PREDICTING STOCK MARKET MOVEMENTS

Group 2:
Daniel Tan (U2121315A)
Aditya Kumar Pugalia(U2123212D)
Pavanraj Selvaraju (U2122616H)

TABLE OF CONTENTS

PRACTICAL MOTIVATION

Dataset

Stock data from Feb 2013 - Feb 2018 for all stocks included in the S&P500 index

- Date
- Day High
- Day Low
- Open Price
- Close Price
- Volume

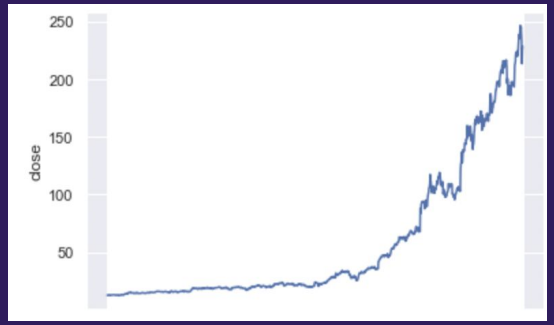
https://www.kaggle.com/datasets/camnugent/sandp500

PROBLEM DEFINITION

Which machine learning model is the most appropriate in predicting price movements in the stock market based on a set of predictors?

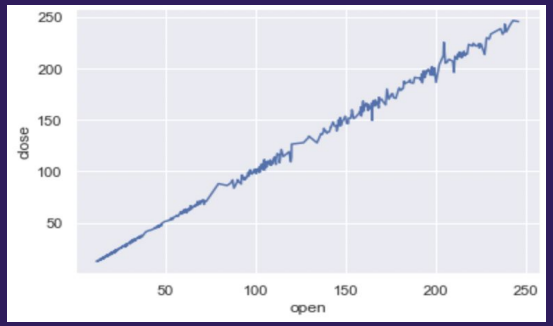
Exploratory Data Analysis

Initial Insights - NVDA



Price by date

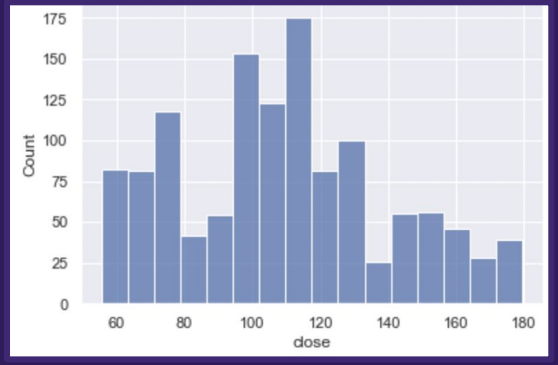
Price was stable for a while, the increased dramatically in recent years. However, short term price movements remained mysterious.



Open vs Close

Daily open and close prices had a nearly perfect linear relationship. However, we could still take advantage of the small fluctuations.

Data Cleaning



Daily close price was reasonably spread out with a slight skew to the left. This indicated that daily price movements were relatively free of anomalies

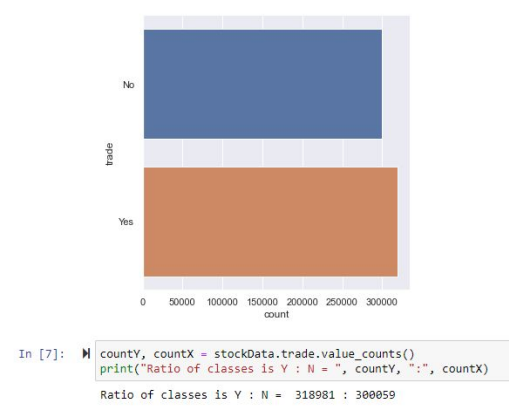
Data Cleaning

Since our motivation was to predict market movements based on data predictors, we cleaned up the data by removing the dates where there was significant market movement due to external macroeconomic shocks.

- 20 June 2013, stock market selloff due to Fed announcing plan to wind down stimulus packages
- 24 January 2014, stock market selloff due to contractions in emerging markets and weak employment data in the US
- 9-15 October 2014, stock market selloff due to Ebola scare and ISIL (or ISIS) aggression in Syria

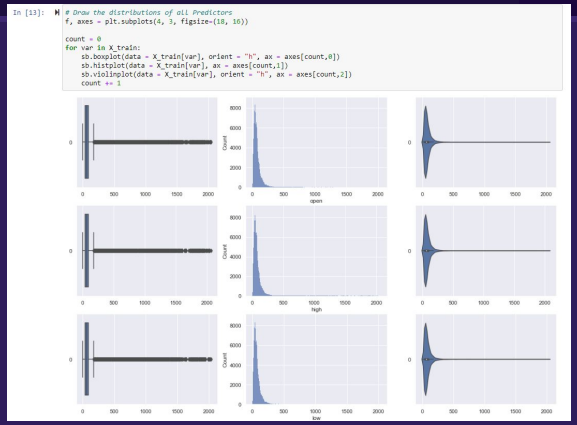
CLASSIFICATION

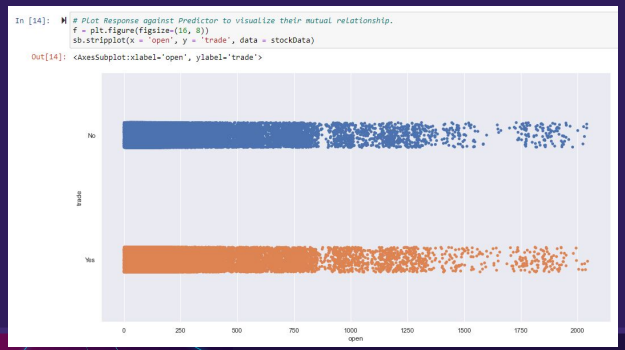
EXPLORATORY DATA ANALYSIS

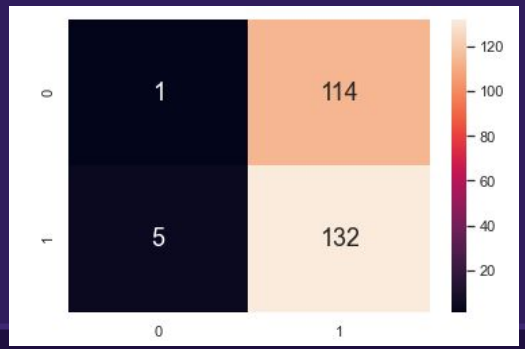



```
In [7]:  countY, countX = stockData.trade.value_counts()
         print("Ratio of classes is Y : N = ", countY, ":", countX)

Ratio of classes is Y : N =  318981 : 300059
```







ANALYSIS

In [92]:

```
# Print the Classification Accuracy
print("Test Data")
print("Accuracy  :\t", tree_CA_SP_d2.score(X_test, y_test))
print()

# Print the Accuracy Measures from the Confusion Matrix
cmTest = confusion_matrix(y_test, y_test_pred)
tpTest = cmTest[1][1] # True Positives : Y (1) predicted Y (1)
fpTest = cmTest[0][1] # False Positives : N (0) predicted Y (1)
tnTest = cmTest[0][0] # True Negatives : N (0) predicted N (0)
fnTest = cmTest[1][0] # False Negatives : Y (1) predicted N (0)

print("TPR Test :\t", (tpTest/(tpTest + fnTest)))
print("TNR Test :\t", (tnTest/(tnTest + fpTest)))
print()

print("FPR Test :\t", (fpTest/(fpTest + tnTest)))
print("FNR Test :\t", (fnTest/(fnTest + tpTest)))
```

```
Test Data
Accuracy  :      0.5277777777777778

TPR Test :       0.9635036496350365
TNR Test :       0.008695652173913044

FPR Test :       0.991304347826087
FNR Test :       0.0364963503649635
```

TIME SERIES FORECASTING

Why Time Series Analysis?

- Extrapolating future values based on historical time stamped Data
- Can help predict how stock prices move with time and see if there are any seasonal trends.
- Can be used to forecast on which days the stocks will yield profits

Method(1): preparing Data for forecasting

- Extracting the opening and closing prices of the data for the Stock (NVDA) from the original dataframe.
- Creating a new dataframe with a column for price difference which is the closing - opening price.

```
NVDA_temp= stockdata.loc[stockdata['Name'] == 'NVDA']
del NVDA_temp['Unnamed: 0']

NVDA = pd.DataFrame(NVDA_temp[['date', 'open', 'close']]).reset_index().drop(['index'], axis = 1)
diff = []
for i in range(1259):
    diff.append(NVDA.iat[i,2] - NVDA.iat[i, 1])
difference = pd.DataFrame(diff, columns = ['Diff'])
NVDA_clean = pd.concat([NVDA, difference], axis = 1)
NVDA_clean.tail()
```

	date	open	close	Diff
1254	2018-02-01	238.52	240.50	1.98
1255	2018-02-02	237.00	233.52	-3.48
1256	2018-02-05	227.00	213.70	-13.30
1257	2018-02-06	204.40	225.58	21.18
1258	2018-02-07	229.58	228.80	-0.78

Stationarity

- Stationarity means that the statistical properties of the process do not change over time.
- Stationary data ensures easier time series analysis.
- Stationarity in data can be checked with the Dickey-Fuller test

Checking for stationarity in Data

```
from statsmodels.tsa.stattools import adfuller
```

```
adft = adfuller(NVDA_clean.Diff,autolag="AIC")
output_df = pd.DataFrame({"Values":[adft[0],adft[1],adft[2],adft[3], adft[4]['1%'], adft[4]['5%'], adft[4]['10%']]
                                                    "critical value (1%)", "critical value (5%)", "critical valu
print(output_df)
```

```
         Values                      Metric
0 -8.388985e+00              Test Statistics
1  2.395100e-13                     p-value
2  1.900000e+01            No. of lags used
3  1.239000e+03  Number of observations used
4 -3.435639e+00          critical value (1%)
5 -2.863876e+00          critical value (5%)
6 -2.568013e+00         critical value (10%)
```

Forecasting using ARIMA

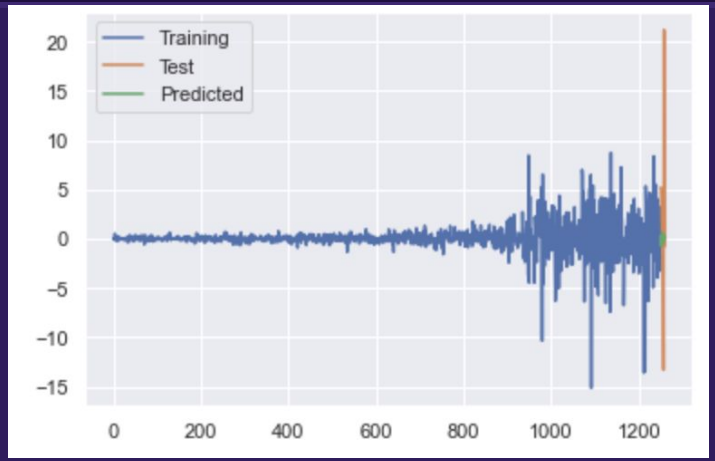
- An autoregressive integrated moving average, or ARIMA, is a statistical analysis model that predicts future values based on past values.

- The Arima function has 3 Parameters:
  - *p*: the lag order.
  - *d*: the degree of differencing.
  - q: order of the moving average.

```
from pmdarima.arima import auto_arima
model = auto_arima(train, trace=True, error_action='ignore', suppress_warnings=True)
model.fit(train)
forecast = model.predict(n_periods=len(test))
forecast = pd.DataFrame(forecast,index = test.index,columns=['Prediction'])
```

Results

	Diff	prediction
1250	5.21	0.327943
1251	4.11	−0.734904
1252	1.61	−0.146496
1253	0.03	0.589701
1254	1.98	0.027337
1255	−3.48	−0.460037
1256	−13.30	0.045916
1257	21.18	0.349083
1258	−0.78	−0.086394

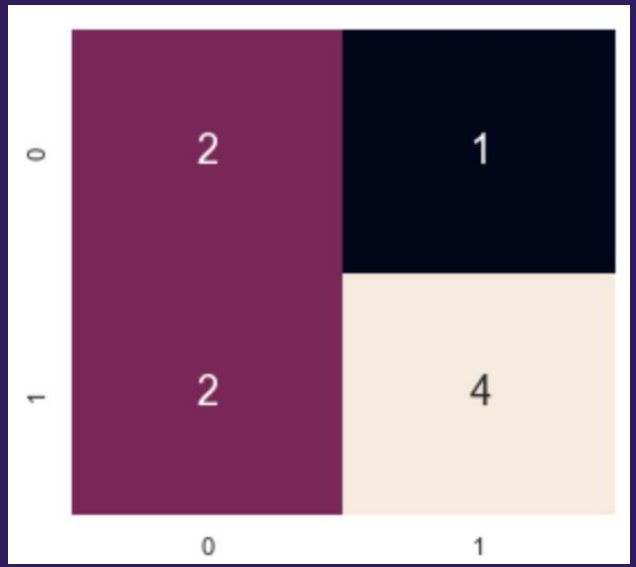


RMSE= 8.67

R^2 = -0.027

```
test_bool = []
forecast_bool = []
for i in range(9):
 if (test.iat[i,0] >= 0):
    test_bool.append(True)
 else:
    test_bool.append(False)
for i in range(9):
 if (test.iat[i,1] >= 0):
    forecast_bool.append(True)
 else:
    forecast_bool.append(False)
print("Test Profit:", test_bool)
print("Predicted Profit:", forecast_bool)
```

```
Test Profit: [True, True, True, True, True, False, False, True, False]
Predicted Profit: [True, False, False, True, True, False, True, True, False]
```



REGRESSION ANALYSIS

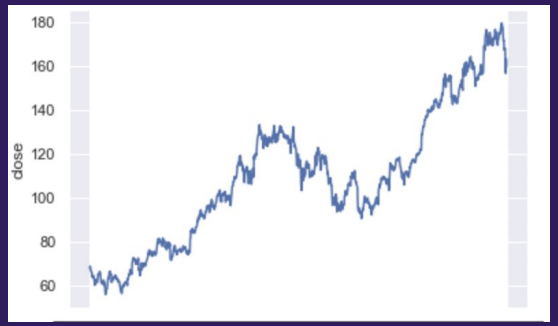
METHODOLOGY

- We aim to use the previous 9 days of data to predict the next day's price movement.
- We operate on the assumption that in short term fluctuations in price will be corrected towards the mean.
- Price movement is calculated by comparing the current day's close price with the previous day's close price.
- This way, we can build a model that predicts if the next day's close price will be higher or lower than the current day's close price.

VARIABLES

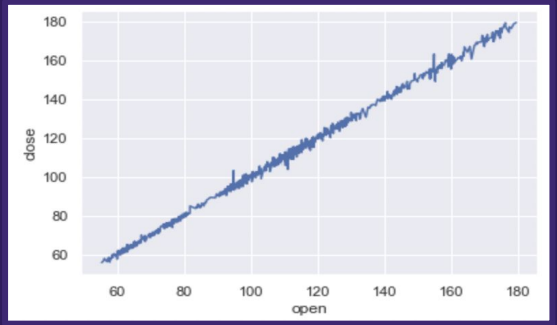
- From the original data given in the dataset, there was no discernible pattern.
- Hence we required new variables that could be useful predictors for price.

VARIABLES



Relative Strength Index[1]

RSI is a momentum indicator commonly used in trading. An RSI greater than 70 signals that a stock is overbought, and and RSI less than 30 signals that a stock is oversold.

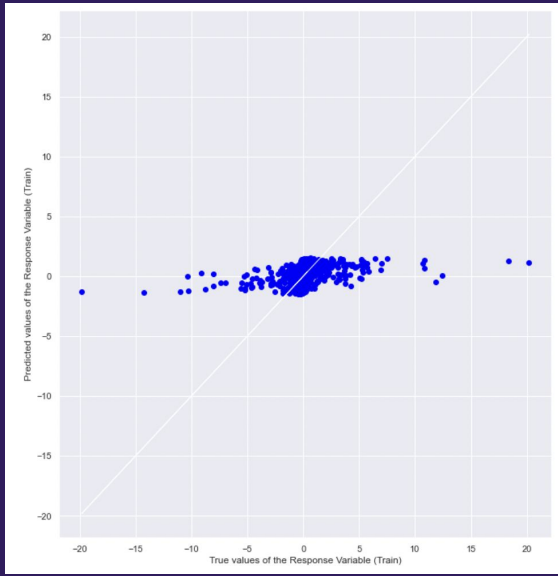


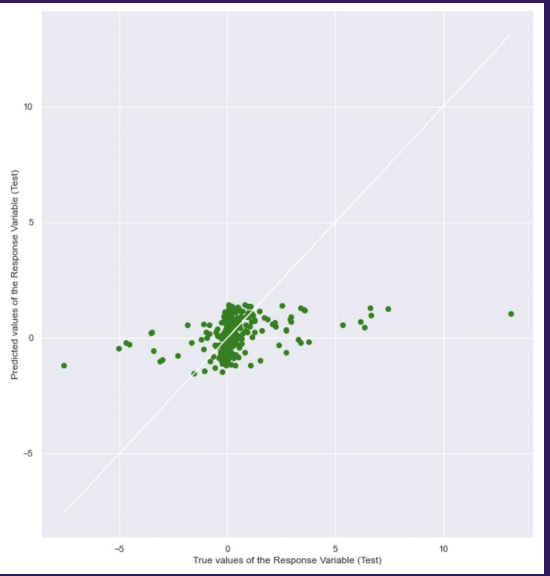
Deviation from EMA

Exponential Moving Average is a smoothed-out price curve. Deviation from EMA signals that a stock may not be trading at a fair price.

1 Fernando, J. (2022, March 18). *Relative strength index (RSI).* Investopedia. Retrieved April 24, 2022, from https://www.investopedia.com/terms/r/rsi.asp

RSI - Results





R^2: 0.113
MSE: 4.10

(Train)

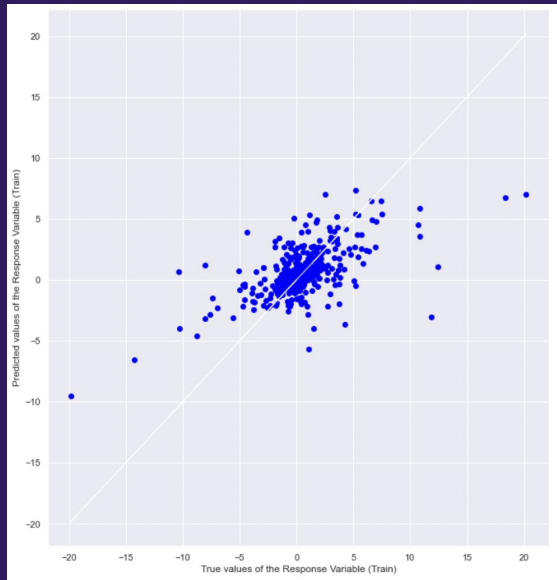
R^2: 0.120
MSE: 2.81

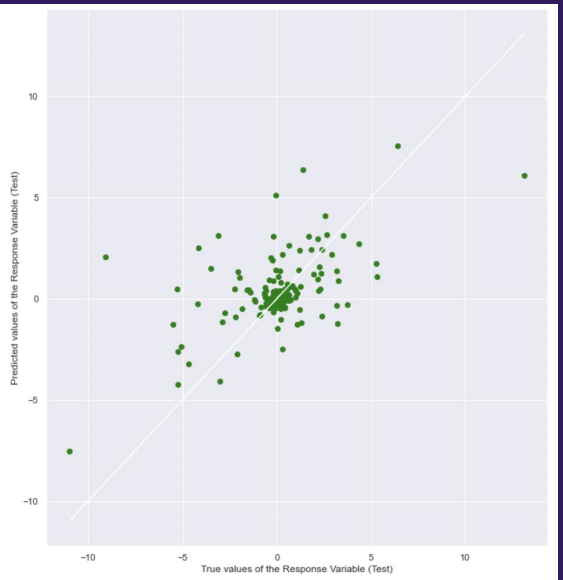
(Test)

RSI - Results

- The model did not manage to predict the price movements given RSI very well.
- We suspected that this was because RSI is a normalised variable: It can only take values between 0 and 100.
- Price movements, on the other hand, can take virtually any values due to the volatility of the market.
- Furthermore, RSI is a momentum indicator: Its current value depends on previous values.
- Price movements, on the other hand, are episodic. Any day's price movements is technically independent from the previous day's.

DEV FROM EMA - Results





R^2: 0.376
MSE: 2.79

(Train)

R^2: 0.274
MSE: 2.76

(Test)

DEV FROM EMA - Results

- The model did not manage to predict the price movements given deviations from EMA very well.
- However, this predictor did fare better than RSI.
- We suspected that this was because EMA is dependent on the stock's price, which is much larger than the price movements.
- Hence, a large percentage change in price movement, leads to a small percentage change in price and EMA.
- This gave us the idea of removing the magnitude from these predictions.

REMOVING MAGNITUDE

- In order to remove the effects of magnitude, we created 2 new variables corresponding to DEV from EMA and Price Movement:
  - Movement sign: 1 if price movement is positive, -1 if negative, 0 otherwise
  - High or Low: 1 if the DEV from EMA is positive, -1 if negative, 0 otherwise

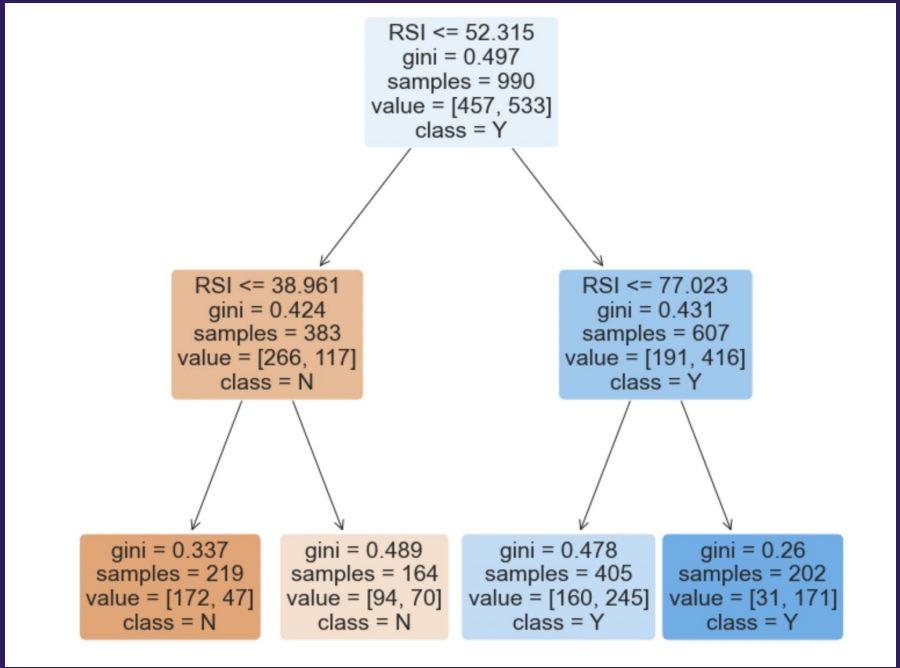
HIGH OR LOW - RESULTS

Movement sign has a negative correlation with the deviation of price from EMA. Hence we look at FPR and FNR for accurate predictions.

Prediction accuracy: ~70%

KEY OUTCOMES AND INSIGHTS

CLASSIFICATION 2.0



Accuracy : 0.722

TPR : 0.771

TNR : 0.654

ANSWERING PROBLEM DEFINITION

The final classification model that classified a day as buy or sell based on RSI yielded the best results

CONCLUSION

- The 3 models used gave us an insight when it comes to predicting stock market movements.
- Ultimately, other factors come into play as well.
- Insights from the regression models gave ideas for classification that yielded better results.