Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Engineering
Academic Year 2021-2022

STOCK PRICE PREDICTOR

Machine Learning Laboratory

By

Jay Dedhia 60004190050
Mayank Gohil 60004190065

Guide(s):

Dr. Chetashri Bhadane

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Engineering
Academic Year 2021-2022

Group Members:
Jay Dedhia – 60004190050
Mayank Gohil – 60004190065

Stock Price Predictor

1. PROBLEM STATEMENT

Predicting how the stock market will perform is one of the most difficult things to do. There are so many factors involved in the prediction – physical factors vs. psychological, rational and irrational behavior, etc. All these aspects combine to make share prices volatile and very difficult to predict with a high degree of accuracy. Using features like the latest announcements about an organization, their quarterly revenue results, etc., machine learning techniques have the potential to unearth patterns and insights we didn't see before, and these can be used to make unerringly accurate predictions.

2. INTRODUCTION

a. Need:

Stock market prediction aims to determine the future movement of the stock value of a financial exchange. The accurate prediction of share price movement will lead to more profit investors can make. Predicting how the stock market will move is one of the most challenging issues due to many factors that involved in the stock prediction, such as interest rates, politics, and economic growth that make the stock market volatile and very hard to predict accurately. The prediction of shares offers huge chances for profit and is a major motivation for research in this area; knowledge of stock movements by a fraction of a second can lead to high profits. Since stock investment is a major financial market activity, a lack of accurate knowledge and detailed information would lead to an inevitable loss of investment. The prediction of the stock market is a difficult task as market movements are always subject to uncertainties. Stock market prediction methods are divided into two main categories: technical and fundamental analysis. Technical analysis focuses on analyzing historical stock prices to predict future stock values (i.e. it focuses

on the direction of prices). On the other hand, fundamental analysis relies mostly on analyzing unstructured textual information like financial news and earning reports. Many researchers believe that technical analysis approaches can predict the stock market movement. In general, these researches did not get high prediction results as they depend heavily on structured data neglecting an important source of information that is the online financial news and social media sentiments. These days more and more critical information about the stock market has become available on the Web. Examples include BBC, Bloomberg, and Yahoo Finance. It is hard to manually extract useful information out of these resources. This draws a picture of the significance of Machine Learning techniques to automatically extract meaningful information for analyzing the stock market.

b. Working:

We will first load the dataset and define the target variable for the problem There are multiple variables in the dataset – date, open, high, low, last, close, total_trade_quantity, and turnover.

- The columns Open and Close represent the starting and final price at which the stock is traded on a particular day.
- High, Low and Last represent the maximum, minimum, and last price of the share for the day.
- Total Trade Quantity is the number of shares bought or sold in the day and Turnover (Lacs) is the turnover of the company on a given date.

The profit or loss calculation is usually determined by the closing price of a stock for the day, hence we will consider the closing price as the target variable. Let's plot the target variable to understand how it's shaping up in our data

We first use linear regression model.

For our problem statement, we do not have a set of independent variables. We have only the dates instead. Let us use the date column to extract features like – day, month, year, mon/fri etc. and then fit a linear regression model. We will first sort the dataset in ascending order and then create a separate dataset so that any new feature created does not affect the original data.

This creates features such as:

'Year', 'Month', 'Week', 'Day', 'Dayofweek', 'Dayofyear', 'Is_month_end', 'Is_month_start', 'Is_quarter_end', 'Is_quarter_start', 'Is_year_end', and 'Is_year_start'.

We will now split the data into train and validation sets to check the performance of the model.

Another interesting ML algorithm that one can use here is kNN (k nearest neighbours). Based on the independent variables, kNN finds the similarity between new data points and old data points. Using the same train and validation set from the last section we check the performance of the model

Finally we use Long Short Term Memory

LSTMs are widely used for sequence prediction problems and have proven to be extremely effective. The reason they work so well is because LSTM is able to store past information that is important, and forget the information that is not. For now, let us implement LSTM as a black box and check it's performance on our particular data.

c. Applications:

- Our project applications include stock price prediction of any stock of any country whose historical data of stock prices is available.
- Our application also provides the analysis of different machine learning models to give the accuracy of each and chose the most appropriate model.
- Our project can also be simply modified to find a solution to any time series problem statement.

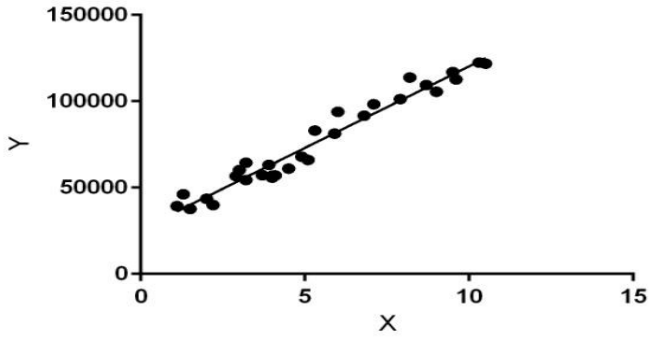
3. ALGORITHMS USED

Linear Regression

a. Theory/Working:

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering,and the number of independent variables getting

used.



Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression. In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Hypothesis function for Linear Regression:

$$y = \theta_1 + \theta_2.x$$

While training the model we are given :
x: input training data (univariate – one input variable(parameter))
y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.
θ1: intercept
θ2: coefficient of x

Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

Cost Function (J):
By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is

minimum. So, it is very important to update the θ1 and θ2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

$$minimize\frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2 \quad J = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Cost function(J) of Linear Regression is the Root Mean Squared Error (RMSE) between predicted y value (pred) and true y value (y).

Gradient Descent:

To update θ1 and θ2 values in order to reduce Cost function (minimizing RMSE value) and achieving the best fit line the model uses Gradient Descent. The idea is to start with random θ1 and θ2 values and then iteratively updating the values, reaching minimum cost.

b. Applications

- Marks scored by students based on number of hours studied (ideally)- Here marks scored in exams are independent and the number of hours studied is independent.
- Predicting crop yields based on the amount of rainfall- Yield is a dependent variable while the measure of precipitation is an independent variable.
- Predicting the Salary of a person based on years of experience- Therefore, Experience becomes the independent while Salary turns into the dependent variable.

K Nearest Neighbours

a) Theory/Working

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

KNN 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.

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Below are some points to remember while selecting the value of K in the K-NN algorithm

- There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.

- 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.
- Large values for K are good, but it may find some difficulties.

b) Applications

- KNN can be used for Recommendation Systems. Although in the real world, more sophisticated algorithms are used for the recommendation system. KNN is not suitable for high dimensional data, but KNN is an excellent baseline approach for the systems. Many companies make a personalized recommendation for its consumers, such as Netflix, Amazon, YouTube, and many more.
- KNN can search for semantically similar documents. Each document is considered as a vector. If documents are close to each other, that means the documents contain identical topics.
- KNN can be effectively used in detecting outliers. One such example is Credit Card fraud detection.

Long Short Term Memory

a) Theory/Working

Long Short Term Memory is a kind of recurrent neural network. In RNN output from the last step is fed as input in the current step. LSTM was designed by Hochreiter & Schmidhuber. It tackled the problem of long-term dependencies of RNN in which the RNN cannot predict the word stored in the long-term memory but can give more accurate predictions from the recent information. As the gap length increases RNN does not give an efficient performance. LSTM can by default retain the information for a long period of time. It is used for processing, predicting, and classifying on the basis of time-series data.

LSTM has three gates:

- The input gate: The input gate adds information to the cell state
- The forget gate: It removes the information that is no longer required by the model

- The output gate: Output Gate at LSTM selects the information to be shown as output

b) Applications

I. Language Modelling

II. Machine Translation

III. Image Captioning

IV. Handwriting generation

V. Question Answering Chatbots

4. IMPLEMENTATION

a. Important screen shots

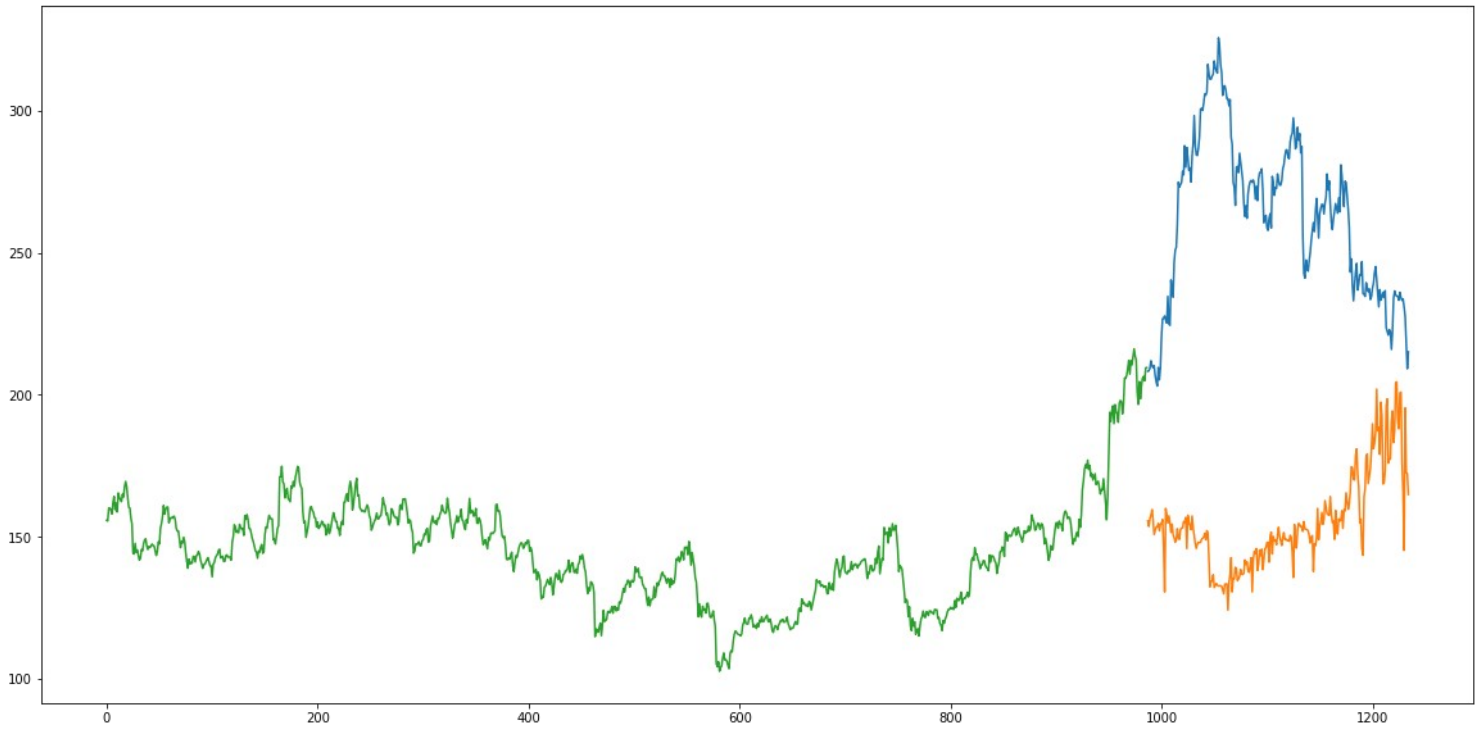
Linear Regression:


```
121.16291596523156
```

The RMSE value is high which clearly shows that linear regression has performed poorly. Let's look at the plot and understand why linear regression has not done well: As seen from the plot above, for January 2016 and January 2017, there was a drop in the stock price. The model has predicted the same for January 2018. A linear regression technique can perform

well for problems such as Big Mart sales where the independent features are useful for determining the target value.

KNN:




```
115.17086550026721
```

There is not a huge difference in the RMSE value which clearly shows that knn has performed poorly, but a plot for the predicted and actual values should provide a more clear understanding.

LSTM:


```
11.772259608962642
```

We can clearly note that the RMSE value for this model is far less than the other two models. Hence we see that the graph is highly accurate for this model and appropriate for our problem statement.

b. Code of important functions

```
import pandas as pd
import numpy as np

#to plot within notebook
import matplotlib.pyplot as plt
%matplotlib inline

#setting figure size
from matplotlib.pylab import rcParams
rcParams['figure.figsize'] = 20,10

#for normalizing data
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))

#read the file
```

```
df = pd.read_csv('NSE-TATAGLOBAL(1).csv')

#print the head
df.head()
```

#setting index as date

```
df['Date'] = pd.to_datetime(df.Date,format='%Y-%m-%d')
df.index = df['Date']

#plot
plt.figure(figsize=(16,8))
plt.plot(df['Close'], label='Close Price history')
```

Linear Regression-

setting index as date values

```
df['Date'] = pd.to_datetime(df.Date,format='%Y-%m-%d')
df.index = df['Date']

#sorting
data = df.sort_index(ascending=True, axis=0)

#creating a separate dataset
new_data = pd.DataFrame(index=range(0,len(df)),columns=['Date', 'Close'])

for i in range(0,len(data)):
    new_data['Date'][i] = data['Date'][i]
    new_data['Close'][i] = data['Close'][i]
```

#create features

```
from fastai.structured import  add_datepart
add_datepart(new_data, 'Date')
new_data.drop('Elapsed', axis=1, inplace=True)  #elapsed will be the time stamp
new_data['mon_fri'] = 0
for i in range(0,len(new_data)):
    if (new_data['Dayofweek'][i] == 0 or new_data['Dayofweek'][i] == 4):
        new_data['mon_fri'][i] = 1
    else:
        new_data['mon_fri'][i] = 0
```

split into train and validation

```
train = new_data[:987]
valid = new_data[987:]
```

```
x_train = train.drop('Close', axis=1)
y_train = train['Close']
x_valid = valid.drop('Close', axis=1)
y_valid = valid['Close']

#implement linear regression
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)
```

```
#make predictions and find the rmse
preds = model.predict(x_valid)
rms=np.sqrt(np.mean(np.power((np.array(y_valid)-np.array(preds)),2)))
rms
```

```
#plot
valid['Predictions'] = 0
valid['Predictions'] = preds

valid.index = new_data[987:].index
train.index = new_data[:987].index

plt.plot(train['Close'])
plt.plot(valid[['Close', 'Predictions']])
```

KNN:

importing libraries

```
from sklearn import neighbors
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))
```

scaling data

```
x_train_scaled = scaler.fit_transform(x_train)
x_train = pd.DataFrame(x_train_scaled)
x_valid_scaled = scaler.fit_transform(x_valid)
x_valid = pd.DataFrame(x_valid_scaled)

#using gridsearch to find the best parameter
params = {'n_neighbors':[2,3,4,5,6,7,8,9]}
knn = neighbors.KNeighborsRegressor()
model = GridSearchCV(knn, params, cv=5)
```

```
#fit the model and make predictions
model.fit(x_train,y_train)
preds = model.predict(x_valid)
```

```
#rmse
rms=np.sqrt(np.mean(np.power((np.array(y_valid)-np.array(preds)),2)))
rms
```

```
#plot
valid['Predictions'] = 0
valid['Predictions'] = preds
plt.plot(valid[['Close', 'Predictions']])
plt.plot(train['Close'])
```

LSTM:

```
#importing required libraries
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM

#creating dataframe
data = df.sort_index(ascending=True, axis=0)
new_data = pd.DataFrame(index=range(0,len(df)),columns=['Date', 'Close'])
for i in range(0,len(data)):
    new_data['Date'][i] = data['Date'][i]
    new_data['Close'][i] = data['Close'][i]

#setting index
new_data.index = new_data.Date
new_data.drop('Date', axis=1, inplace=True)

#creating train and test sets
dataset = new_data.values

train = dataset[0:987,:]
valid = dataset[987:,:]

#converting dataset into x_train and y_train
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(dataset)
```

```
x_train, y_train = [], []
for i in range(60,len(train)):
    x_train.append(scaled_data[i-60:i,0])
    y_train.append(scaled_data[i,0])
x_train, y_train = np.array(x_train), np.array(y_train)

x_train = np.reshape(x_train, (x_train.shape[0],x_train.shape[1],1))

# create and fit the LSTM network
model = Sequential()
model.add(LSTM(units=50, return_sequences=True,
input_shape=(x_train.shape[1],1)))
model.add(LSTM(units=50))
model.add(Dense(1))

model.compile(loss='mean_squared_error', optimizer='adam')
model.fit(x_train, y_train, epochs=1, batch_size=1, verbose=2)

#predicting 246 values, using past 60 from the train data
inputs = new_data[len(new_data) - len(valid) - 60:].values
inputs = inputs.reshape(-1,1)
inputs  = scaler.transform(inputs)

X_test = []
for i in range(60,inputs.shape[0]):
    X_test.append(inputs[i-60:i,0])
X_test = np.array(X_test)

X_test = np.reshape(X_test, (X_test.shape[0],X_test.shape[1],1))
closing_price = model.predict(X_test)
closing_price = scaler.inverse_transform(closing_price)
```

```
rms=np.sqrt(np.mean(np.power((valid-closing_price),2)))
```

```
rms
```

```
#for plotting
```

```
train = new_data[:987]
valid = new_data[987:]
valid['Predictions'] = closing_price
plt.plot(train['Close'])
plt.plot(valid[['Close','Predictions']])
```

5. PERFORMANCE ANALYSIS

To analyse the performance and accuracy of the three machine learning algorithms used we calculate their root mean square error(RMSE). The values we got were:

1. Linear regression-121.16291596523156
2. KNN-115.17086550026721
3. LSTM-11.772259608962642

Thus we can see that the RMSE values for linear regression and K nearest neighbours is similar, which is very high in comparison to the long short term memory algorithm.

Thus we conclude that long short term memory algorithm has performed best in all of the three algorithms for our problem statement.

6. CONCLUSION

Linear regression is a simple technique and quite easy to interpret, but there are a few obvious disadvantages. One problem in using regression algorithms is that the model overfits to the date and month column. Instead of taking into account the previous values from the point of prediction, the model will consider the value from the same date a month ago, or the same date/month a year ago.

Like linear regression, kNN also identified a drop in January 2018 since that has been the pattern for the past years. We can safely say that regression algorithms have not performed well on this dataset.

The LSTM model can be tuned for various parameters such as changing the number of LSTM layers, adding dropout value or increasing the number of epochs. The predictions from LSTM are best to identify whether the stock price will increase or decrease.

Stock price is affected by the news about the company and other factors like demonetization or merger/demerger of the companies. There are certain intangible factors as well which can often be impossible to predict beforehand.