Nov 2022

ALGORITHMIC TRADING

Submitted by

Satyam Singh

Roll No. : 201500625

Akshat Maan Singh

Roll No.:201500067

Supra Vyas

Roll No. : 201500718

Akhil Jaiswal

Roll No.:201500063

Bachelor of Technology

In

Computer Engineering and Applications

GLA University

Mathura - 281406, INDIA

BONAFIDE CERTIFICATE

Certified that the project report "**ALGORITHMIC TRADING** " is the bonafide work of Akshat Maan Singh , Satyam Singh, Supra Vyas, Akhil Jaiswal who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. Rohit Aggrawal	Ms. Ruchi Talwar
HEAD OF THE DEPARTMENT	**SUPERVISION**
	Technical Trainer
Computer Engg. And Applications	Training And Development

Submitted for the project viva-voce examination held

Acknowledgement

I would like to thank GLA University for proposing the mini Project in our curriculum so that we can learn new concepts easily by doing hands-on. I would like to thank B.Tech Department for encouraging us to learn more by doing practical. I would like to
extend my regards to my mentor and my project guide **Ms. Ruchi Talwar** who has helped us in getting this project ready. They have always stood by us, whenever we needed them, and they have done a lot for us in order to get this project completed. Without their variable help, this project would not have been completed. I would also like to thank the Internet for providing a large source of information that we needed for completing this project. Once again I would like to thank all the people involved in this project from the bottom of my heart, and please forgive us if we have forgotten to mention any names. This project is completed with your help only.

TABLE OF CONTENTS

Abstract

In this present Era of modernization, Technology plays a major role in people's life and in almost every field including the Education sector, Business sector, Scientific Research, Military Operation, Trading, etc.

Nowadays we are having solutions to almost all problems, we have machines to support our lives. One of the problems that I know about the Stock Exchange is when to buy or sell. When its price increases or decreases to get profit. So we are here to facilitate a process for the customer to know when the price will increase or decrease. We are going to use Machine Learning and apply some classes-function algorithms. An example of this is Linear regression, KNN, etc. The Machine Learning model is going to predict whether a stock price is going to increase or decrease based on certain criteria and the capability of decision-making is going to come by training the model with the Kaggle datasets. We are also going to compare the performance of our model with the other Machine Learning algorithms.

Keywords: Data, Machine Learning, Training, Predict, Data sets.

CHAPTER 1

INTRODUCTION

1.1 Need Identification

- One of the biggest problems faced when buying or selling a stock is when its price goes up or down.
- Buying high and selling low is also a very big problem.
- Not knowing the true performance of your investment.
- Market crashes are a problem for customers.
- Lack of the knowledge for the newcomers for the stock market.

1.2 Identification of Problem

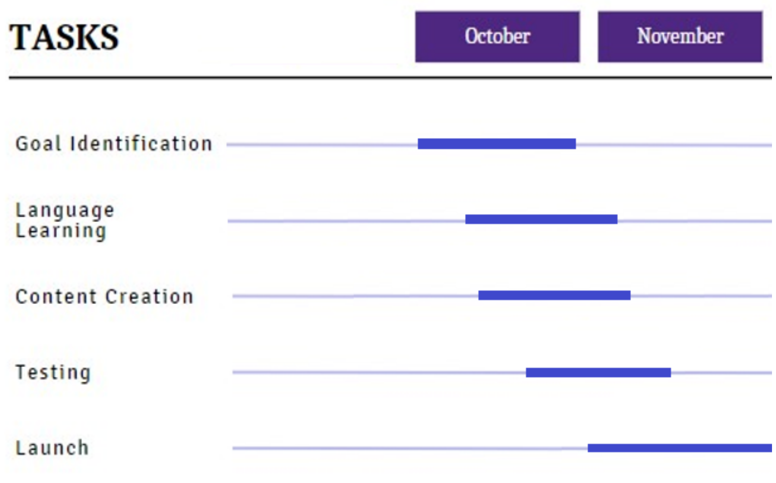
Stocks offer investors the greatest potential for growth (capital appreciation) over the long haul. Investors willing to stick with stocks over long periods of time, say 15 years, generally have been rewarded with strong, positive returns.

But stock prices move down as well as up. There's no guarantee that the company whose stock you hold will grow and do well, so you can lose money you invest in stocks.

1.3 Identification of Tasks

- This proposed model will distinguish the nature of the customer on the basis of the record of company data.
- These records are taken from the company and create a data set. With the help of these data sets and training machine learning model, we are going to predict the price of the stock.

1.4 Timeline



1.5 Organization of the report

Task is divided in some chapters which are in our report

Introduction:

The first chapter is about what people are facing to buy or sell any stock.

People want a solution to this problem.

Building the solution

Timeline with gantt chart

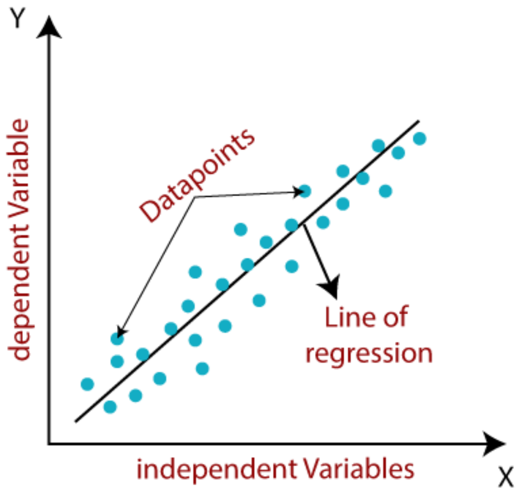
CHAPTER 2

BACKGROUND STUDY

The Models we are going to use for the prediction **purpose are as follows:-**

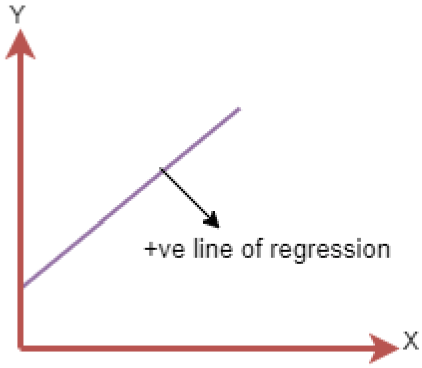
2.1 Linear Regression :-

- It comes under the umbrella of Supervised Machine Learning.
- It is a statistical method that is used for predictive analysis.
- Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.
- Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called linear regression.
- It is used for solving Regression Problems.
- The linear regression model provides a sloped straight line representing the relationship between the variables.
- A linear line showing the relationship between the dependent and independent variables is called a regression line.



- A regression line can show two types of relationship:
    1. Positive Linear Relationship:

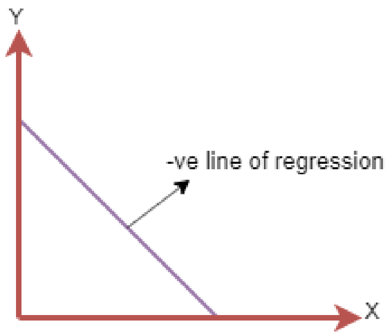
       If the dependent variable increases on the Y-axis and the independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



The line equation will be: **$Y = a_0 + a_1 x$**

2. Negative Linear Relationship:

   If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.



The line of equation will be: **$Y = -a_0 + a_1 x$**

Assumption of Linear Regression:

- The Two variables Should be in a Linear Relationship.
- All the variables should be Multivariate Normal.
- There should be No Multicollinearity in the data.
- There should be No Autocorrelation in the data.
- There should be Homoscedasticity among the data.

Linear Regression Equation:

- $$Y_i = f(X_i, \beta) + e_i$$

Steps in Linear Regression Algorithm :

- Reading and understanding the data.

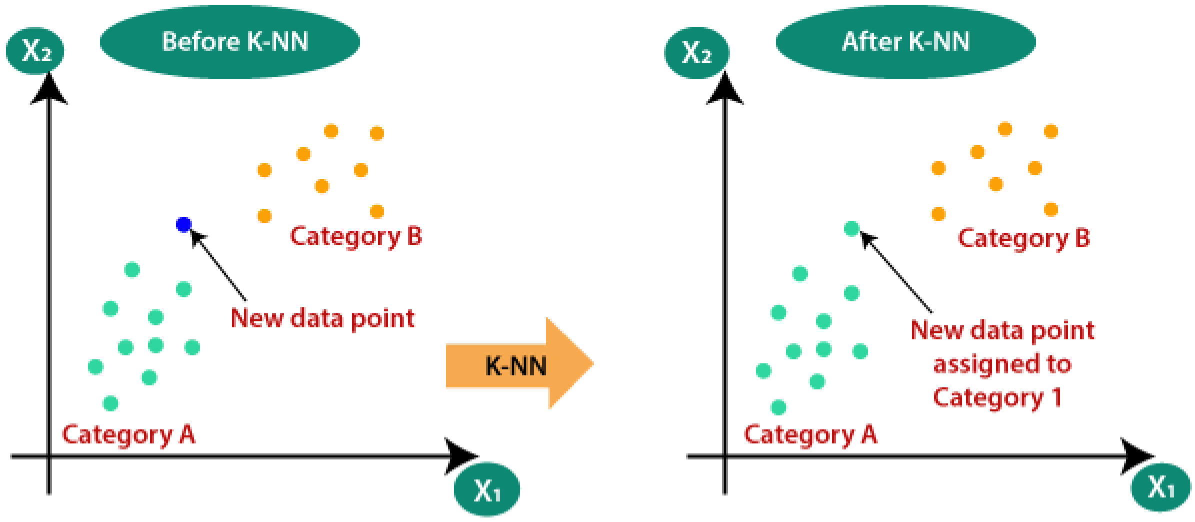
- Visualizing the data.
- Data Preparation
- Splitting the data into training and test sets.
- Building the linear model.
- Residual analysis of the train data.
- Making predictions using the final model and evaluation.

2.2 KNN (K-Nearest Neighbors):-

- It comes under the umbrella of Supervised Machine Learning.
- The K-NN algorithm assumes similarity between the new case or data and available cases and puts the new case into the category that is most similar(more close to new data or case) to the available categories.
- It is a non-parametric algo, which means it does not make any assumption or underlying data.
- Also known as lazy learner algo because it does learn from the training set immediately in place of that it stores dataset and at time of classification , it performs required action on dataset.
- The K-NN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

2.2.1 Why do we need a K-NN Algorithm:

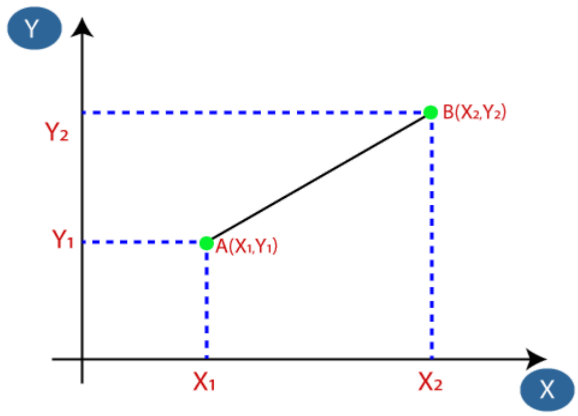
Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



How does K-NN work?:

- The K-NN working can be explained on the basis of the below algorithm.
    1. Select the number K of the neighbors.
    2. Calculate the Euclidean distance of **K number of neighbors**.
    3. Take the K nearest neighbors as per the calculated Euclidean distance.
    4. Among these k neighbors, count the number of the data points in each category.
    5. Assign the new data points to that category for which the number of the neighbor is maximum.
    6. Our model is ready.

Euclidean Equation:



Euclidean Distance between two A and B dist((x,y),(a,b))=$\sqrt{(x - a)^2 + (y - b)^2}$

How to select the value of K in the K-NN Algorithm?

- There is no specified way of determining the best value for ‘K’ , so we need to go for a hit and trail. By default we go for the value of ‘k’=5(most preferred).
- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
- Quite Large values for K can lead to underfitting.
- Medium values(neither too small nor too large) work fine with the model.

2.3 Naive Baye’s Classifier:-

Baye’s Theorem

$$P(A|B) = P(B|A)\, P(A) / P(B)$$

Where,

- **P(A):**Class Prior Probability
- **P(B):** Likelihood
- **P(A|B):** Posterior Probability
- **P(A):** Predictor Prior Probability

Naïve Baye’s Classifier:

It is a set of algo. Based on baye’s theorem.It is kind of algo. Which uses the bayes theorem.

Then what is Baye's theorem:

- In mathematics it is a probability theory ,
- It is the most important part of probability.
- This theory is named after Thomas bayes.
- It describes the probability of any events based on previous knowledge of those events.

Steps:-

1. As the first step toward prediction using naïve bayes, you will have to estimate frequency of each and every attribute.
2. Calculate possibilities of each attribute.
3. Normalizing or returning the standard condition of the values.

P(YES) = Probability of YES / Probability of YES + Probability of NO

P(NO) = Probability of NO / Probability of YES + Probability of NO

Why Naive Bayes?

- It is very fast and efficient to use on discrete as well as continuous data.
- It helps us to compute the conditional probability of an event based on previous probabilities of two or more events.
- It requires the least amount of training data from which we trained our ML model.
- It is naïve bayes it assumes that the occurrence of certain things or features are independent of each other like identifying a fruit by their particular shape, taste, colour and weight.

Apply Naïve Baye's Classifier on text data in NLP (Natural Language Processing):

- NLP helps computers to communicate with humans with their own language.

- In NLP we usually perform pre-processing steps

a. STOP WORD
b. STEMMING
c. BAG OF WORDS
d. TF-IDF

After apply these steps we got vectors of specific sentences and we also have output features.

Like: P(YES|GIVEN SENTENCE)
GIVEN SENTENCE may be x1,x2,x3,x4,x5,xn

Vector:- It is used to represent numerical characteristics of data.

- It is probability of X and Y is already happened.

2.4 Disadvantage:

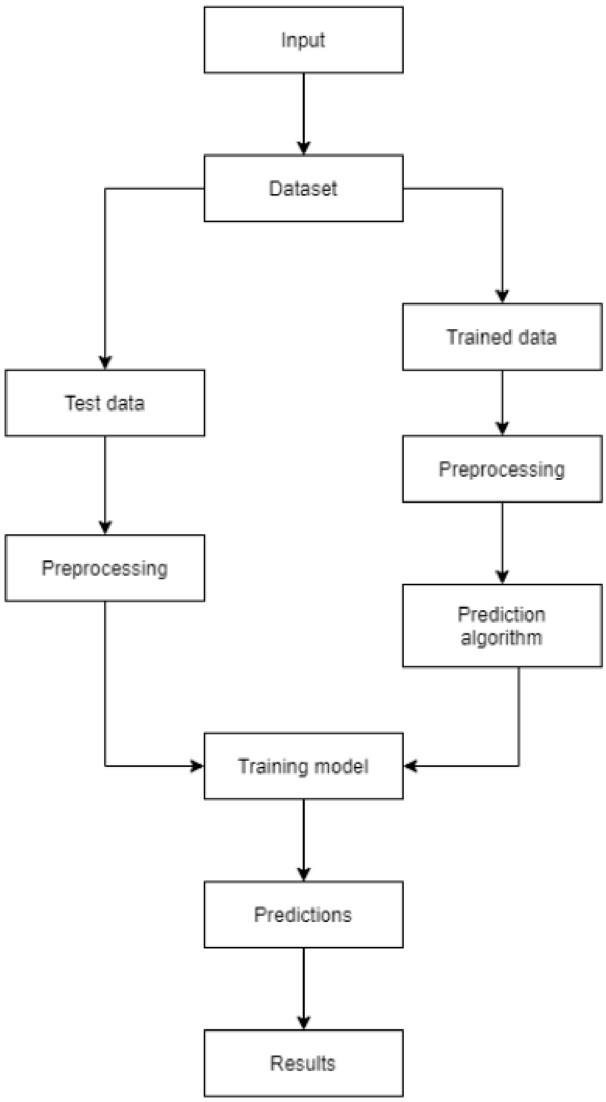
- Its biggest disadvantage is that it implicitly assumes that all the attributes are unrelated to each other or this is not seen or happens in real life.

CHAPTER 3

DESIGN FLOW

3.1 Structure Chart

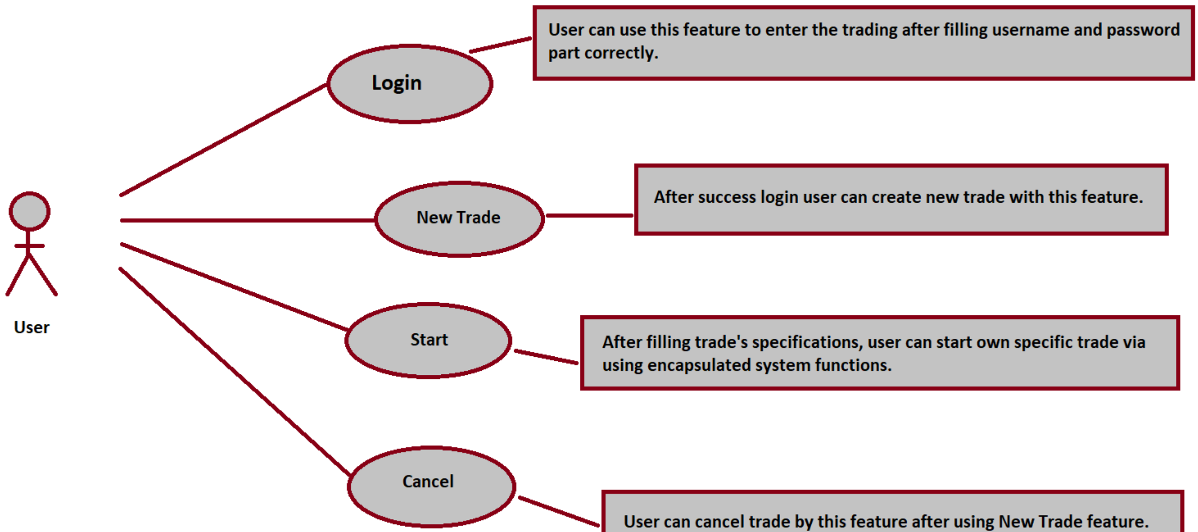
A structure chart (SC) in software engineering and organizational theory is a chart which shows the breakdown of a system to its lowest manageable levels. They are used in structured programming to arrange program modules into a tree. Each module is represented by a box, which contains the module's name.



3.2 USE CASE DIAGRAM

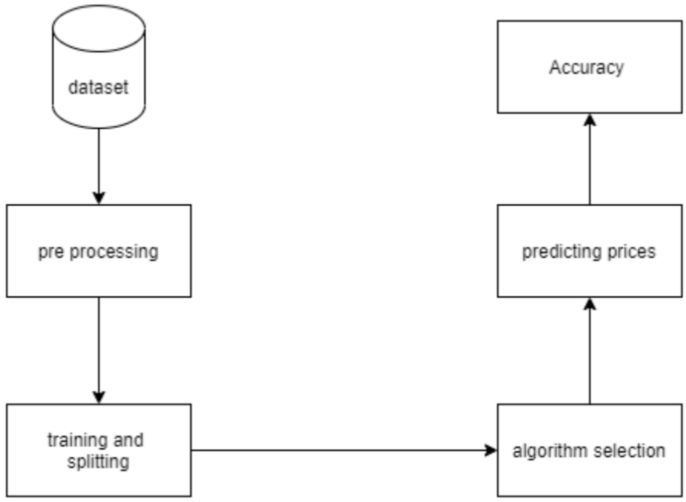
In the Unified Modelling Language (UML), a use case diagram can summarise the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialised symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which our system or application interacts with people, organizations, or external systems.
- Goals that our system or application helps those entities (known as actors) achieve.
- The Scope of the System.



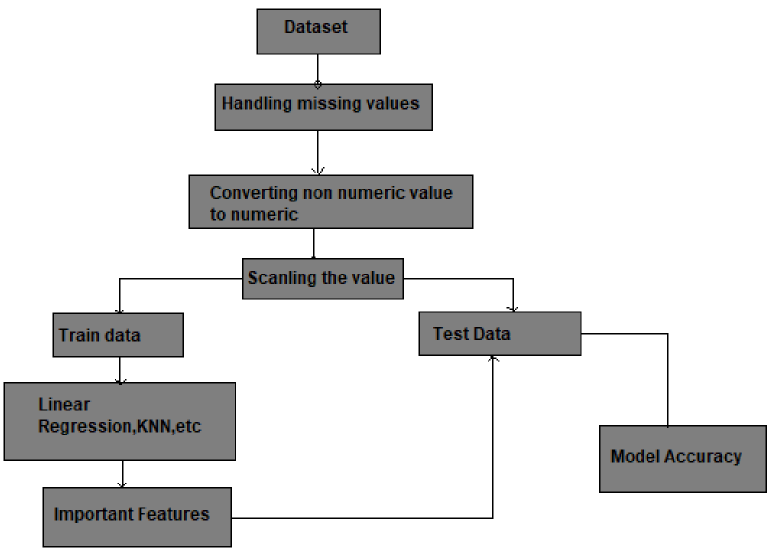
3.3 Component Diagram:

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.



3.4 FLOW CHART

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows.



CHAPTER 4

RESULT ANALYSIS AND VALIDATION

4.1 Hardware

- RAM : 4 GB
- Storage: 20 GB or more
- CPU : 2 Ghz or faster
- Architecture : 32 bit or 64 bit

4.2 Software Requirement

- Python 3.0 in Google Colab is used in data pre- processing , model training and prediction.
- Operating System: windows 7 & above or Linux based OS or MAC OS

4.3 Required Tool

Pandas
Numpy
sklearn
Matplotlib
K-NN Classifier

4.4 System configuration:

This project can run on commodity hardware. We ran the entire project on an Intel I5 processor with 8 GB Ram, 2 GB Nvidia Graphic Processor, It also has 2 cores which run at 1.7 GHz, 2.1 GHz respectively. First part is the training phase which takes 10-15 mins of time and the second part is the testing part which only takes a few seconds to make predictions and calculate accuracy.

4.5 Data Analysis

- Most important question is that on what aspects, we are going to analyse whether the price of the stock is going up or down. We have to target some variables on that aspect. We are going to predict it.
- We have to check the previous data of the company.
- Check the difference of the open and closing of the price.

4.6 Sample Code:

4.6.1 Linear Regression:

```
[1]  import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     %matplotlib inline
```

```
[2]  #import chart_studio.plotly as py
     import plotly.graph_objs as go
     from plotly.offline import plot

     # for offline plotting
     from plotly.offline import download_plotlyjs,init_notebook_mode,plot,iplot
     init_notebook_mode(connected=True)
```

```
[3]  apple=pd.read_csv("apple_stock.csv")
```

```
[ ]  apple.head() #to see the top 5 rows from the dataset
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	01-11-2012	21.365000	21.535713	18.062500	20.902857	17.924156	12929851200
1	01-12-2012	21.201786	21.235357	17.901072	19.006071	16.372108	12132752800
2	01-01-2013	19.779285	19.821428	15.535714	16.267500	14.013056	13123423600
3	01-02-2013	16.396786	17.319286	15.630714	15.764286	13.579582	9344034000
4	01-03-2013	15.642857	16.783930	14.964286	15.809286	13.697714	9176876800

```
[4] apple.info().  #To find the no. of ros and columns data type
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 120 entries, 0 to 119
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Date       120 non-null    object
 1   Open       120 non-null    float64
 2   High       120 non-null    float64
 3   Low        120 non-null    float64
 4   Close      120 non-null    float64
 5   Adj Close  120 non-null    float64
 6   Volume     120 non-null    int64
dtypes: float64(5), int64(1), object(1)
memory usage: 6.7+ KB
```

Converting data column into datetime format using pandas library

```
apple['Date']=pd.to_datetime(apple['Date'])
```

```
[ ] print(f'DataFrame Contains stock prices between {apple.Date.min()}{apple.Date.max()}')
```

```
DataFrame Contains stock prices between 2012-01-11 00:00:002022-01-10 00:00:00
```

```
[ ] print(f'Total days={(apple.Date.max()-apple.Date.min()).days} days')
```

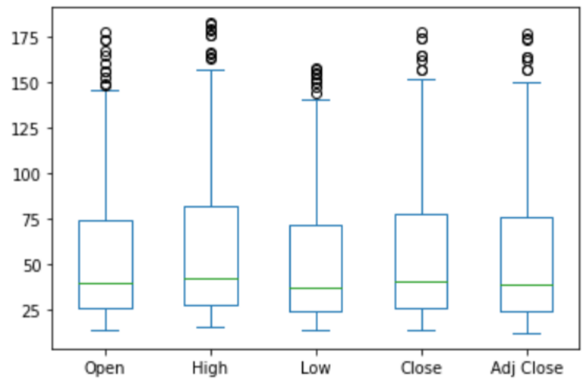
```
Total days=3652 days
```

```
[ ] apple.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	120.000000	120.000000	120.000000	120.000000	120.000000	1.200000e+02
mean	59.868991	64.208304	56.100420	60.694619	58.914108	3.770999e+09
std	47.866219	51.450434	44.312807	48.266903	48.848531	2.456733e+09
min	14.381786	15.901786	13.753571	14.161786	12.351480	1.257109e+09
25%	26.018126	27.593125	23.975000	26.472500	24.220844	2.178167e+09
50%	39.961250	42.331249	37.196251	41.157500	39.135750	2.890443e+09
75%	74.563748	81.844376	71.893752	77.904377	76.494040	4.333296e+09
max	177.830002	182.940002	157.800003	177.570007	176.838242	1.312342e+10

```
apple[['Open','High','Low','Close','Adj Close']].plot(kind='box')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f3acdaad390>



Setting the layout for our plot

```
layout=go.Layout(
    title='Stock Prices Of Apple',
    xaxis=dict(
        title='Date',
        titlefont=dict(
            family='Courier New monospace',
            size=18,
            color='#7f7f7f'
            )
    ),
    yaxis=dict(
        title='Price',
        titlefont=dict(
        )
    )
)
apple_data=[{'x':apple['Date'],'y':apple['Close']}]
plot=go.Figure(data=apple_data,layout=layout)
```

```
#plot(plot) #plotting offline
iplot(plot)
```

```python
#Building the regression model
from sklearn.model_selection import train_test_split

#for preprocessing
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
```

```python
#for model evaluation
from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import r2_score
```

```python
[34] #Split the data into train and test sets
X=np.array(apple.index).reshape(-1,1);
Y=apple['Close']
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state=101)
```

```python
[35] #feature scaling
scaler=StandardScaler().fit(X_train)
```

```python
[36] from sklearn.linear_model import LinearRegression
```

```python
[37] #Creating a linear model
lm=LinearRegression()
lm.fit(X_train,Y_train)
```

```
LinearRegression()
```

```python
#plot actual and predicted values from Train dataset
trace0=go.Scatter(
    x=X_train.T[0],
    y=Y_train,
    mode='markers',
    name='Actual'
)
trace1=go.Scatter(
    x=X_train.T[0],
    y=lm.predict(X_train).T,
    mode='lines',
    name='Predicted'

)
```

+ Code + Text

Add code cell
⌘/Ctrl+M B

```python
[39] apple_data=[trace0,trace1]
layout.xaxis.title.text='Day'
plot2=go.Figure(data=apple_data,layout=layout)
```

```python
[40] iplot(plot2)
```

```python
[41] #Calculate scores for model evaluation
scores=f'''
{'Metric'.ljust(10)}{'Train'.center(20)}{'Test'.center(20)}
{'r2_score'.ljust(10)}{r2_score(Y_train,lm.predict(X_train))}\t{r2_score(Y_test,lm.predict(X_test))}
{'MSE'.ljust(10)}{mse(Y_train,lm.predict(X_train))}\t{mse(Y_test,lm.predict(X_test))}
'''
```

```python
[42] print(scores)
```

```
Metric            Train               Test
r2_score  0.7609938505446694     0.8122651520235734
MSE       536.7367560267173      459.96548228929805
```

4.6.2 KNN Algorithm Implementation

Importing the library

```
[1] import numpy as np
    import matplotlib.pyplot  as plt
    import pandas as pd
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.metrics import accuracy_score

    from pandas_datareader import data as pdr
```

Fetching the data from the dataset

```
dataset=pd.read_csv("/content/apple_stock.csv")
dataset=dataset.dropna()
dataset=dataset[['Open','High','Low','Close','Adj Close']]
dataset.head()    #Print the first 5 rows of the dataset
```

	Open	High	Low	Close	Adj Close
0	21.365000	21.535713	18.062500	20.902857	17.924156
1	21.201786	21.235357	17.901072	19.006071	16.372108
2	19.779285	19.821428	15.535714	16.267500	14.013056
3	16.396786	17.319286	15.630714	15.764286	13.579582
4	15.642857	16.783930	14.964286	15.809286	13.697714

```
[3] dataset['Open-Close']=dataset.Open-dataset.Close
    dataset['High-Low']=dataset.High-dataset.Low
    dataset=dataset.dropna()
    X=dataset[['Open-Close','High-Low']]
    X.head()
```

	Open-Close	High-Low
0	0.462143	3.473213
1	2.195715	3.334285
2	3.511785	4.285714
3	0.632500	1.688572
4	-0.166429	1.819644

```
Y=np.where(dataset['Close'].shift(-1)>dataset['Close'],1,-1)
```

Splitting the dataset into training and testing

```
[5] split_percentage=0.7
    split=int(split_percentage*len(dataset))
    X_train=X[:split]
    Y_train=Y[:split]
    X_test=X[split:]
    Y_test=Y[split:]
```

Now we are instantiating the knn model

```
knn=KNeighborsClassifier(n_neighbors=50)

#Now we will fit the model
knn.fit(X_train, Y_train)
```

```
KNeighborsClassifier(n_neighbors=50)
```

Accuracy Score

```
[7] accuracy_train=accuracy_score(Y_train,knn.predict(X_train))
    accuracy_test=accuracy_score(Y_test,knn.predict(X_test))

     #Print the training and training scores
    print('Train_data_accuracy : %.2f' %accuracy_train)
    print('Testing_data_accuracy : %.2f'  %accuracy_test)
```

```
Train_data_accuracy : 0.60
Testing_data_accuracy : 0.64
```

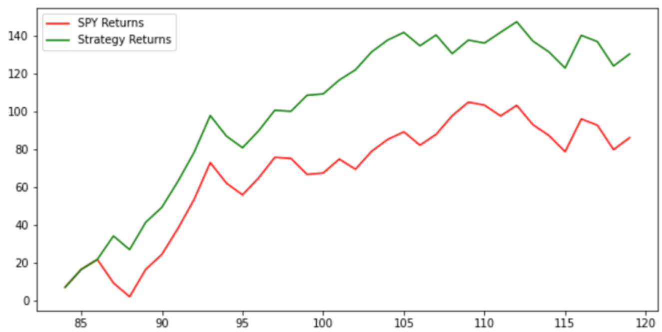
Creating the training stategies using knn model

```
[8] #predicted signal
    dataset['Predicted_signal']=knn.predict(X)

    #SRY Cumulative returns
    dataset['SPY_returns'] = np.log(dataset['Close']/dataset['Close'].shift(1))
    Cumulative_SPY_returns = dataset[split:]['SPY_returns'].cumsum()*100

    # Cumulative Strategy Returns
    dataset['Startegy_returns'] = dataset['SPY_returns']* dataset['Predicted_signal'].shift(1)
    Cumulative_Strategy_returns = dataset[split:]['Startegy_returns'].cumsum()*100
```

```
#plot the results to visualize the performance
plt.figure(figsize=(10,5))
plt.plot(Cumulative_SPY_returns, color='r',label = 'SPY Returns')
plt.plot(Cumulative_Strategy_returns, color='g', label = 'Strategy Returns')
plt.legend()
plt.show()
```



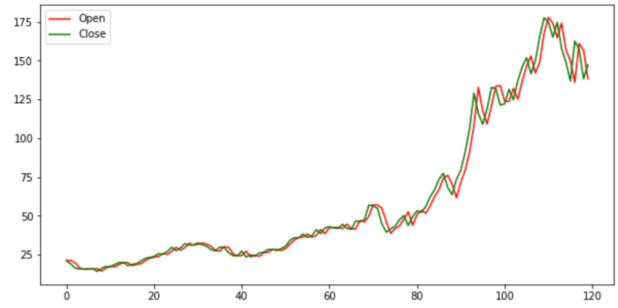
```
#Now we will calculate the sharpe ratio

#Calculating the standard deviation
Stde=Cumulative_Strategy_returns.std()
```

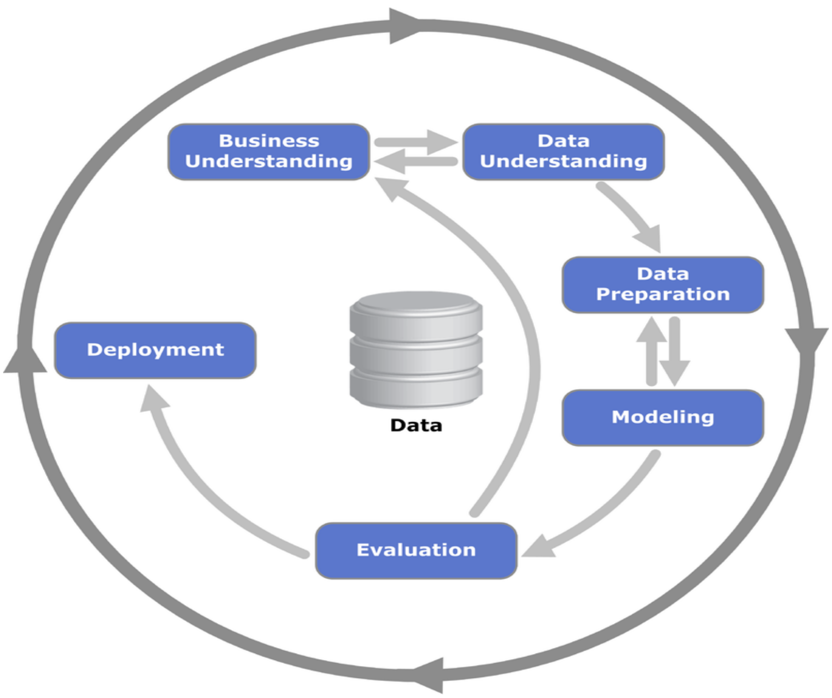
```
[11] Sharpe=(Cumulative_Strategy_returns-Cumulative_SPY_returns)/Stde
     Sharpe=Sharpe.mean()
     print('Sharpe ratio: %.2f'%Sharpe)

     Sharpe ratio: 0.81
```

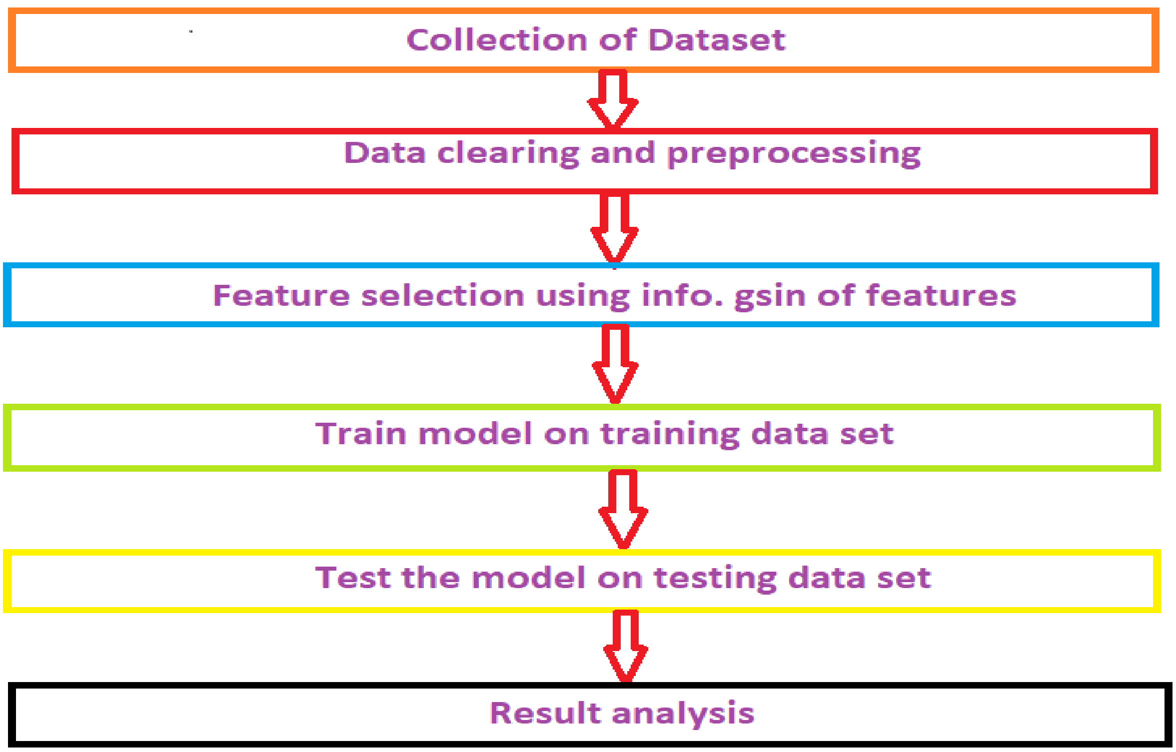
```
from matplotlib.backend_bases import CloseEvent
Op=dataset['Open']
Cls=dataset['Close']
plt.figure(figsize=(10,5))
plt.plot(Op, color='r',label = 'Open')
plt.plot(Cls, color='g', label = 'Close')
plt.legend()
plt.show()
```



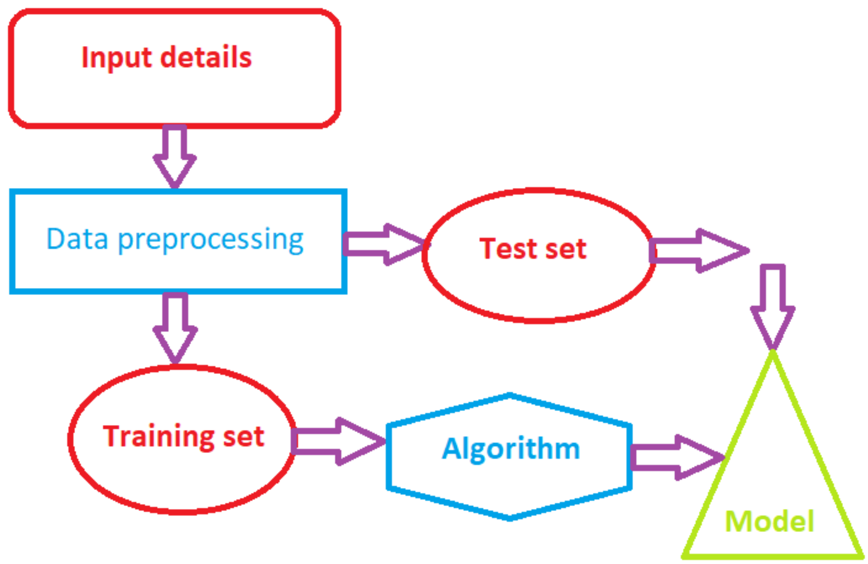
4.7 Algorithmic Trading



4.8 Process Diagram



4.9 Architectural Diagram of System



4.10 Backed databases

```
[*]: import pandas as pd
     df=pd.read_csv("apple_stock2.csv")
     df.head()
```

```
[*]: import mysql.connector as msql
     from mysql.connector import Error
     try:
         conn = msql.connect(host='localhost', user='root',
                             password='root')#give ur username, password
         if conn.is_connected():
             cursor = conn.cursor()
             cursor.execute("CREATE DATABASE apple_stock2")
             print("Database is created")
     except Error as e:
         print("Error while connecting to MySQL", e)
```

```
]: import mysql.connector as msql
   from mysql.connector import Error
   try:
       conn = msql.connect(host='localhost', database='apple_stock2', user='root',
       if conn.is_connected():
           cursor = conn.cursor()
           cursor.execute("select database();")
           record = cursor.fetchone()
           print("You're connected to database: ", record)
           cursor.execute('DROP TABLE IF EXISTS apple_stock2;')
           print('Creating table....')
   # in the below line please pass the create table statement which you want #to cr
           cursor.execute("CREATE TABLE apple(Date varchar(255),open int, high int,
           print("Table is created....")
           #loop through the data frame
           for i,row in df.iterrows():
               #here %S means string values
               sql = "INSERT INTO apple_stock2.apple VALUES (%s,%s,%s,%s,%s,%s,%s,%
               cursor.execute(sql, tuple(row))
               print("Record inserted")
               # the connection is not auto committed by default, so we must commit
               conn.commit()
   except Error as e:
               print("Error while connecting to MySQL", e)
```

```
[ ]: # Execute query
     sql = "SELECT * FROM apple_stock2.apple"
     cursor.execute(sql)
     # Fetch all the records
     result = cursor.fetchall()
     for i in result:
         print(i)
```

```
[ ]: import mysql.connector
```

```
[ ]: pip install xlrd
```

```
[ ]: import xlrd
```

```
[ ]: conn = msql.connect(host='localhost', database='apple_stock2', user='root', pass
```

```
+------------+------+-------+-----------+
| Date       | Open | Close | Predicted |
+------------+------+-------+-----------+
| 01-12-2022 |  138 |   147 |       160 |
| 01-01-2023 |  147 |   163 |       180 |
| 01-02-2023 |  180 |   186 |       190 |
+------------+------+-------+-----------+
```

```
+------------+------+-------+-----------+
| Date       | Open | Close | Predicted |
+------------+------+-------+-----------+
| 01-12-2022 |  390 |   398 |       400 |
| 01-01-2023 |  405 |   410 |       450 |
| 01-02-2023 |  467 |   510 |       570 |
+------------+------+-------+-----------+
```

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

- According to this research paper prediction accuracy is sweet for datasets.
- This research paper can find out that the company's stock price would increase or decrease and the accuracy is very good.

Model	Train Score	Test Score
Linear Regression	76.09 %	81.24 %
K-Nearest Neighbors	85.09%	86.77%
Naive Bayes	80.03%	77.91%
MSE	536.74	459.97

5.2 Future Work

- We will make a web page in which users make an input of the company and through the best algorithm it will predict the future stock price of the particular company.
- We will also make an app which will do the same work and the app will be linked with the web page

REFERENCES

Website:

- www.javatpoint.com
- www.geeksforgeeks.com
- www.w3schools.com
- www.kaggle.com
- www.finance.yahoo.com

Books:

- Python

  Python Crash Course:A Hands-On

- Machine Learning

  Hands-On ML with Scikit -Learn, Keras & TensorFlow

  Lluís A. Belanche Muñoz