



# STOCK RECOMMENDATION SYSTEM

Jhalaa Chinoy Tejasvi, Jing Wang, Kit Yup Yip & Qi Zhang

## a. ABSTRACT

This paper discusses the building of a stock recommendation system using data mining techniques. The recommendation is implemented by analyzing the associations between different stocks. If a stock symbol is provided as an input, the system will return a group of stocks which are similar to the original stock. This comparison is done on basis of three parameters - strong association rules, in the same group and similar price variation trend. Recommendations are also made by regression modeling. The system also contains a forecasting tool with built in seasonal modeling to reflect the future trends of a particular stock symbol.

Our results show that associative modeling techniques along with regression and forecast modeling can be used to give stock recommendations. The system could find a group of related stocks based on user's input and different prediction models. Additionally, the system could return top 5 stocks with promising trend for people to choose if the input is empty. These results will provide useful guidelines to develop a more advanced version of recommendation system that a user can input a combination of stocks (may permit to set weight for each stock) and the system will return the results using some sort of windowing effect that is set by users to manage risk of each investment.

## b. INTRODUCTION

### [I] Statement

Our system provides an easy and understandable solutions to users. This helps users who are new to the stock market and helps them make tangible decisions on buying and selling stock to increase their wealth.

Our system filters stock market data and financial web articles to identify healthy businesses. For the most part businesses tend to be seasonal; meaning that holidays, quarterly earnings reports, Q4 sales, etc. tend to affect stock price. Some businesses are also extremely popular and get a lot of attention in the media; this added attention may make stock prices more volatile. There are also associative effects that impact stock prices of groups of businesses, i.e. tech stock like FANG (Facebook Apple Netflix Google). Lastly, there are unforeseen

events such as political changes, natural disasters, new technologies, etc. that affect stock prices. The problem we are trying to solve is how to filter out things like seasonality, volatility, and unforeseen events out of stock market data and financial articles to help investors who may be new to the stock market identify healthy businesses or groups of business. Our system uses co-occurrence from various news articles to determine which stocks get mentioned together. It then generates association rules using apriori to build a map of rules which can be used at a later point in time. The system uses the above map to give a set of 5 similar stocks based on a given stock.

The system also enables the user to use clustering to determine the similar stocks based on the features the users are interested in. The user can choose features from the list Price, Volume, Market Cap, Beta, PE Ratio and EPS.

The system also provides the stock trends by using linear regression and compares different stocks on basis of this and suggests 5 top stocks to invest in. This is a quick fix for users who don't want to do much research and just get a list of stocks to invest in. We also provide a way of forecasting and providing future predictions to the users.

Identifying healthy businesses or groups of businesses in a way is similar to trying to predict how a business will do in the future. The main goal of our system is to identify healthy publicly traded businesses (or associated businesses); a secondary goal is to give users enough information needed to invest and have a method to mitigate risk.

## **[II] Importance of this problem**

There is so much information constantly being generated, keeping up with all of it is overwhelming for anybody. The daily news, economic reports, technology journals, it's all constantly being uploaded to the web. We all have ways of filtering our daily news feeds from visiting our favorite news sites to following different people on Instagram.

Similarly, it is very important that new investors or people who are considering investing for the first time have a method for filtering and sorting through financial information quickly. While the stock market is a really exciting place to be in, it is also very intimidating. Our stock recommendation system will help such users understand the stock market and provide them with a helping hand to make such decisions.

Developing a method to sort through tons of financial data to try and get insight into the overall health of a business is beneficial not only for individual investors but also economists, market analysts, politicians, scholars, students, and

everyday people. Insight into the state of our economy helps everyone make better choices about

their purchases and may even influence daily spending habits, as well as identifying investment opportunities.

## **[III] Background Information**

Predicting stock market performance generally falls into three categories Fundamental Analysis, Technical Analysis, and Machine Learning. Fundamental Analysis is a very involved process; the analyst must know a lot about the particular business they are interested in investing in. A fundamental analyst would analyze a business's profit margins, earnings, losses, direct competitors, production capabilities, even the management and employees. On the other hand, Technical Analysis differs by predicting stock performance by mostly analyzing past market performance data; this method is also involved but it does not rely on understanding the fundamentals of a business. The latest methods for predicting stock prices are in the form of Machine Learning; from simple algorithms to Artificial Neural Networks[10].

The stock market, is very complex and convoluted for a beginner. Political events, market news, quarterly earnings reports, international influence and conflicting trading behaviour, all have a direct impact on how it performs. [7] A lot of studies have shown that predicting stock market returns is a difficult task .[8] A stock is a type of security, which represents ownership in a company, its assets, earnings and dividends. It is an indicator of how the company is performing and what its growth/profits are. Stocks are traded in stock market, where the prices are controlled by traders' bids (buy price) and offers (sell price). Stocks should be recommended to the investor based on his

interests, preferences and trading behaviour. [6] We plan to make this

task less time consuming and easily understandable to the user and provide solutions to the same.

### **c. Methodology**

In order to identify healthy businesses and business groups we will split the problem in three:

1. Find associative characteristics between stocks.
2. Build simple regression models on stocks with coefficient correlations.
3. Use a forecasting tool that models seasonal trends.

With a clear ideal of how stocks are related to each other, stock variation trends, and forecasting methods we will be able to give stock recommendations to investors. Let's discuss each method in more detail.

#### **[I] Find associative characteristics**

We used Yahoo Finance, Investopedia etc to filter stock symbols in article content. we retrieved a list of stocks and also calculate their co-occurrence in the articles. This type of data can be used to generate associations rules.

We also downloaded stock fundamental data like stock category, price, P/E, Volume, market capitalization and EPS et al. Based on these data, we built a vector model of stock features and use clustering techniques (i.g. k-means) to group them.

#### **[II] Regression model(s) on each stock**

We build a simple linear regression model for each symbol and calculate the coefficients representing trends. As what we do in class, we use the linear model.LinearRegression function from scikit-learn library. To run the model, first divide the historical stock price data into training and testing sets, then fit the model on the training set and predict with the fitted model on the testing set.

For fetching historical data from yahoo finance, package pandas\_datareader is used. Since the linear regression model only applies for short time analysis, we only chose the date range from 1/1/2018-4/1/2019 for one year to get the latest trends.

Using popular symbol list filtered from article content, we record coefficient value for every stock. Finally, we could output the top n stocks with best trends to recommend user and also the similar n stocks for a specific input stock symbol.

#### **[III] Forecasting tool that models seasonal trends**

We used historical data from yahoo's finance api's to predict the stock price for a given stock symbol, given other stock symbols. In addition to the normal pandas and numpy libraries, We are also using Prophet, which is a forecasting library released by the Facebook core data science team. It implements a procedure for forecasting trends are fit with yearly, weekly, and daily seasonality, plus holiday effects [11],[12]. This gives us the future trend predictions for the stock provided as input.

In summary, based on the three types of data obtained from the methods above, we will be able to predict specific results out of data. Also, we will build a comprehensive recommendation system out of this data to investors.

### **d.Code**

All our code is in the below github link

<https://github.com/stockrecommendationsystem/python>

First we get all the news links from Investopedia sitemap. On the site, Only sitemap\_1.xml is available. It includes articles in the past 3 months.

```
import requests
import time
import re
import random
from bs4 import BeautifulSoup

class NewsLinkCrawler:
    # Crawler of page webpage
    def __init__(self, url):
        self.rooturl = url
        self.pageurls = []

    # Get the urls of all news articles
    def run(self):
        html = requests.get(self.rooturl)

        soup = BeautifulSoup(html.text) #Parse webpage by using BeautifulSoup
        links = soup.select("urlset > url > loc")
        for link in links:
            url = link.get_text() #Get the text of each tag
            if '/news/' in url:
                self.pageurls.append(url)

        f = open('../data/all_news.txt', 'w')
        f.write('\n'.join(self.pageurls))
        f.close()

def main():
    url = "https://www.investopedia.com/sitemap_1.xml" #Initial url for getting all page tags
    crawler = NewsLinkCrawler(url)
    crawler.run()
    main()
```

After running the above code, we obtained 9,937 news articles. In each article, stock symbols of different companies are included. For example, in <https://www.investopedia.com/news/twitter-now-using-ai-recommend-tweets-twtr/>, TWTR, FB, GOOG and MSFT are the stock symbols of Twitter, Facebook, Google and Microsoft.

Next, we retrieve stocks in each news article, see the codes below (just cut-off screenshot). Here we still use BeautifulSoup library to parse stock symbols from article content. In the article content, each symbol is linked to stock page. We write regular expression to retrieve it based on this feature.

```
import requests
import time
import re
import random
from bs4 import BeautifulSoup
from pathlib import Path

class OccurrenceRetriever:
    # Crawler of page webpage
    def __init__(self, file, start, end):
        self.pageurls = []
        self.occurrences = []
        self.file = file
        self.start = start
        self.end = end

    # Parse stock symbols from news article
    def StockParser(self, url):
        stock_list = []
        html = requests.get(url) #Get the content of page webpage by url
        bsoup = BeautifulSoup(html.text.encode("utf-8"), "lxml")
        body_soup = bsoup.find('div', {'class': 'article-body'})
        #print(body_soup)
        for stock in body_soup.findAll('a', {'href': re.compile('https://www.investopedia.com/markets/stocks/')}):
            matchObj = re.search(r'markets/stocks/(.*)', stock.get('href'), re.M|re.I)
            if matchObj:
                stock_list.append(matchObj.group(1))
        return stock_list

    # Start to retrieve
    def run(self):
```

After running the code above, we can get stock symbols in each article like this:

```
etfc
fb
nflx
mu      amat      lrcx
msft     symc      flt      de      lyb

googl    fb
pm        bti      bti
twtr

amzn
aapl
twtr      fb      goog      msft
unh
```

Each line represents a list of stocks in an article. For example, the 13th line “twtr, fb, goog, msft” indicates that Twitter, Facebook, Google and Microsoft are mentioned in the same article.

Next, we build a list of unique stocks based on the table above and then retrieve financial fundamentals of these stocks from Yahoo Finance. The codes (just cut-off screenshot) are shown as below:

```
import requests
from bs4 import BeautifulSoup
from pathlib import Path
import csv
import time

class StockBasicsCrawler:
    # Crawler of page webpage
    def __init__(self, url):
        self.rooturl = url

    # Get the urls of all news articles
    def run(self):
        file = "../data/unique-stocks.txt"
        stocks = Path(file).read_text().split("\n")
        with open('../data/stock_basics.csv', mode='w') as basics_file:
            basics_writer = csv.writer(basics_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL)
            header = ["Stock", "Price", "Volume", "Market Cap", "Beta", "PE Ratio", "EPS"]
            basics_writer.writerow(header)
            count = 0
            for stock in stocks:
                html = requests.get(self.rooturl + stock)
                soup = BeautifulSoup(html.text, features="lxml") #Parse webpage by using BeautifulSoup
                pricetd = soup.select('td[data-test="PREV_CLOSE-value"]')
                price = pricetd[0].get_text().replace(", ", "") if pricetd else "na"
                volumetd = soup.select('td[data-test="TD_VOLUME-value"]')
                volume = volumetd[0].get_text().replace(", ", "") if volumetd else "na"
                mktcaptd = soup.select('td[data-test="MARKET_CAP-value"]')
                mktcap = mktcaptd[0].get_text().replace(", ", "") if mktcaptd else "na"
```

After running the codes, we can get a list of 1530 unique stocks and related financial fundamentals.

## [I]Association Rules

According to the data we generate in the previous step, we use apriori algorithm to retrieve association rules [9]. The codes of main function is listed below:

```
association_rules = apriori(records,
min_support=0.002, min_confidence=0.1,
min_lift=2, min_length=2)
```

## [II]K-means Clustering

Here we use KMeans function imported from sklearn.cluster to classify different stocks. The core codes of predicting the cluster each stock belongs to is listed below:

```
kmeans =
KMeans(n_clusters=clusterNum,init='k-means++',m
ax_iter=300,n_init=10,random_state=0)
clusterIds = kmeans.fit_predict(features)
```

Different users are concerned about different stock features. In order to resolve this issue and make our recommendation more flexible, here we allow users to specify stock features they are interested in. They can select one or multiple features from 6 as shown below.

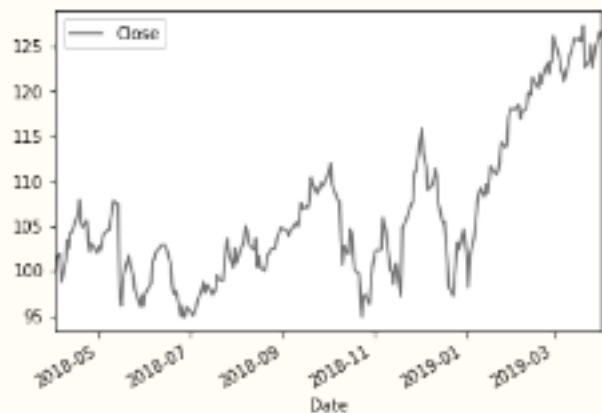
```
# Available feature ID and name:
# 1: "Price",
# 2: "Volume",
# 3: "Market Cap",
# 4: "Beta",
# 5: "PE Ratio"
# 6: "EPS"
```

## [III]Linear Regression Model

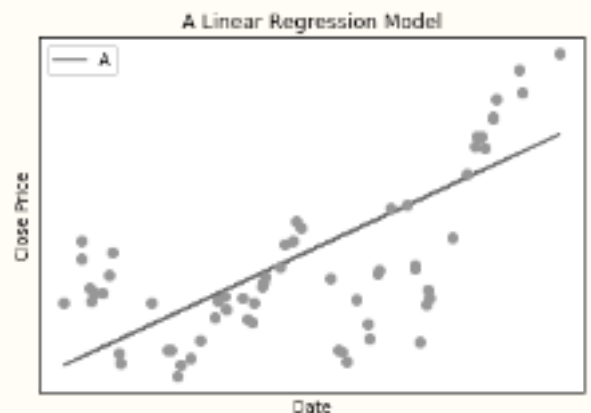
According to the data we generate in the previous step, we can build linear regression Model for each stock symbol. We build the linear regression models of stocks to get the coefficients representing trends (class method). For the consistency of the whole project, read the stock list of all the symbols we needed in the text file. The stock\_list.txt is generated from the first part of the project. Then historical data for a specific stock, using symbol as the parameter is retrieved. For fetching historical data from yahoo finance, an additional package pandas\_datareader are used here, which can be

installed with pip from the command line: pip install pandas-datareader. We can specify the stock symbol, start date, end date and also the data source. There are two exceptions defined for handling error

while fetching data using datareader: error RemoteDataError(when the symbol is not valid and can't be find in the dataset) and error KeyError(when there isn't full data for the input data range(start, end)).



We then build a linear regression model for a specific stock. There are three parameters. Show\_statistics is to indicate if the statistics are printed. Show\_dots is to indicate if dots data needed. Show\_plot is to indicate if the plot is shown.

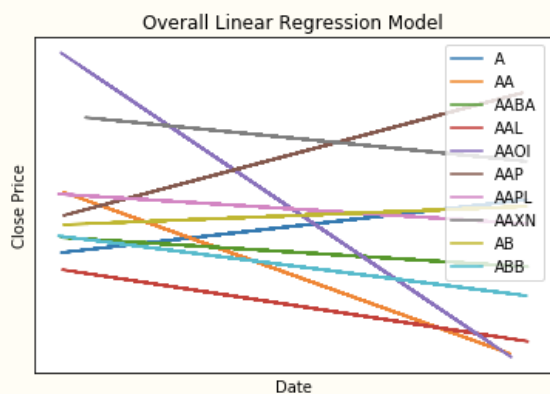


```
Coefficients: [5.68569057]
Mean squared error: 33.39
Variance score: 0.57
features: Date Close_Price
5.685690572599488
```

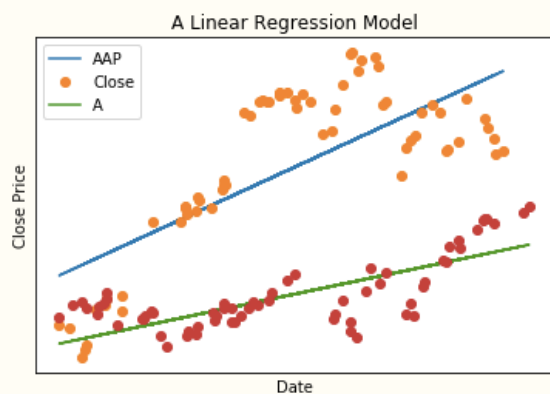


We then calculate the coefficients and record for each symbol in a new text file. The linear regression model is evaluated by mean squared error (the model error) and variance score(the degree to which the model explains the variation present in the data).

These scores are used to measure the efficacy of a particular linear model. 1 variance score means perfectly prediction, so we could use these values to analyze the performance of the model. The visual figure shows how the data points distributed.



We then get top n stocks with biggest coefficients  
You can set any n to get top n stocks from the result coefficient dataset



	Stock	Coefficient
6	AAP	13.3734
0	A	5.68863

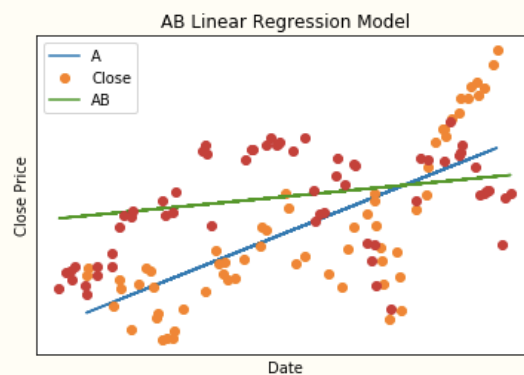
We can also get stocks with same trends

You can set any n offset to get  $n*2-1$  similar stocks from the result coefficient dataset. The  $n=3$  will return 5 similar stocks, 1 will return 1 similar stocks.

	Stock	Coefficient
0	A	5.34906
1	AA	-17.9774
2	AABA	-3.54877
3	AAL	-7.79476
4	AAOI	-33.4569
5	ANZ	NaN
6	AAP	13.2222
7	AAPL	-3.0748
8	AAXN	-5.00924
9	AB	1.72391
10	ABB	-6.34512

Example with coefficient\_data\_test, using symbol 'AAOI'

get\_similar\_stock(coefficient\_data\_test, 'AB', n = 1, show\_dots = True)



#### [IV] Forecasting tool to model seasonal trends

We are using historical data from yahoo's finance api's to predict the stock price for a given stock

symbol, given other stock symbols. In addition to the normal pandas and numpy libraries, We are also using Prophet, which is a forecasting library released by the Facebook core data science team. It implements a procedure for forecasting trends are fit

with yearly, weekly, and daily seasonality, plus holiday effects. We load the data and specify the start and end dates and also the stock symbol. Using this, we can get an array of closing values.

We can use this to forecast the values.

### e. Results

As described in previous sections, each of the three methods for stock recommendation system produces a list of recommended stocks. They are based on associative rules, clusters and linear regression respectively. The related results are described in Part 1 and 2. In addition, our system would predict how each recommended stock may perform in the future, which is introduced in Part 3.

#### 1. Find associative characteristics

After running the codes, 175 association rules are retrieved between stocks. The results (only cutoff screenshot) are shown as below.

```
Rule: aaba -> vz
Support: 0.002735229759299781
Confidence: 0.5263157894736842
Lift: 31.54443485763589
=====
Rule: aal -> dal
Support: 0.0030087527352297594
Confidence: 0.5945945945945946
Lift: 94.51468860164512
=====
Rule: luv -> aal
Support: 0.002324945295404814
Confidence: 0.4594594594594595
Lift: 101.8050778050778
=====
Rule: ual -> aal
Support: 0.0030087527352297594
Confidence: 0.5945945945945946
Lift: 114.41251778093884
```

Here we attempt to conduct a deep analysis by taking the first rule as an example. The rule indicates that “vz” is likely to exist if “aaba” exist.

Association rules are created by searching data for frequent if-then patterns. The criteria support and

confidence are usually used to identify the most important relationships. This rule has the support of 0.00027 and confidence of 0.52. Confidence of 0.52 indicates if “aaba” appears, then there is a possibility (52%) that “vz” appears. Support indicates of how frequently “vz” and “aaba” appear in the dataset. From its value of 0.00027, their occurrence frequency is very low. This may result from the fact that there are a lot of stocks in the market and not all the stocks have high frequency of

media reports. Some hot stocks like AAPL et al. have much more media reports than others. “Aaba corresponds to Altaba Inc. and verizon company. The strong association between these two companies exists probably because of Yahoo! Inc. As we know, Altaba Inc was formed from the remains of Yahoo! Inc after Verizon acquired Yahoo's Internet business.

Similarly, we can analyze other 174 rules obtained. By lowering the values of the parameters (min\_support and min\_confidence) in apriori model, we can obtain more association rules to meet our recommendation needs.

Based on generated association rules above, we can make recommendation easily. When user inputs a stock symbol, the system could search the “from” stock in the rules and retrieve “to” stocks and present them. Here we take Google as an example. Below list 5 companies recommended based on association rules. They are Apple, Amazon, Salesforce, Cisco and Facebook respectively. This result makes sense because these companies are from the same industry and in similar company size (top tech giants) with Google. Therefore, it is

expected that they are often mentioned together in media reports.

```
Stock you have input: googl
-----
Here are a list of stocks you may be interested in:

Stock-1: aapl
Support: 0.004102844638949671
Confidence: 0.5882352941176471
Lift: 35.255544840887175
=====
Stock-2: amzn
Support: 0.004102844638949671
Confidence: 0.5882352941176471
Lift: 35.255544840887175
=====
Stock-3: crm
Support: 0.004102844638949671
Confidence: 0.5882352941176471
Lift: 35.255544840887175
=====
Stock-4: csc
Support: 0.004102844638949671
Confidence: 0.5882352941176471
Lift: 35.255544840887175
=====
Stock-5: fb
Support: 0.004102844638949671
Confidence: 0.5882352941176471
Lift: 35.255544840887175
=====
```

Furthermore, the associations of stocks can be obtained by generating stock clusters. After running k-means clustering codes, we can get a group of clusters (only cutoff screenshot) as below. To clarify, the results listed here is based on the combination of “price” and “volume”. From the results, we can see that the sizes of clusters are diverse. A few clusters have 50+ stocks. In contrast, 18 of 100 clusters have only one stock. This is understandable because the stock market is highly dynamic, especially in some features like stock price and volume. There is no surprise to see some outliers.

```
Here are generated clusters:

cluster-1: avgo,bhp,bks,cern,cs,edu,etn,expe,i
no,leg,maxr,nee,now,okta,paas,payx,pnc,rng,ros
t,sfix,sgen,slca,trco,tsg,ua,unm,wb,zixi
cluster-2: nbev
cluster-3: intc,symc,vz
cluster-4: ge
cluster-5: bac,mbfi
cluster-6: aeo,arnc,cnp,jnj,nlsn,pm,su,uaa,wdc
cluster-7: agro,alrm,amba,bco,bdc,bgne,blcm,ca
sy,cdxs,ceva,cg,cib,conn,drys,egrx,env,eqix,eq
m,eth,flws,gel,gmre,hei,hli,kw,lad,lgih,lgnd,m
ack,med,mpwr,ocn,pcty,reta,rvnc,sbny,site,snx,
tcon,tdy,uclt,vsto
cluster-8: eca,mu,roku
cluster-9: spy
cluster-10: dal,jcp,jpm
```

Similarly, we can make recommendation based on the cluster each stock belongs to. When user inputs a stock symbol, the system could check the clusters one by one to find which cluster includes target stock, then it returns other stocks in the same cluster. Below list 5 companies which are in the same cluster with Amazon.

```
Stock you have input: amzn
Features you are concerned about: Price, Volume
-----
Here are a list of stocks you may be interested in:

Stock-1: agnc
=====
Stock-2: amrn
=====
Stock-3: bidu
=====
Stock-4: cbs
=====
Stock-5: disca
=====
```

When different stock features are selected, the recommendation results are changed. For example, when selected stock features are Beta and PE Ratio, the results are very different, which are shown as below.



```

Stock you have input: amzn
Features you are concerned about: Beta, PE Ratio
-----
Here are a list of stocks you may be interested in:

Stock-1: cci
=====
Stock-2: cybr
=====
Stock-3: keys
=====
Stock-4: mrcy
=====
Stock-5: nxp
=====

```

By comparison with the results of “price” and “volume”, we can see that there is no common stocks in two recommended lists. This indicates that users’ actual concerns are very important to data mining and they will affect the design of recommendation systems.

## 2. Build simple regression models on stocks with coefficient co-relations.

The Input is stock\_list.txt including a stock list. The stock\_list.txt is generated from the first part of the project, and it should be put in the same directory as this ipynb file. Users could customize a text file with all the symbols they want to analyze. Here we only analyze the stock list in the stock\_list.txt. Since linear regression model only applies short-time evaluation with relatively consistent trend, the date range for fetching stock historical data is set for only one year from 04/01/2018 to 04/01/2019. The code for setting is as following:

```

# Considering data from 2018 to 2019 for
consistency
start = datetime.datetime(2018, 4, 1)
end = datetime.datetime(2019, 4, 1)

```

### Result 1: Coefficient dataset and export file

Code:

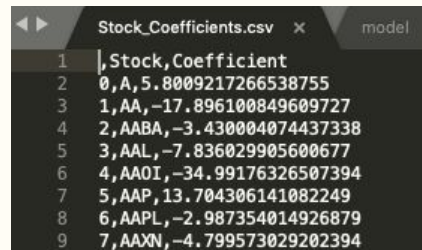
```

# Using function 4 get_coefficient_dataset
coefficient_data =
get_coefficient_dataset('stock_list.txt',
show_plot = False)

```

coefficient\_data.head()

Calculated coefficients for symbols in coefficient\_data for further analysis. The function get\_coefficient\_dataset also generated a text file Stock\_Coefficients.csv as record. Users could use the file freely. The file should be in the same folder as the ipython notebook.



	Stock	Coefficient
1	0,A	5.8009217266538755
2	1,AA	-17.896100849609727
3	2,AABA	-3.430004074437338
4	3,AAL	-7.836029905600677
5	4,AAOI	-34.99176326507394
6	5,AAP	13.704306141082249
7	6,AAPL	-2.987354014926879
8	7,AAVN	-4.799573029202394

The invalid input symbols from the input list are indicated by the error messages printed, which will

remind the user to change the symbols or date range to valid ones.

```

No search result : 'ANZ'
Date range not supported : 'ELNK'
Date range not supported : 'ESRX'
No search result : 'ETP'
Date range not supported : 'GIMO'
Date range not supported : 'HAR'
Date range not supported : 'LNKD'
Date range not supported : 'MFS'

```

The output stock\_coefficient table in the ipython notebook is as below, sorted by Stock name and only show the head of the table. The positive value means increasing trend and the negative value means decreasing trend. Bigger value means more quick change. AA is decreasing more quickly than the AABA and A is increasing. So referring the coefficient, the system would recommend “A” representing Agilent Technologies, Inc. in these 5 stocks.

Stock Coefficient

0	A	5.80092
1	AA	-17.8961
2	AABA	-3.43
3	AAL	-7.83603
4	AAOI	-34.9918

Result 2: Top five stocks from the stock list

Code:

```
# Get top five stocks from the stock list
```

```
get_top_stock(coefficient_data, n = 5)
```

Using the function 5 get\_top\_stock. We sorted the stock\_coefficient table by coefficient value by descending order and selected the first 5 stocks in the table. The number 5 is selected by passing parameter n as 5.

The output printed a result stock list for direct view.

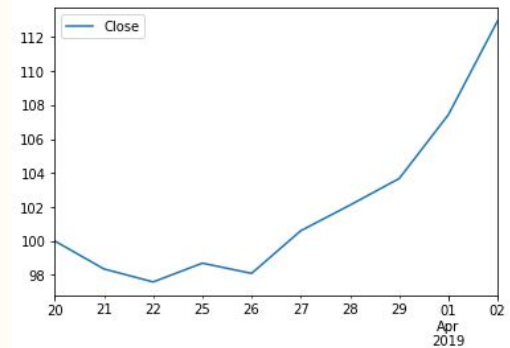
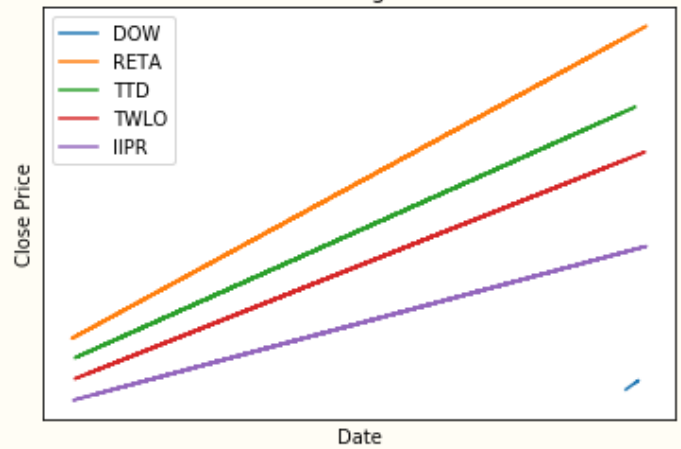
```
The 5 stocks with best trends are:
['DOW', 'RETA', 'TTD', 'TWLO', 'IIPR']
```

The output printed a table including the coefficient value for top 5 stocks.

	Stock	Coefficient
418	DOW	123.861
1133	RETA	81.598
1347	TTD	68.4666
1354	TWLO	60.4618
702	IIPR	43.0513

The output also printed a graph for visual view with all top 5 stocks. We can see they increase dramatically with big coefficient value. The Dow is short at right corner since its historical data is started only from 20/03/2019 to 02/04/2019. It's new but promising, the plot of dow is presented at right.

Overall Linear Regression Model



Result 3: Similar five stocks for input symbol

'AMZN'

Code:

```
# Get similar five stocks from the stock list, using
'AMZN' as example
```

```
# Change it to any input stock symbol
```

```
symbol_test = 'AMZN'
```

```
get_similar_stock(coefficient_data, symbol_test,
n = 3)
```

Using function 6 get\_similar\_stock, we get 5 similar stocks for the symbol 'AMZN' as amazon company. The same as in get\_top\_stock, it printed a literal stock list, a table with coefficient value and visual graph for comparison. Users could change the symbol to any stock they like.

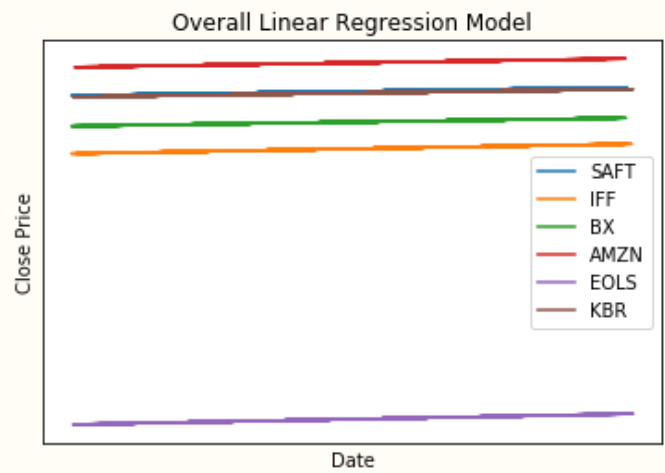
The printed stock list is as following, the five similar stocks for 'AMZN' are 'SAFT', 'IFF', 'BX', 'ELOS' and 'KBR'.

The 5 simialr stocks are:  
['SAFT', 'IFF', 'BX', 'EOLS', 'KBR']

The printed table is as following, we can see the coefficient values are very close and around 1:

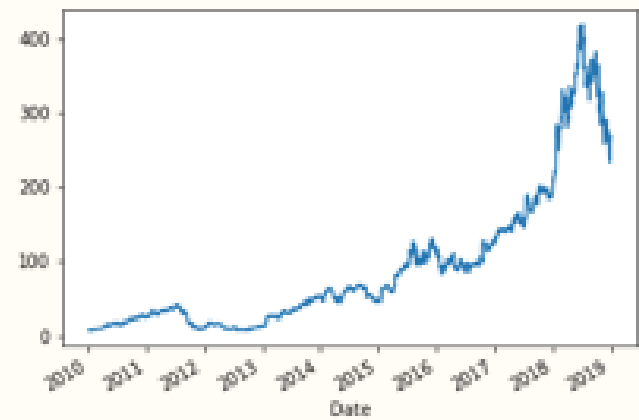
	Stock	Coefficient
1171	SAFT	1.04309
699	IFF	0.992022
218	BX	0.990293
86	AMZN	0.96194
470	EOLS	0.961843
766	KBR	0.95323

The printed plot with similar stocks for 'AMZN' and 'AMZN' itself. We can see the lines are parallel, which means they have very similar trends. The line 'KBR' and 'SAFT' is overlapped, so looks like there is only one line.



3. Use a forecasting tool that models seasonal trends.
- In order to make specific forecasting predictions that filter out seasonal trends we decided to

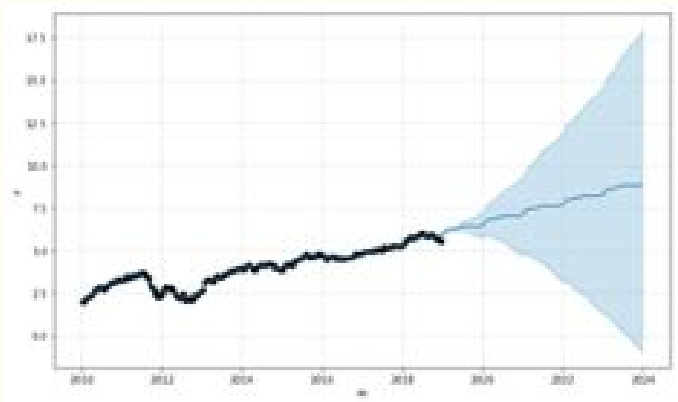
look at Netflix stock closing values as an example.



NFLX closing values (\$USD) from Jan 1, 2010 to Jan 1, 2019

The data is then normalized (take log of all closing values to normalize the skewed data) and we can create a Prophet model with seasonality and can make a forecast prediction which allows an investor to track overall behavior of stock investments and

decide if the model is no longer applicable if the value at any point is outside the confidence interval bounds.



NFLX normalized closing values (\$USD) from Jan 1, 2010 to Jan 1, 2019 with predictive modeling out to 5 yrs using Prophet

Once the Prophet model is created we determined that it was a fairly accurate model  $\hat{y}$  follows actual data closely with confidence also plotted on the subsequent slide.



Overall, we feel that these three methods do provide a method for investors to identify healthy stock options as well as groups of stock similar to a specific stock symbol as shown by each of the individual results in each category.

## f. Discussion

There were two basic philosophical approaches we took to help solve the issue of aiding investors in stock purchases. One was that past behavior of stock prices is an indicator of future behavior (data modeling) and the second was that the best way to predict tomorrow's stock value is to look at today's price as well as the price of similar stock (associative modeling). We feel that both of these do help guide investors.

The hard part of making predictions in such a volatile environment is that there are so many factors that affect stock price on a day to day basis. We tried to correct for some of the more obvious ones like seasonality and some amount of volatility and unforeseen events by taking both data modeling techniques along with associative modeling techniques.

Our results showed that it is possible to correlate different stocks based on factors like linear regression modeling coefficients as well as being

able to make models to forecast future performance, and this data can all be used to give guidance to potential investors. However, it's difficult to tell which stock to invest in.

## g. Future work

The conclusions that we can make at this point is that our system gives guidance, however we feel that a more complete solution would be the aggregation of these methods into a system that provides a single set of investment opportunities. The selection of possible stock investments is still heavily relied upon the user to make.

One possible approach, which was mentioned in passing previously, would be to select the top performing related stocks based on their positive coefficient results. Take the set of the high performing stocks and apply the forecasting Prophet model to each of them (and perhaps permit users to

set a different weight for each stock). Allow users to set an investment window in which if at any point the stocks perform outside of the lower (or upper) confidence bound, then cash out and re-run the analysis; since the modeling is not correct and needs updating. This would be sort of like a windowed self-correcting modeling system.

As a tool for aiding investors we feel that this is a tool that can be used as is, however more work is needed to give a more comprehensive approach to a new investor to the stock market.

---

## LITERATURE CITED

- [1] <https://www.kaggle.com>
- [2] <https://finance.yahoo.com>
- [3] <https://www.investopedia.com>
- [4] <https://github.com/WillKoehrsen/Data-Analysis/blob/master/stocker/Stock%20Prediction%20Usage.ipynb>
- [5] <https://www.pythonforfinance.net/2018/02/08/stock-clusters-using-k-means-algorithm-in-python/>
- [6] <http://ceur-ws.org/Vol-1606/paper02.pdf>
- [7] <https://www.sciencedirect.com/science/article/pii/S1062976917300443#bib0570>
- [8] <https://www.sciencedirect.com/science/article/pii/S1062976917300443#bib0540>
- [9] <https://github.com/NUOEL/cs6220>
- [10] [https://en.wikipedia.org/wiki/Stock\\_market\\_prediction](https://en.wikipedia.org/wiki/Stock_market_prediction)
- [11] [https://facebook.github.io/prophet/docs/quick\\_start.html](https://facebook.github.io/prophet/docs/quick_start.html)
- [12] [https://facebook.github.io/prophet/docs/quick\\_start.html](https://facebook.github.io/prophet/docs/quick_start.html)