

IST-652 Final Project

Stock Market Analysis

-Mihir Deshpande

**Abstract**

Stock market is one area that captures the interests of many. Although making serious money consistently through stock market remains mere a dream for a lot of people. Many People make their trading decisions based on the word of mouth either friends, co-workers or family and then complain of losing money to this big bad gamble that is the stock market. I wanted to use this final project as an opportunity to seriously investigate what strategies professional traders use in order to make their decisions and use python and automate some of those strategies. Finally, I wanted to use twitter sentiment analysis to get a correlation between online sentiment and price movement for stocks. I would also like to apologize in advance for the longish report as some of these technical concepts needed some explaining to do. I also tried to incorporate your advice and tried to frame analysis questions around a couple of companies as against the one size fits all generalized approach that I originally took.

**Index**

1. **The Data ………………………………………………………………………………………………………………….2**
2. **Analysis objectives and questions…………………………………………………………………………….5**
3. **Techincal Indicators………………………………………………………………………………………………… 6**
   * + - 1. **Average Trdaing Volume………………………………………………………………….6**
         2. **Simple Moving Average……………………………………………………………………9**
         3. **Exponential Moving Average………………………………………………………….10**
         4. **Moving Average Convergence Divergence………………………………………11**
         5. **Relative Strength Index…………………………………………………………………..13**
4. **Trading Strategy………………………………………………………………………………………………………15**
5. **Twitter Sentiment Analysis………………………………………………………………………………………18**
6. **Conclusion and Future Scope……………………………………………………………………………………22**

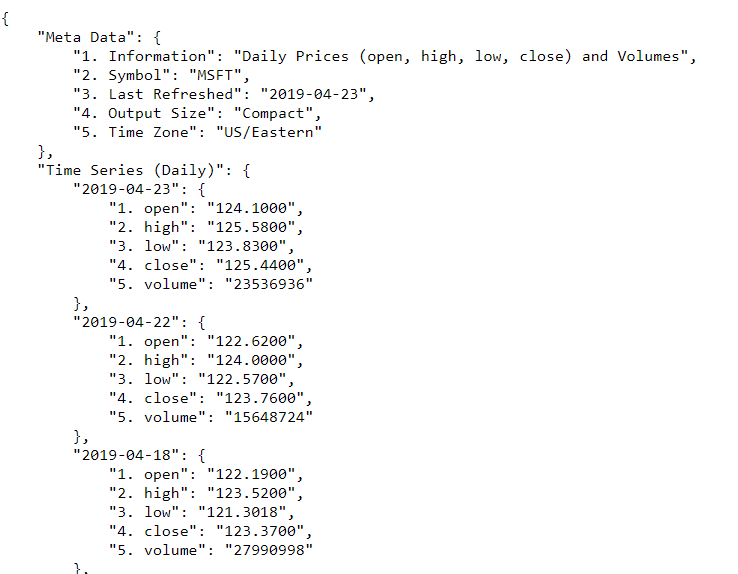
**The Data**

For the final project, I have chosen stock market data for my analysis. The data was sourced from <https://www.alphavantage.co/> which provides near real-time data for a stock. The data can be requested in json format or a downloadable csv format. For the purpose of this project I have chosen the json format.

Sample call to the API:

[https://www.alphavantage.co/query?**function**=TIME\_SERIES\_DAILY&**symbol**=MSFT&**apikey**=demo](https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=MSFT&apikey=demo)

This returns the stock data for Microsoft (given by the parameter symbol) in the query.



This json file contains meta data and the actual Time series (daily) values. Time series (daily) will give us the daily values for that stock as against hourly, weekly or monthly. This specification is to be done within the API query.

To get this data in the python environment packages ‘requests’ and ‘json’ were used. json. loads () represents the json data as a python object(dictionary). The required data is under the ‘Time Series (Daily)’ key and these values were stored as a pandas data frame for further analysis.



Some were pre-processing steps need to be done to ensure that the data is in the right format for the analysis.

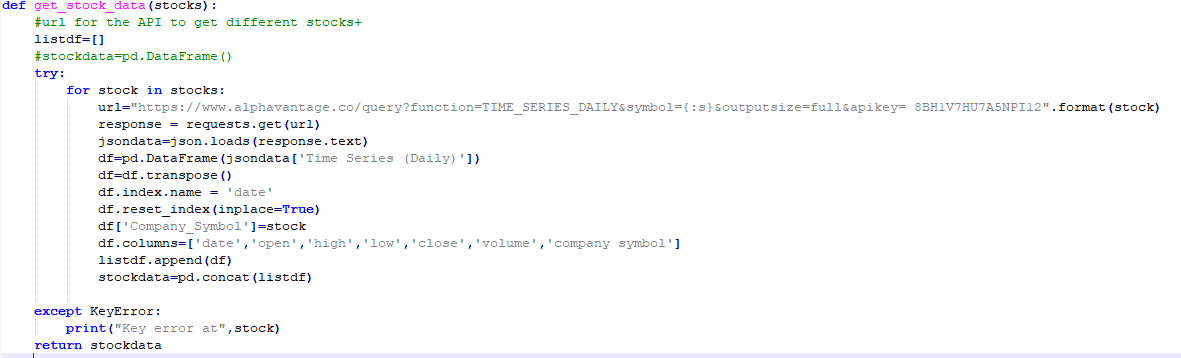
For the purpose of this project I collected the stock data for 50 largest companies by market-cap. For this I used another dataset that had the company symbol and its market-cap. I then sorted the data frame in the ascending order of the market-cap and filtered the top 50 values from it and then extracted the company symbol from it and stored it as a list. It looked something like this.



‘stocklist’ is the list of company symbols for the top 50 companies with largest market-cap.

Since the API query returns the data for only one company at a time, I wrote a function to make multiple API calls, get the data, process and store it to a data frame. The function takes a list (of company symbol) as the argument.

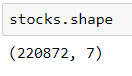
Here is the function:



I used string formatting to modify the API query and get the data for different companies. It makes multiples data frames for multiple companies and adds them to a list. The final step is to concatenate all in the data frames in a single data frame for further analysis. I had to run this function 10 times or sets of 5 companies each and merge the data frames later as the there is a restriction on the number of calls permitted in a time frame. I then saved the combined data frame as a csv file to disk using the pd.to\_csv () function. I then can access this file whenever necessary.



The dataframe has 220872 rows and 7 columns



***Columns:***

**Date: -** Date for the stock data price point

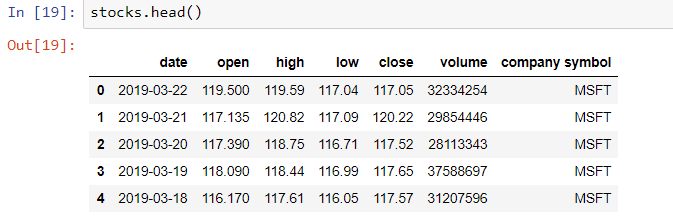
**Open: -** Opening price of the stock for the day

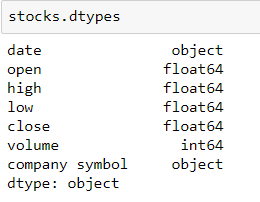
**High: -**Highest price point in the day

**Low: -** Lowest price point in the day

**Close: -** Closing price of the stock for the day (This is the most important column for our analysis)

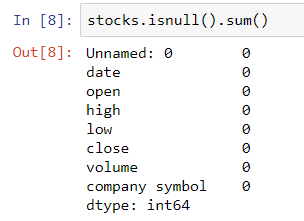
**Company Symbol: -** NASDAQ symbol for the company





All the variables are numeric except for date and company symbol. Date is later converted to datetime object using pd.to\_datetime () method in pandas.

I further checked for the missing values and found that there were none





**Analysis objectives and Questions: -**

* To calculate and plot various technical indicators
* Get two companies, one for which the average yearly traded Volume was continuously increasing and other decreasing
* What were some good buy and sell points from Tesla in the recent past using the indicators?
* To implement a trading strategy using the indicators
* What were some good buy and sell points from Tesla in the recent past using the trading strategy?
* perform sentiment analysis on the tweets and try to correlate it with the close price of the stock
* Is there any correlation between twitter sentiment and price movement for Tesla?

**Technical Indicators: -**

Technical indicators are a **fundamental** part of technical analysis and are typically plotted as a chart pattern to try to predict the **market trend.** They use historical data like close and volume and apply mathematical calculations to analyze market trend. Technical indicators covered in this project were Simple Moving Average, Exponential Moving Average, Moving Average Convergence Divergence (MACD) and Relative Strength Index (RSI). These will be discussed later.

**Average Trading Volume**: Although, average volume is not a technical indicator, I was curious to look at the average traded volume for the last 5 years for different companies. For this, I extracted the year from the date column and stored it as another column.

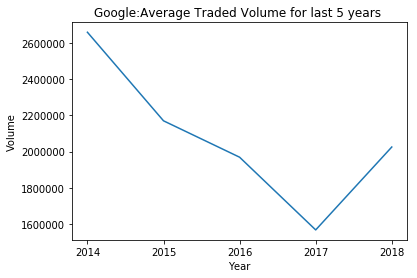


Then I used pandas groupby () to group the data by company and year and took the mean of the volume

grouped=stocks.groupby(['year','company symbol'])

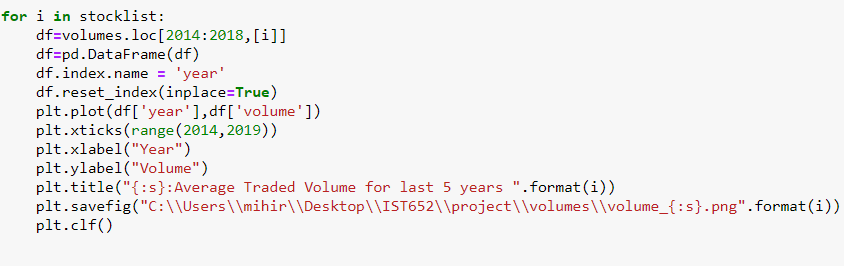
volumes=grouped['volume'].mean()

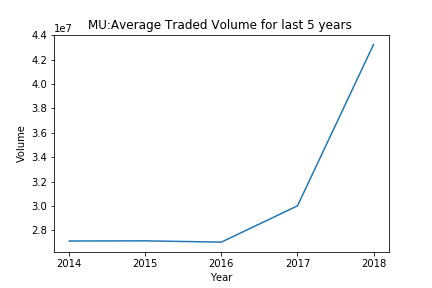
I then filtered the output to get the data for the years 2014 to 2018. Finally I used matplotlib to plot the result.



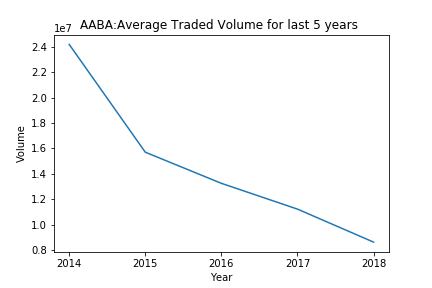
The yearly average trading volume for Google came down till 2017 and then has started going up.

I ran a for loop to get these plots for all the companies and save to disk.





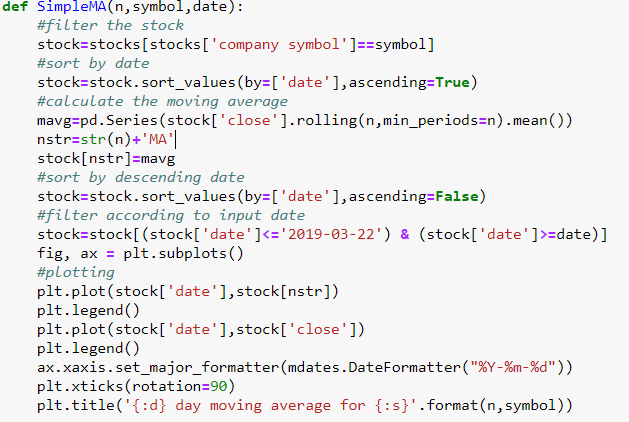
The chart for Micron Technologies (MU) is an interesting one and we can see that the average traded volume has risen incrementally in these years. This means that the trader interest in the stock has picked up over the years.



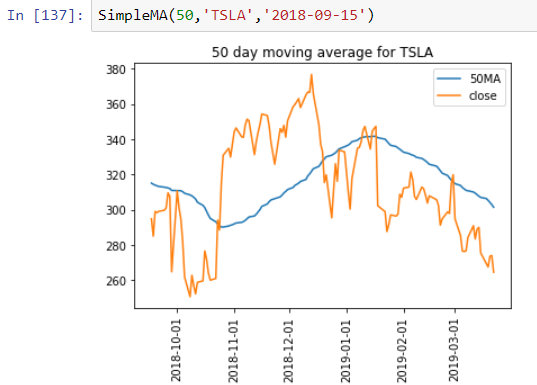
The Yearly average traded Volume is continuously decreasing for Albata Inc. from 2014 to 2018. This means that the trader interest in this company’s stock has continuously decreased over the past 5 years.

Simple Moving Average:

A simple moving average (SMA) is an arithmetic [moving average](https://www.investopedia.com/terms/m/movingaverage.asp) calculated by adding recent closing prices and then dividing that by the number of time periods in the calculation average. For instance, a 50 day Moving average is calculated by adding the close price for the last 50 days and dividing it by 50 and doing this for all the data points. I have written a function that takes in the time-period for calculation, company name and date from which the calculations are to be done, calculates the Simple moving average for those specifications and plots it using matplotlib.



Here is how I used the function to get 50-day moving average for Tesla from the date 09-15-2018. I get the following output. This function can be used similarly to get the moving average for any company for any period.



A good buy point for Tesla using the Simple Moving Average is around October 2018 when the stock price for Tesla crossed the 50 day Moving average and a good sell point for Tesla is around end of December 2018 when the close price of Tesla was trading below the 50 day moving average.

Exponential Moving Average:

The [exponential moving average](https://www.investopedia.com/terms/e/ema.asp) (EMA) is a [weighted moving average](https://www.investopedia.com/terms/l/linearlyweightedmovingaverage.asp) (WMA) that gives more weighting, or importance, to recent price data than the [simple moving average](https://www.investopedia.com/terms/s/sma.asp) (SMA) does.

The implementation of EMA is very similar to that of SMA except for the calculation of the moving average where the ewm method is used instead of the rolling method .

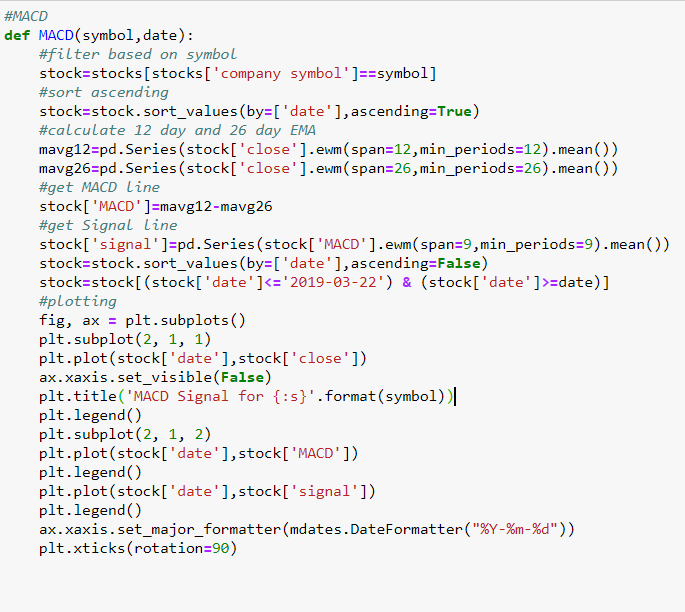


Exponential Moving average will generate similar buy and sell calls for Tesla but with more Refinement.

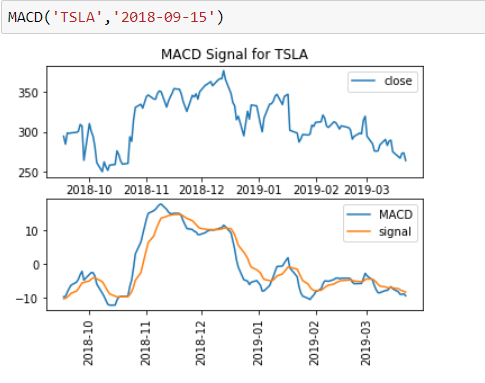
Moving Average Convergence Divergence (MACD)

Moving average convergence divergence is a momentum indicator that shows relationship between two moving averages of a stocks price. The MACD is calculated by subtracting the 26-period Exponential Moving Average (EMA) from the 12-period 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

I have written a function that does all these calculations and plots the MACD and signal line for any stock from the fifty stocks.



Here is how the function can be used to get the MACD for any stock

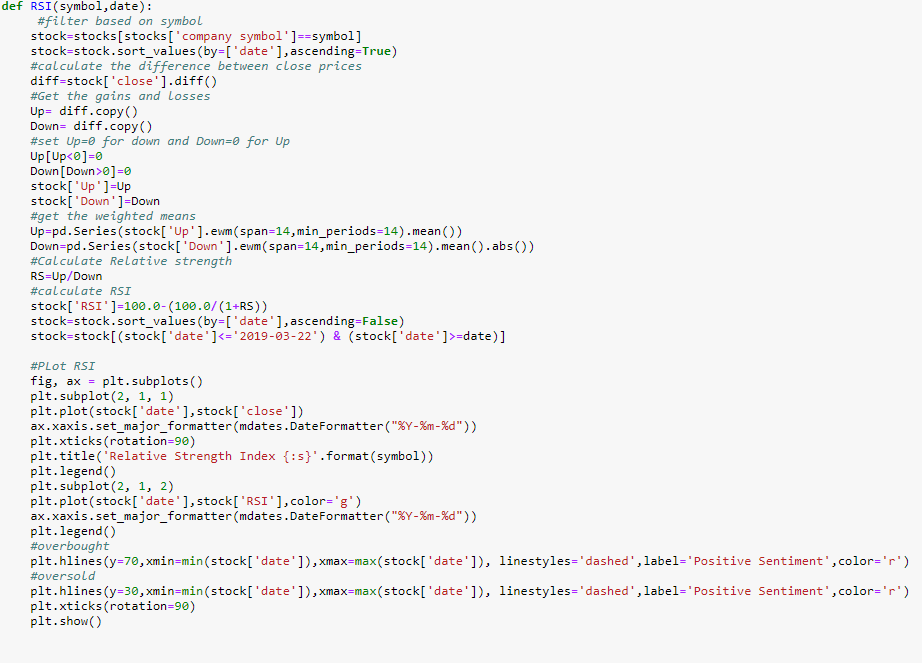


As visible from the plot the MACD line(blue) crossed the signal line(orange) around October-2018 generating a buy signal. The price of the stock went up from there. A good Selling point will be around end-December 2018 when the MACD line crosses the signal line from Above. This validates our findings with the moving averages.

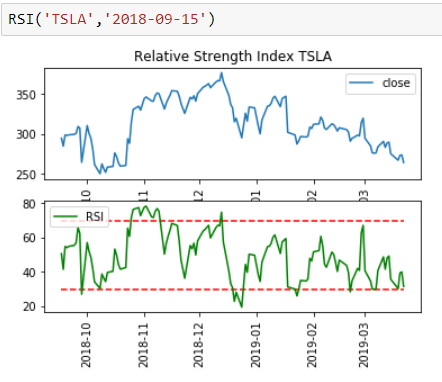
Relative Strength Index

The relative strength index (RSI) is a momentum indicator that measures the magnitude of recent price changes to evaluate overbought or oversold conditions in the price of a stock or other asset. The RSI is displayed as an oscillator (a line graph that moves between two extremes) and can have a reading from 0 to 100. Traditional interpretation and usage of the RSI is that values of 70 or above indicate that a security is becoming overbought or overvalued and may be primed for a trend reversal or corrective pullback in price. An RSI reading of 30 or below indicates an oversold or undervalued condition. It is calculated by deriving average gains and losses for the last 14 days and applying some mathematical calculations on it.

I implemented a function to calculate RSI and plot it for any stock



Here is how it can be used: -

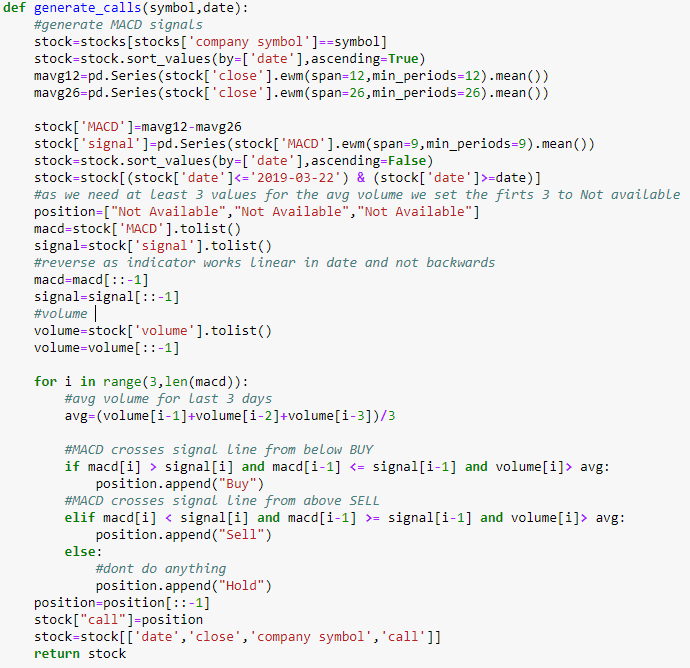


The values above and below the two red lines indicate overbought and oversold levels. This information can be used to make decisions for entering the market. According to RSI, Good points to enter or exit the market for Tesla are in November 2018 and end-December 2018 when the RSI goes above and below the threshold levels of 30 and 70.

Trading Strategy

I implemented a trading strategy to generate buy and sell signals. This strategy uses a combination of volume and MACD to generate the signals. If the daily volume is greater than average volume for last 3 days and the MACD line crosses the signal line from below, it generates a buy signal. If the daily volume is greater than average volume for last 3 days and the MACD line crosses the signal line from above, it generates a sell signal. If none of these the above condition is met generate a hold signal.

Here is the function:-



This is output form the function.



The strategy generates 3 calls from 19th Feb to 22nd March and the prices after the signals have validated the strategy.

|  |  |  |  |
| --- | --- | --- | --- |
| date | close | company symbol | call |
| 3/15/2019 | 275.43 | TSLA | Sell |
| 3/4/2019 | 285.36 | TSLA | Sell |
| 2/27/2019 | 314.74 | TSLA | Buy |
| 2/21/2019 | 291.23 | TSLA | Sell |
| 1/18/2019 | 302.26 | TSLA | Sell |
| 12/17/2018 | 348.42 | TSLA | Sell |
| 11/20/2018 | 347.49 | TSLA | Sell |
| 10/5/2018 | 261.95 | TSLA | Sell |
| 8/17/2018 | 305.5 | TSLA | Sell |
| 8/2/2018 | 349.54 | TSLA | Buy |

These are the buy and sell calls generated by the strategy from August 2018 to March 2019 for Tesla. A total of 10 trading calls were generated.

Price movement for Tesla after buy and sell calls for one instance each: -

Price movement after sell call on 17th December

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Price | Company | Call |
| 12/24/2018 | 295.39 | TSLA | Hold |
| 12/21/2018 | 319.77 | TSLA | Hold |
| 12/20/2018 | 315.38 | TSLA | Hold |
| 12/19/2018 | 332.97 | TSLA | Hold |
| 12/18/2018 | 337.03 | TSLA | Hold |
| 12/17/2018 | 348.42 | TSLA | Sell |

As Evident from above, the price dropped around 50 USD in a week after the strategy predicted a Sell call.

Price Movement after Buy call on 2nd August

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Price | | Company | | Call |
| 8/7/2018 | | 379.57 | | TSLA | Hold |
| 8/6/2018 | | 341.99 | | TSLA | Hold |
| 8/3/2018 | | 348.17 | | TSLA | Hold |
| 8/2/2018 | | 349.54 | | TSLA | Buy |

As evident from above, the price rose by around 30 Dollars in 5 days after the strategy generated a buy call.

Twitter Sentiment Analysis:

I used twitter API and vader sentiment to analyze the sentiment for stocks and map it to the respective price movement for the stock. I wrote a bunch of helper functions to assist in pulling the tweets, tokenizing, removing stop words, getting the vader sentiment score and finally plotting it.

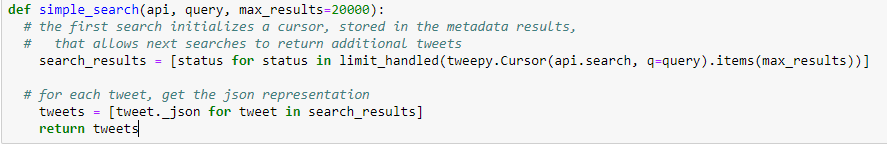
VADER SENTIMENT:

VADER Sentiment Analysis. VADER (Valence Aware Dictionary and sentiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media and works well on texts from other domains.

The compound score is computed by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized to be between -1 (most extreme negative) and +1 (most extreme positive). This is the most useful metric if you want a single unidimensional measure of sentiment for a given sentence. Calling it a 'normalized, weighted composite score' is accurate.

1. **positive sentiment**: compound score >= 0.05
2. **neutral sentiment**: (compound score > -0.05) and (compound score < 0.05)
3. **negative sentiment**: compound score <= -0.05

Step 1 : Pulling the tweets



This will take the api object, the query and max\_results as inputs and returns a list of dictionaries containing the tweet and associated information.

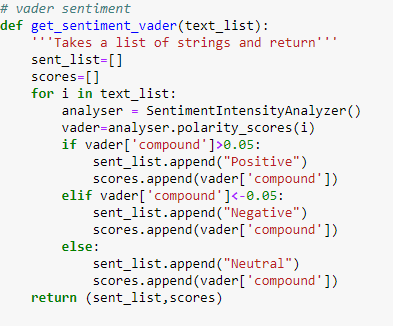
Step 2: Text processing

I used the nltk tweet tokenizer for tokenizing, removed stopwords from the nltk stopwords corpus, applied the alpha filter (from the class) to remove special symbols. The function takes in the list of dict of tweets and returns a processed list of strings, where each string is a tweet.

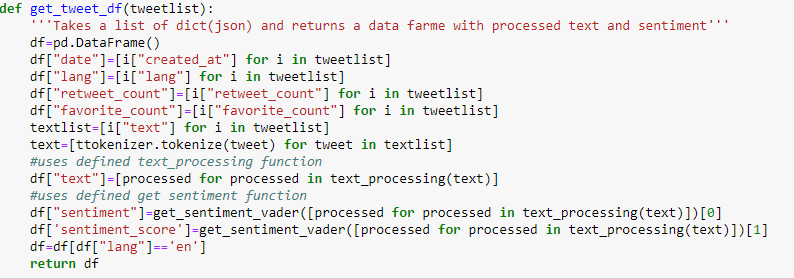


Step 3: Sentiment analysis

Getting the sentiment score from the Vader sentiment analyzer. I used the compound score as with the thresholds mentioned above. I wrote the helper function to get the sentiment. It takes a list of strings and returns a tuple of lists with sentiment and sentiment scores.



This function uses the helper functions mentioned above, takes a list of tweets(json) and returns a pandas with the sentiment, sentiment score , text, date etc.



Step 4: Get latest data for a stock

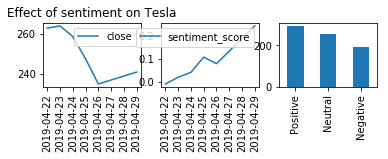
I wrote the helper function to get the latest data for a stock. It takes the symbol of the company and returns a data frame with the latest values.



Step 5: Get the plots



This function takes the name of the stock as input and returns the sentiment plot as output along with the close price. It basically uses the get\_stock() and get\_tweet\_df() functions to get the stock price and sentiment data frames, merges them on date, groups them on date and gets the daily average sentiment score.



There seems to be no direct correlation(positive) between the price movement and twitter sentiment for Tesla. Although this might be because of the number of tweets that are analyzed which are low as the free version of twitter API gives access to a limited number of tweets.

**Note: - Running these functions will give a different output every time as we are dealing with API’s**

Conclusion and Future scope

All the technical indicators were successfully implemented, the trading strategy was also successfully implemented so was the sentiment analysis part. Historical Buy and Sell points were obtained for Tesla. In future, price forecasting can be achieved using time series modelling to decide exit points based on probable future prices. The trading strategy can also be refined, and alternative strategies can be developed. Sentiment analysis can be used as a feature along with historical data to make a predictive model to determine price movement. Further, this script can be scheduled to run daily, and emails can be sent out using the SMTP protocol based on the Buy/Sell calls for companies.