE WORK

In Future since other investing methods such as Fixed deposits, Gold and Land are not likely to cope up with the market inflation rate they will not be considered as a viable investing option thus significantly increasing the amount of people willing to invest. So in Future we can work on our app so that we can directly view the stock prices of different algorithms. We can also create a recommender system for someone who does not know a lot about investing. We can add various other functionalities such as creation of Demat Account and option of holding and buying of different stocks. The app we create can tell an individual about different prediction algorithms and all their accuracy so a person can act accordingly.

If we talk about Genetic and Reinforcement models we clearly observed that with an increase in the amount of dataset it increases the accuracy of prediction so if we could arrange for more data the accuracy of our prediction is likely to increase.

Chapter 05

REFERENCES

1. https://fortunly.com/statistics/stock-market-statistics/#gref

2. Khan ZH (2011) Price prediction of share market using artificial neural network. Int J Comput Appl (IJCA)

3. Das SP, Padhy S (2014) Support vector machines for prediction of futures prices in the Indian stock market. Int J Comput Appl (IJCA) \

4. Chavan PS (2013) Parameters for stock market prediction. Int J Comput Technol Appl (IJCTA)

5. Sheta AF (2015) A comparison between regression, artificial neural networks and support vector machines for predicting stock market index. Int J Adv Res Artif Intell (IJARAI)

6. Preethi G, Santhi B (2012) Stock market forecasting techniques: a survey. J Theor Appl Inf Technol (JTAIT)

7. Aparna Nayak, M. M. Manohara Pai, Radhika M. Pai, Prediction Models for Indian Stock Market, Procedia Computer Science, Volume 89, 2016, Pages 441-449, ISSN 1877-0509,

8. Y Li, Y., Pan, Y. A novel ensemble deep learning model for stock prediction based on stock prices and news. *Int J Data Sci Anal* (2021). https://doi.org/10.1007/s41060-021-00279-9

9. Khan, Wasiat & Malik, Usman & Ghazanfar, Mustansar ali & Azam, Muhammad Awais & Alyoubi, Khaled & Alfakeeh, Ahmed. (2020). Predicting stock market trends using machine learning algorithms via public sentiment and political situation analysis. Soft Computing. 24. 10.1007/s00500-019-04347-y.

10. Khan, Wasiat & Malik, Usman & Ghazanfar, Mustansar ali & Azam, Muhammad Awais & Alyoubi, Khaled & Alfakeeh, Ahmed. (2020). Predicting stock market trends using machine learning algorithms via public sentiment and political situation analysis. Soft Computing. 24. 10.1007/s00500-019-04347-y.

11. Y. Shynkevich, T. M. McGinnity, S. Coleman and A. Belatreche, "Stock price prediction based on stock-specific and sub-industry-specific news articles," 2015 International Joint Conference on Neural Networks (IJCNN), 2015, pp. 1-8, doi: 10.1109/IJCNN.2015.7280517.

12. Kim, Dongyoung, Park, Jae-Won, & Choi, Jae-Hyun. (2014). A Comparative Study between Stock Price Prediction Models Using Sentiment Analysis and Machine Learning Based on SNS and News Articles. *Journal of Information Technology Services*, *13*(3), 221–233. https://doi.org/10.9716/KITS.2014.13.3.221

13. Ding, G., Qin, L. Study on the prediction of stock price based on the associated network model of LSTM. *Int. J. Mach. Learn. & Cyber.* 11, 1307–1317 (2020). https://doi.org/10.1007/s13042-019-01041-1

14. Ya Gao, Rong Wang, Enmin Zhou, "Stock Prediction Based on Optimized LSTM and GRU Models", *Scientific Programming*, vol. 2021, Article ID 4055281, 8 pages, 2021. https://doi.org/10.1155/2021/4055281

16. https://www.intechopen.com/chapters/72381

17. https://natureofcode.com/book/chapter-9-the-evolution-of-code/

18. G L. Rabiner and B. Juang, "An introduction to hidden Markov models," in IEEE ASSP Magazine, vol. 3, no. 1, pp. 4-16, Jan 1986, doi: 10.1109/MASSP.1986.1165342.

19. H Grossi, Enzo & Buscema, Massimo. (2008). Introduction to artificial neural networks. European journal of gastroenterology & hepatology. 19. 1046-54. 10.1097/MEG.0b013e3282f198a0.

20. I Fattah, Jamal & Ezzine, Latifa & Aman, Zineb & Moussami, Haj & Lachhab, Abdeslam. (2018). Forecasting of demand using ARIMA model. International Journal of Engineering Business Management. 10. 184797901880867. 10.1177/1847979018808673.

21. J Tsay, Ruey. (2010). Linear Time Series Analysis and Its Applications. 10.1002/9780470644560.ch2.

22. K Salehinejad, Hojjat & Sankar, Sharan & Barfett, Joseph & Colak, Errol & Valaee, Shahrokh. (2017). Recent Advances in Recurrent Neural Networks.

23. M Valakevicius, Eimutis & Bražėnas, Mindaugas. (2015). Application of Seasonal Holt-Winter model for the prediction of exchange rate volatility. Engineering Economics. 26. 10.5755/j01.ee.26.4.5210.