TABLE OF CONTENTS

Abstract

The Stock market process is full of uncertainty and is affected by many factors. Hence the Stock market analysis is one of the important exertions in finance and business. There are two types of analysis possible for prediction, technical and fundamental. In this project we will discuss about technical analysis they are Simple Moving Average (SMA), Exponential Moving Average (EMA), Bollinger Bands (BBands), RSI and MACD. The Bollinger Bands are indicated volatility by upper and lower band of cost for simple moving average. Relative Strength Index is a momentum oscillator that indicates change of price movement and speed of stock. The moving average convergence divergence (MACD) is one of the most well-known and used indicators in technical analysis. This indicator is comprised of two moving averages, which help to measure momentum in the security. Now the study two types of moving average measures which are: Exponential moving average and Simple moving average.

Introduction

Stock market is the bone of fast emerging economies such as India. Major of capital infusion for companies across the country was made possible only thru shares sold to people. So, our country growth is tightly bounded with the performance of our stock market. The purpose of Bollinger Band is to determine high and low of stocks. Prices of stocks are high at upper band and low at lower band. Investor can take systematic trading decision by recognize the pattern of the stock. Relative Strength Index is a technical indicator determines strength and weakness of stock on closing price of a recent trading period. RSI measures volatility and magnitude of price movement direction of stock. The moving average convergence divergence (MACD) is one of the most well-known and used indicators in technical analysis. This indicator is comprised of two moving averages, which help to measure momentum in the security. We will compare Exponential Moving Average based MACD with Simple Moving Average based MACD of technical analysis explains which method is better for the technical analysis based on moving average.

Objectives

The Main Objective is to analyse the historical stock market data using technical analysis methods like Simple Moving Average (SMA), Exponential Moving Average (EMA), Bollinger Bands (BBands), Moving Average Convergence Divergence (MACD).

We will Analyse as a comparative study between those technical analytic methods and suggest the best method with proper reasons for future analysis and prediction on stock market.

Problem Statement

The basic strategy is to buy 100 stocks of Nike when the strategy generates a buy signal for any given indicator and sell 100 stocks after 'n' days. Technical analysis is done for predicting short term price momentum of any asset class; hence we've decided to hold the stock only for a short term (60 days). You can go for any number of days and see what works out best. We will demonstrate how to implement and back-test technical analysis trading strategies and not any investment advice.

Literature survey

s.no	Title	Author/journal	Techniques	Results
1.	A Comparative Study on Technical Analysis by Bollinger Band and RSI	Shah Nisarg Pinakin International Journal in Management and Social Science	BB (Bollinger Band)., RSI (Relative Strength Index).	Out of two methods Bollinger Band comparatively give good return and profit than RSI.
2.	Comparison Between Exponential Moving Average Based MACD with Simple Moving Average Based MACD of Technical Analysis	Naik Parth Pradipbhai IJSR - International Journal Of Scientific Research.	Exponential Moving Average based MACD, Simple Moving Average based MACD	EMA based MACD generates maximum profit and return compare to SMA based MACD.
3.	Stock Market Prediction using Data Mining	Govinda.K Institute of Electrical and Electronics Engineers (IEEE) 2017	Random Forest Model,Support Vector Machine	On the basis of the results obtained, we can say that both the models exhibited notable performance in predicting the direction of the stock index. The Random Forest model using a 1-gram model for text analysis produced an accuracy of 84.3% and on using a 2-gram model produced an accuracy of 86.2%.The linear Support Vector Machine using 1gram model and 2-gram model for text analysis produced predictions with an accuracy of 82.2% and 84.6%, while the nonlinear Support Vector Machine produced predictions with an accuracy of 85.1% for both 1-gram and 2-gram models.

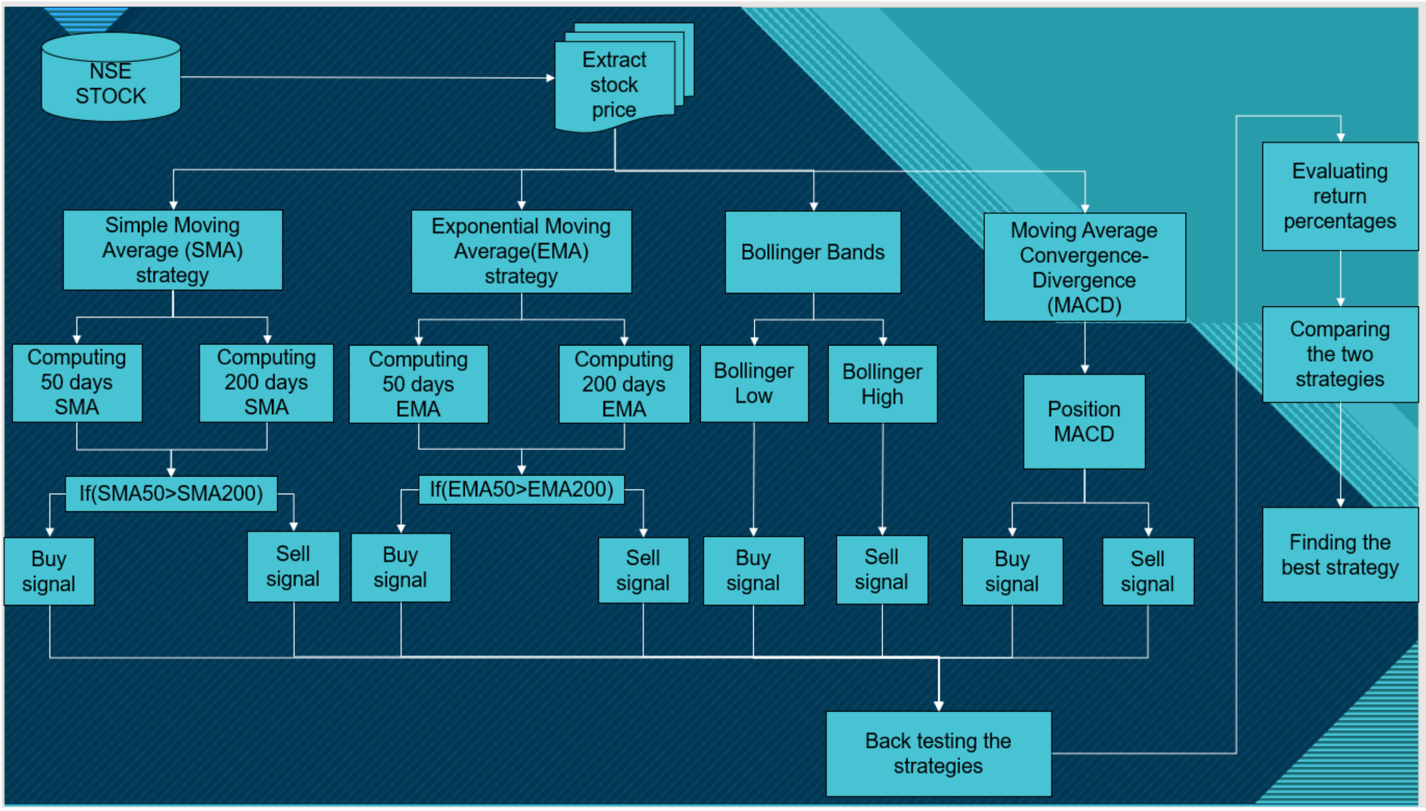
4.	Stock Market Prediction: A Big Data Approach	Girija V Attigeri Institute of Electrical and Electronics Engineers (IEEE) 2015	Big Data Analysis, Sentiment Analysis , Machine Learning	A prediction model has been built that uses big data analytical capabilities, social media analytics and machine learning to periodically predict the trend about stock markets. Model shows that sentiment analysis of the social data complements proven technical analysis methods such as regression analysis. It shows that volatility of the markets and the future performance of the system is affected by the economic and political news and influence of the social media.
5.	An analysis on Stock Market Prediction	Dr.D.Ezhilmaran International Journal of Computer Science & Engineering Technology (IJCSET) 2016	Fuzzy Neural Networks, Kmeans clustering	Limited work is done with fundamental approaches which give plenty of opportunity for further research. Since the stock data is highly volatile and unpredictable it needs the intelligence of human for effective prediction. Also it needs rigorous training of old data for analysis. This temperament of stock data makes data mining and AI techniques as suitable once. Back propagation algorithm for training and suitable AI technique applied on some fundamental approaches may render promising results.
6.	Deep Learning Based Forecasting in Stock Market	Gozde Sismanoglu Institute of Electrical and Electronics Engineers (IEEE) 2019	Deep Neural Networks (DNNs),Recurrent Neural Networks (RNNs),LSTM.	LSTM network is applied to a large-scale stock market NYSE, NASDAQ, and NYSE MKT, ranging from January 2, 1968, to April 09, 2018 for predicting the future values. With the use of LSTM architectures, some hidden dynamics of the market can be captured, and efficient predictions can be possible. Proposed model results in an acceptable root mean square error (RMSE) value as 0.04.

Tools & Technologies

Jupyter Notebook is a web application for creating and sharing documents that contain code, visualizations, and text. It can be used for data science, statistical modeling, machine learning, and much more.

Excel: Analyze Data in Excel empowers you to understand your data through natural language queries that allow you to ask questions about your data without having to write complicated formulas.

System Architecture



DESCRIPTION OF MODULES:

In our project we are using different technical indicators.

Technical Indicators The technical indicators we're going to be using here are:

- Simple Moving Average
- Exponential Moving Average
- Bollinger Bands
- Moving Average Convergence Divergence (MACD)

1. Simple Moving Average (SMA)

Simple Moving Average is a technical indicator which calculates the average of 'Close' price of an underlying stock for a given period of time. eg. SMA_50 calculates the average of close price for the last 50 days.

Basic SMA strategy is that if the short term (fast) MA > long term (slow) MA, it's a signal to buy the stock. Similarly, if the short term (fast) MA < long term (slow) MA, it's a signal to sell the stock.

2. Exponential Moving Average (EMA)

The exponential moving average (EMA) is a technical chart indicator that tracks the price of an investment (like a stock or commodity) over time. The EMA is a type of weighted moving average (WMA) that gives more weighting or importance to recent price data.

The exponential moving average is designed to improve on the idea of a simple moving average (SMA) by giving more weight to the most recent price data, which is considered to be more relevant than older data. Since new data carries greater weight, the EMA responds more quickly to price changes than the SMA.

3. Bollinger Bands

In the 1980s, John Bollinger, a long-time technician of the markets, developed the technique of using a moving average with two trading bands above and below it. Unlike a percentage calculation from a normal moving average, Bollinger Bands simply add and subtract a standard deviation calculation.

When stock prices continually touch the upper Bollinger Band, the prices are thought to be overbought, triggering a sell signal; conversely, when they continually touch the lower band, prices are thought to be oversold, triggering a buy signal.

4. Moving Average Convergence-Divergence (MACD)

MACD is a trend following momentum-indicator which shows relationship between the moving avearges of a security's market price. It is calculated by subtracting 26-day EMA (exponential moving average) from 12-day EMA.

The result of that calculation is the MACD line. A nine-day EMA of the MACD called the "signal line," is then plotted on top of the MACD line, which can function as a trigger for buy and sell signals. Traders may buy the security when the MACD crosses above its signal line and sell, or short the security when the MACD crosses below the signal line. Moving average convergence divergence (MACD) indicators can be interpreted in several ways, but the more common methods are crossovers, divergences, and rapid rises/falls.

Back-testing

SMA Crossover Strategy,

EMA Crossover Strategy,

Bollinger Bands Strategy,

MACD strategy

Our general strategy using **SMA, EMA, Bollinger Bands** and **MACD** is to buy and hold 100 stocks of Nike when a bullish signal is generated and sell after 60 days.

Complete program/CODE with outputs

n [4]:
```
# importing necessary libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import pandas_datareader as web
import datetime
from datetime import date

%matplotlib inline
```

n [5]:
```
# Fetching Nike data from Yahoo! Finance
Nike = web.get_data_yahoo('NKE',start = datetime.datetime(2018, 1, 2),
                              end = date.today())
```

[216]:
```
#Nike.to_csv('C:\\Users\\sree\\Desktop\\FDA_project\\Nike.csv', index=True)
```

n [6]:
```
Nike.head()
```

ut[6]:

Date	High	Low	Open	Close	Volume	Adj Close
2018-01-02	63.490002	62.849998	62.849998	63.490002	6511000.0	60.613392
2018-01-03	63.660000	62.759998	63.480000	63.480000	6091100.0	60.603848
2018-01-04	63.549999	62.549999	63.400002	63.439999	5780500.0	60.565659
2018-01-05	64.300003	63.470001	63.700001	63.980000	11632300.0	61.081196
2018-01-08	64.709999	63.980000	64.150002	64.550003	11905700.0	61.625370

In [8]:
```
Nike.index
```

Out[8]:
```
DatetimeIndex(['2018-01-02', '2018-01-03', '2018-01-04', '2018-01-05',
               '2018-01-08', '2018-01-09', '2018-01-10', '2018-01-11',
               '2018-01-12', '2018-01-16',
               ...
               '2022-10-24', '2022-10-25', '2022-10-26', '2022-10-27',
               '2022-10-28', '2022-10-31', '2022-11-01', '2022-11-02',
               '2022-11-03', '2022-11-04'],
              dtype='datetime64[ns]', name='Date', length=1221, freq=None)
```

In [9]:
```
Nike = Nike.drop(columns=['Volume', 'Adj Close'])
Nike.head()
```

Out[9]:

Date	High	Low	Open	Close
2018-01-02	63.490002	62.849998	62.849998	63.490002
2018-01-03	63.660000	62.759998	63.480000	63.480000
2018-01-04	63.549999	62.549999	63.400002	63.439999
2018-01-05	64.300003	63.470001	63.700001	63.980000
2018-01-08	64.709999	63.980000	64.150002	64.550003

In [9]:
```
# Plotting Nike Close Price over the years
fig = go.Figure()
fig.add_trace(go.Scatter(x= Nike.index, y= Nike.Close, name='Close'))
fig.update_layout(title='Price of Nike over the Years', xaxis_title='Date', yaxis_title='Price')
fig.show()
```

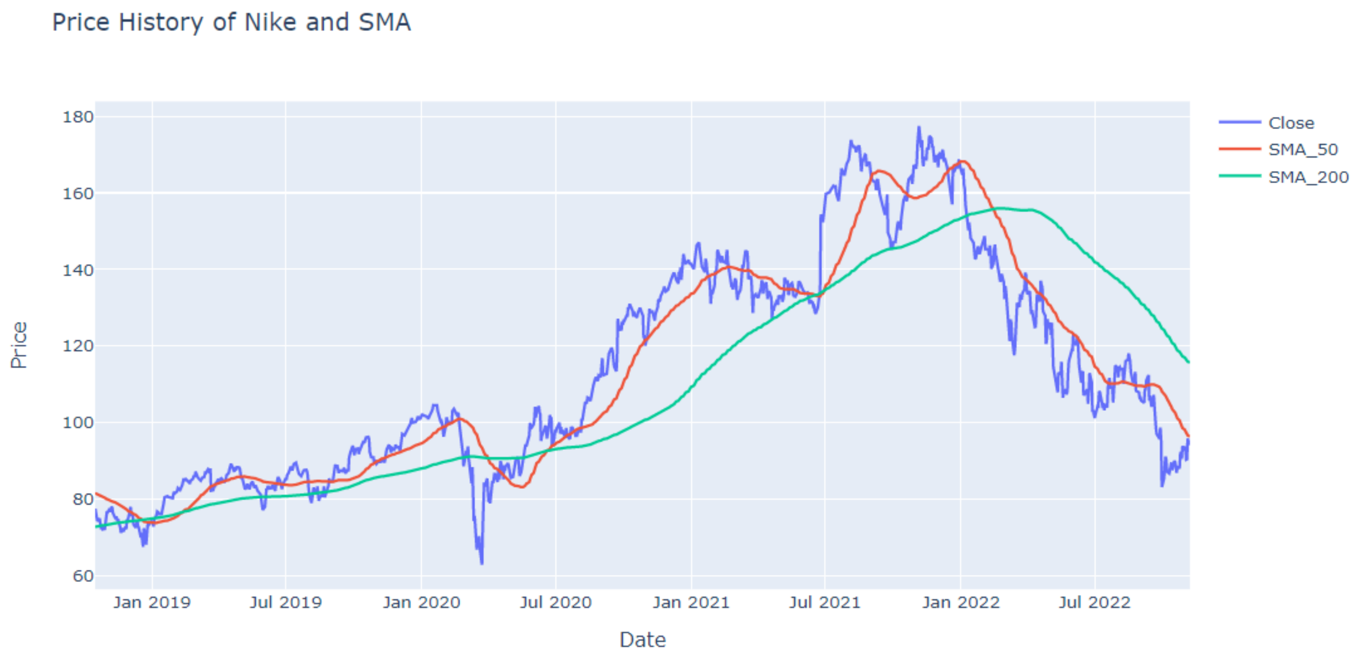
Simple Moving Average (SMA)

6]:
```
# Fast SMA (50 days)
Nike['SMA_50'] = Nike['Close'].rolling(50).mean()

# Slow SMA (200 days)
Nike['SMA_200'] = Nike['Close'].rolling(200).mean()

Nike = Nike.dropna()
```

7]:
```
Nike.head()
```

7]:

	High	Low	Open	Close	SMA_50	SMA_200
Date						
2018-10-16	77.650002	75.320000	75.599998	77.480003	81.534800	72.71425
2018-10-17	77.800003	75.940002	77.519997	76.480003	81.453801	72.77920
2018-10-18	77.410004	75.000000	77.400002	75.599998	81.355800	72.83980
2018-10-19	76.050003	74.139999	75.830002	74.209999	81.214800	72.89365
2018-10-22	75.209999	74.300003	74.650002	74.900002	81.098200	72.94825

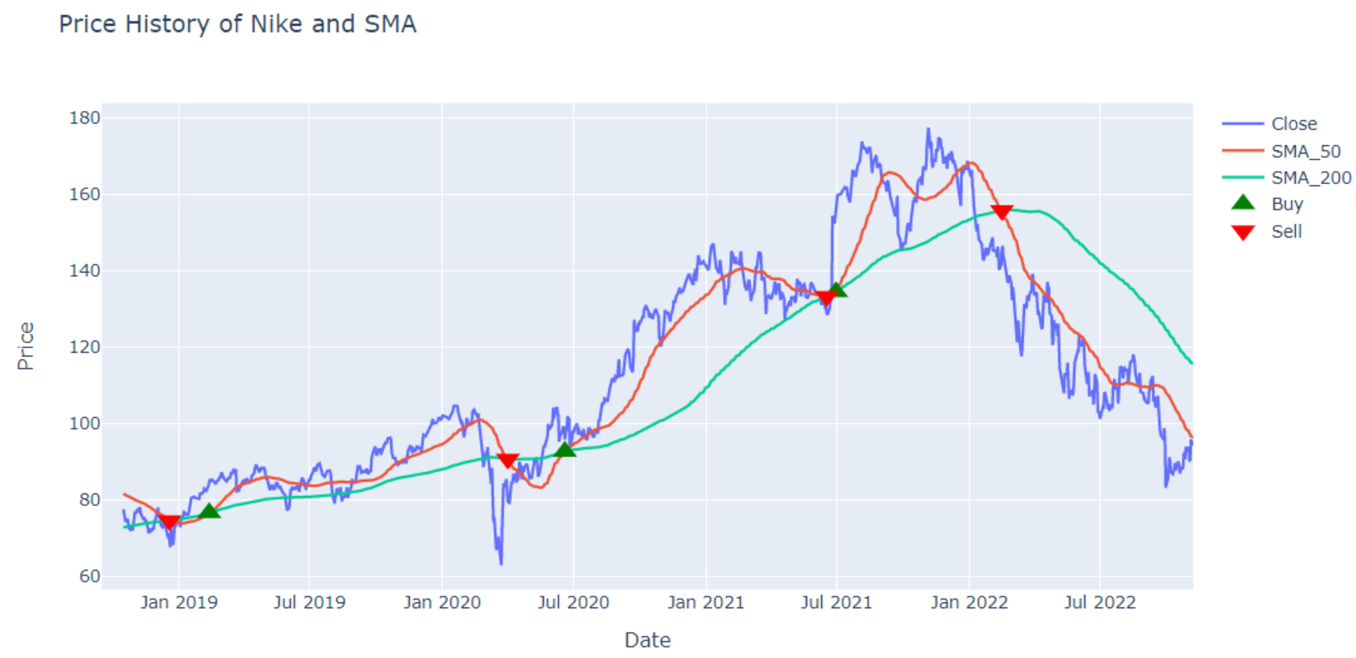


Generating Trade Signals using SMA

```python
# 'Buy' signal if Short SMA > Long SMA else 'Sell'
Nike['Signal_SMA'] = np.where(Nike['SMA_50'] > Nike['SMA_200'], 1.0, 0.0)
Nike['Position_SMA'] = Nike['Signal_SMA'].diff()
Nike.head(250)
```

Date	High	Low	Open	Close	SMA_50	SMA_200	Signal_SMA	Position_SMA
2018-10-16	77.650002	75.320000	75.599998	77.480003	81.534800	72.71425	1.0	NaN
2018-10-17	77.800003	75.940002	77.519997	76.480003	81.453801	72.77920	1.0	0.0
2018-10-18	77.410004	75.000000	77.400002	75.599998	81.355800	72.83980	1.0	0.0
2018-10-19	76.050003	74.139999	75.830002	74.209999	81.214800	72.89365	1.0	0.0
2018-10-22	75.209999	74.300003	74.650002	74.900002	81.098200	72.94825	1.0	0.0
...	...	...	...	...	...	...	...	...
2019-10-08	92.709999	90.410004	90.449997	91.750000	85.927000	84.05945	1.0	0.0
2019-10-09	93.050003	91.820000	92.410004	92.519997	86.033400	84.16020	1.0	0.0
2019-10-10	93.500000	92.720001	93.500000	93.000000	86.172800	84.28470	1.0	0.0
2019-10-11	94.570000	93.559998	94.000000	93.879997	86.387999	84.38905	1.0	0.0
2019-10-14	95.250000	94.050003	94.199997	94.879997	86.662799	84.49510	1.0	0.0

250 rows × 8 columns



Back-testing SMA Strategy

```
32]: Nike['Position_SMA'].value_counts()
```

```
32]:  0.0    1015
     -1.0       4
      1.0       3
     Name: Position_SMA, dtype: int64
```

```
6]: buyAmt = 0
    sellAmt = 0
    buyDates = np.array([])

    for i in range(Nike.shape[0]):
        if Nike.iloc[i, 7]==1:
            buyAmt = buyAmt + Nike.iloc[i, 3]*100
            buyDates = np.append(buyDates, i)

    print('Total Amount Invested:',buyAmt)

    for i in range(Nike.shape[0]):
        for j in buyDates:
            if i == j:
                if (int(j)+60 < Nike.shape[0]):
                    sellAmt = sellAmt + Nike.iloc[int(j+60), 3]*100
                else:
                    sellAmt = sellAmt + Nike.iloc[int(j+10), 3]*100

    total_SMA = sellAmt - buyAmt

    print('Total Cumulative Returns:',total_SMA)
    print('Returns in %:',(sellAmt/buyAmt)*100)
```

```
Total Amount Invested: 33498.00033569336
Total Cumulative Returns: 1685.9992980957031
Returns in %: 105.03313416084485
```

Exponential Moving Average (EMA)

38]:
```
# Fast EMA (50 days)
Nike['EMA_50'] = Nike['Close'].ewm(span= 50, adjust=False).mean()

# Slow EMA (200 days)
Nike['EMA_200'] = Nike['Close'].ewm(span= 200, adjust=False).mean()

Nike.head()
```

38]:

Date	High	Low	Open	Close	SMA_50	SMA_200	Signal_SMA	Position_SMA	EMA_50	EMA_200
2018-10-16	77.650002	75.320000	75.599998	77.480003	81.534800	72.71425	1.0	NaN	77.480003	77.480003
2018-10-17	77.800003	75.940002	77.519997	76.480003	81.453801	72.77920	1.0	0.0	77.440788	77.470053
2018-10-18	77.410004	75.000000	77.400002	75.599998	81.355800	72.83980	1.0	0.0	77.368600	77.451446
2018-10-19	76.050003	74.139999	75.830002	74.209999	81.214800	72.89365	1.0	0.0	77.244733	77.419192
2018-10-22	75.209999	74.300003	74.650002	74.900002	81.098200	72.94825	1.0	0.0	77.152783	77.394126

Generating Trade Signals using EMA

39]:
```
# 'Buy' signal if Short EMA > Long EMA else 'Sell'
Nike['Signal_EMA'] = np.where(Nike['EMA_50'] > Nike['EMA_200'], 1.0, 0.0)
Nike['Position_EMA'] = Nike['Signal_EMA'].diff()
Nike.head()
```

39]:

Date	High	Low	Open	Close	SMA_50	SMA_200	Signal_SMA	Position_SMA	EMA_50	EMA_200	Signal_EMA	Position_EMA
2018-10-16	77.650002	75.320000	75.599998	77.480003	81.534800	72.71425	1.0	NaN	77.480003	77.480003	0.0	NaN
2018-10-17	77.800003	75.940002	77.519997	76.480003	81.453801	72.77920	1.0	0.0	77.440788	77.470053	0.0	0.0
2018-10-18	77.410004	75.000000	77.400002	75.599998	81.355800	72.83980	1.0	0.0	77.368600	77.451446	0.0	0.0
2018-10-19	76.050003	74.139999	75.830002	74.209999	81.214800	72.89365	1.0	0.0	77.244733	77.419192	0.0	0.0
2018-10-22	75.209999	74.300003	74.650002	74.900002	81.098200	72.94825	1.0	0.0	77.152783	77.394126	0.0	0.0

Back-testing EMA Strategy

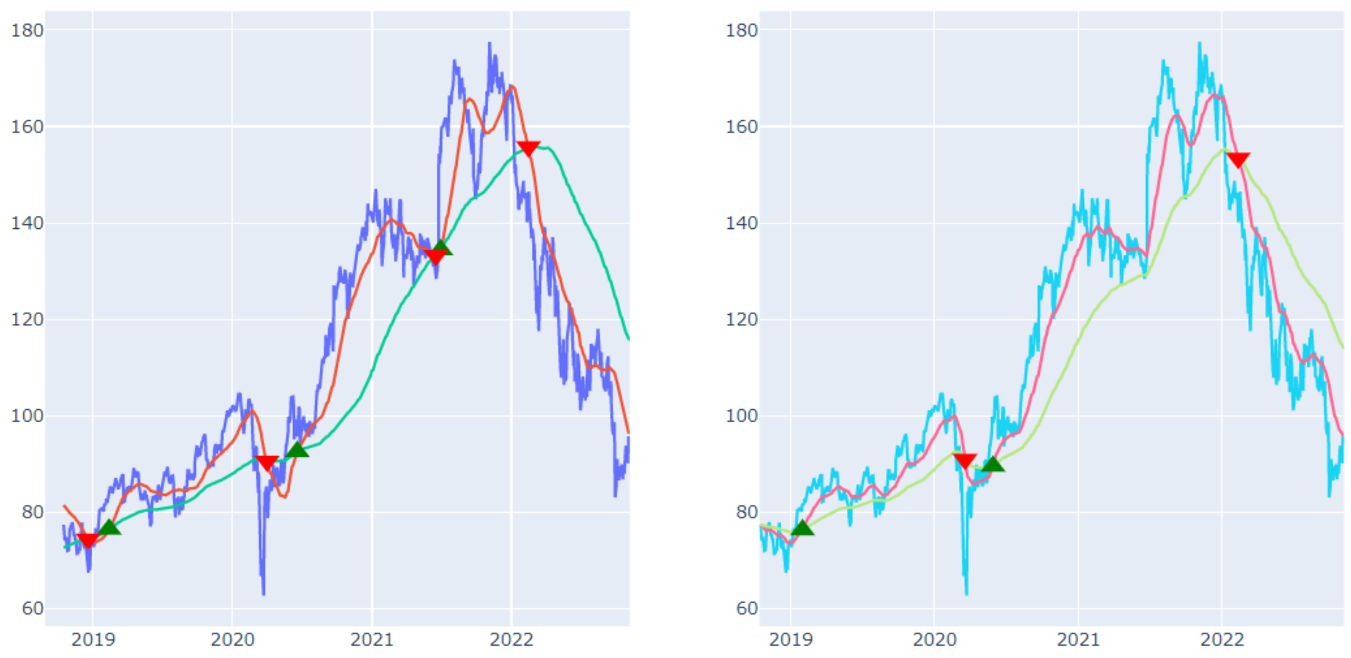
```
1]: Nike.columns.get_loc('Position_EMA')
```

```
1]: 11
```

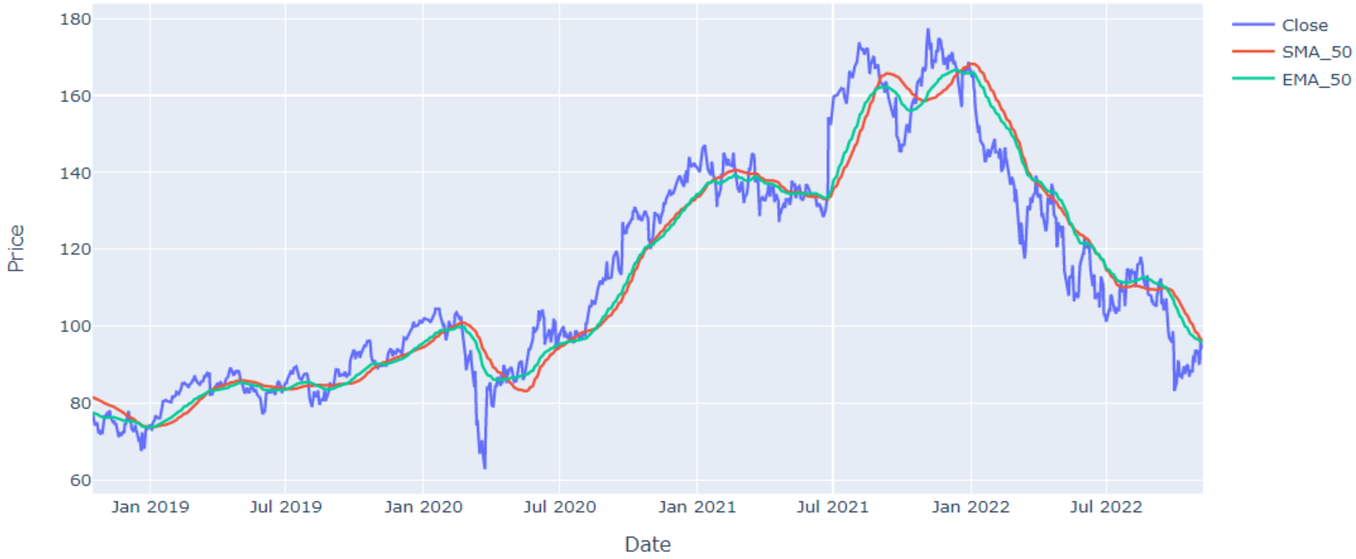
```
2]: buyAmt = 0
sellAmt = 0
buyDates = np.array([])

for i in range(Nike.shape[0]):
    if Nike.iloc[i, 11]==1:
        buyAmt = buyAmt + Nike.iloc[i, 3]*100
        buyDates = np.append(buyDates, i)

print('Total Amount Invested:',buyAmt)

for i in range(Nike.shape[0]):
    for j in buyDates:
        if i == j:
            if (int(j)+60 < Nike.shape[0]):
                sellAmt = sellAmt + Nike.iloc[int(j+60), 3]*100
            else:
                sellAmt = sellAmt + Nike.iloc[int(j+10), 3]*100

total_EMA = sellAmt - buyAmt

print('Total Cumulative Returns:',total_EMA)
print('Returns in %:',(sellAmt/buyAmt)*100)
```

```
Total Amount Invested: 18045.999908447266
Total Cumulative Returns: 1963.00048828125
Returns in %: 110.87775960456688
```

SMA and EMA Trading Strategies Comparison



Price History of Nike, SMA_50 and EMA_50



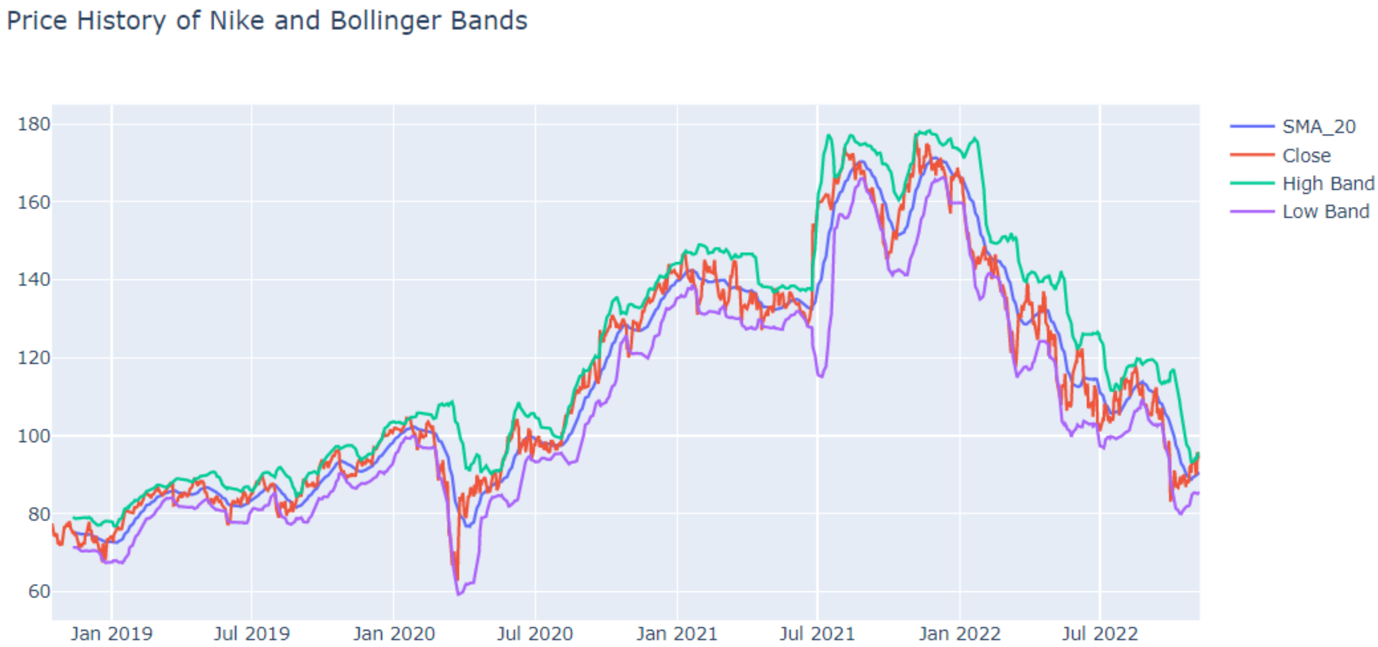
Price History of Nike, SMA_200 and EMA_200

Bollinger Bands

]:
```python
# Calculating Bollinger Bands considering 20 days and 2 standard deviations

Nike['Rolling Mean'] = Nike['Close'].rolling(20).mean()
Nike['Rolling Std'] = Nike['Close'].rolling(20).std()

Nike['Bollinger High'] = Nike['Rolling Mean'] + (Nike['Rolling Std']*2)
Nike['Bollinger Low'] = Nike['Rolling Mean'] - (Nike['Rolling Std']*2)
Nike.tail()
```

]:

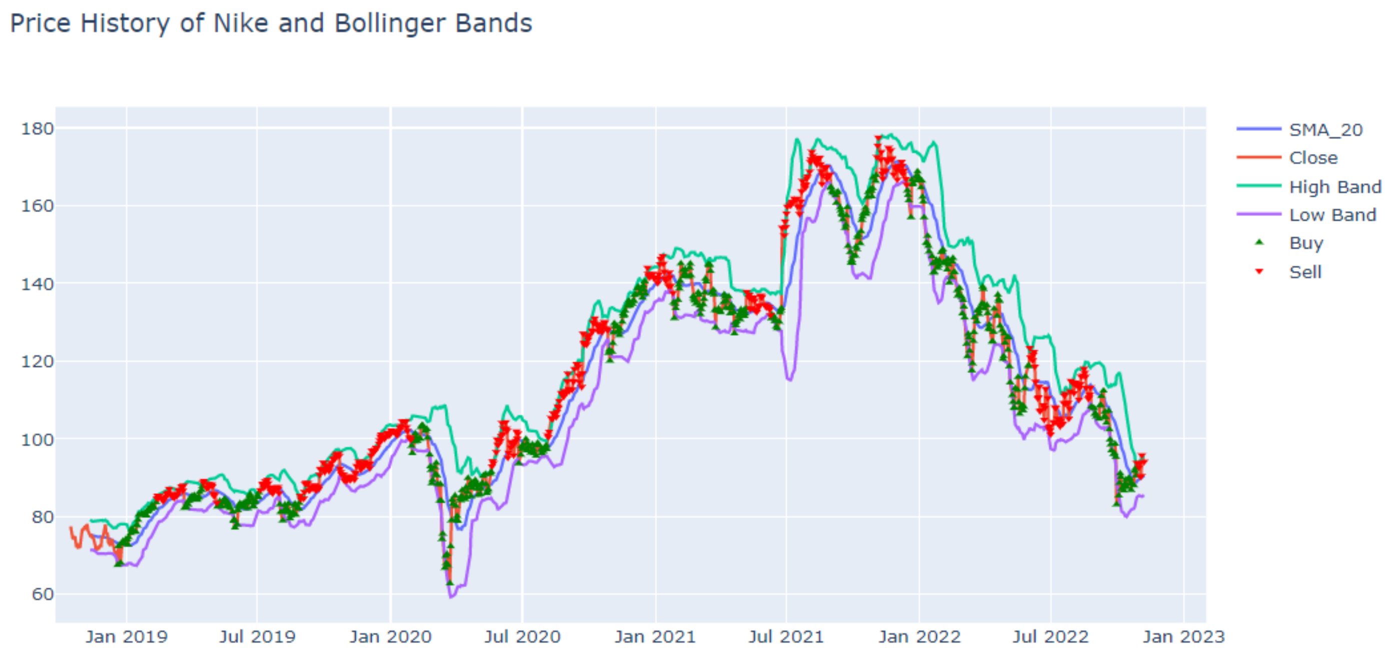
ɔpen	Close	SMA_50	SMA_200	Signal_SMA	Position_SMA	EMA_50	EMA_200	Signal_EMA	Position_EMA	Rolling Mean	Rolling Std	Bollinger High	Bollinger Low
0002	93.769997	97.7962	116.709850	0.0	0.0	96.504700	114.806535	0.0	0.0	89.761000	2.236161	94.233322	85.288678
9997	90.300003	97.3886	116.426600	0.0	0.0	96.261379	114.562689	0.0	0.0	89.721001	2.218030	94.157061	85.284940
0001	90.400002	96.9684	116.153050	0.0	0.0	96.031521	114.322265	0.0	0.0	89.732501	2.221075	94.174651	85.290351
9997	95.790001	96.6198	115.918350	0.0	0.0	96.022049	114.137864	0.0	0.0	90.164001	2.513982	95.191965	85.136036
9998	94.114998	96.3365	115.674175	0.0	0.0	95.947263	113.938631	0.0	0.0	90.535250	2.522188	95.579627	85.490874



Generating trade signals using Bollinger Bands

[49]:
```
Nike['Position_BB'] = None

# Fill our newly created position column - set to sell (-1) when the price hits the upper band, and set to
# buy (1) when it hits the lower band
for row in range(len(Nike)):

    if (Nike['Close'].iloc[row] > Nike['Bollinger High'].iloc[row]) and (Nike['Close'].iloc[row-1] < Nike['Bollinger High'].iloc[
        Nike['Position_BB'].iloc[row] = -1

    if (Nike['Close'].iloc[row] < Nike['Bollinger Low'].iloc[row]) and (Nike['Close'].iloc[row-1] > Nike['Bollinger Low'].iloc[ro
        Nike['Position_BB'].iloc[row] = 1

# Forward fill our position column to replace the "None" values with the correct long/short
# positions to represent the "holding" of our position forward through time
Nike['Position_BB'].fillna(method='ffill',inplace=True)
Nike.tail()
```

se	SMA_50	SMA_200	Signal_SMA	Position_SMA	EMA_50	EMA_200	Signal_EMA	Position_EMA	Rolling Mean	Rolling Std	Bollinger High	Bollinger Low	Position_BB
97	97.7962	116.709850	0.0	0.0	96.504700	114.806535	0.0	0.0	89.761000	2.236161	94.233322	85.288678	-1.0
03	97.3886	116.426600	0.0	0.0	96.261379	114.562689	0.0	0.0	89.721001	2.218030	94.157061	85.284940	-1.0
02	96.9684	116.153050	0.0	0.0	96.031521	114.322265	0.0	0.0	89.732501	2.221075	94.174651	85.290351	-1.0
01	96.6198	115.918350	0.0	0.0	96.022049	114.137864	0.0	0.0	90.164001	2.513982	95.191965	85.136036	-1.0
98	96.3365	115.674175	0.0	0.0	95.947263	113.938631	0.0	0.0	90.535250	2.522188	95.579627	85.490874	-1.0



Back-testing Bollinger Bands Strategy

1]:
```
Nike.columns.get_loc('Position_BB')
```

1]:
```
16
```

2]:
```
Nike['Position_BB'].value_counts()
```

2]:
```
 1.0    542
-1.0    436
Name: Position_BB, dtype: int64
```

3]:
```
buyAmt = 0
sellAmt = 0
buyDates = np.array([])

for i in range(Nike.shape[0]):
    if Nike.iloc[i, 16]==1:
        buyAmt = buyAmt + Nike.iloc[i, 3]*100
        buyDates = np.append(buyDates, i)

# print(buyDates[1])
print('Total Amount Invested:',buyAmt)

for i in range(Nike.shape[0]):
    for j in buyDates:
        if i == j:
            if (int(j)+60 < Nike.shape[0]):
                sellAmt = sellAmt + Nike.iloc[int(j+60), 3]*100
            else:
                sellAmt = sellAmt + Nike.iloc[int(j+5), 3]*100

total_BB = sellAmt - buyAmt

print('Total Cumulative Returns:',total_BB)
print('Returns in %:',(sellAmt/buyAmt)*100)
```

```
Total Amount Invested: 6183209.003067017
Total Cumulative Returns: 93182.01179504395
Returns in %: 101.50701701574091
```

Moving Average Convergence-Divergence (MACD)

]:
```
macd = Nike[['Close']]
macd.head()
```

]:

	Close
Date	
2018-10-16	77.480003
2018-10-17	76.480003
2018-10-18	75.599998
2018-10-19	74.209999
2018-10-22	74.900002

]:
```
macd['MACD'] = macd['Close'].ewm(span=12, adjust= False).mean() - macd['Close'].ewm(span=26, adjust= False).mean()
macd['Signal'] = macd['MACD'].ewm(span=9, adjust= False).mean()

Nike[['Signal_9', 'MACD']] = macd[['Signal', 'MACD']]
Nike.head()
```

SMA_200	Signal_SMA	Position_SMA	EMA_50	EMA_200	Signal_EMA	Position_EMA	Rolling Mean	Rolling Std	Bollinger High	Bollinger Low	Position_BB	Signal_9	MACD
72.71425	1.0	NaN	77.480003	77.480003	0.0	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000
72.77920	1.0	0.0	77.440788	77.470053	0.0	0.0	NaN	NaN	NaN	NaN	NaN	-0.015954	-0.079772
72.83980	1.0	0.0	77.368600	77.451446	0.0	0.0	NaN	NaN	NaN	NaN	NaN	-0.055076	-0.211562
72.89365	1.0	0.0	77.244733	77.419192	0.0	0.0	NaN	NaN	NaN	NaN	NaN	-0.128719	-0.423289
72.94825	1.0	0.0	77.152783	77.394126	0.0	0.0	NaN	NaN	NaN	NaN	NaN	-0.208836	-0.529305

Generating trade signals using MACD

```
Nike['Signal_MACD'] = np.where(Nike.loc[:, 'MACD'] > Nike.loc[:, 'Signal_9'], 1.0, 0.0)
Nike['Position_MACD'] = Nike['Signal_MACD'].diff()
Nike.head()
```

\	Position_SMA	EMA_50	EMA_200	...	Position_EMA	Rolling Mean	Rolling Std	Bollinger High	Bollinger Low	Position_BB	Signal_9	MACD	Signal_MACD	Position_MACD
)	NaN	77.480003	77.480003	...	NaN	NaN	NaN	NaN	NaN	NaN	0.000000	0.000000	0.0	NaN
)	0.0	77.440788	77.470053	...	0.0	NaN	NaN	NaN	NaN	NaN	-0.015954	-0.079772	0.0	0.0
)	0.0	77.368600	77.451446	...	0.0	NaN	NaN	NaN	NaN	NaN	-0.055076	-0.211562	0.0	0.0
)	0.0	77.244733	77.419192	...	0.0	NaN	NaN	NaN	NaN	NaN	-0.128719	-0.423289	0.0	0.0
)	0.0	77.152783	77.394126	...	0.0	NaN	NaN	NaN	NaN	NaN	-0.208836	-0.529305	0.0	0.0

Price History of Nike and MACD

Back-testing MACD strategy

8]:
```
Nike.columns.get_loc('Position_MACD')
```

8]:
```
20
```

9]:
```
Nike['Position_MACD'].value_counts()
```

9]:
```
 0.0    949
 1.0     37
-1.0     36
Name: Position_MACD, dtype: int64
```

0]:
```
buyAmt = 0
sellAmt = 0
buyDates = np.array([])

for i in range(Nike.shape[0]):
    if Nike.iloc[i, 20]==1:
        buyAmt = buyAmt + Nike.iloc[i, 3]*100
        buyDates = np.append(buyDates, i)

print('Total Amount Invested:', buyAmt)
for i in range(Nike.shape[0]):
    for j in buyDates:
        if i == j:
            if (int(j)+60 < Nike.shape[0]):
                sellAmt = sellAmt + Nike.iloc[int(j+60), 3]*100
            else:
                sellAmt = sellAmt + Nike.iloc[int(j+8), 3]*100

total_MACD = sellAmt - buyAmt

print('Total Cumulative Returns:',total_MACD)
print('Returns in %:',(sellAmt/buyAmt)*100)
```

```
Total Amount Invested: 427870.0019836426
Total Cumulative Returns: 11871.997833251953
Returns in %: 102.77467403141429
```

Result and discussion

- If we had followed this same strategy using SMA from 2018 till date, our total returns would be 105.03% i.e $10,000 invested using the same strategy would have generated $20,503 (as on 07/11/22).
- If we had followed this same strategy using EMA from 2018 till date, our total returns would be 110.87% i.e $10,000 invested using the same strategy would have generated $21,087 (as on 07/11/22).
- We can also see here that EMA generated better results as compared to SMA. This is because EMA gives more weightage to recent prices while in SMA we calculate the average of all 'Close' prices over a period of time.
- If we had followed this same strategy using Bollinger Bands from 2018 till date, our total returns would be 101.50% i.e $10,000 invested using the same strategy would have generated $20,150 (as on 07/11/22).
- If we had followed this same strategy using MACD from 2018 till date, our total returns would be 102.77% i.e $10,000 invested using the same strategy would have generated $20,277 (as on 07/11/22).

Conclusion

In this project, we have learnt how to predict stock prices using some popular technical indicators like Simple Moving Average (SMA), Exponential Moving Average (EMA), Bollinger Bands (BBands), Moving Average Convergence Divergence (MACD) and back-tested to calculate returns over 4 years (2018-Present). We can also consider pairing other external factors with these technical indicators to improvise on our trading strategy and get much better returns.

Limitations

- Technical analysis only works in a free market, and when the market is being manipulated, it's no longer free
- In times of extreme fear or extreme optimism, don't expect technical analysis to help you earn any money
- Technical analysis has the same limitation of any strategy based on particular trade triggers. The chart can be misinterpreted. The formation may be predicated on low volume. The periods being used for the moving averages may be too long or too short for the type of trade you are looking to make.
- In some cases, one of the technical indicators will show a buy signal and another indicator will show a sell signal. This causes confusion in trading decisions.
- For example, when a possible entry or exit point for a stock is suggested, it doesn’t guarantee a successful trade. Stock may decrease after the entry. Stock can also rise after the exit.

Future work

As part of our future work, we will extend our work by exploring the information and we will also try to find other different types of analysis and without any limitations will try to perform analysis which can be useful for the traders to invest in stocks.

References

https://finance.yahoo.com/news/getting-cheap-nike-inc-nyse-120148820.html

https://corporatefinanceinstitute.com/resources/equities/bollinger-bands/

https://www.investopedia.com/ask/answers/difference-between-simple-exponential-moving-average/