STOCK PRICE PREDICTION OF NIFTY 50 COMPANIES

Presented By :

Bhopatrao Nihal (Roll no. 5)

Kumbhare Sanket (Roll no. 36)
Wankhede Shashank (Roll no. 59)

Guide By :

Prof. A. S. Kunte

Problem Statement:

Creating a website which will be able to predict future price of certain company out of Nifty 50 companies using deep learning techniques in the form of table or graph

DESCRIPTION

- What is Nifty 50?
  A stock is the certain amount of ownership in company.
  Basic goal of this project is to use deep learning to make prediction of stock price.
  This project focuses on NIFTY 50 companies only.
  We are going to implement deep learning algorithm know as long short-term memory (LSTM) which is based on Recurrent Neural Network (RNN).

OBJECTIVES

1. To implement LSTM model which will be train on available historical data.
2. To forecast the next 30 days price of stock of particular NIFTY 50 listed company in which user is interested.
3. To integrate this model in order to develop simple system through which user can interact and perform predictions as per his requirements i.e company name and number of predictions.
4. To provide insights of data in graphical format such as bar charts,line chart,comparison between predicted vs actual,accuracy of prediction etc.

SCOPE

1. The project is focuses on applications of model(LSTM) for stock price prediction.
2. In this project we are considering only NIFTY 50 listed companies as their historical data and real time data are easy to fetch.
3. To predict stock price user will have option of uploading dataset of that company
4. User will be able to make prediction of next 30 days.

Time Series Forecasting

Time Series Forecasting is type of problem in machine learning or deep learning
When we want to make prediction over the period of time we consider it as time series forecasting problem and these predictions may be sales or revenue after 1 -2 months or price of stock of a company or any arbitrary value which make some sense in real world

Time Series Forecasting Algorithms

- Autoregression (AR)
- Moving Average (MA)
- Autoregressive Moving Average (ARMA)
- Autoregressive Integrated Moving Average (ARIMA)
- Seasonal Autoregressive Integrated Moving-Average (SARIMA)
- Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX)
- Vector Autoregression (VAR)
- Vector Autoregression Moving-Average (VARMA)
- Vector Autoregression Moving-Average with Exogenous Regressors (VARMAX)
- Simple Exponential Smoothing (SES)
- Holt Winter's Exponential Smoothing (HWES)
- Naïve, Snaïve
- Exponential smoothing
- TBATS
- Prophet
- NNETAR
- LSTM

Summarising

- After reading several articles and watching some youtube videos we conclude that there are various algorithms for time series forecasting out of them LSTM performed well for stock price prediction

- Linear regression being linear in nature does not perform well with respect to time however still we can predict price using this algorithm

- Auto ARIMA or ARIMA is useful for stationary dataset and because our problem is non-stationary it will not much efficient

- For predicting stock price of tommorrow we need to provide today’s opening price,closing price etc which means we need to use previous output as input which is possible in RNN i.e LSTM

Implementation

```
[ ]  start = dt.datetime(2011,1,17)
     end = dt.datetime(2020,12,31)
     ds = web.DataReader("WIPRO.NS","yahoo",start,end)
```

```
[ ]  ds.tail()
```

Date	High	Low	Open	Close	Volume	Adj Close
2020-12-28	386.399994	382.000000	383.450012	382.899994	4725879.0	382.041077
2020-12-29	390.500000	383.100006	384.000000	385.000000	11459126.0	384.136383
2020-12-30	386.600006	382.799988	385.000000	384.399994	7188435.0	383.537720
2020-12-31	387.600006	381.200012	381.200012	386.250000	6394605.0	385.383575
2021-01-01	390.750000	385.049988	385.049988	388.100006	5042336.0	387.229431

Implementation

```
[ ]  past_days = 30
```

```
[ ]  # preparing independent and dependent features
     def prepare_data(timeseries_data, n_features):
       X, y =[],[]
       for i in range(len(timeseries_data)):
         # find the end of this pattern
         end_ix = i + n_features
         # check if we are beyond the sequence
         if end_ix > len(timeseries_data)-1:
           break
         # gather input and output parts of the pattern
         seq_x, seq_y = timeseries_data[i:end_ix], timeseries_data[end_ix]
         X.append(seq_x)
         y.append(seq_y)
       return np.array(X), np.array(y)
```

Implementation

```
plt.figure(figsize=(16,8))
plt.plot(model.predict(X),color='red',label='Predicted')
plt.plot(ds['Close'].values,color='green',label='Actual')
```

```
[<matplotlib.lines.Line2D at 0x7f36707ede10>]
```

```
start = dt.datetime(2011,1,17)
end = dt.datetime(2020,12,31)
df = web.DataReader("WIPRO.NS","yahoo",start,end)
dataset=df
dataset=dataset['Close'].values
dataset=dataset[len(dataset)-30:]

n_steps=30

x_input = np.array(dataset.tolist())
temp_input=list(x_input)
lst_output=[]
i=0
while(i<31):

    if(len(temp_input)> past_days):
        x_input=np.array(temp_input[1:])
        print("{} day input {}".format(i,x_input))
        #print(x_input)
        x_input = x_input.reshape((1, n_steps, n_features))
        #print(x_input)
        yhat = model.predict(x_input, verbose=0)
        print("{} day output {}".format(i,yhat))
        temp_input.append(yhat[0][0])
        temp_input=temp_input[1:]
        #print(temp_input)
        lst_output.append(yhat[0][0])
        i=i+1
    else:
        x_input = x_input.reshape((1, n_steps, n_features))
        yhat = model.predict(x_input, verbose=0)
        print(yhat[0])
        temp_input.append(yhat[0][0])
        lst_output.append(yhat[0][0])
        i=i+1
```

[30]
```
compare_df = pd.DataFrame({'Actual':df2['Close'].values,
                           'Predicted':lst_output,
                          'Diff':df2['Close'].values-lst_output
                       })
compare_df
```

	Actual	Predicted	Diff
0	388.100006	371.843872	16.256134
1	396.399994	373.580780	22.819214
2	406.299988	379.351135	26.948853
3	406.399994	378.447083	27.952911
4	406.750000	379.498901	27.251099
5	430.200012	383.427185	46.772827
6	446.799988	384.531677	62.268311
7	457.700012	387.585114	70.114899
8	459.000000	388.239197	70.760803
9	454.350006	388.512024	65.837982
10	438.549988	382.440308	56.109680

[30]

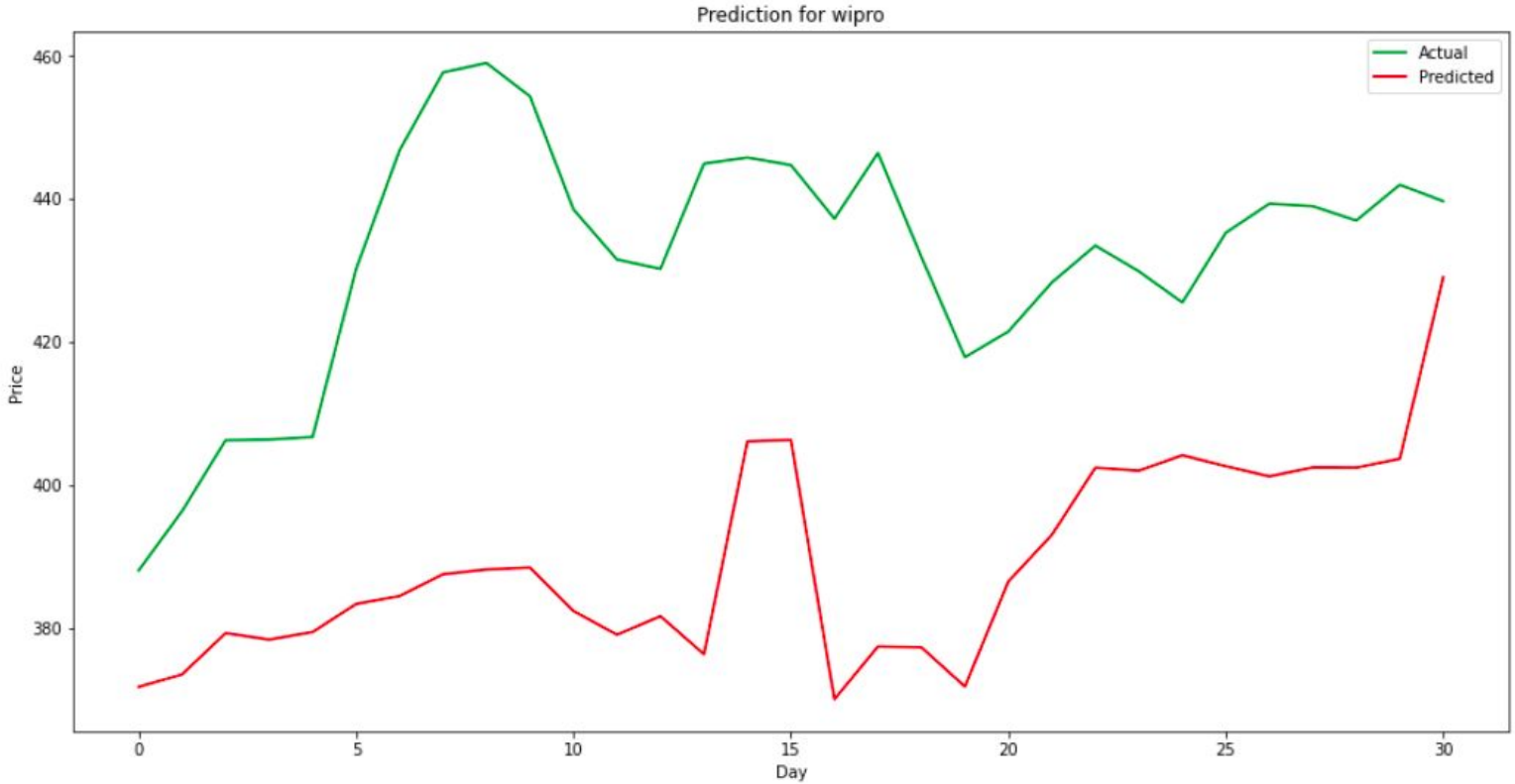
10	438.549988	382.440308	56.109680
11	431.549988	379.128723	52.421265
12	430.250000	381.722076	48.527924
13	444.950012	376.400665	68.549347
14	445.799988	406.152527	39.647461
15	444.750000	406.340454	38.409546
16	437.250000	370.118927	67.131073
17	446.450012	377.479034	68.970978
18	431.899994	377.365448	54.534546
19	417.899994	371.889557	46.010437
20	421.500000	386.574707	34.925293
21	428.350006	393.086090	35.263916
22	433.500000	402.467743	31.032257
23	429.899994	402.045288	27.854706
24	425.549988	404.204681	21.345306
25	435.299988	402.673401	32.626587
26	439.350006	401.254395	38.095612
27	439.000000	402.501770	36.498230
28	437.000000	402.469879	34.530121
29	442.000000	403.717041	38.282959
30	439.700012	429.104248	10.595764

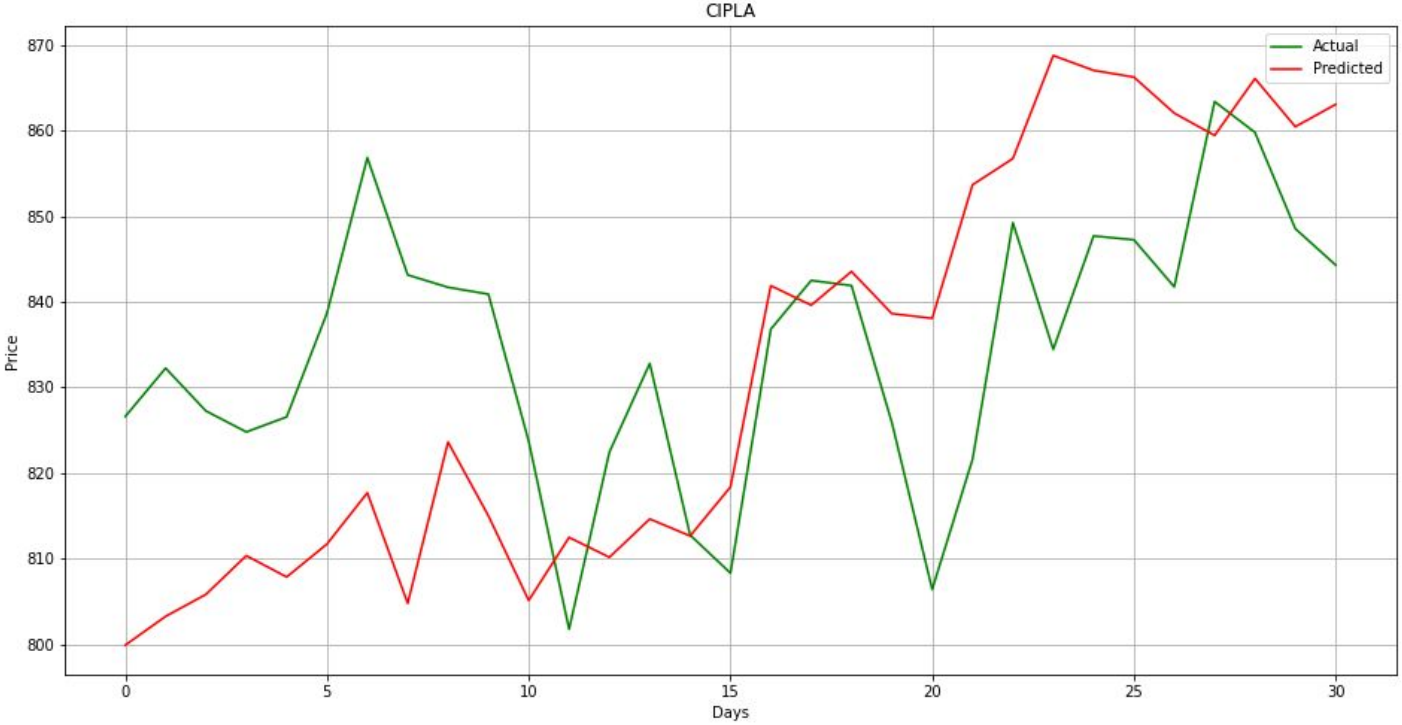
```
[46] #Evaluting
     rmse = np.sqrt(np.mean(compare_df['Diff'])**2)
     rmse
```

```
49.07242313508065
```

```
plt.plot(compare_df['Predicted'],color='red',label='Predicted')
plt.title('Prediction for wipro')
plt.xlabel('Day')
plt.ylabel('Price')
plt.legend()
```

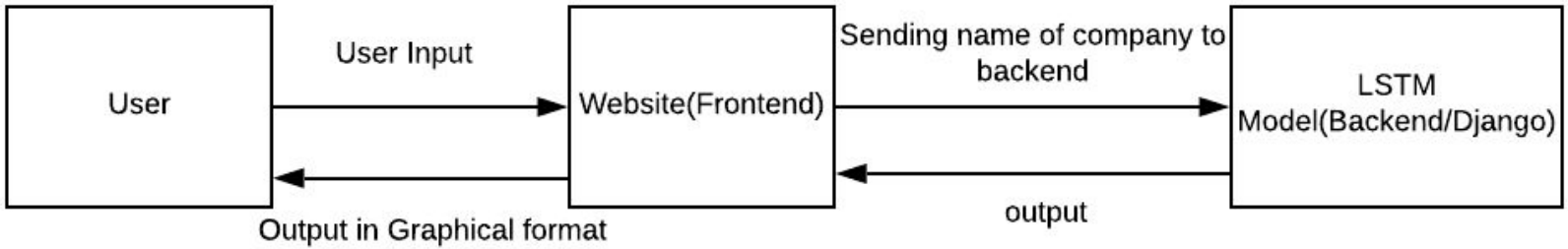
```
<matplotlib.legend.Legend at 0x7ff188e4a350>
```




CIPLA
Actual
Predicted
Price
Days
870
860
850
840
830
820
810
800
0
5
10
15
20
25
30

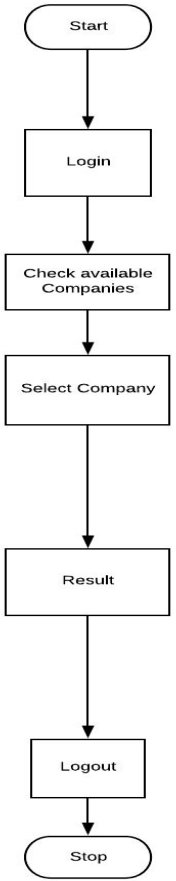
System Design:

- Block Diagram

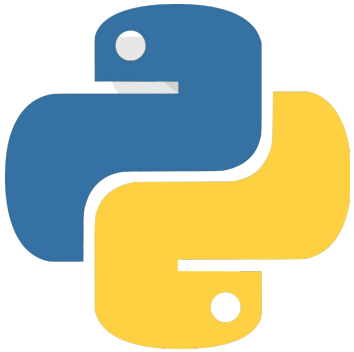


System Design:

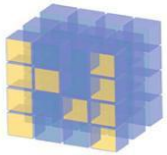
- Activity Diagram



Overview of Our System



django



NumPy

Keras

NSEpy

SQLite

Chart.js

pandas

Source:Google

1. **Framework for Backend:**

Django is a Python-based free and open-source web framework that follows the model-view-controller architectural pattern.

2. Processing input:

Django based website will take input i.e historical data from user and process it in real time using predefined deep-learning model

3. Graphical Representation:

Website will produce value of stock in graphical format such as bar charts,line chart,comparison between predicted vs actual,accuracy of prediction etc

Website User Interface

Nifty50 Prediction

News About Register

Login

Username

Password

Login

Create account for Predictions

Please login or register for making predictions.

Top Stories

Nifty50 Prediction

News About Gaitonde

Choose Company

Submit

Predcition for next 30 days

Day	Closing Price
1	3971.75
2	3940.06
3	3999.65
4	3951.54
5	4014.18
6	4029.35
7	4025.73
8	3957.95
9	3957.45
10	3910.93
11	3969.09

Choose Company Submit

Predcition for next 30 days

Day	Closing Price
1	652.61
2	636.92
3	665.64
4	683.73
5	682.81
6	686.83
7	678.62
8	681.56
9	656.72
10	649.60
11	652.08

Top Stories

REPEAT -- Victoria Gold Adopts Shareholder Rights Plan

2 hours ago

Considering the Gold sovereign bond offer? Here's what you can do

2 hours ago

Gold bond opens for subscription, issue price at discount to market rates

2 hours ago

Discovery, AT&T, Tesla, Palantir: What to Watch When the Stock Market Opens Today

3 hours ago

Wynn Interactive to Become Public Company via SPAC

3 hours ago

Saudi Arabia's Jabal Omar reports steeper first-quarter loss amid Covid-19 restrictions

3 hours ago

InnerScope Hearing Technologies (OTC: INND) Retains Skyline Corporate Communications Group, LLC to P...

3 hours ago

India Inc has 50% independent directors on its boards, ratio in US is 85%

3 hours ago

Reference

- https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/
- https://www.youtube.com/watch?v=Vfx1L2jh2Ng
- https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/
- https://towardsdatascience.com/an-overview-of-time-series-forecasting-models-a2fa7a358fcb
- https://en.wikipedia.org/wiki/Recurrent_neural_network
- https://www.analyticsvidhya.com/blog/2017/12/introduction-to-recurrent-neural-networks/
- https://www.youtube.com/watch?v=ndz8WKFqu8o
- https://www.researchgate.net/publication/340636297_Stock_Market_Prediction_Using_LSTM_Recurrent_Neural_Network


Thank
You!