A drawing of a cartoon character

Description automatically generated

Kent Appiah, Seán Hayes & Jeff Levesque

https://github.com/jeff1evesque/ist-736 | Final Project

IST-736: Market Sentiment

PRofessor gates

Table of Contents

[Introduction 2](#_Toc11272572)

[Analysis 2](#_Toc11272573)

[Data Preparation 2](#_Toc11272574)

[Twitter API 2](#_Toc11272575)

[Quandl API 3](#_Toc11272576)

[Joining Data 4](#_Toc11272577)

[Exploratory 4](#_Toc11272578)

[Word Clouds 4](#_Toc11272579)

[Sentiment Analysis 4](#_Toc11272580)

[Stop Words 4](#_Toc11272581)

[Baseline Results 4](#_Toc11272582)

[Time series 4](#_Toc11272583)

[Granger Causality 5](#_Toc11272584)

[Classification 5](#_Toc11272585)

[Results 5](#_Toc11272586)

[Conclusions 5](#_Toc11272587)

# Introduction

# Analysis

# Data Preparation

## Twitter API

The Twitter API[[1]](#footnote-1) was implemented using an approved Twitter developer[[2]](#footnote-2) account. A python template file (config--TEMPLATE.py) was created containing dummy text representing the secret key and tokens provided with the Twitter developer account. This file was copied as config.py, with values properly substituted. Changes were a requirement of the general application implementing the Twython[[3]](#footnote-3) package. Within the application, two main twitter functionalities were streamlined. The first allowed general querying through a set of parameters[[4]](#footnote-4), while the second allowed querying content for specified twitter screen names. This functionality was provided using the user timeline component[[5]](#footnote-5).

Five screen names were queried:

* Jimcramer
* ReformedBroker
* TheStalwart
* LizAnnSonders
* SJosephBurns

Corresponding code generated dataframe structures, for each of the above screen names, then outputted to an associated csv file[[6]](#footnote-6). On future executions, if the corresponding csv file already exists, then the twitter api did not duplicate exiting files.

Furthermore, the parameters collected from the twitter accounts were screen\_name, created\_at, and full\_text. Each account was collected using a rate\_limit=900[[7]](#footnote-7). This ensured that the maximum number of tweets could be collected per screen name. However, due to the request limit, roughly 15 minutes needed to transpire before re-executing, to obtain the maximum content for the successive screen name. Thus, a little over 1.5 hours was required to initially generate local csv files.

Finally, a default start\_date = datetime(3000, 12, 25) and end\_date = datetime(1000, 12, 25) was defined. This definition was created to represent the datetime range for a given twitter screen name. Specifically, the initial start\_date was compared to each tweet for a given user. If a tweet exists with an earlier datetime, this was set as the new start\_date. This type of logic was extended similarly for the end\_date. This maximized value allows the functional tweet domain to accurately map to the quandl historical range.

## Quandl API

Like the Twitter API, the python Quandl API[[8]](#footnote-8) was utilized to acquire market data, including the Nasdaq index. An account was needed to obtain the associated API key, and the same config.py was utilized, respectively. Moreover, the date range was maximized[[9]](#footnote-9) in order to obtain the largest possible dataset. While obtaining data was not as restricted by the same rate limit as Twitter, a local csv file was created. This ensures integrity and optimization in case a future study extends with additional datasets. While five different columns were returned, only the Index Value was utilized for successive calculations, described in sections below:

* Trade Date
* Index Value
* High
* Low
* Total Market Value
* Dividend Market Value

Finally, as described earlier, market data range, was predicated on the maximized tweet domain.

|  |
| --- |
|  |
| Figure 1: domain mapping from Twitter API (X) and Quandl Data f(x)[[10]](#footnote-10). |

## Joining Data

To simplify processing, tweets were aggregated by created\_at and screen\_name. If an account tweeted multiple times a given day, each full\_text instance was concatenated to a single string. This allowed sentiment measure to be computed as a time series. Furthermore, each twitter account data was merged on Trade Date = created\_at column. Moreover, later classification tasks were predicated on comparing the current day market value with the previous day. Thus, some edge cases needed to be considered:

1. If the first Index Value is nan, drop the instance
2. If successive (n+1) index has a previous step nan, skip and do nothing
3. If successive (n+1) index is nan, set market values to previous day and concatenate current full\_text with previous day.

Additionally, if a given day contained an empty string for full\_text, this instance was dropped, and the dataframe index was reset.

# Exploratory

Initial exploration was performed for each twitter screen name, and overall aggregation. Specifically, word clouds, vader sentiment measures, and topic modeling was determined for each twitter screen name. Finally, word clouds and sentiment measures were repeated on the overall dataset.

## Word Clouds

## Sentiment Analysis

## Stop Words

A rich set of stop words was created[[11]](#footnote-11), utilized prior to vectorization:

# Baseline Results

## Time series

## Granger Causality

## Classification

# Results

# Conclusions

1. <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets> [↑](#footnote-ref-1)
2. <https://developer.twitter.com/en/apps> [↑](#footnote-ref-2)
3. <https://twython.readthedocs.io/en/latest/> [↑](#footnote-ref-3)
4. <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.html> [↑](#footnote-ref-4)
5. <https://developer.twitter.com/en/docs/tweets/timelines/api-reference/get-statuses-user_timeline.html> [↑](#footnote-ref-5)
6. <https://github.com/jeff1evesque/ist-736-hw/tree/master/data> [↑](#footnote-ref-6)
7. <https://developer.twitter.com/en/docs/basics/rate-limiting.html> [↑](#footnote-ref-7)
8. <https://www.quandl.com/tools/python> [↑](#footnote-ref-8)
9. <https://github.com/jeff1evesque/ist-736/blob/9652d7aa79dc576ca5ad671effbb76362beaa72a/app.py#L227> [↑](#footnote-ref-9)
10. <https://en.wikipedia.org/wiki/Domain_of_a_function> [↑](#footnote-ref-10)
11. <https://github.com/jeff1evesque/ist-736/blob/master/utility/stopwords.py> [↑](#footnote-ref-11)