Final Project (24.3)

Stock Predictor – Price & Media

Northwestern

Agenda and Team Members

- Brian Young
- Carl Coffman
- Push Palani
- Steven Crutcher

Final Project - Stocks

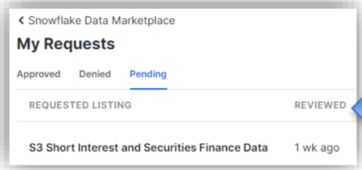
Project Scope

- Predict future stock price and how it may be influenced by social media noise – such as what had happened recently to GameStop and AMC.

Site Hosts

- Stock Predictor (ccc-gh.github.io)
  https://ccc-gh.github.io/FinalProject-Stock/

- GitHub Code Repository
  https://github.com/CCC-GH/FinalProject-Stock



‹ Snowflake Data Marketplace

My Requests

Approved Denied Pending

REQUESTED LISTING	REVIEWED
S3 Short Interest and Securities Finance Data	1 wk ago

Key Data Sources

Used:

- yfinance - # Yahoo! Finance market data
  $ pip install yfinance
  import yfinance

- pandas-datareader – Tweet data feed
  $pip install pandas-datareader
  import pandas_datareader

- Reddit – user-generated bulletins
  https://www.reddit.com/r/datasets/

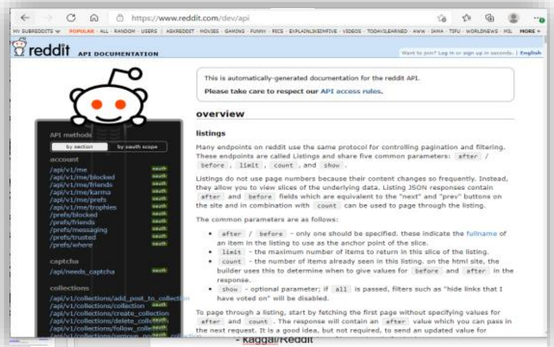
- Kaggle – WallStreetBest posts
  https://www.kaggle.com/datasets

Reviewed:

- snowflake – great data cloud connectivity store/retrieve, but no access given to stock-shorting data.

Social Media and Cloud Data

Reddit – https://www.reddit.com/r/datasets/


https://www.reddit.com/dev/api
MY SUBREDDITS · POPULAR · ALL · RANDOM · USERS | ASKREDDIT · MOVIES · GAMING · FUNNY · PICS · EXPLAINLIKEIMFIVE · VIDEOS · TODAYILEARNED · AWW · IAMA · TIFU · WORLDNEWS · MIL · MORE »
reddit API DOCUMENTATION
Want to join? Log in or sign up in seconds. | English
API methods
by section
by oauth scope

account

/api/v1/me oauth
/api/v1/me/blocked oauth
/api/v1/me/friends oauth
/api/v1/me/karma oauth
/api/v1/me/prefs oauth
/api/v1/me/trophies oauth
/prefs/blocked oauth
/prefs/friends oauth
/prefs/messaging oauth
/prefs/trusted oauth
/prefs/where oauth

captcha

/api/needs_captcha oauth

collections

/api/v1/collections/add_post_to_collection
/api/v1/collections/collection oauth
/api/v1/collections/create_collection
/api/v1/collections/delete_collection oauth
/api/v1/collections/follow_collection oauth

This is automatically-generated documentation for the reddit API.

Please take care to respect our API access rules.

overview

listings

Many endpoints on reddit use the same protocol for controlling pagination and filtering. These endpoints are called Listings and share five common parameters: `after` / `before`, `limit`, `count`, and `show`.

Listings do not use page numbers because their content changes so frequently. Instead, they allow you to view slices of the underlying data. Listing JSON responses contain `after` and `before` fields which are equivalent to the "next" and "prev" buttons on the site and in combination with `count` can be used to page through the listing.

The common parameters are as follows:

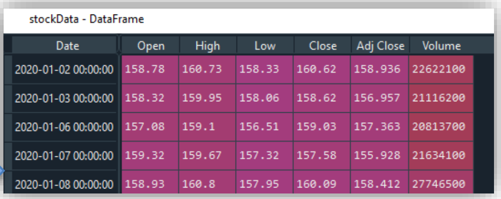
- `after` / `before` - only one should be specified. these indicate the fullname of an item in the listing to use as the anchor point of the slice.
- `limit` - the maximum number of items to return in this slice of the listing.
- `count` - the number of items already seen in this listing. on the html site, the builder uses this to determine when to give values for `before` and `after` in the response.
- `show` - optional parameter; if `all` is passed, filters such as "hide links that I have voted on" will be disabled.

To page through a listing, start by fetching the first page without specifying values for `after` and `count`. The response will contain an `after` value which you can pass in the next request. It is a good idea, but not required, to send an updated value for

- kaggal/Reddit

Data Challenges

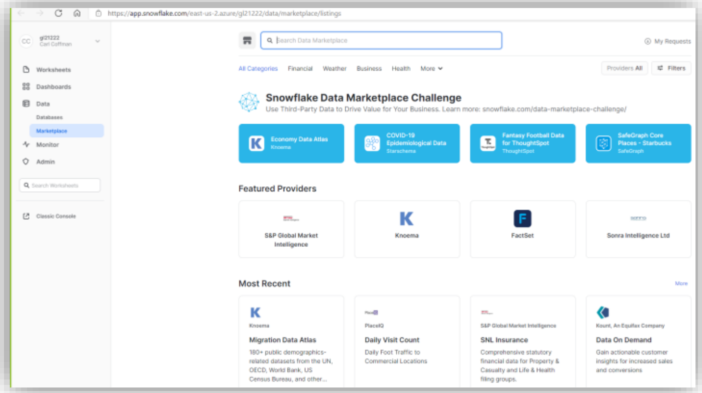
- No significant, historical warehouse for social media data.
- Finding "free" historical data source for a particular stock's short activity – current (.info) and basic-historical is made available; Open, High, Low, Close, Volume



stockData - DataFrame

Date	Open	High	Low	Close	Adj Close	Volume
2020-01-02 00:00:00	158.78	160.73	158.33	160.62	158.936	22622100
2020-01-03 00:00:00	158.32	159.95	158.06	158.62	156.957	21116200
2020-01-06 00:00:00	157.08	159.1	156.51	159.03	157.363	20813700
2020-01-07 00:00:00	159.32	159.67	157.32	157.58	155.928	21634100
2020-01-08 00:00:00	158.93	160.8	157.95	160.09	158.412	27746500

snowflake – Cloud Data/warehouse & Market Data



Snowflake Data Marketplace Challenge

Use Third-Party Data to Drive Value for Your Business. Learn more: snowflake.com/data-marketplace-challenge/

Economy Data Atlas Knoema	COVID-19 Epidemiological Data Starschema	Fantasy Football Data for ThoughtSpot ThoughtSpot	SafeGraph Core Places - Starbucks SafeGraph

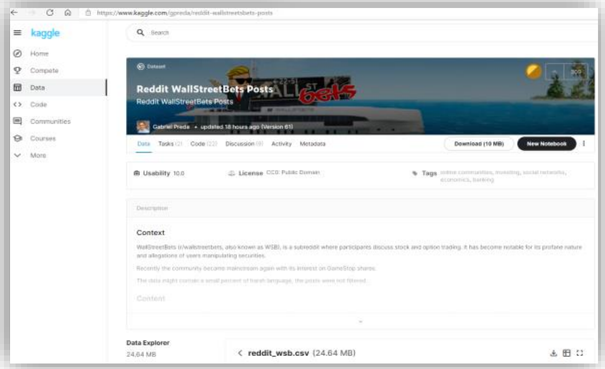
Featured Providers

S&P Global Market Intelligence	Knoema	FactSet	Sonra Intelligence Ltd

Most Recent More

Knoema **Migration Data Atlas** 180+ public demographics-related datasets from the UN, OECD, World Bank, US Census Bureau, and other...	PlaceIQ **Daily Visit Count** Daily Foot Traffic to Commercial Locations	S&P Global Market Intelligence **SNL Insurance** Comprehensive statutory financial data for Property & Casualty and Life & Health filing groups.	Kount, An Equifax Company **Data On Demand** Gain actionable customer insights for increased sales and conversions

Kaggle –
WallStreetBest posts



Reddit WallStreetBets Posts

Reddit WallStreetBets Posts

Gabriel Preda • updated 18 hours ago (Version 61)

Usability 10.0 License CC0: Public Domain Tags online communities, investing, social networks, economics, banking

Description

Context

WallStreetBets (r/wallstreetbets, also known as WSB), is a subreddit where participants discuss stock and option trading. It has become notable for its profane nature and allegations of users manipulating securities.

Recently the community became mainstream again with its interest on GameStop shares.

The data might contain a small percent of harsh language, the posts were not filtered.

Content

Data Explorer
24.64 MB

reddit_wsb.csv (24.64 MB)

Libraries Used

from sklearn:

- linear_model import LinearRegression
- tree import DecisionTreeRegressor
- model_selection import train_test_split
- metrics import mean_squared_error

***ARIMA** - A popular and widely used statistical method for **time series forecasting**.

- from statsmodels.tsa.arima_model import ARIMA

Other:

- import pandas as pd
- import numpy as np
- import matplotlib.pyplot as plt
- from pandas.plotting import lag_plot
- from pandas.tseries.holiday import USFederalHolidayCalendar
- from pandas.tseries.offsets import CustomBusinessDay
- from pandas_datareader.data import DataReader
- import yfinance as yf
- import datetime

Linear Regression - Time Series Forecasting

***ARIMA -** AutoRegressive Integrated Moving Average:

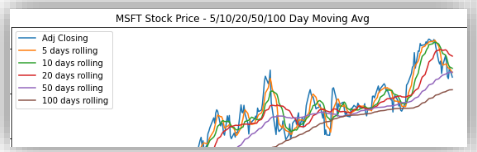
- **AR:<Auto Regressive>** uses the dependent relationship between an observation and some predefined number of lagged observations (also known as “time lag” or “lag”)
- **I:< Integrated >** model employs differencing of raw observations (e.g. it subtracts an observation from an observation at the previous time step) in order to make the time-series stationary
- ***MA: < Moving Average >*** model exploits the relationship between the residual error and the observations

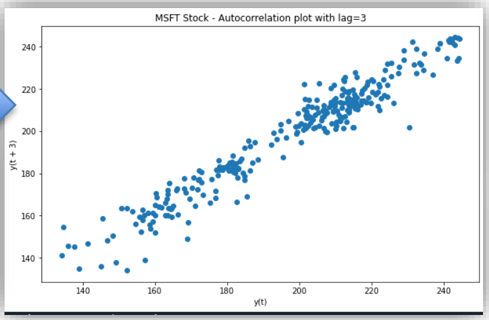
Model parameters - expect as input parameters 3 arguments (p,d,q)

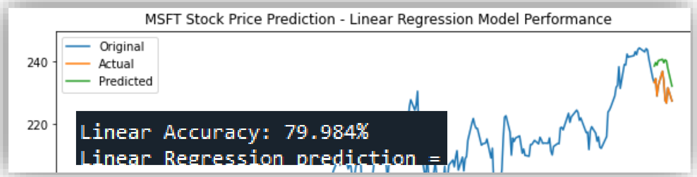
- **p** is the number of lag observations *(using 4 lags)*
- **d** is the degree of differencing *(not-stationary, days, so placed “1” in model)*
- **q** is the size/width of the moving average window

ARIMA / Linear Regression

ARIMA (or Box-Jenkins method) is a good model to be applied to this type of data (there is auto-correlation in the data).

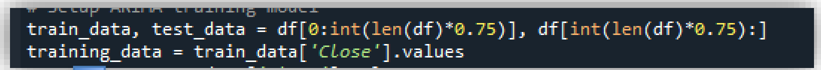






```
Linear Accuracy: 79.984%
Linear Regression prediction =
```

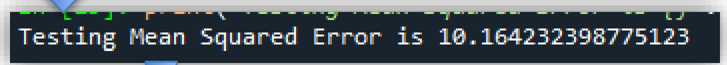
ARIMA Test - 75% data split test model to train/fit model *10.2 MSE



```
# setup ARIMA training model
train_data, test_data = df[0:int(len(df)*0.75)], df[int(len(df)*0.75):]
training_data = train_data['Close'].values
```

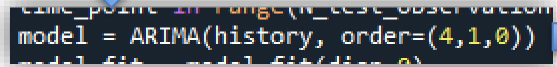
ARIMA MSE - *10.2 MSE

**note: avg squared value across all the test set predictions*

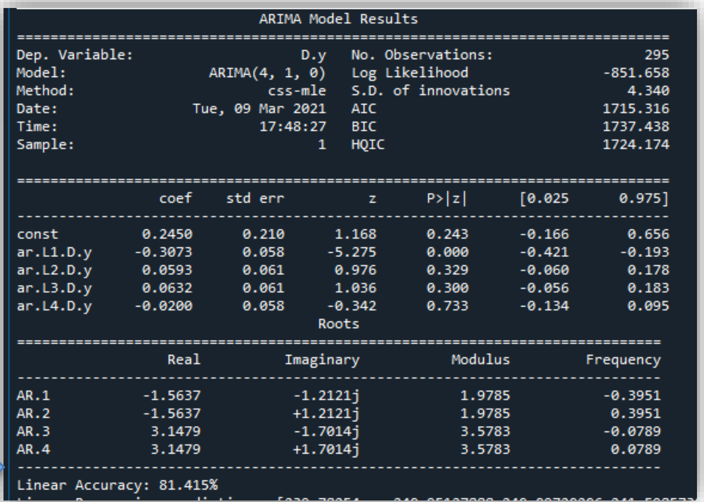


```
Testing Mean Squared Error is 10.164232398775123
```

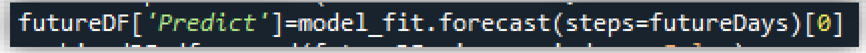
order: p=4 lags, d=1 (not stationary)



```
time_point in range(N_test_observation
model = ARIMA(history, order=(4,1,0))
model_fit = model.fit(disp=0)
```

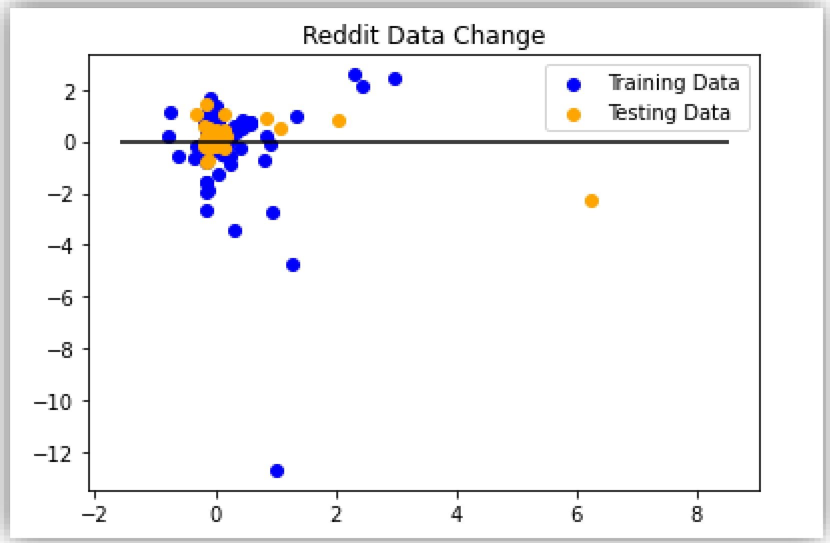


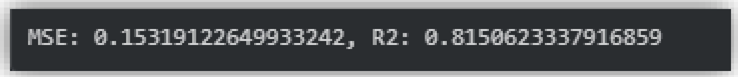
```
                             ARIMA Model Results
==============================================================================
Dep. Variable:                    D.y   No. Observations:                  295
Model:                 ARIMA(4, 1, 0)   Log Likelihood                -851.658
Method:                       css-mle   S.D. of innovations              4.340
Date:                Tue, 09 Mar 2021   AIC                           1715.316
Time:                        17:48:27   BIC                           1737.438
Sample:                             1   HQIC                          1724.174

==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2450      0.210      1.168      0.243      -0.166       0.656
ar.L1.D.y     -0.3073      0.058     -5.275      0.000      -0.421      -0.193
ar.L2.D.y      0.0593      0.061      0.976      0.329      -0.060       0.178
ar.L3.D.y      0.0632      0.061      1.036      0.300      -0.056       0.183
ar.L4.D.y     -0.0200      0.058     -0.342      0.733      -0.134       0.095
                                    Roots
==============================================================================
                  Real          Imaginary           Modulus         Frequency
------------------------------------------------------------------------------
AR.1           -1.5637           -1.2121j            1.9785           -0.3951
AR.2           -1.5637           +1.2121j            1.9785            0.3951
AR.3            3.1479           -1.7014j            3.5783           -0.0789
AR.4            3.1479           +1.7014j            3.5783            0.0789
------------------------------------------------------------------------------
Linear Accuracy: 81.415%
```



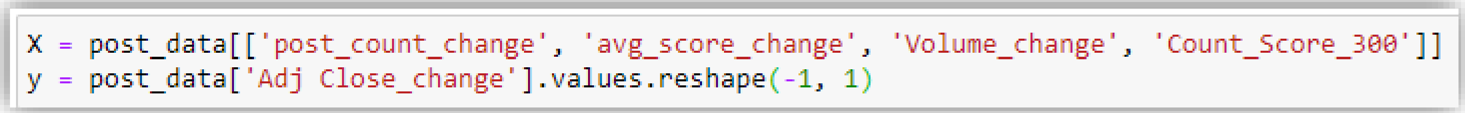
```
futureDF['Predict']=model_fit.forecast(steps=futureDays)[0]
```

Reddit Data



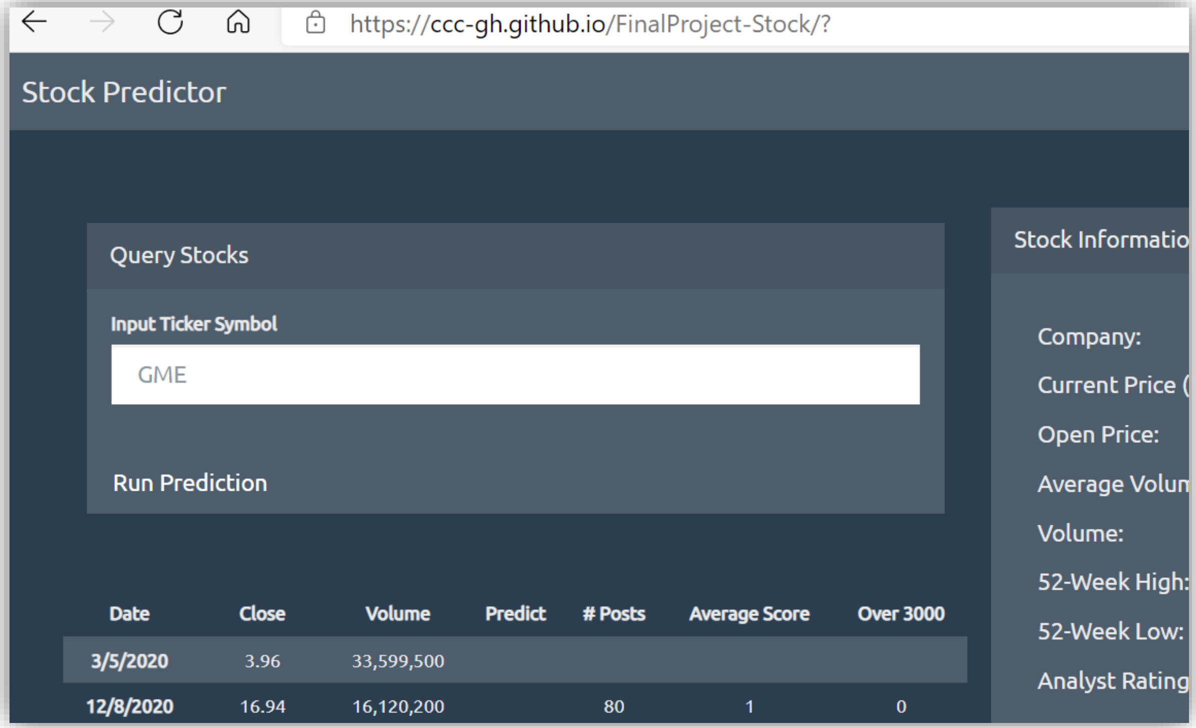


```
MSE: 0.15319122649933242, R2: 0.8150623337916859
```



```
X = post_data[['post_count_change', 'avg_score_change', 'Volume_change', 'Count_Score_300']]
y = post_data['Adj Close_change'].values.reshape(-1, 1)
```

Stock Predictor Web Page



Company:

Current Price (

Open Price:

Average Volum

Volume:

52-Week High:

52-Week Low:

Analyst Rating

Date	Close	Volume	Predict	# Posts	Average Score	Over 3000
3/5/2020	3.96	33,599,500				
12/8/2020	16.94	16,120,200		80	1	0

Potential Next Steps

- Additional historical stock data such as short activity to focus on potential GameStop(GME) or AMC activity.
- If data can’t be found, warehouse daily to build missing data – screen-scaping(beautiful-soup) is an option from sites such as shortsqueeze and/or Daily Short Sale Volume.
- Further investigation of ARIMA model – test/track various P/D/Q configs
- Reduce time grain to hourly, minute for automated trading/prescription reporting (take automatic action on report).
- Build additional charts and dashboards to track multiple stocks.

QUESTIONS?