“Stock Market Prediction Using Machine Learning”

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

By

Yashraj Mishra (151287)

Under the supervision of

Dr. Ekta Gandotra

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled "Framework on Automated Trade Systems using Time-Series Data and Machine Learning Classifiers" in partial full filment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2018 to May 2019 under the supervision of Dr. Ekta Gandotra Assistant Professor(Senior Grade).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Yashraj Mishra, 151287

This is to certify that above statement made by the candidate is true to the best of my knowledge.

Dr. Ekta Gandotra

Assistant Professor(Senior Grade)
Computer Science Department
Dated:

Acknowledgement

I would like to express my deepest appreciation to all those who provided us the possibility to complete this report. A special gratitude we give to our final year project supervisor, Dr. Ekta Gandotra, whose contribution in stimulating suggestions and encouragement, helped me and my partner to coordinate our project well especially in writing this report.

Furthermore we would also like to acknowledge with much appreciation the crucial role of Jaypee University of Information Technology, who gave the permission to use all required equipment and the necessary materials to complete the project A framework on automated trade system using time-series data and machine learning classifiers .A special thanks goes to my supervisor, Dr. Ekta Gandotra, who help me to assemble the parts and gave suggestion about the project "Algorithmic Trading" he have invested his full effort in guiding us for achieving the goal. We have to appreciate the guidance given by other supervisor as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advices.

Thanking you,
Yashraj Mishra (151287)

Abstract

Algorithmic trades (AT) and their activity in the esteem revelation process on the S&P 500 summary associations are being assessed. Extraction of the association tickers and their individual stock data is being done. Gotten some answers concerning different AI classifiers and their importance in Algorithmic Trading .Algorithmic trading act purposely by checking the market for liquidity and deviations of expense from focal regard. Algorithmic Trading chooses the three fundamental conditions of the securities trade whether to buy, sell, or hold a stock. Different data controls were done and numerous abilities were made which were mapped to different names and using classifiers endeavored to anticipate the three conditions of the securities trade.

Overall, budgetary trade estimate is an extraordinarily confounding system, to control stocks as demonstrated by your necessities, incorporates cumbersome data of stocks and how these stocks can change their advancements and by the sum they will climb or down in light of some financial circumstances. Issue is that, can a Machine foresee these advancements and devise a sort of trading strategy according to the given data using particular AI models.

Different shippers would look at different degrees of a particular approach, a couple of sellers likely won't put confidence in trading using machines, as they trust exhibit renders itself once there is a monstrous addition or lows. Vendors may use different classes with different parameters, and attempt to devise a strategy that best fits the given enlightening gathering.

One way to deal with develop a procedure is believing that, in a market a couple of associations are significantly related, unfavorably related, and some presumably won't be associated in any way shape or form. Using the associations machine can settle on a choice on what basics the data has been given, given the Classifiers, in the wake of removing feature sets and mapping them to names, the Classifiers take those abilities and fit the given data according to the foreordained coordinated.

Table of Contents

List of Figures

Chapter 1

Introduction

Problem Definition:

The Stock market check is an exceptionally fascinating errand which joins high substances of how the budgetary exchange limits, and what unconventionalities can be prompted in a market in light of different conditions. While a few venders may battle that the market itself is functional, and that if there is new check or any assortment from the standard in a market it charms it by auditing itself, thusly making no space for conjectures, while several vendors may battle that on the off chance that the information is orchestrated well, by then machine can make a sort out of procedure that is persuading can affect high continue exchanging or HFT, which is just conceivable through Algorithmic Trading Systems or Automated Systems of Trade.

Money related authorities think about the expression, Buy low, Move high yet this does not give enough setting to settle on proper endeavor decisions. Before an investigator places assets into any stock, He should realize how money markets continues .Setting assets into a wonderful stock regardless at a horrible time can have awful results, while vitality for a common stock at the fortunate time can hold up under focal points. Cash related monetary pros of today are going toward this issue of trading as they don't for the most part understand concerning which stocks to buy or which stocks to offer with the authentic objective to get impeccable focal points. Envisioning whole game plan estimation of the stock is commonly clear than foreseeing on day-to-day premise as the stocks change rapidly reliably subject to world events.

The answer for this issue requests the utilization of instruments and advances identified with the field of information mining, design acknowledgment, machine learning and information forecast. The application will foresee the stock costs for the following exchanging day. The necessities and the usefulness of this application corresponds it to the class.

Project Overview and Specifications

Artificial intelligence (AI) to play an integral role in our day to day life applications whether it be home environment applications like Alexa or financial applications like trading, it is a development towards a new era of technology. This project comprises of an application of AI on financial data, known as algorithmic trading.

Automated trading systems involves the use of complex AI systems to make extremely fast trading decisions like buy, hold, or sell. It involves high frequency trading or HFT to make millions of trade in a day.

Machine learning is a subset of AI and generally provides solutions which learn from experience without being explicitly programmed.

In simple words, just the machine learning models are selected and fed with data the model then automatically adjusts its parameters and improves its outcome.

Hardware Specifications

Automated Trading systems may sound too complex but requires very few hardware costs, you just need a good computer with a good editor and you’re ready to go, not much requirement of extra hardware specifications.

Software Specifications

• Python Idle3.6.5

It is a python supervisor wherein we will really execute the calculation to separate informational indexes, perform controls, and anticipate the normal outcomes with exactness.

• NumPy

numpy is basically a module or you can say a library that is available in python for scientific computing now it contains a lot of things it contains a powerful dimensional array object then tools for integrating with C C++ it is also very useful in linear algebra Fourier transform and random number capabilities now let me tell you guys numpy can also be used as an efficient multi-dimensional container for data for generic data now let me tell you what exactly is multi-dimensional array now over here this picture actually depicts multi-dimensional array so we have various elements that are stored in their respective memory locations so we have one two threes in their own memory locations now why is it two dimensional it is two dimensional because it has rows as well as columns so you can see we have three columns and we have four rows available so that is the reason why it becomes a two dimensional array so if I would have had only one row then I would have said that it is a one dimensional array but since it contains rows as well as columns that is it is represented in a matrix form that is why we call this as a two dimensional array so I hope we are clear with what exactly two dimensional arrays

- **Matplotlib**

Matplotlib is a very useful library for the Python programming language and its statistical discipline extension NumPy. It gives an critique settled API to implanting plots into submissions developing totally beneficial.

- **Pandas**

It is likewise a Python library for information control and examination. Specifically, it offers information structures and activities for controlling numerical table and time arrangement information.

- **Pandas-Data Reader**

Use to remove information from large number of web sources.

- **BeautifulSoup4**

BeautifulSoup4 is a python bundle for analyzing HTML and XML documents. It produces a parse tree for described pages that can be used to obtain data from web pages which is supportive in web scraping.

• Scikit-Learn/SkLearn

It is Machine learning python library which chiefly includes arrangement, relapse, and bunching calculations like help vector machines or SVM, inclination boosting, irregular woods, and k-NearestNeighbour

Chapter 2

Literature Survey

Existing Systems

Money related trade judgment making is a strengthening and difficult errand of fiscal data guess. Figure about securities trade with high exactness improvement return advantage for examiners of the stocks. In perspective on the snare of budgetary trade financial data, expansion of productive models for forecast conclusion is very difficult, and it must be precise. This consider attempted to make models for guess of the securities trade and to pick whether to buy/hold the stock using data mining and AI techniques. The AI framework like Naive Bayes, k-Nearest neighbors (k-NN), Support Vector Machine(SVM), Artificial Neural Network(ANN) and Random Forest has been used for progressing of gauge model. Particular pointers are resolved from the stock prices

set up on timetable data and it is used as commitments of the proposed guess models. Ten years of securities trade data has been used for sign gauge of stock. Based on the instructive accumulation, these models can make buy/hold signal for monetary trade as a yield. The rule target of this errand is to deliver yield signal(buy/hold) as per customers essential like mean be contributed, time term of endeavor, least advantage, most prominent hardship, using data mining and AI frameworks.

Forecasting the way of stock prices is a widely deliberate subject in many fields including trading, finance, statistics and computer science. Depositors in the stock market can maximize their yield by export or selling their investment if they can determine when to enter and exit a position. Specialized traders typically use essential and/or technical analysis to inspect stocks in making venture decisions. Vital analysis involves a study of company fundamentals such as proceeds and profits, market position, growth rates, etc. Technical analysis, on the other hand, is based on the study of historical price oscillations. Due to the nature of market forces, economies tend to follow a pattern of expansion and contraction, over long periods of time. The stocks trade within an over arching environment where economy moves from one phase of the business cycle to the next.

Compared to the existing work, this project analyses the stocks trading decisions utilizing the technical conduct of the trading patterns within the context of the changeable economic and business environment.

The objective function is to maximize medium to longer term profits based on S&P500 stock market index. The inputs are the technical pointers data and the economic indicators data. Three models (neural network, soft max logistic regression, decision forest) are then used to predict the buy/sell decisions.

proposed Systems

As debated overhead stock market forecast is a huge subject and has a lot parts on which we can investigation upon, but one object all models have in common is their check on correctness of how well the models practical can fit to a given dataset and is it identical the results and forecasting correctly or not. Still each model has a few effects in common, they all need a list of companies of any stock exchange to forecast upon the three basic situations of market buy, hold, and sell and to do this the stock market data for each company against their tickers was stored in machine (to avoid larger accessing time) and data manipulations were performed in order to prepare the dataset for additional machine learning classifiers which will ultimately forecast the marks and deliver the output.

Feasibility Study

To plaid the practicability of the overhead model the given productivity will be plaid and coordinated alongside the graph of the definite company for that period of time and distinguish the patterns.

As a future Scope in our project we will further use quantopiana online platform for emerging trading approaches and back testing them, we will use it to advance a plan on quantopian and back test it to check the possibility of the tactic.

Chapter 3
System Analysis and Design

Technical Analysis

Technical Analysis is helpful to estimate the future economic stock movement based on stock historical movement. Technical limitations do not forecast stock price, but based on historical analysis, technical limits can forecast the stock movement on existing market condition over time. Technical examination help depositor to forecast the stock price movement (up/down) in that specific time interval. Technical examination habits a wide diversity of charts that show price over period.

The company tickers of S&P 500 list from Wikipedia is being saved and abstraction of stock data in contradiction of every company ticker is being done.

Then close index of every company is taken into account and put it into one data frame and try to find a connection between each company and then pre-processing the data and creating different technical parameters built on stock price, bulk and close worth and based on the movement of prices will progress technical meters that will aid set a target percentage to foretell buy, sell, hold.

Data Extraction

To initiate, a list of companies is essential for which we could progress statistical models and forecast future conditions of stock. Every company will have its private record of stock data from 1/1/2000 to 31/12/2017. Firstly, a list of companies is desired, so the S&P 500 list is mined from Wikipedia, there the S&P list is in a table format which is easy to handle.

Appointment and hit the Wikipedia page, and the response is given, which comprises our source code, to treat it any way you want you admittance the .text attribute of it which you turn into a soup object using BeautifulSoup4 which is HTML parsing library. Once the soup has been mined the table of ticker data is found by simply searching for ‘wiki table sortable classes’ and retrieving the table 0th column and first row of every entry to get the ticker symbol for 500 companies settled affording to their total market cap.

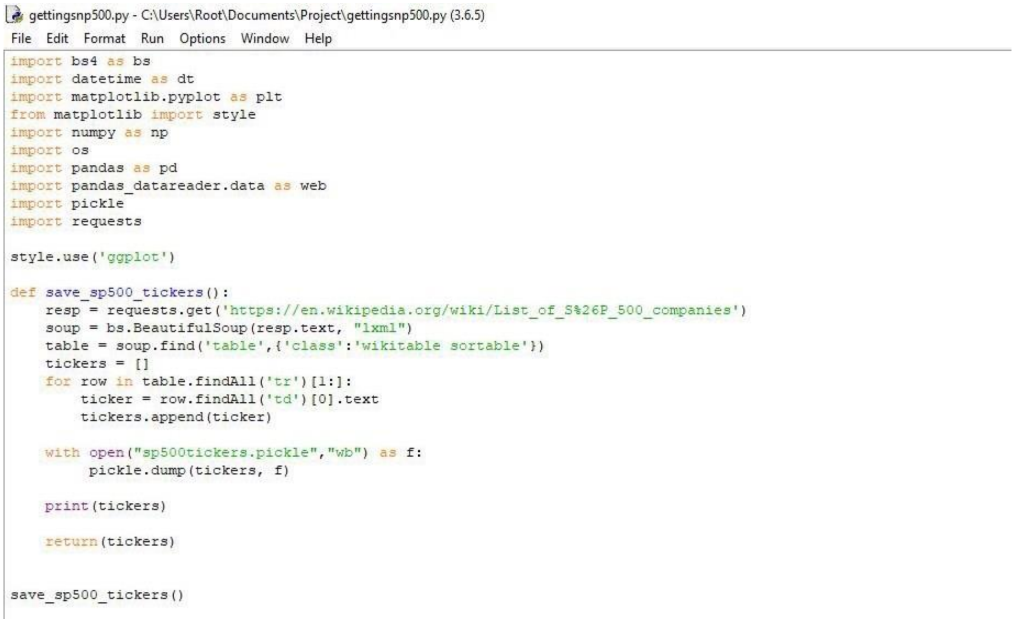
For each table row, ticker is the table data, clutch the .text of it and attach this ticker to the list, to save the list use pickle and if the list changes, modify it to check for specific periods of time. Redeemable the list of tickers, so not to hit Wikipedia again and again every time the script is ride.

Have tickers of 500 companies, need the stock estimating data of each company. Abstract the stock estimating data of the first 15 companies, each company has stock data to around 6000 admissions for each company. The companies which were started after 2000 and has vacant values, their entries of nan are replaced by zero.

Excerpt the data using pandas-data reader, a python mining library.

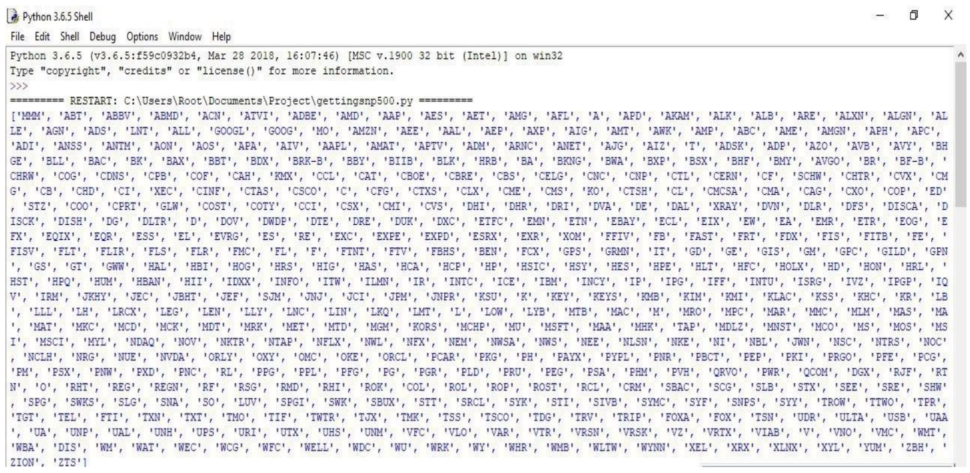
Excerpt the data from Morningstar and saved the data locally in .csv format and the data is used for further influences.

Now that, stock data and company tickers are deposited, can move further to data guidance and to know what indexes do we have in our data.


gettingsnp500.py - C:\Users\Root\Documents\Project\gettingsnp500.py (3.6.5)
File Edit Format Run Options Window Help

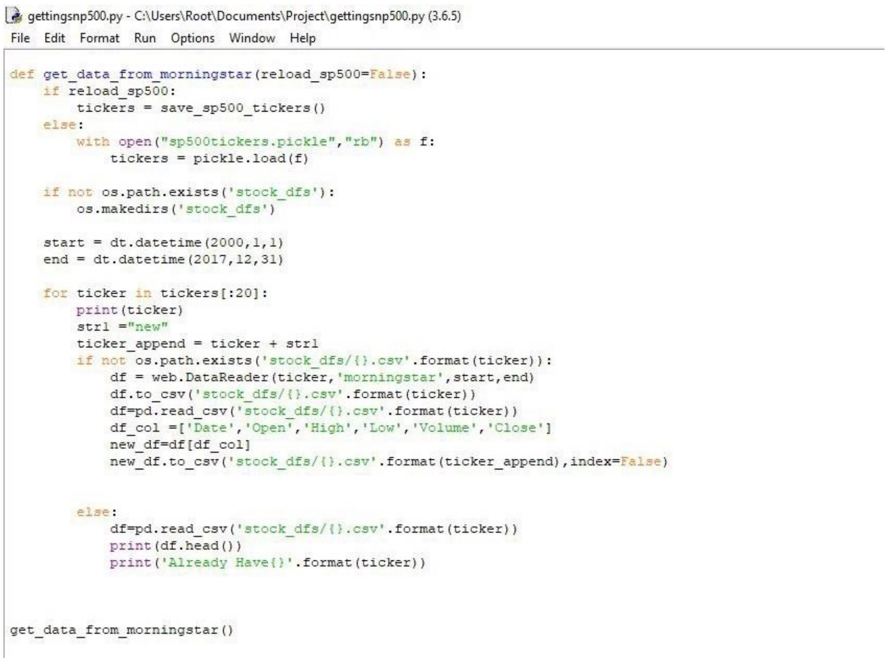
```python
import bs4 as bs
import datetime as dt
import matplotlib.pyplot as plt
from matplotlib import style
import numpy as np
import os
import pandas as pd
import pandas_datareader.data as web
import pickle
import requests

style.use('ggplot')

def save_sp500_tickers():
    resp = requests.get('https://en.wikipedia.org/wiki/List_of_S%26P_500_companies')
    soup = bs.BeautifulSoup(resp.text, "lxml")
    table = soup.find('table',{'class':'wikitable sortable'})
    tickers = []
    for row in table.findAll('tr')[1:]:
        ticker = row.findAll('td')[0].text
        tickers.append(ticker)

    with open("sp500tickers.pickle","wb") as f:
         pickle.dump(tickers, f)

    print(tickers)

    return(tickers)


save_sp500_tickers()
```

Fig 1: Code to Grab S&P 500tickers



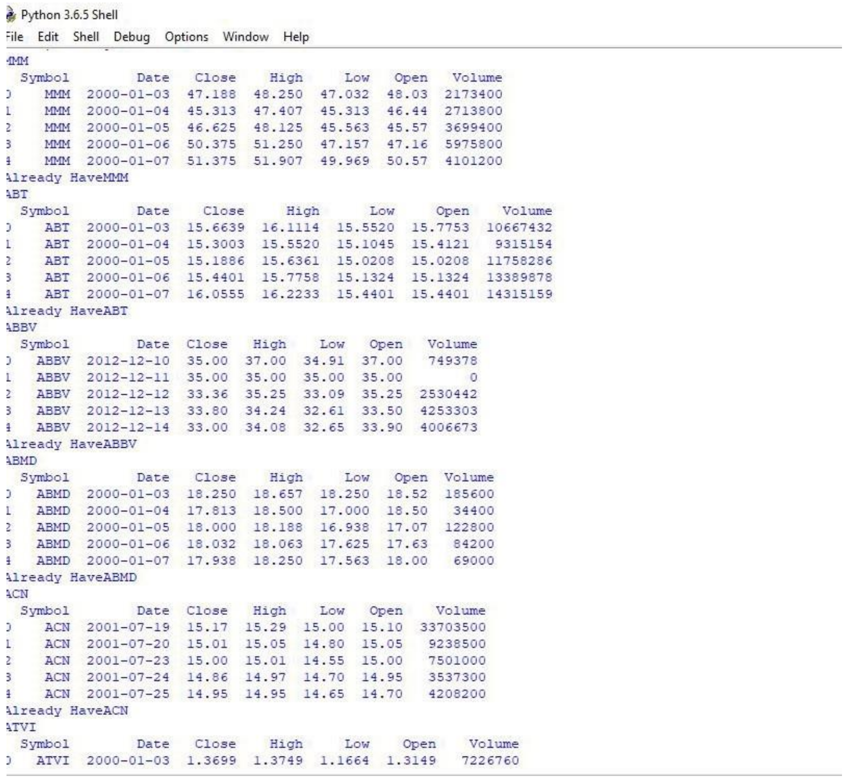
```
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 16:07:46) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
========= RESTART: C:\Users\Root\Documents\Project\gettingsnp500.py =========
['MMM', 'ABT', 'ABBV', 'ABMD', 'ACN', 'ATVI', 'ADBE', 'AMD', 'AAP', 'AES', 'AET', 'AMG', 'AFL', 'A', 'APD', 'AKAM', 'ALK', 'ALB', 'ARE', 'ALXN', 'ALGN', 'AL
LE', 'AGN', 'ADS', 'LNT', 'ALL', 'GOOGL', 'GOOG', 'MO', 'AMZN', 'AEE', 'AAL', 'AEP', 'AXP', 'AIG', 'AMT', 'AWK', 'AMP', 'ABC', 'AME', 'AMGN', 'APH', 'APC',
'ADI', 'ANSS', 'ANTM', 'AON', 'AOS', 'APA', 'AIV', 'AAPL', 'AMAT', 'APTV', 'ADM', 'ARNC', 'ANET', 'AJG', 'AIZ', 'T', 'ADSK', 'ADP', 'AZO', 'AVB', 'AVY', 'BH
GE', 'BLL', 'BAC', 'BK', 'BAX', 'BBT', 'BDX', 'BRK-B', 'BBY', 'BIIB', 'BLK', 'HRB', 'BA', 'BKNG', 'BWA', 'BXP', 'BSX', 'BHF', 'BMY', 'AVGO', 'BR', 'BF-B', '
CHRW', 'COG', 'CDNS', 'CPB', 'COF', 'CAH', 'KMX', 'CCL', 'CAT', 'CBOE', 'CBRE', 'CBS', 'CELG', 'CNC', 'CNP', 'CTL', 'CERN', 'CF', 'SCHW', 'CHTR', 'CVX', 'CM
G', 'CB', 'CHD', 'CI', 'XEC', 'CINF', 'CTAS', 'CSCO', 'C', 'CFG', 'CTXS', 'CLX', 'CME', 'CMS', 'KO', 'CTSH', 'CL', 'CMCSA', 'CMA', 'CAG', 'CXO', 'COP', 'ED'
, 'STZ', 'COO', 'CPRT', 'GLW', 'COST', 'COTY', 'CCI', 'CSX', 'CMI', 'CVS', 'DHI', 'DHR', 'DRI', 'DVA', 'DE', 'DAL', 'XRAY', 'DVN', 'DLR', 'DFS', 'DISCA', 'D
ISCK', 'DISH', 'DG', 'DLTR', 'D', 'DOV', 'DWDP', 'DTE', 'DRE', 'DUK', 'DXC', 'ETFC', 'EMN', 'ETN', 'EBAY', 'ECL', 'EIX', 'EW', 'EA', 'EMR', 'ETR', 'EOG', 'E
FX', 'EQIX', 'EQR', 'ESS', 'EL', 'EVRG', 'ES', 'RE', 'EXC', 'EXPE', 'EXPD', 'ESRX', 'EXR', 'XOM', 'FFIV', 'FB', 'FAST', 'FRT', 'FDX', 'FIS', 'FITB', 'FE', '
FISV', 'FLT', 'FLIR', 'FLS', 'FLR', 'FMC', 'FL', 'F', 'FTNT', 'FTV', 'FBHS', 'BEN', 'FCX', 'GPS', 'GRMN', 'IT', 'GD', 'GE', 'GIS', 'GM', 'GPC', 'GILD', 'GPN
', 'GS', 'GT', 'GWW', 'HAL', 'HBI', 'HOG', 'HRS', 'HIG', 'HAS', 'HCA', 'HCP', 'HP', 'HSIC', 'HSY', 'HES', 'HPE', 'HLT', 'HFC', 'HOLX', 'HD', 'HON', 'HRL', '
HST', 'HPQ', 'HUM', 'HBAN', 'HII', 'IDXX', 'INFO', 'ITW', 'ILMN', 'IR', 'INTC', 'ICE', 'IBM', 'INCY', 'IP', 'IPG', 'IFF', 'INTU', 'ISRG', 'IVZ', 'IPGP', 'IQ
V', 'IRM', 'JKHY', 'JEC', 'JBHT', 'JEF', 'SJM', 'JNJ', 'JCI', 'JPM', 'JNPR', 'KSU', 'K', 'KEY', 'KEYS', 'KMB', 'KIM', 'KMI', 'KLAC', 'KSS', 'KHC', 'KR', 'LB
', 'LLL', 'LH', 'LRCX', 'LEG', 'LEN', 'LLY', 'LNC', 'LIN', 'LKQ', 'LMT', 'L', 'LOW', 'LYB', 'MTB', 'MAC', 'M', 'MRO', 'MPC', 'MAR', 'MMC', 'MLM', 'MAS', 'MA
', 'MAT', 'MKC', 'MCD', 'MCK', 'MDT', 'MRK', 'MET', 'MTD', 'MGM', 'KORS', 'MCHP', 'MU', 'MSFT', 'MAA', 'MHK', 'TAP', 'MDLZ', 'MNST', 'MCO', 'MS', 'MOS', 'MS
I', 'MSCI', 'MYL', 'NDAQ', 'NOV', 'NKTR', 'NTAP', 'NFLX', 'NWL', 'NFX', 'NEM', 'NWSA', 'NWS', 'NEE', 'NLSN', 'NKE', 'NI', 'NBL', 'JWN', 'NSC', 'NTRS', 'NOC'
, 'NCLH', 'NRG', 'NUE', 'NVDA', 'ORLY', 'OXY', 'OMC', 'OKE', 'ORCL', 'PCAR', 'PKG', 'PH', 'PAYX', 'PYPL', 'PNR', 'PBCT', 'PEP', 'PKI', 'PRGO', 'PFE', 'PCG',
'PM', 'PSX', 'PNW', 'PXD', 'PNC', 'RL', 'PPG', 'PPL', 'PFG', 'PG', 'PGR', 'PLD', 'PRU', 'PEG', 'PSA', 'PHM', 'PVH', 'QRVO', 'PWR', 'QCOM', 'DGX', 'RJF', 'RT
N', 'O', 'RHT', 'REG', 'REGN', 'RF', 'RSG', 'RMD', 'RHI', 'ROK', 'COL', 'ROL', 'ROP', 'ROST', 'RCL', 'CRM', 'SBAC', 'SCG', 'SLB', 'STX', 'SEE', 'SRE', 'SHW'
, 'SPG', 'SWKS', 'SLG', 'SNA', 'SO', 'LUV', 'SPGI', 'SWK', 'SBUX', 'STT', 'SRCL', 'SYK', 'STI', 'SIVB', 'SYMC', 'SYF', 'SNPS', 'SYY', 'TROW', 'TTWO', 'TPR',
'TGT', 'TEL', 'FTI', 'TXN', 'TXT', 'TMO', 'TIF', 'TWTR', 'TJX', 'TMK', 'TSS', 'TSCO', 'TDG', 'TRV', 'TRIP', 'FOXA', 'FOX', 'TSN', 'UDR', 'ULTA', 'USB', 'UAA
', 'UA', 'UNP', 'UAL', 'UNH', 'UPS', 'URI', 'UTX', 'UHS', 'UNM', 'VFC', 'VLO', 'VAR', 'VTR', 'VRSN', 'VRSK', 'VZ', 'VRTX', 'VIAB', 'V', 'VNO', 'VMC', 'WMT',
'WBA', 'DIS', 'WM', 'WAT', 'WEC', 'WCG', 'WFC', 'WELL', 'WDC', 'WU', 'WRK', 'WY', 'WHR', 'WMB', 'WLTW', 'WYNN', 'XEL', 'XRX', 'XLNX', 'XYL', 'YUM', 'ZBH', '
ZION', 'ZTS']
```

Fig2: Output of Grabbed 500 tickers



```
def get_data_from_morningstar(reload_sp500=False):
    if reload_sp500:
        tickers = save_sp500_tickers()
    else:
        with open("sp500tickers.pickle","rb") as f:
            tickers = pickle.load(f)

    if not os.path.exists('stock_dfs'):
        os.makedirs('stock_dfs')

    start = dt.datetime(2000,1,1)
    end = dt.datetime(2017,12,31)

    for ticker in tickers[:20]:
        print(ticker)
        str1 ="new"
        ticker_append = ticker + str1
        if not os.path.exists('stock_dfs/{}.csv'.format(ticker)):
            df = web.DataReader(ticker,'morningstar',start,end)
            df.to_csv('stock_dfs/{}.csv'.format(ticker))
            df=pd.read_csv('stock_dfs/{}.csv'.format(ticker))
            df_col =['Date','Open','High','Low','Volume','Close']
            new_df=df[df_col]
            new_df.to_csv('stock_dfs/{}.csv'.format(ticker_append),index=False)


        else:
            df=pd.read_csv('stock_dfs/{}.csv'.format(ticker))
            print(df.head())
            print('Already Have{}'.format(ticker))



get_data_from_morningstar()
```

Fig 3: Code to Grab stock data from Morningstar



```
MMM
  Symbol        Date   Close    High     Low   Open   Volume
0    MMM  2000-01-03  47.188  48.250  47.032  48.03  2173400
1    MMM  2000-01-04  45.313  47.407  45.313  46.44  2713800
2    MMM  2000-01-05  46.625  48.125  45.563  45.57  3699400
3    MMM  2000-01-06  50.375  51.250  47.157  47.16  5975800
4    MMM  2000-01-07  51.375  51.907  49.969  50.57  4101200
Already HaveMMM
ABT
  Symbol        Date    Close     High      Low     Open    Volume
0    ABT  2000-01-03  15.6639  16.1114  15.5520  15.7753  10667432
1    ABT  2000-01-04  15.3003  15.5520  15.1045  15.4121   9315154
2    ABT  2000-01-05  15.1886  15.6361  15.0208  15.0208  11758286
3    ABT  2000-01-06  15.4401  15.7758  15.1324  15.1324  13389878
4    ABT  2000-01-07  16.0555  16.2233  15.4401  15.4401  14315159
Already HaveABT
ABBV
  Symbol        Date  Close   High    Low   Open   Volume
0   ABBV  2012-12-10  35.00  37.00  34.91  37.00   749378
1   ABBV  2012-12-11  35.00  35.00  35.00  35.00        0
2   ABBV  2012-12-12  33.36  35.25  33.09  35.25  2530442
3   ABBV  2012-12-13  33.80  34.24  32.61  33.50  4253303
4   ABBV  2012-12-14  33.00  34.08  32.65  33.90  4006673
Already HaveABBV
ABMD
  Symbol        Date   Close    High     Low   Open  Volume
0   ABMD  2000-01-03  18.250  18.657  18.250  18.52  185600
1   ABMD  2000-01-04  17.813  18.500  17.000  18.50   34400
2   ABMD  2000-01-05  18.000  18.188  16.938  17.07  122800
3   ABMD  2000-01-06  18.032  18.063  17.625  17.63   84200
4   ABMD  2000-01-07  17.938  18.250  17.563  18.00   69000
Already HaveABMD
ACN
  Symbol        Date  Close   High    Low   Open    Volume
0    ACN  2001-07-19  15.17  15.29  15.00  15.10  33703500
1    ACN  2001-07-20  15.01  15.05  14.80  15.05   9238500
2    ACN  2001-07-23  15.00  15.01  14.55  15.00   7501000
3    ACN  2001-07-24  14.86  14.97  14.70  14.95   3537300
4    ACN  2001-07-25  14.95  14.95  14.65  14.70   4208200
Already HaveACN
ATVI
  Symbol        Date   Close    High     Low    Open   Volume
0   ATVI  2000-01-03  1.3699  1.3749  1.1664  1.3149  7226760
```

Fig 4 : Output Stock data of companies.

Data Manipulation

Presently that, stock estimating information of organizations is put away, unite this information in one information outline. While, the majority of the information available to us is now there, may need to really get to the information together. To do this, join the majority of the stock datasets together. Every one of the stock records right now accompany: Open, High, Low, Close, Volume. At any rate to begin, simply intrigued by the nearby for the present.

Draw our recently made rundown of tickers and start with an unfilled information outline, and be prepared to peruse in each stock's information outline, for the most part intrigued by the Close segments, and the segment has been renamed to whatever the ticker name is and a common information outline begins building.

Use pandas outer join to combine the data frame and if there’s nothing in the main d/f, then start with the current d/f, otherwise use pandas join.

Now, check if any interesting correlation data is found. To do this, visualize it, since it’s a lot of data. Use matplot lib with numpy to fulfill this task.

Utilize the mapping style 'ggplot', building a connection table in pandas is very basic, perform it by simply including main combine _data frame. Correlation, and it will naturally take a gander at the whole information outline, and decide the relationship of each section to each segment.
Now, by creating this painting, heatmap, this is a very simple hot map made in Map plot, but the tool has been made as it has worked. To do this, you will need to edit the actual data. This will give you an example of the relevant mathematical price related prices and we will do our calculations and indicators.

Heat map is designed using a variety of colours, which can be a range of anything to anything, the color balance is gotten from the c-map, use RdY1Gn, which is a color -record that goes from red on the shortened side to yellow in the middle and green for the progressive part of the scale, which will give red for rejections affairs, green for positive and yellow for no connections. Added adjacent bar that is a colour bar as a sort of scale for considerate. Now, regular x and y axis rocks so there is a way to know which concerns are which, since there is just plotted the data.

Used ‘overturn_axis()’ and ‘tick_top’, this will dismissive the axis so the graph is a tiny easier

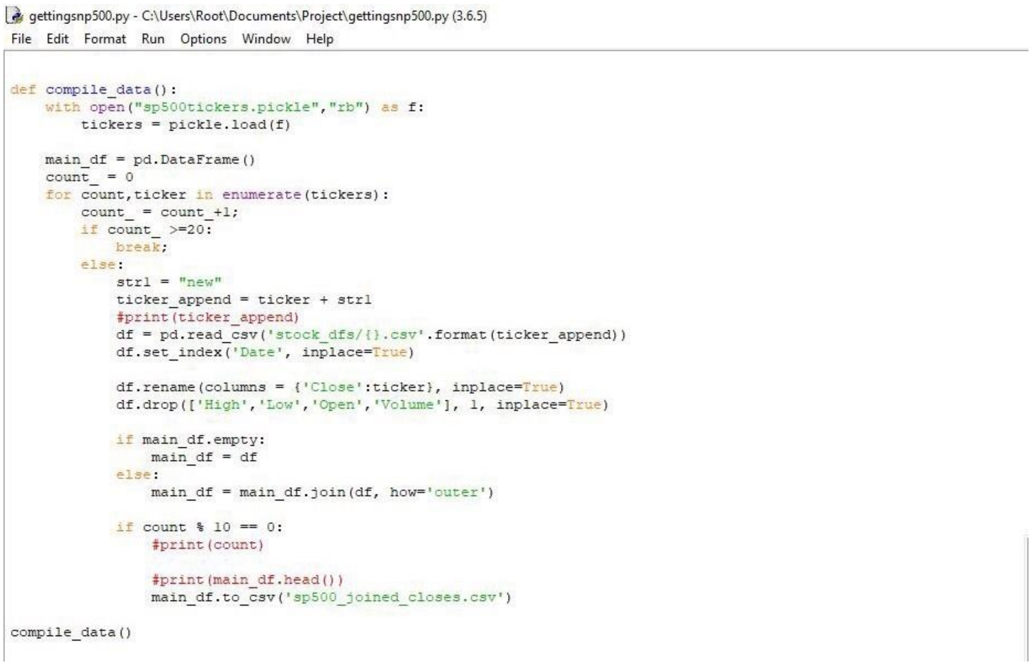
To deliver, since there will be specific space between the x’s and y’s. Generally matplotlib foliage room on the extreme ends of graph since this inclines to make graphs calmer to read, but, in this case, it doesn’t. Then also flippant the x axis to be at the top of the graph, rather than the customary bottom, again just to make this more comparable a correlation table should be, and add the company names to the currently-nameless ticks.

In this case, the particular same list from both sides could be used, since column_labels and row_labels should be matching lists, this won’t always be true for all heat maps.

Also rotate the x ticks, which are the definite tickers themselves, since normally they’ll be written out horizontally. There are over 500 labels here, so rotate them 90 degrees so they’re vertical. It’s still a graph that’s going to be far too large to really see everything zoomed out,

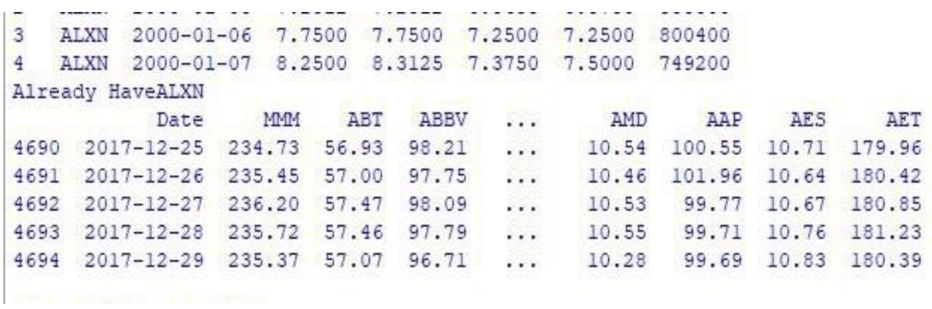
but that’s fine, the line that pronounces heatmap1.set_clim(-1,1) just tells the color map that the range is going to be from -1 to +1.

Looking at the correlations, see that there are various associations. The majority of companies are, expectedly positively related. There are a few that are strongly linked there are some that are negatively correlated and some are not correlated at all.


gettingsnp500.py - C:\Users\Root\Documents\Project\gettingsnp500.py (3.6.5)
File Edit Format Run Options Window Help

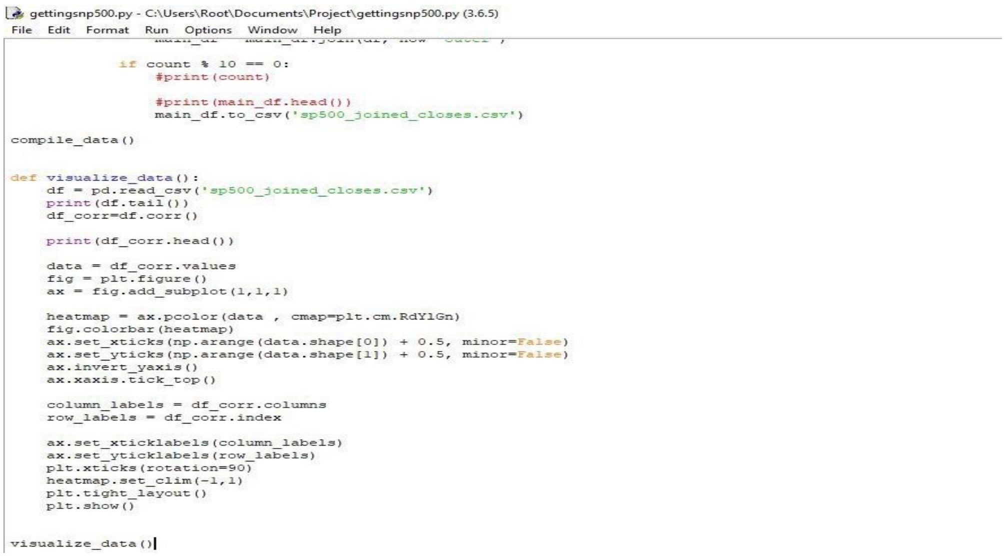
```
def compile_data():
    with open("sp500tickers.pickle","rb") as f:
        tickers = pickle.load(f)

    main_df = pd.DataFrame()
    count_ = 0
    for count,ticker in enumerate(tickers):
        count_ = count_+1;
        if count_ >=20:
            break;
        else:
            str1 = "new"
            ticker_append = ticker + str1
            #print(ticker_append)
            df = pd.read_csv('stock_dfs/{}.csv'.format(ticker_append))
            df.set_index('Date', inplace=True)

            df.rename(columns = {'Close':ticker}, inplace=True)
            df.drop(['High','Low','Open','Volume'], 1, inplace=True)

            if main_df.empty:
                main_df = df
            else:
                main_df = main_df.join(df, how='outer')

            if count % 10 == 0:
                #print(count)

                #print(main_df.head())
                main_df.to_csv('sp500_joined_closes.csv')

compile_data()
```

Fig 5: Code to compile all close index of company in one data frame.



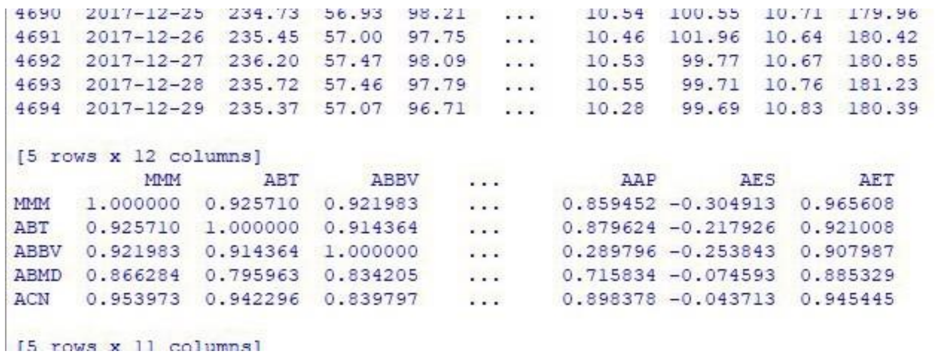
```
3   ALXN  2000-01-06  7.7500  7.7500  7.2500  7.2500  800400
4   ALXN  2000-01-07  8.2500  8.3125  7.3750  7.5000  749200
Already HaveALXN
            Date     MMM    ABT   ABBV   ...     AMD     AAP    AES     AET
4690  2017-12-25  234.73  56.93  98.21   ...   10.54  100.55  10.71  179.96
4691  2017-12-26  235.45  57.00  97.75   ...   10.46  101.96  10.64  180.42
4692  2017-12-27  236.20  57.47  98.09   ...   10.53   99.77  10.67  180.85
4693  2017-12-28  235.72  57.46  97.79   ...   10.55   99.71  10.76  181.23
4694  2017-12-29  235.37  57.07  96.71   ...   10.28   99.69  10.83  180.39
```

Fig 6: Output close index of all companies together in one data frame.


gettingsnp500.py - C:\Users\Root\Documents\Project\gettingsnp500.py (3.6.5)
File Edit Format Run Options Window Help

```
                main_df = main_df.join(df, how='outer')

            if count % 10 == 0:
                #print(count)

                #print(main_df.head())
                main_df.to_csv('sp500_joined_closes.csv')

compile_data()


def visualize_data():
    df = pd.read_csv('sp500_joined_closes.csv')
    print(df.tail())
    df_corr=df.corr()

    print(df_corr.head())

    data = df_corr.values
    fig = plt.figure()
    ax = fig.add_subplot(1,1,1)

    heatmap = ax.pcolor(data , cmap=plt.cm.RdYlGn)
    fig.colorbar(heatmap)
    ax.set_xticks(np.arange(data.shape[0]) + 0.5, minor=False)
    ax.set_yticks(np.arange(data.shape[1]) + 0.5, minor=False)
    ax.invert_yaxis()
    ax.xaxis.tick_top()

    column_labels = df_corr.columns
    row_labels = df_corr.index

    ax.set_xticklabels(column_labels)
    ax.set_yticklabels(row_labels)
    plt.xticks(rotation=90)
    heatmap.set_clim(-1,1)
    plt.tight_layout()
    plt.show()

visualize_data()
```

Fig 7: Code to find and visualize correlations.



```
4690  2017-12-25  234.73  56.93  98.21   ...     10.54  100.55  10.71  179.96
4691  2017-12-26  235.45  57.00  97.75   ...     10.46  101.96  10.64  180.42
4692  2017-12-27  236.20  57.47  98.09   ...     10.53   99.77  10.67  180.85
4693  2017-12-28  235.72  57.46  97.79   ...     10.55   99.71  10.76  181.23
4694  2017-12-29  235.37  57.07  96.71   ...     10.28   99.69  10.83  180.39

[5 rows x 12 columns]
           MMM       ABT      ABBV    ...          AAP       AES       AET
MMM   1.000000  0.925710  0.921983    ...     0.859452 -0.304913  0.965608
ABT   0.925710  1.000000  0.914364    ...     0.879624 -0.217926  0.921008
ABBV  0.921983  0.914364  1.000000    ...     0.289796 -0.253843  0.907987
ABMD  0.866284  0.795963  0.834205    ...     0.715834 -0.074593  0.885329
ACN   0.953973  0.942296  0.839797    ...     0.898378 -0.043713  0.945445

[5 rows x 11 columns]
```

Fig 8: Output Of the correlation table.

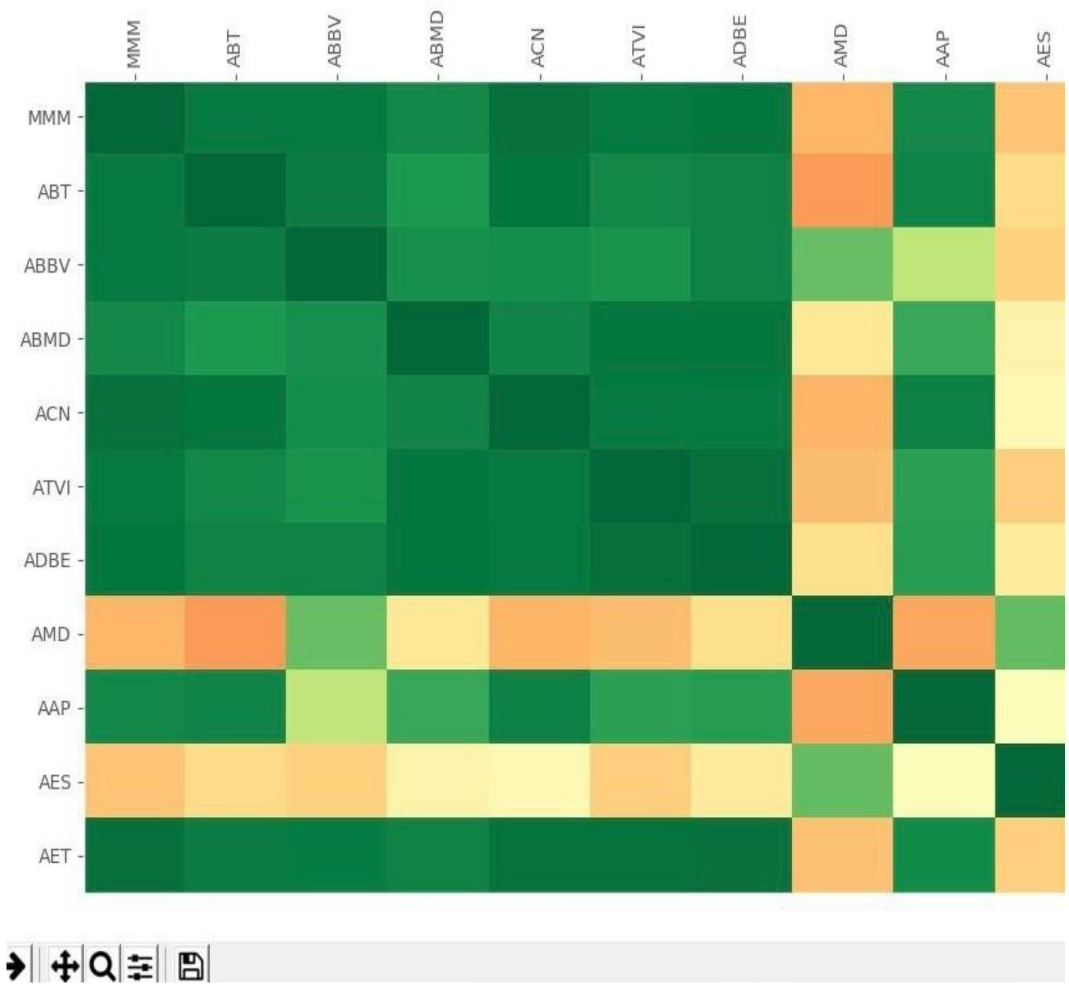
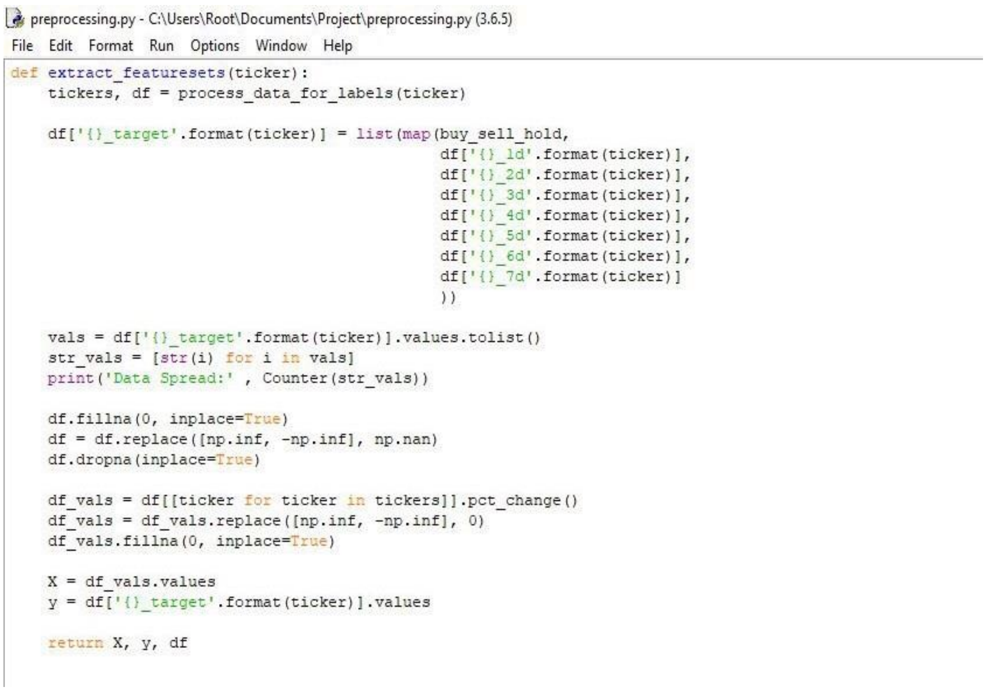


Fig 9: Heatmap of the correlations.

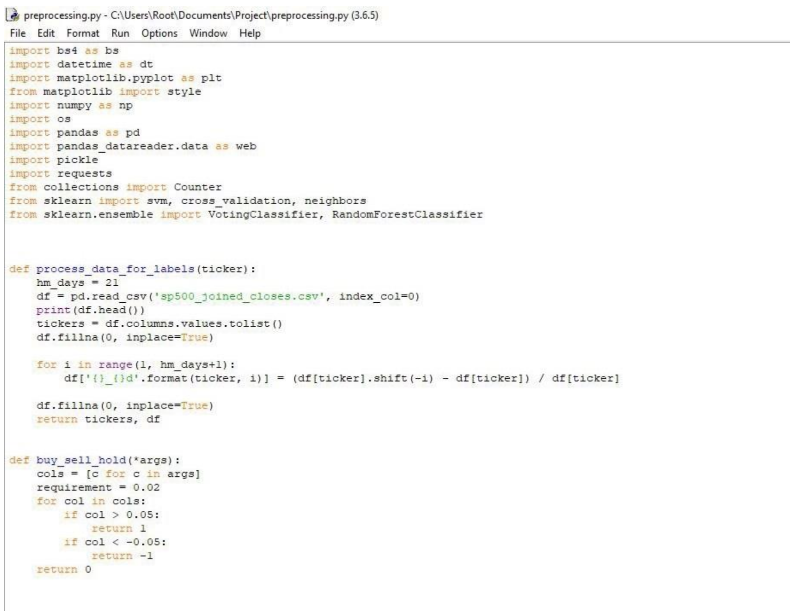
Preprocessing the Data to prepare for Machine Learning

First thing this is one of the major steps of data mining what has to be done before data mining so normally data mining consists of a set of algorithms which are used in order to mine data and get patterns out of it so this is a collection of very intelligent algorithms which are being designed by researcher and normally when we think of data mining we only think of these algorithms but generally a lot of work needs to be done before we can apply these algorithms and there's a lot of work that has to be done after we apply the algorithms which includes visualization of the patterns and so on but normally we think that this middle part of the work which is actually applying the algorithms is the major portion but it turns out that these remaining parts is actually 90% of the work of typical application the only 10% of the work consists in actually applying the algorithms 90% of the work goes into doing data pre-processing and the post data mining task which is visualization so what is data pre-processing data pre-processing is actually preparing the data in in a particular way into a format that is suitable for actually doing the data mining algorithms so data pre-processing generally consists of several steps sometimes data is not always present in a single place it might have to be gathered from lots of different places for example if you are doing some web application so the data might have to be gathered from several different web sites and collected into one single format before you can actually do data mining or in a business scenario if you are the owner of let's say a chain of let's say supermarkets or some other kind of stores then you have branches all over the city or a core in several different cities in each branch you might have a separate database which stores the data of that branch and what you need is to collect all this data into a single location so what you need is data gathering or data collection from multiple different sources into a single repository and that sometimes they call as data warehousing okay so where you collect data from lots of different places clean it clean the data so after you gather the data you have to clean the data now cleaning sounds like a very mundane kind of work that you know it's not very pleasant kind of work but actually lots of researchers have actually focused on data cleaning aspects and they've developed some nice algorithms which can do data cleaning automatically okay so it's not actually a very manual intensive job so after there is still some manual effort but there are lots of tools that help you to do the data cleaning automatically ok so once you do data gathering and cleaning you have to convert the data into a suitable format and the format that you need to convert it into depends on what kind of gait of mining algorithms you want to apply after that ok so depending on whether you want to do association rules or classification or clustering you need to convert the available data into suitable format so for that you might want to ignore certain aspects of the data certain attributes you might want to include only certain attributes you might want to combine certain attributes into other attributes and then use the resulting format has input to the data mining algorithms and finally you also need to do what is called as normalization consists of making the data into a it's the kind of averaging of all the data into a uniform range okay so for example suppose you have some attribute which is having a range between let's say 0 and 1 million ok and you might think that this range is too large ok so you might want to normalize all the values that occur for this attribute into the range lecture 0 to 1 so there will be fractional values between 0 and 1 okay so in all to do that obviously in this case is very simple you just need to divide each of the available values by one million and then you will get the corresponding fractional values but in other cases data normalization might be a little more tricky things like for example in some applications for example think of a scenario where let's say you're the judge of some music competition okay and students are the participants are singing different songs and you're not a really America highly musical expert but you can actually evaluate the music and judge whether it's good or not so let's say the first participant comes and sings okay and since you have no baseline in your mind you don't know what score to give him you have to give us code between lecture zero and ten and you don't know what's go to give him because there's a first person who assigned so let's say you kind of be lenient and you give him a score of eight okay so then the next person comes and sings and maybe he sings better than the previous person so you have to give a higher further than this so let's say you give him nine next person comes he sings even better so you let us say give him nine point five okay so like this you and the next person who comes maybe he sings poorer than this person so you give him seven okay so in that way you keep giving scores to all the different people so you have a collection of data like this but this data is not very clean data it's not data that is that has been thought about nicely because here it doesn't mean that this person thanked exactly one unit better than this person and this person sang one unit less than this person all that you had psychologically was this person standing a little bit less and this person sound a little bit better than this person so after once you have this data you will need to do some kind of processing on this data to make sure that it's all well normalized for example suppose the person who had the worst score here was fight so then you one nice thing to do would be to normalize all these different values into a range between zero and one and so for that maybe you want to subtract each of these values by five so subtract this by five subtract each of these things by five because five is a minimum value here and then once you get the corresponding values you divide those values by the maximum value that you get so that you get everything between a range between zero and one so these kinds of things that you do is called normalization to bring all the data into common format so these are the kinds of tasks that you need to do for data pre-processing and this is a essential prerequisite for actually applying intelligent data mining algorithm .


preprocessing.py - C:\Users\Root\Documents\Project\preprocessing.py (3.6.5)
File Edit Format Run Options Window Help

```
def extract_featuresets(ticker):
    tickers, df = process_data_for_labels(ticker)

    df['{}_target'.format(ticker)] = list(map(buy_sell_hold,
                                              df['{}_1d'.format(ticker)],
                                              df['{}_2d'.format(ticker)],
                                              df['{}_3d'.format(ticker)],
                                              df['{}_4d'.format(ticker)],
                                              df['{}_5d'.format(ticker)],
                                              df['{}_6d'.format(ticker)],
                                              df['{}_7d'.format(ticker)]
                                              ))

    vals = df['{}_target'.format(ticker)].values.tolist()
    str_vals = [str(i) for i in vals]
    print('Data Spread:' , Counter(str_vals))

    df.fillna(0, inplace=True)
    df = df.replace([np.inf, -np.inf], np.nan)
    df.dropna(inplace=True)

    df_vals = df[[ticker for ticker in tickers]].pct_change()
    df_vals = df_vals.replace([np.inf, -np.inf], 0)
    df_vals.fillna(0, inplace=True)

    X = df_vals.values
    y = df['{}_target'.format(ticker)].values

    return X, y, df
```

Fig 10: Code to set trading conditions and data processing for labels.


preprocessing.py - C:\Users\Root\Documents\Project\preprocessing.py (3.6.5)
File Edit Format Run Options Window Help

```python
import bs4 as bs
import datetime as dt
import matplotlib.pyplot as plt
from matplotlib import style
import numpy as np
import os
import pandas as pd
import pandas_datareader.data as web
import pickle
import requests
from collections import Counter
from sklearn import svm, cross_validation, neighbors
from sklearn.ensemble import VotingClassifier, RandomForestClassifier



def process_data_for_labels(ticker):
    hm_days = 21
    df = pd.read_csv('sp500_joined_closes.csv', index_col=0)
    print(df.head())
    tickers = df.columns.values.tolist()
    df.fillna(0, inplace=True)

    for i in range(1, hm_days+1):
        df['{}_{}d'.format(ticker, i)] = (df[ticker].shift(-i) - df[ticker]) / df[ticker]

    df.fillna(0, inplace=True)
    return tickers, df


def buy_sell_hold(*args):
    cols = [c for c in args]
    requirement = 0.02
    for col in cols:
        if col > 0.05:
            return 1
        if col < -0.05:
            return -1
    return 0
```

Fig 11: Code to extract feature sets and map them to labels.

SYSTEM DEVELOPMENT

Framework improvement is a procedure of making a product model, testing it, changing over it to a total model and reapplying different testing calculations to it lastly making the whole programming.

Mathematics plays vital role in developing a model using RNN(Deep Machine Learning). We require measured methods to:

- Obtain the correct algorithm on the mathematical data available.
- Obtaining correct features and parameters for the model so that the future estimates are precise.
- To evaluate the model for over fitting and under fitting of the data set.
- Expecting the ambiguity of the project model.

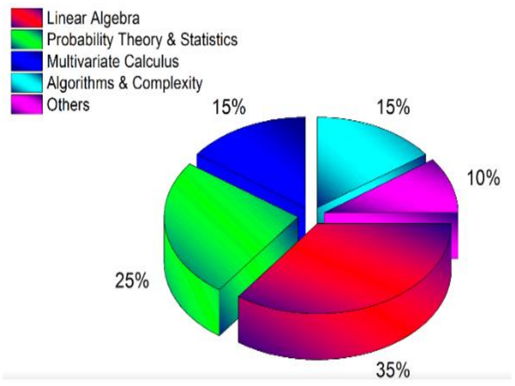


Fig 12: Pie chart

Functions in mathematical models to estimate faults:

Different functions are:

- **Mean Square Error:**

It is the mean of all the squared errors attained by an algorithm in logistic regression. The error is the change between the actual value and the calculated value, which is intended between numerous data points.

$$\sum_{i=1}^{n} \frac{\left(w^T x(i) - y(i)\right)^2}{n}$$

Here :wt is the weight connected,

x(i) is forecast value,

y(i) is the definite value

- **Euclidean distance metric:**

This is an extra very important performance metric used in ML algorithms to compute the distance among to consequences. It takes the feature set as input and put on the formula to the values of the parameters of the point to analyze the distance. Euclidean space can also become a capable metric in space. This metric is also known as Pythagorean metric.

$$dist((x, y), (a, b)) = \sqrt{(x - a)^2 + (y - b)^2}$$

Here: (x,y) are values of two features of point 1.

(a,b) are values of tw0 features of point 2.

- **Manhattan Distance:**

Manhattan distance is also a distance metric helpful in machine learning algorithms. It varies by Euclidian distance very faintly. This metric is used we very rapidly Want to find the distance between them.

ALGORITHMS

So, basically the algorithm we used here depends upon the recurrent neural network where we consider long short-term memory cell. Neural Network mimics the behavior of the brain and sometimes attains the superhuman capabilities. A neural network consists of 3 main parts i.e. the input layer, a hidden layer and output layer along with some activation functions. So let's discuss some basic algorithmic structure.

Basic algorithmic structure

Input layer consists of input features that are basically derived from the old output of stock market prediction. The input layer consists of many old features of the stock market. The input layer will be connected to some hidden layers. The hidden layer will pass on to some more hidden layers and in the end the output layers. The output layer will basically predict the probability. The probability will be determined by sigmoid activation function or rather by a softmax activation function. So, the input tensor will be the shape of input and in between, there are many hidden layers. It is important to give the shape of input to the tensor otherwise it will not be able to predict the output. The input layer is basically judged by the recurrent neural network. The recurrent neural network consists of long short-term memory cell. The data is contained inside the CSV file which is old stock market predictions. Info layer comprises information includes that are fundamentally gotten from the old yield of securities exchange forecast. The info layer comprises of numerous old highlights of a securities exchange. The information layer will be associated with some concealed layers. The shrouded layer will pass on to some more concealed layers and at last the yield layers. The yield layer will fundamentally foresee the likelihood. The likelihood will be dictated by sigmoid initiation work or rather by softmax actuation work. In this way, the info tensor will be the state of information and in the middle of there are many concealed layers. It is essential to give the state of contribution to the tensor else it won't have the capacity to anticipate the yield. The information layer is fundamentally made a decision by the intermittent neural system. The intermittent neural system comprises a long transient memory cell. The information is contained inside the CSV document which is old securities exchange forecasts.

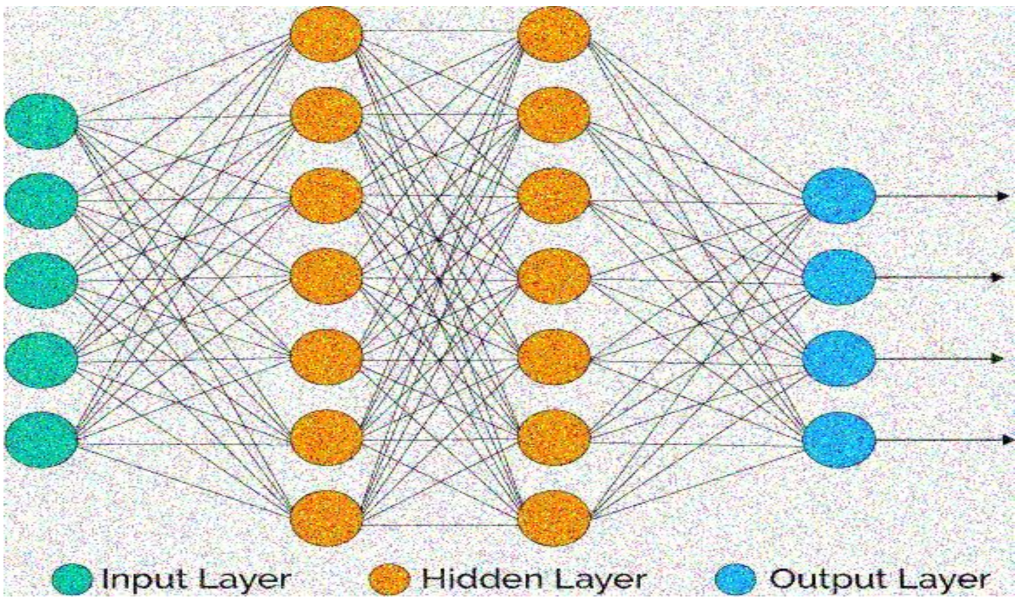


Fig 13: Layer diagram

Activation Functionplays an important role in determination of useful output. Let us suppose a case in which a negative output is not possible. We, will apply RELU (Rectified Linear Unit) activation function if we just want to consider positive part. Examples of Activation Function are RELU, Sigmoid Function, Softplus, Gaussian, Softmax etc. We will mostly use RELU activation function.

Activation Function is fundamentally used to scale the yield esteems or now and then used to change over the high qualities to likelihood so we can recognize the most plausible yield. For our situation, we have for the most part utilized a RELU activation work that changes over negative qualities to zero. Activation capacities can likewise be utilized to expel non-vital qualities which can be destructive to our classifier. In Recurrent Neural system generally utilized activation capacities are relu and sigmoid however there are a wide range of activation capacities like softmax, tan hyperbolic, exponential, delicate in addition to and so on. In our model activation capacities have assumed an exceptionally good job. Activation capacity can be of an extraordinary significance in the event that we talk about the important qualities by simply scaling or evacuating the non-vital qualities. This can lead the exponential increment in precision of model if there should arise an occurrence of preparing information and testing information. Along, with preparing and testing information approval information will diminish the over fitting of model.

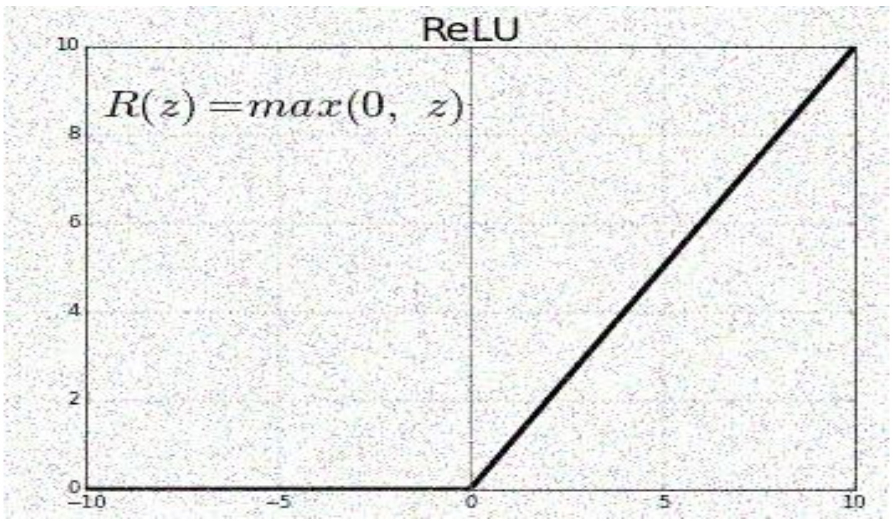


Fig 14: ReLU function

Reccurent Neural Networks (RNN) :

RNNs are an amazing and strong sorts of Neural Systems. In a RNN, the data goes through a circle/loop. When it settles on a choice, it mulls over the current info and furthermore what it has gained from the sources of info it got beforehand or previously.

A Recurrent Neural Network feeds itself with two inputs, the present input and the recent past. This is imperative and Crucial because the succession/sequence of information contains urgent data about what's coming straightaway and that's the reason RNN can do things other algorithms can not.

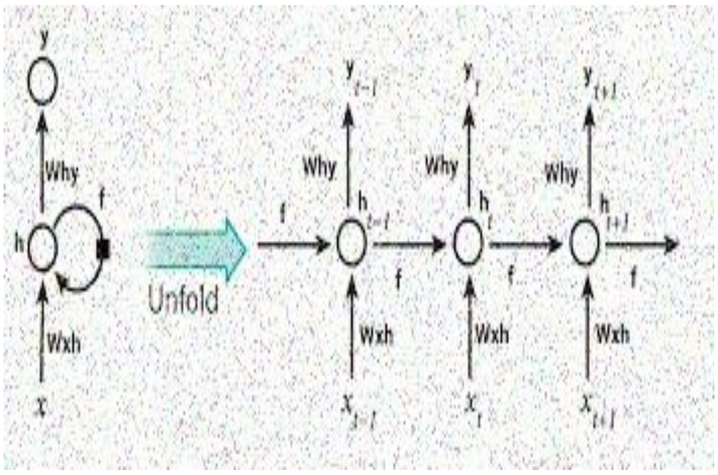


Fig 15: Neural network

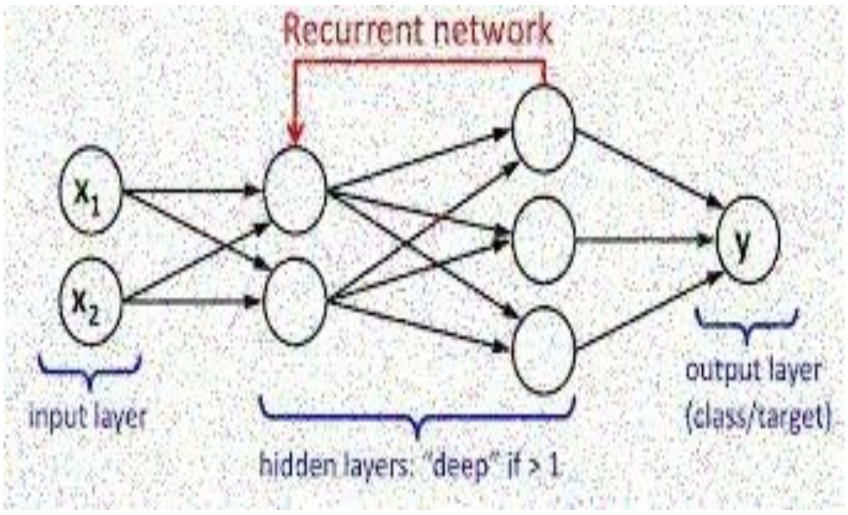


Fig 16: Recurrent neural network

Introduction To Classifiers

Classification can be classified as a major process that includes predicting the class of given data sets for prediction of different real time applications, classes are also sometimes referred to or known as targets or labels or categories. Classification includes predictive Modelling which is the task of approximating a given mapping function (f) from the input variables (X) to discrete output variables(y). Consider a example, spam detection in our email service providers can be identified to as a classification problem, it includes binary classification as there are only 2 classes of spam and not spam. A classifier will utilize some training data in a given percentage to understand how the given input variables will relate to the class. In this particular case, that is spam and non-spam, e-mails have to be used to provide training data to machine. When the classifier will be trained accurately it can be used to detect an unknown e- mail.

Classification generally belongs to the category of supervised learning where the targets or labels or categories are also provided with the input data. There can be many applications in classification in many domains such as in stock market, recommender systems, target marketing etc.

There are two types of learners in classification as lazy learners and
eager learners. 1. Lazy learners
Lazy learners have a simple task to store the training data and wait until the testing data appears. When the testing data appears, classification is conducted that is based on the most related information there is in the put away preparing information. Contrasted with the excited students, the apathetic students have exceptionally less preparing time however has additional time in foreseeing.

Examples. k-nearestneighbour, Case-based thinking

2. Enthusiastic students

Enthusiastic students more often than not build an arrangement model that depends on the given preparing information before accepting the information for grouping. It ought to have the option to focus on a solitary theory that will cover the whole example space there is. Because of this model development, energetic students as a rule set aside a long effort for train and extremely less time to anticipate.

Models. Choice Tree, Naive Bayes, Artificial Neural Networks

Various Types Of Classifier Used

- **K-nearestneighbour**

The k-nearestneighbour calculation otherwise called k-NN is a non-parametric strategy or approach utilized for characterization and relapse. In both the instances of characterization and relapse the info comprises of the k nearest preparing models that are available in the element space. The yield it significantly relies upon whether the k-NN is utilized for grouping or it is being utilized for relapse.

In k-NN when it is utilized for arrangement, the yield is commonly a class participation. An article is arranged on the off chance that there is a larger part of vote from its neighbors and, at that point the item being doled out to the class that is most regular among its entire k closest neighbors where k is a positive number commonly little. In the event that k = 1, at that point the item is essentially relegated to the class of its single closest neighbor.
K nearest neighbors is what that stands for is one of the simplest supervised machine learning algorithms mostly used for classification so we want to know is this a dog or it's not a dog is it a cat or not a cat it classifies a data point based on how its neighbors are classified KN in stores all available cases and classifies new cases based on a similarity measure and here we gone from cats and dogs right into wine another favorite of mine KN in stores all available cases and classifies new cases based on a similarity measure and here you see we have a measurement of sulfur dioxide versus the chloride level and then the different wines they've tested and where they fall on that graph based on how much sulfur dioxide and how much chloride k and KN is

a perimeter that refers to the number of nearest neighbors to include in the majority of the voting process and so if we add a new glass of wine there red or white we want to know what the neighbors are in this case we're going to put k equals five and we'll talk about K in just a minute a data point is classified by the majority of votes from its five nearest neighbors here the unknown point would be classified as red since four out of five neighbors are red so how do we choose K how do we know K equals five I mean that's what is the value we put in there and so we can talk about it how do we choose the factor K knn algorithm is based on feature similarity choosing the right value of K is a process called parameter tuning and is important for
better accuracy so at K equals three we can classify we have a question mark in the middle as either a as a square or not is it a square or is it in this case a triangle and so if we set K equals to three we're going to look at the three nearest neighbors we're going to say this is a square and if we put K equals
to seven we classify as a triangle depending on what the other data is around and you can see is the K changed pending on where that point is that drastically changes your answer and we jump here we go how do we choose the factor of K you'll find this in all machine learning choosing these factors that's the face you get he's like oh my gosh you'd say choose the right K did it right my values in whatever machine learning tool you're looking at so that you don't have a huge bias in one direction or the other and in terms of K n n the number of K if you choose it too low the bias is based on it's just too noisy it's right next to a couple things and it's going to pick those things and you might get asked you to answer and if your K is too big then it's going to take forever to process so you're going to run into processing issue sand resource issues so what we do the most common use and there's other options for choosing K is to use the square root of n so it is a total number of values you have you take the square root of it in most cases you also if it's an even number so if you're using like in this case squares and triangles if it's even you want to make your K value odd that helps it select better so in other words you're not going to have a balance between two different factors that are equal so you usually take the square root of n and if it's even you add one to it or subtract one from it and that's where you get the K value from that is the most common use and it's pretty so lid it works very well when do we use KNN we can use K n when data is labeled so you need a label on it we know we have a group of pictures with dogs cats data is noise free and so you can see here when we have a class that we have like underweight 140 23 Hello Kitty normal that's pretty confusing we have a variety of data coming in so it's very noisy and that would cause an issue Dana said is small so we're usually working

with smaller data sets where you might get into gig of data if it's really clean it doesn't have a lot of noise because K and N is a lazy learner i.e. it doesn't learn a discriminative function from the training set so it's very lazy so if you have very complicated data and you have a large amount of it you're not
going to use the KNN but it's really great to get a place to start even with large data you can sort out a small sample and get an idea of what that looks like using the KNN and also just using for smaller data sets works really good how does a knn algorithm work consider a data set having two variables height and centimeters and weight in kilograms and each point is classified as normal or under weight so we can see right here we have two variables you know true/false or either normal or they're not they're underweight on the basis of the given data we have to classify the below set as normal or under weight using KNN so if we have new data coming in this says 57 kilograms and 177 centimeters is that going to be normal or underweight to find the nearest neighbors we'll calculate the Euclidean distance according to the Euclidean distance formula the distance between two points in the plane with the coordinates XY and a B is given by instance D equals the square root of x minus a squared plus y minus B squared and you can remember that from the two edges of a triangle we're computing the third edge since we know the x side and the y side let's calculate it to understand clearly so we have our unknown point and we placed it there in red and we have our other points where the data is scattered around the distance d1is a square root of 170 minus 167squared plus 57 minus 51 squared which is about six point seven and distance two is about 13 and distance three is about 13 point four similarly we will calculate the Euclidean distance of unknown data point from all the points in the data set and because we're dealing with small amount of data that's not that hard to do it's actually pretty quick for a computer.

• Support Vector Machines

SVM is specific to supervised learning machine learning model learns from the past input data and makes future predictions as output so we teach the model we teach it what a strawberry is and once the model is trained it can identify up straw berry that's what's mean by supervised learning in the larger picture of the machine learning model and under supervised learning you can see that the support vector fits in under classification deciding what yes-and-no is and there is also a regression version but it is primarily used for classification let's take a detour and see if we can connect us to the human experience and find out why support vector machine so in this example last week my son and I visited a fruit shop dad is that an apple or a strawberry so the question comes up what fruit do they just pick up from the fruit stand after a couple of seconds you can figure out that it was strawberry so let's take this model a step further and let's why not build a model which can predict an un knowing data and in this we're going to be look in gat some sweet strawberries or crispy apples we want it to be able to label those two and decide what the fruit is and we do that by having data already put in so we already have a bunch of strawberries we know our strawberries and they're already labeled as such we already have a bunch of apples we know our apples and are labeled as such then once we train our model that model then can be given the new data and the new data is this image in this case you can see a question mark on it and it comes through and goes it's a strawberry in this case we're using the support vector machine model SVM is a supervised learning method that looks at data and sorts it into one of two categories and in this case we' resorting the strawberry into the strawberry site at this point you should

be asking the question how does the prediction work before we dig into an example with numbers let's apply this to our fruit scenario we have our support vector machine we've taken it and we taken labeled sample of data strawberries and apples and we drawn a line down the middle between the two groups this split now allows us to take new data in this case an apple and a strawberry and place them in the appropriate group based on which side of the line they fall in and that way we can predict the unknowing as color full and tasty as a fruit example is let's take a look at another example with some numbers involved and we can take a closer look at how the math works in this example we're going to be classifying men and women and we're going to start with a set of people with a different height and a different weight and to make this work we'll have to have a sample data set a female where you have their height weight 174 65 174 88and so on and we'll need a sample dataset of the male they have a height 17991 82 80 and so on let's go ahead and put this on a graph so have a nice visual so you can see here we have two groups based on the height versus the weight and on the left side we're going to have the women on the right side we're going to have the men now if we're going to create a classifier let's add a new data point and figure out if it's male or female so before we can do that we need to split our data first we can split our data by choosing any of these lines in this case we draw in two lines through the data in the middle that separates the men from the women but to predict the gender of a new data point we should split the data in the best possible way and we say the best possible way because this line has a maximum space that separates the two classes here you can see there's a clear split between the two different classes and in this one there's not so much a clear split this doesn't have the maximum space that separates the two that is why this line best splits the data we don't want to just do this by eye balling it and before we go further we need to add some technical terms to this we can also say that the distance between the points and the line should be as far as possible in technical terms we can say the distance between the support vector and the hyper plane should be as far as possible and this is where the support vectors are the extreme points in the data set and if you look at this data set they have circled two points which seem to be right on th eout skirts of the women and one on the outskirts of the min and hyper plane has a maximum distance to the support vectors of any class now you'll see the line down the middle and we call this the hyper plane because when you're dealing with multiple dimensions it's really not just a line but a plane of intersections and you can see here where the support vectors have been drawn in dashed lines the math behind this is very simple we take D plus the shortest distance to the closest positive point which would be on the men's side and D minus is the shortest distance to the closest negative point which is on the women's side the sum of D plus and D minus is called the distance margin or the distance between the two support vectors that are shown in the dashed lines and then by finding the largest distance margin we can get the optimal hyper plane once we've created an optimal hyper plane we can easily see which side the new data fits in and based on the hyper plane we can say the new data point belongs to the male gender hope fully that's clear and how that works on a visual level as a data scientist you should also be asking what happens if the hyper plane is not optimal if we select a hyper plane having low margin then there is a high chance of miss classification this particular SVM model the one we discussed so far is also called or referred to as the lsv so far so clear but a question should becoming up we have our sample data set but instead of looking like this what if it looked like this where we have two sets of data but one of them occurs in the middle of another set you can see here where we have the blue and the yellow and then blue again on the other side of our data line in this data set we can't use a hyper plane so when you see data like this it's necessary to move away from a 1d view of the data to a two-dimensional view of the data and for the transformation we use what's called a kernel function the kernel function will take the 1d input and transfer it to a two-dimensional output as you can see in this picture here the1d when transferred to a two-dimensional makes it very easy to draw a line between the two data sets what if we make it even more complicated how do we perform an SVM for this type of data se there you can see we have at two-dimensional data set where the data is in the middle surrounded by the green data on the outside in this case we're going to segregate the two classes we have our sample data set and if you draw a line through it's obviously not an optimal hyper plane in there so to do that we need to transfer the 2d to a 3darray and when you translate it into a3-dimensional array using the kernel you can see we can place a hyper plane right through it and easily split the data before we start looking at a programming example and dive into the script let's look at the advantage of the support vector machine we'll start with high dimensional input space or sometimes referred to as the curse of dimensionality we looked at earlier one dimension to dimension three dimension when you get to a thousand dimensions a lot of problems start occurring with most algorithms that have to be adjusted for the SVM automatically does it in high dimensional space one of the high dimensional space one high dimensional space that we work on is sparse document vectors this is where we tokenize the words in document so we can run our machine learning algorithms over though I've seen ones get as high as 2.4million different tokens that's a lot of vectors to look at and finally we have regularization parameter the realization parameter or lambda is a parameter that helps figure out whether we're going to have a bias or over fitting of the data whether it's going to be over fitted to a specific instance or is going to be biased to a higher low value with the SVM it naturally avoids the over fitting and bias problems that we see in many other algorithms these three advantages of the support vector machine make it a very powerful tool to add to your repertoire of machine learning tools now we did promise you a use case study we're actually going to dive in to some Python programming and so we're going to go into a problem statement and start off with the zoo so in the zoo example we have family members going to the zoo we have the young child going dead is that a group of crocodiles or alligators well that's hard to differentiate and zoos are a great place to start looking at science and understanding how things work especially as a young child and so we can see the parents in here thinking well what is the difference between a crocodile and an alligator well one crocodiles are larger in size alligators are smaller in size snout width the crocodiles have a narrow snout and alligators have a wider snout and the course in the modern day and age the father's Hittner is thinking how can It turn this into a lesson for my son and e goes let a support vector machine segregate the two groups I don't know if my dad ever told me that but that would be funny now in this example we're not going to use actual measurements and data we're just using that for imagery and hat's very common and a lot of machine learning algorithms and setting them up by let's roll up our sleeves and we'll talk about that more in just a moment as we break into our Python script so here arrive in our actual coding and I'm going to move this into a Python editor in just a moment but let's talk a little bit about what we're going to cover first we're going to cover in the code the setup how to actually create our SVM and you're going to find that there's only two lines of code that actually create it and the rest of it is done so quick and fast that it's all here in the first page and we'll show you what that looks like as far as our data so we're going to create some data I talked about creating data just a minute ago and so we'll get into the creating data here and you'll see this nice correction of our two blobs and we'll go through that in just a second and then the second part is we're going to take this and we're going to bump it up a notch we're going to show you what it looks like behind the scenes but let's start with actually creating our setup I like to use the Anaconda Jupiter notebook because it's very easy to use but you can use any of your favorite Python editors or setups and go in there but let's go ahead and switch over there see what that looks like so here we are in the anaconda Python notebook or anaconda Jupiter notebook with Python we're using Python 3 I believe this is3.5 but it should be work in any of your3x versions and you'd have to look at the SK learn and make sure if you're using a 2x version or an earlier version let's go ahead and put our code in there and one of the things I like about the Jupiter notebook is I go up to view and I'm going to go ahead and toggle the line numbers on to make it a little bit easier to talk about and we can even increase the size because this is that I did in in this case I'm using Google Chrome Explorer and that's how it open up for the editor although anyone any like I said any editor will work now the first step is going to be our imports and we're going to import four different parts the first two I want you to look at our line 1 and line 2 our numpy as NP and matt plot library pie plot as p LT now these are very standardized imports when you're doing work the first one is the numbers python we need that because part of the platform we're using uses out for the numpy array and i'll talk about that in a minute so you can understand why we want to use a numpy array versus a standard python array and normally it's pretty standard setup to use NP for numpy the map plot library is how we're going to view our data so this has do you need the NP for the SK learn module but the map plot library is purely for our use for visualization and so you really don't need that for the SVM but we're going to put it there so you have a nice visual aid and we can show you what it looks like that's really important at the end when you finish everything so you have a nice display for everybody to look at and then finally we're going to I'm going to jump one ahead to line number four that's the SK learned dataset samples generator import and make blobs and told you that we were going to make up data and this is a tool that's in the SK learn to make up data I personally don't want to go to the zoo get in trouble for jumping over the fence and probably beaten by the crocodiles or alligators as I work on measuring their snouts and with them lengths instead we're just going to make up some data and that's what that make blobs is it's a wonderful too lif you're ready to test your setup and you're not sure about what data are going to put in there you can create this blob and it makes it real easy to use and finally we have our actual svm the SK learn import SVM online 3 so that covers all our imports we're going to create remember I use the make blobs to create data and we're going to create a capital X and a lowercase y equals make blobs in samples equals 40 so we're going to make 40 lines of data it's going to have two centers with the random state.

Random Forest Classifier

unfriend enforce the random forest algorithm is one of the most popular and most powerful supervised machine learning algorithm that is capable of performing both regression and classification tasks as the name suggests this algorithm creates the forest with a number of decision trees in general the more trees in the forest the more robust the prediction and thus higher accuracy to model multiple decision trees to create the forest you are going to use the same method of constructing the decision what the information gain or Gini index approach amongst other algorithms if you're not aware of the concepts of decision tree classifier please check out my other lecture on intuition tree God for machine learning you'll need to know how the decision tree classifier works before you can learn the working nature of the random forest algorithm so how does it work in random forests we grow multiple trees as opposed to a single tree in court model to classify a new object based on attributes each tree gives a classification and we save the tree votes for that class the forest choose the classification having the most votes over all the other trees in the forest and in the case of regression takes the average of the outputs by different trees so let's look at the advantages of the random forest so the same random forest algorithm or random forest classifier can be used for both classification and regression tasks then enforce classifier will handle the missing values and maintain accuracy when a large proportion of the data are missing when we have more trees in the forest random classifies won't over fit the model it has the power to handle large data sets with higher dimensionality if you look at the disadvantages of rainforests however it surely does a good job at classification but it's not as good as for regression problems as it does not give precise continuous nature predictions in the case of regression it doesn't predict beyond the range train data and they may over fit data sets that are particularly noisy random phones can feel like a black box approach for statistical models you have very little control of what the model does you can try at best try different parameters and random seeds so what are the applications our friend Forrest so let's check a few of them down below so we can use them in the banking sector so these are for finding loyal customers and finding the fraud customers it can be using medicine where we identify the correct combination of components to validate listen learn enforce algorithms also help for identifying disease by analyzing the patient's medical records in the stock market random forest algorithm is used to identify the stock behavior as well as the expected loss or profit by purchasing a particular stock in e-commerce the rain forest is used in a small segment of the recommendation engine for identifying the likelihood of a customer liking the recommended products and this is based on some Lacan's customers in computer vision the random forest is used for image classification Microsoft have used random forests for body parts classification for Xbox Kinect and other applications involves lip-reading as well as for its classification let's take a look at the random forest serial code and how it works so it works in the following manner where each tree is planted and grown as follows so assume announced cases in the training set is in then the sample of these end cases is taken at random but what replacement the sample will be the training set for growing the tree if they are M input variables or features a number of M smaller than M is specified such that each node M variables are selected at random out of DM the best list on these M input variables is used to split the node the value of M is held constant while we grow the forest each tree is grown to the largest extent possible and is no pruning and then we predict data by aggregating that protections of the entries which means majority vote for classification and average for regression so to perform the predictions using the Train random forest algorithm we need to pass the test features through the rules of each randomly created trees suppose let's say we formed a thousand random decision trees deform the random forest say we're taking if an image contains a hand each random forest will predict a different outcome or class for the same test feature it small subset of the forest look at a random set of features for example a finger suppose 100 random decision trees predict some three unique targets such as a finger thumb or maybe the meal then the votes of finger is tell it out of hundreds and M decisions and likewise for the other two targets if finger is getting highest votes then the final random forest returns the finger as its predicted target this concept of voting is known as majority voting to select elections the same applies to the rest of the fingers of the hand if the algorithm predicts the rest of the fingers to be fingers then the high-level decision tree can vote that the image is a hand and this is why random forests are known as ensemble machine learning algorithm ensembles are a divide-and-conquer approach used to improve performance the main principle behind ensemble methods is that a group of weak learners can come together to form a strong learner each classifier is individually a weak learner while the classifiers taking together are a strong learner and thus ensemble methods reduce the variance and improve performance before we end the lecture let's take a look at some terms and definitions that you might come across such as guiding and boosting bootstrap aggregating is also known as baggy which is a machine learning ensemble meta algorithm designed to improve the stability and accuracy of the machine learning algorithms used in physical classification and progression it also reduces variance and helps to avoid over fitting boosting is a machine learning ensemble after algorithm for primarily reducing bias and also fails in supervised learning and a family of machine learning algorithms which also convert weak learners into strong ones algorithms that achieve hypothesis boosting quickly becomes simply known as boosting .

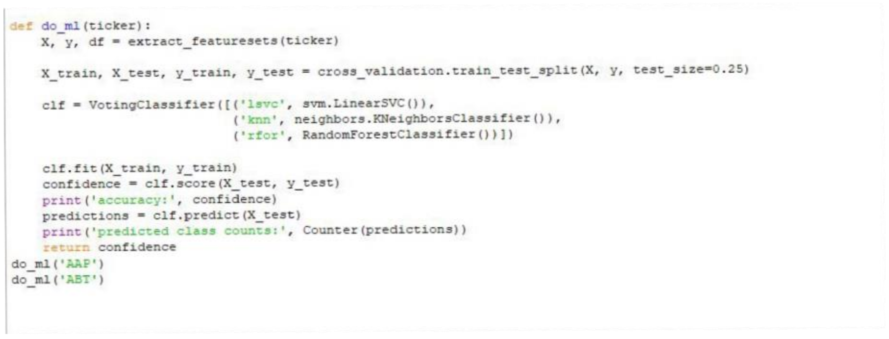
Performing Machine Learning Against Companies

Sk learn is a machine learning framework. The svm is being imported for a Support Vector Machine classifier, cross_validation will easily let us create shuffled training and testing samples for classification . The neighbours that is imported is for K Nearest Neighbors Algorithm. Then, there is also a Voting Classifier and Random Forest Classifier. The voting classifier is just what it sounds like, basically, it's a classifier that will combine many classifiers, and allow them to each get a "vote" on what they think the class of the feature sets is. The random forest classifier is just another classifier. Used these three classifiers in the voting classifier. Now, the feature sets and labels are available, and it’s time to shuffle them, train, and then test.

What this does (refer fig 3.4.1), is shuffle our data (so its not in any specific order any more), and then create training and testing samples for prediction. Don’t try to "test" this algorithm on the same data that is being trained against. If that happens, chances are it would do a lot better than it would be in reality. Hence, test the algorithm on data that it has never seen before to see if there is actually a model that works.

The line clf.fit() will take the X data, and fit to the y data, for each of the pairs of X's and y's that are present. Once that's done, it can be tested.

If this model is indeed successful, this can be saved with pickle, and can be loaded at any time to feed it some feature sets and get a prediction out of it, with clf. Predict which will predict either a single value from a single feature set, or a list of values from a list of feature sets.



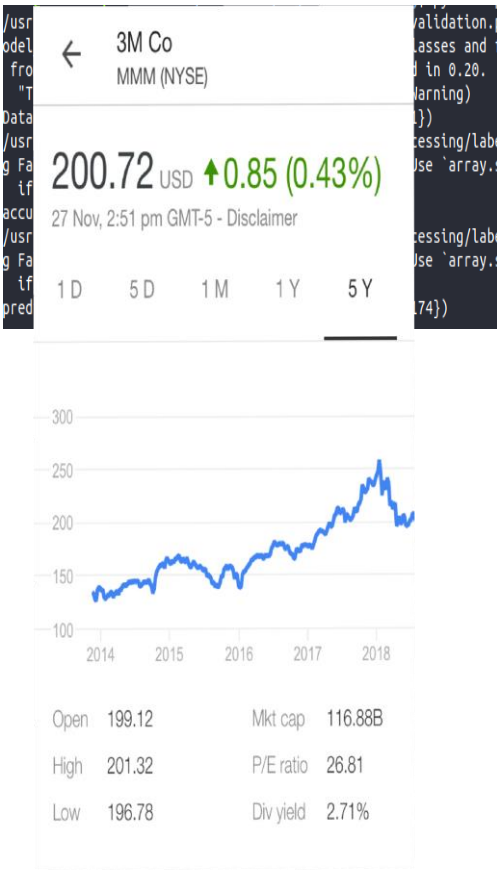
```python
def do_ml(ticker):
    X, y, df = extract_featuresets(ticker)

    X_train, X_test, y_train, y_test = cross_validation.train_test_split(X, y, test_size=0.25)

    clf = VotingClassifier([('lsvc', svm.LinearSVC()),
                            ('knn', neighbors.KNeighborsClassifier()),
                            ('rfor', RandomForestClassifier())])

    clf.fit(X_train, y_train)
    confidence = clf.score(X_test, y_test)
    print('accuracy:', confidence)
    predictions = clf.predict(X_test)
    print('predicted class counts:', Counter(predictions))
    return confidence
do_ml('AAP')
do_ml('ABT')
```

Fig 17: Code of implementing Classifiers and performing Machine learning.

Chapter 4

Results and Outputs

As it can be grasped in the figure given underneath, one side it demonstrations the forecast counter spread of the company future prices, and additional figure demonstrations the graph of the company at that particular time of year in terms to the forecast and it can be detected that much of the outcomes are precise. As it can be perceived that the data spread is habitually saying buy the stock, it can be incorrect on the hold condition because the teaching data will never be perfectly stable ever, so supposedly if the model forecast buy then this would be 1722 correct out of 4527 which is still good and actually a better score than it attained, and it still is getting the above accuracy mark of 33% which is decent in a stock market analysis. Many situations will static be there which machines can miss out, supposedly this has circumstances to buy, sell, hold and sometimes the model can be penalised, say the model predictable a 2% rise in the following seven days, but the growth only went up to 1.5% and departed 2% the next day, then the model will forecast (buy, hold) rendering to the 1.5% rise in the seven days and give the predictable spread. A model can also be penalised if supposedly the growth went 2% up and then suddenly falls 2% short the next day, this sort of outcomes in real trading would be thoughtful and same goes for the classical of it turns out to be highly precise. Now observing at the spread and the graph of the company notice around the era of 2017 the company was growing in the market so therefore there were actually more buys, which rapidly fallen in 2018, but the data we mined was till 31, December 2017 and it displays that at the starting of the year it had lot of buys, hence 1722 out of 4527 which speedily was sold just in a tiny time hence a lot of sells more than the holds, giving 1424 out 4527, the model may not be perfectly accurate but has a very close range of decisions which can be accepted in real trading or using algorithms to trade.



```
/usr                                              validation.
odel                                              asses and
 fro                                              in 0.20.
  "T                                              Warning)
Data                                              })
/usr                                              essing/labe
g Fa                                              Use `array.
  if
accu
/usr                                              essing/labe
g Fa                                              Use `array.
  if
pred                                              74})
```

3M Co
MMM (NYSE)

200.72 USD ↑0.85 (0.43%)
27 Nov, 2:51 pm GMT-5 - Disclaimer

Open	199.12	Mkt cap	116.88B
High	201.32	P/E ratio	26.81
Low	196.78	Div yield	2.71%

Fig 18: Output data spread and predicted spread

Chapter 5
Conclusions and Recommendations

Hereby, it can be proposed that no trading algorithm can be 100% effective, not only 100%, it will typically never be close to 70% but to attain even an accuracy of 40% or 35% is still good sufficient to get a good forecast spread. Although extreme attained accurateness was 39%, it was still able to closely forecast the predictable outcome and have coordinated against the company graph. To make our expectation more efficient, it can be done by including bulky data sets that have millions of entries and could train the machine more powerfully. Different activities of stocks can lead to diverse raises or lows in the forecast price, use these movements to magistrate whether a company should be traded in or not. No training Data can ever be stable, hence there are always some unevenness which can be seen in the above data spread, but to still forecast close to an consequence will also lead to a good approach if it has greater than 33% accuracy. While, developing a strategy trader should always think to always have nominal imbalance while still being above 33%accurate.

It can also be determined that in a stock market, there is probable that some companies might not be associated at all, and mostly can be associated to each other, and can help justice movements of stock accordingly, we can scale affairs and see how much in percentages they are correlated.

Including gigantic data sets, to increase more effectiveness, and in data set if had nan values in tables, because of two simple reasons either a specific company wasn’t opened during that time of year, or the data is not readily obtainable, in both the cases replace the null values with 0 , which is somewhat that trader might want to change while developing a trading tactic.

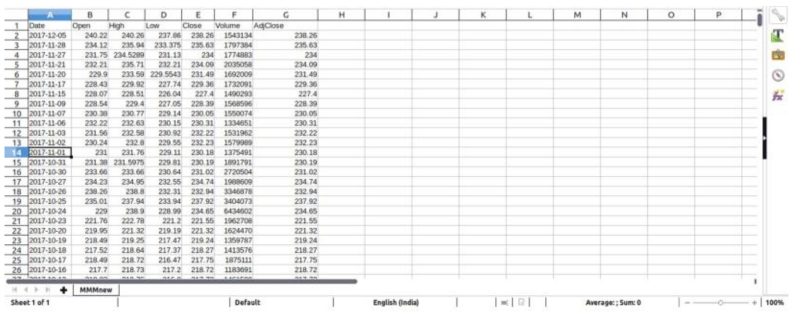
Further more, there can be back testing of the trading strategy, using zip line and quantopian a python platform for testing trading strategies and can see how well can a model fit into some random data of stock, and can the model from this random data of stock develop relations and correlations, and predict on terms of change.

References

- https://www.researchgate.net/publication/
- http://cs229.stanford.edu/proj2017/final-reports/5234854.pdf
- https://pythonprogramming.net
- https://pypi.org/project/pandas/
- https://matplotlib.org
- https://www.google.co.in/amp/s/www.geeksforgeeks.org/numpy-in-python-set-1-introduction/amp/
- https://pypi.org/project/beautifulsoup4/

Appendix (Data Set Snap Shot)

MMM data setss



	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Volume	AdjClose
2	2017-12-05	240.22	240.26	237.86	238.26	1543134	238.26
3	2017-11-28	234.12	235.94	233.375	235.63	1797384	235.63
4	2017-11-27	231.75	234.5289	231.13	234	1774883	234
5	2017-11-21	232.21	235.71	232.21	234.09	2035058	234.09
6	2017-11-20	229.9	233.59	229.5543	231.49	1692009	231.49
7	2017-11-17	228.43	229.92	227.74	229.36	1732091	229.36
8	2017-11-15	228.07	228.51	226.04	227.4	1490293	227.4
9	2017-11-09	228.54	229.4	227.05	228.39	1568596	228.39
10	2017-11-07	230.38	230.77	229.14	230.05	1550074	230.05
11	2017-11-06	232.22	232.63	230.15	230.31	1334651	230.31
12	2017-11-03	231.56	232.58	230.92	232.22	1531962	232.22
13	2017-11-02	230.24	232.8	229.55	232.23	1579989	232.23
14	2017-11-01	231	231.76	229.11	230.18	1375491	230.18
15	2017-10-31	231.38	231.5975	229.81	230.19	1891791	230.19
16	2017-10-30	233.66	233.66	230.64	231.02	2720504	231.02
17	2017-10-27	234.23	234.95	232.55	234.74	1988609	234.74
18	2017-10-26	238.26	238.8	232.31	232.94	3346878	232.94
19	2017-10-25	235.01	237.94	233.94	237.92	3404073	237.92
20	2017-10-24	229	238.9	228.99	234.65	6434602	234.65
21	2017-10-23	221.76	222.78	221.2	221.55	1962708	221.55
22	2017-10-20	219.95	221.32	219.19	221.32	1624470	221.32
23	2017-10-19	218.49	219.25	217.47	219.24	1359787	219.24
24	2017-10-18	217.52	218.64	217.37	218.27	1413576	218.27
25	2017-10-17	218.49	218.72	216.47	217.75	1875111	217.75
26	2017-10-16	217.7	218.73	217.2	218.72	1183691	218.72