Pandemic Pandemonium: An Examination of the Economic and Political Effects of COVID-19 on the US

Aaron Carmichael

Oklahoma State University

Table of Contents

Table of Figures

I. Abstract

During the writing of this paper, the world, and more specifically, the United States, is facing an unprecedented set of circumstances due to the outbreak of the SARS-CoV-2 virus, also known as the coronavirus. This pandemic has led to many new measures being put into place, including social distancing measures, shelter-in-place orders, and mask mandates. As these efforts have been made and the case numbers and death toll have risen, it has had severe negative effects across the nation, especially on the economy. The communication services, consumer cyclical, healthcare, and industrials sectors have all seen varying degrees of impact, but for the most part, they have recovered after the initial economic impact of the pandemic on their average adjusted closing stock prices. During these times, the eyes of the nation have also consistently been on federal politicians. The increased popularity of Twitter over the past several years has led to an increased political presence on the platform, allowing politicians to quickly post information and opinions on current events. A sentiment analysis of tweets spanning from February 2020 until the end of June from 11 politicians representing both major political parties revealed no trends that were initially obvious, but did indicate a slight negative trend over time from both parties. Considering the negative sentiment trends and the overall stock market recovery, this paper determines that there is no correlation between political sentiment and the pandemic or stock market behavior.

II. Introduction

Back in November of 2019, the first cases of SARS-CoV-2, colloquially known as the Coronavirus, appeared in Wuhan, China. (Ma, 2020) As of July 31, 2020, the United States alone has approximately 4.4 million active cases and 150,000 deaths (CDC, 2020), and these numbers have grown to their current totals in the span of just over six months, as the first US case appeared on January 21st, 2020. (Schumaker, 2020) During this time, the response to the virus has primarily been left to state governments, and these responses have been different across the nation. However, many states have seen measures such as mask mandates, social distancing implementations, and shelter-in-place orders. As a result of this disease and the responses, the US has seen a record spike in unemployment, with a peak at 14.7%, a rate not seen since the Great Depression. (Rugaber, 2020) These measures have greatly disrupted daily life, and have wreaked havoc on not just the US economy, but economies around the globe as well. Due to businesses not being able to operate normally, the US stock market saw abysmal drops as the thoughts of a recession became more real. Around March 17th, the Dow Jones Industrial Average dropped 12.9%, the S&P 500 dropped 11.9%, and the Nasdaq dropped 12.3%, with all three markets eventually ending 25% lower from their highs. (BBC, 2020)

During times of national emergencies, people often turn to their leaders for reassurance, inspiration, or guidance. The rise of social media in recent years has led to most US politicians having an increased online presence. More specifically, Twitter has become a popular platform for quick sharing of information by anyone, including politicians at all levels. During this pandemic and related quarantines, social media usage increased, with survey results indicating that the majority of respondents have increased their time spent on YouTube and Facebook, with other sites such as Instagram, Twitter, and Pinterest having increased usage as well. (Clement, 2020)

The purpose of this paper is to take a deeper look at the economic impact of the Coronavirus outbreak in the United States, as well as to take a deeper look into political communication during this time, and finally, to examine if there is any apparent correlation between the two. In order to do this, stock prices from four different stock sectors have been collected and visualized to see the impact over time. These sectors are communications, consumer cyclical, healthcare, and industrials. As businesses and education were forced to move operations online, and as social media and media streaming use increased, the communications sector may have seen significant change. Restaurants and similar services have been arguably impacted worse than most other businesses, as shelter-in-place and social distancing orders significantly negatively impacted them, and some have even been forced to close. In fact, according to the Washington Post, economists project that 100,000 small businesses have permanently closed because of the pandemic. "Their latest data suggests at least 2 percent of small businesses are gone, according to a survey conducted May 9 to 11. The carnage has been even higher in the restaurant industry, where 3 percent of restaurant operators have gone out of business, according to the National Restaurant Association." (Long, 2020) With the nature of the pandemic, it stands to reason that hospitals and the healthcare industry as a whole has seen a higher amount of activity than normal. Lastly, the industrials sector contains many diverse industries that have all been affected differently. Among these are airlines, which have suffered greatly due to the restrictions on travel, and logistics, which have also suffered due to closing businesses and lower demand for various goods.

Additionally, using the free Twitter API, tweets from different politicians from both the republican and democratic parties have been collected from February 2020 to the end of June. This paper examines the sentiment expressed in these tweets over time in order to determine how it changed in response to outside events, with a focus on the COVID-19 pandemic.

III. Methods and Data Preparation

A. Financial Data

After choosing the four sectors of the stock market to focus on, a list of the related stocks was needed to actually find financial information. FinViz, a website focusing on financial visualizations, contains what is essentially a database of traded stocks, where users can look up different stocks based on ticker, industry, and sector. As the website only publicly presents historical information in a visualized format, it was not an ideal source of actual prices. However, a web scraper was written to interface with the website and pull the symbols and associated company name for every stock in the four chosen sectors. The actual financial data was scraped from Yahoo Finance, which publicly contained historical highs, lows, opens, and closes for every stock. The tickers scraped from FinViz were used to search for historical stock data on Yahoo Finance, and the results were combined into one table. Finally, some companies split their stocks during the observed period, which resulted in significant drops in prices. These companies were removed for the analysis. The data dictionary for financial data can be seen below:

Table 1: Data Dictionary - fin_full.csv

Variable Name	Description	Data Type	Source	Example
Date	The date of the associated prices	datetime	Yahoo Finance	6/23/2020
Open	Opening stock price of the day	float	Yahoo Finance	16.4
High	Highest price of the day	float	Yahoo Finance	16.49
Low	Lowest price of the day	float	Yahoo Finance	15.88
Close	Closing stock price of the day	float	Yahoo Finance	15.96
Adj. Close	Closing price adjusted to reflect what is considered to be the true price	float	Yahoo Finance	15.96
Volume	Number of shares traded on that day	int	Yahoo Finance	96765
Symbol	Stock ticker symbol for the associated company	string	Finviz	BOMN
Company Name	Name of the company associated with the ticker	string	Finviz	Boston Omaha Corporation
Sector	Sector containing the company	string	Finviz	communicationservices
Industry	Industry containing the company	string	Finviz	advertisingagencies

In addition to this dataset, a second one was constructed to provide a higher view of how each individual industry performed over time. Specifically, its purpose is to illustrate the relationship between where the industry started and where it ended, in terms of adjusted closing price, as well as if and/or how well the industry has recovered.

Table 2: Data Dictionary - ind_pct_changes.csv

Variable Name	Description	Data Type	Example
Sector	Sector that the industry belongs to	Str	healthcare
Industry	Industry containing the associated metrics	Str	internetretail
Start	First adjusted closing price for the industry	Float	33.116
End	Final adjusted closing price for the industry	Float	78.22
Min	Lowest adjusted closing price for the industry	Float	78.67
Pct_change	Percent change from start to end	Float	11.86271
Recovery	Percentage of the initial drop in adjusted stock price that was recovered after the initial drop	Float	128.167

One factor to keep in mind while examining this dataset is to know that some companies were on a significant positive trend at the start of the date range, meaning that their adjusted closing price increased, sometimes significantly, before plummeting as the pandemic came to be.

B. Twitter Data

Twitter offers various API’s that allow approved users to interact with tweets and profiles on the website. Most API’s charge in various different ways, but Twitter does offer a free API that is subject to certain limitations. Primarily, a request for tweets would only return tweets up to seven days old from the time of the request. For this reason, a script was written to begin before Super Tuesday, the day with the largest number of states holding their presidential primary elections, that ran weekly that collected any tweets from the past week that were written by specified Twitter profiles. The profiles that were examined are as follows:

- @AOC – Alexandria Ocasio-Cortez (D), New York House Representative
- @AyannaPressley, Ayanna Pressley (D), Massachusetts House Representative
- @BernieSanderse, Bernie Sanders (D), Vermont Senator
- @GOPLeader, Kevin McCarthy (R), California House Representative, House Minority Leader
- @Ilhan, Ilhan Omar (D), Minnesota House Representative
- @JoeBiden, Joe Biden (D), former US Vice-President, 2020 Presidential Candidate

- @RashidaTlaib, Rashida Tlaib (D), Michigan House Representative
- @realDonaldTrump, Donald Trump (R), current US President
- @senatemajldr, Mitch McConnell (R), Kentucky Senator, Senate Majority Leader
- @SenSchumer, Chuck Schumer (D), New York Senator, Senate Minority Leader
- @SpeakerPelosi, Nancy Pelosi (D), California House Representative, House Majority Leader

The script utilizing the Twitter API was originally written to examine political sentiment during an election year, but its purpose changed with the COVID-19 pandemic's occurrence. With the first tweets occurring near Super Tuesday, the data provides a look at tweets just before the pandemic, as well as during the pandemic and its evolution over time.

The Twitter API provides information on just about anything about a tweets or the associated profile, so most variables either did not apply to this analysis or were irrelevant. Variables such as URLs for attached photos, a user's Twitter bio – the text included at the top of their profile, were not included in the final dataset.

C. Sentiment Analysis

In order to get a clear picture of how sentiment changed over time, a bag-of-words approach utilizing multiple lexicons was used in order to prevent the biases of one lexicon impact the overall picture. The lexicons used were:

AFINN

The AFINN lexicon is a list of English words curated by Finn Årup Nielsen of the Technical University of Denmark. In the construction of this list, Nielsen manually "scored for valence, leaving out, e.g., subjectivity/objectivity, arousal and dominance." (Nielsen, 2011) Nielson's goal for this lexicon was to be a new and improved version of an existing lexicon, ANEW, by adding and focusing on internet slang and obscene words. While the inclusion of obscenity would not necessarily impact the analysis of political tweets, the inclusion of internet slang could potentially allow for any slang possibly used, perhaps by the younger politicians in the data, to be taken into account.

NRC

The NRC word-emotion lexicon was created by Saif Mohammad and Peter Turney of the Institute for Information Technology at the National Research Council of Canada. This particular lexicon focuses on the emotions and sentiments expressed in text, as well as the polarity of the text. In order to do this, this lexicon scores individual words based on eight basic emotions from Plutchik's wheel of emotions: anger, anticipation, disgust, fear, joy, sadness, surprise, and trust. It also bases measures polarity based on positivity and negativity. By using this lexicon, the data provides a deeper look into the specific emotions expressed in the tweets over time, and it is not limited to just being positive or negative. The data for this lexicon was gathered by crowdsourcing via Amazon's Mechanical Turk, where the creators asked either native English speakers or people that were very fluent in English to annotate the words. Mohammad and Turney noted that there are two main issues presented by crowdsourcing: 1) cheaters, who would input random information, and malicious annotators, who would intentionally input wrong information, and 2) ensuring that the "turkers" (respondents) had the same connotation and definition of the words in mind while responding. In order to address this, the survey began with synonym questions, where the goal is to guide the turkers to the intended meaning of the particular word. Additionally, for those questions, of the four choices, three of them were distractors, meaning that cheaters and malicious annotators would more than likely have their responses removed. (Mohammad & Turney, 2013)

In the context of this project, however, the NRC lexicon does present another problem that cannot be addressed; the data lacks the context of who specifically annotated these words. Demographic information could provide more information as to why different words are assigned their respective values in this lexicon. This is important for analyzing Twitter data, as the vernacular as seen on the internet is consistently changing over time, and words will be used in different ways. Given that these tweets are all from politicians, this should not be as prevalent of an issue, given that they are more likely to read more formally than tweets from random users. However, given the generational difference between some of the politicians in the data, there is a possibility that different words could be used in different ways, and there is no way of knowing which way the NRC lexicon is better suited for.

VADER

The VADER (Valence Aware Dictionary for sentiment Reasoning) lexicon is another crowd-sourced lexicon created by C.J. Hutto and Eric Gilbert at the Georgia Institute of Technology. While the NRC lexicon focus primarily on specific emotions, the VADER lexicon focuses entirely on positivity, neutrality, and negativity. Additionally, it is fine tuned for microblogging and social media, making it a perfect candidate to use in an analysis on Twitter data. In a similar fashion to the NRC lexicon, Hutto and Gilbert addressed the issues that arise with crowdsourcing by presenting contributors with a reading comprehension test and a training and orientation session for sentiment rating. Since the VADER lexicon focuses only on three variables, being positivity, neutrality, and negativity, it is less prone to the various biases that could be caused by demographic variance in the contributors than the NRC lexicon, which provides 10 variables. (Hutto & Gilbert, 2014)

Textblob

Textblob is an NLP Python package very similar to NLTK in that it includes modules for part-of-speech tagging, tokenization, as well as other common NLP functions, but it also includes a measure for the polarity of a "blob" of text, which acts as a good measure of the positivity and negativity of the text. Unlike NRC and VADER, it is not crowdsourced, rather it is a public project hosted on GitHub and PyPI, which eliminates some of the problems posed by the creation of crowdsourced lexicons.

D. Data Consolidation and Transformation

All of the aforementioned lexicons provide one or several different measures for the sentiment of a given text. However, the primary issue posed by using many different lexicons is that all of them are on different scales. For that reason, almost every variable has a second corresponding variable that has been standardized to a scale of either -100 to 100 or 0 to 100. The latter scale was used for measures such as the NRC emotion scores, where the measure was simply measuring the presence of that emotion. The former scale was used for dichotomous variables, such as the VADER compound score, where a negative integer represents a negative sentiment, and a positive integer represents a positive sentiment. Additionally, specifically for the NRC lexicon, additional variables were created that combined the different measured emotions into positive and negative emotions, which allows the data to provide a more general overview on emotional content rather than a more granular approach. Lastly, a final variable was created to represent the combined sentiment values from every lexicon to be used as a main measure of sentiment in every tweet. The data dictionary for this dataset can be seen below:

Table 3: Data Dictionary - tweets_final.csv

Variable Name	Description	Data Type	Source	Example
Created_at	Creation date and time of the tweet	Datetime	Twitter API	3/2/2020 0:07
Screen_name	Screen name of tweet author	Str	Twitter API	AOC
Favorite_count	No. of favorites	int	Twitter API	537
Retweet_count	No. of retweets	int	Twitter API	415
Quote_count	No. of quotes	int	Twitter API	16
Reply_count	No. of replies	int	Twitter API	5
Hashtags	List of hashtags in tweet	List(Str)	Twitter API	GreenNewDe al
Quoted_text	If tweet quotes another, text of original tweet	Str	Twitter API	
Retweet_text	If tweet is a retweet, text of original tweet	Str	Twitter API	Yes, there IS a…
Party	Political affiliation	Str	Manual	Democrat
Afinn_source	Afinn score of text in the original tweet	int	Afinn	5
Afinn_source_category	Category of afinn_source score	Str	Afinn	positive
Afinn_quote	Afinn score of quoted tweet, if applicable	int	Afinn	0
Afinn_quote_category	Category of afinn_quote score	Str	Afinn	neutral

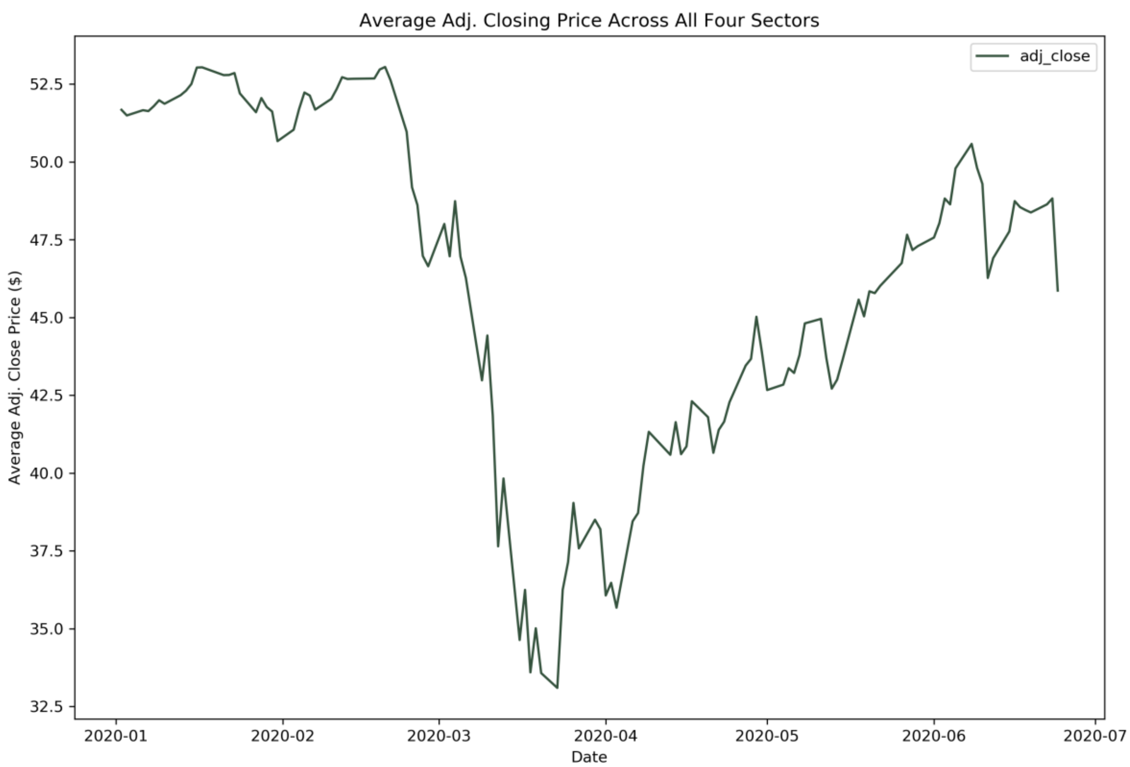
Afinn_retweet	Afinn score of retweet text, if applicable	int	Afinn	-5
Afinn_retweet_category	Category of afinn_retweet score	str	Afinn	negative
Fear_score	NRC score of fear content	int	NRC	4
Fear_freq	NRC frequency % of fear	float	NRC	0.105263
Anger_score	NRC score of anger content	int	NRC	3
Anger_freq	NRC frequency % of anger	float	NRC	0.078947
Anticip_score	NRC score of anticipation content	int	NRC	3
Anticip_freq	NRC frequency % of anticipation	float	NRC	0.078947
Trust_score	NRC score of trust content	int	NRC	4
Trust_freq	NRC frequency % of trust	float	NRC	0.105263
Surprise_score	NRC score of surprise content	int	NRC	3
Surprise_freq	NRC frequency % of surprise	float	NRC	0.078947
Pos_score	NRC score of positive content	int	NRC	5
Pos_freq	NRC frequency % of positive	float	NRC	0.131579
Neg_score	NRC score of negative content	int	NRC	4
Neg_freq	NRC frequency % of negative	float	NRC	0.105263
Sad_score	NRC score of sad content	int	NRC	4
Sad_freq	NRC frequency % of sad	float	NRC	0.105263
Disgust_score	NRC score of disgust content	int	NRC	4
Disgust_freq	NRC frequency % of disgust	float	NRC	0.105263
Joy_score	NRC score of joy content	int	NRC	4
Joy_freq	NRC frequency % of joy	float	NRC	0.105263
Vader_positive	VADER score of positivity	float	Vader	0.375
Vader_neutral	VADER score of neutrality	float	Vader	0.625
Vader_negative	VADER score of negativity	float	Vader	0.147
Vader_compound	VADER compound score	float	Vader	0.9271
Textblob_polarity	Textblob score of polarity	float	Textblob	0.9
Virality	Sum of interactions (favorites, retweets, quotes, replies) with a tweet	int	Twitter API	431
Afinn_std	Afinn_source standardized to -100-100	float	Afinn	27.27273
Fear_std	Fear_score standardized to -100-100	float	NRC	12
Anger_std	Anger_score standardized to -100-100	float	NRC	8
Sad_std	Sad_score standardized to -100-100	float	NRC	12.5
Disgust_std	Disgust_score standardized to -100-100	float	NRC	12.5
Anticip_std	Anticip_score standardized to -100-100	float	NRC	8.695652
Trust_std	Trust_score standardized to -100-100	float	NRC	12
Surprise_std	Surprise_score standardized to -100-100	float	NRC	8.695652
Joy_std	Joy_score standardized to -100-100	float	NRC	12

Pos_std	Pos_score standardized to -100-100	float	NRC	14.81481
Neg_std	Neg_score standardized to -100-100	float	NRC	11.53846
Nrc_sentiment	Difference between pos_std and neg_std	float	NRC	3.276353276
Nrc_PosEmo	Sum of positive NRC emotions	float	NRC	41.3913
Nrc_NegEmo	Sum of negative NRC emotions	float	NRC	45
Nrc_EmoScore	Difference between nrc_PosEmo and nrc_NegEmo	float	NRC	-3.6087
Vader_positive_std	Vader_positive standardized to -100-100	float	Vader	42.13483146
Vader_negative_std	Vader_negative standardized to -100-100	float	Vader	18.01471
Vader_neutral_std	Vader_neutral standardized to -100-100	float	Vader	57.86517
Vader_compound_std	Vader_compound standardized to -100-100	float	Vader	93.884895544
Textblob_polarity_std	Textblob_polarity standardized to -100-100	float	Textblob	90
Sentiment_compound	Average of textblob_polarity_std, vader_compound_std, nrc_sentiment, and afinn_std	float	Manual	42.15787

IV. Observations

A. Financial Data

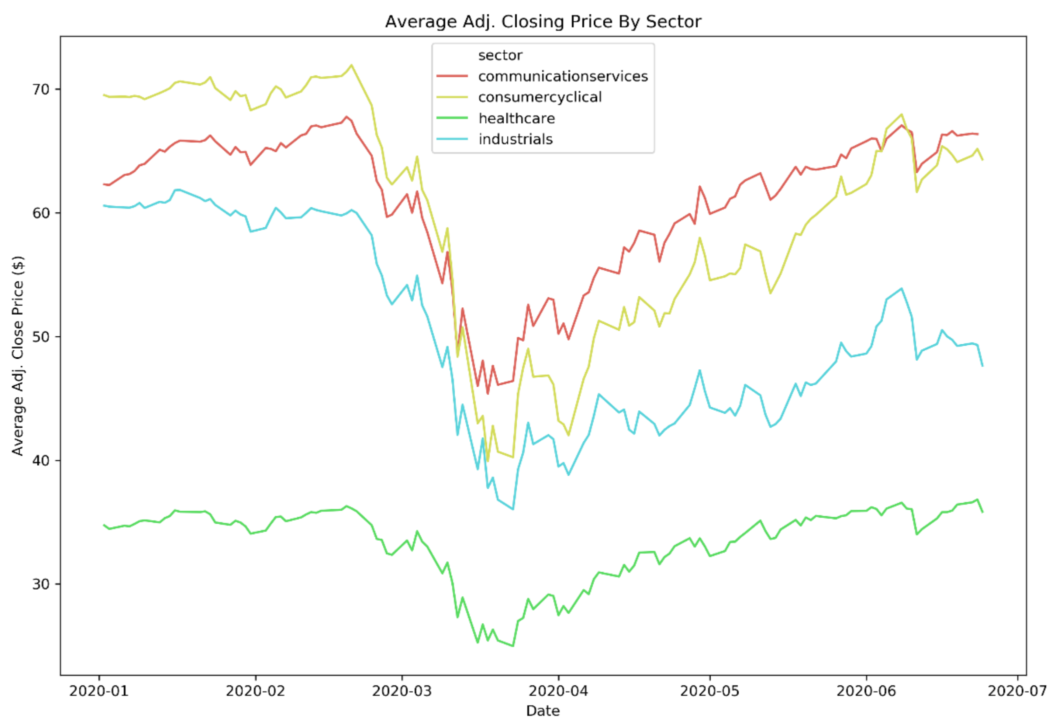
Figure 1: Average Adj. Closing Price Across All Four Sectors



As expected, the overall adjusted closing price for stocks across all four chosen sectors shows a significant drop that correlates with the increased prevalence and severity of the COVID-19 pandemic. The significant decline lines up with many precautionary measures taken by the various states. According to USA Today, by March 30th, all but 8 states (Arkansas, Iowa, Nebraska, North Dakota, Oklahoma, South Dakota, Utah, Wyoming) had issued shelter-in-place orders. (Ortiz & Hauck, 2020) In addition to shelter-in-place orders, universities across the nation began moving their operations and instruction entirely online in order to preserve student safety. Paul Basken noted in an article for The World

University Rankings that as of March 10th, the number of closed institutions was quickly growing, affecting hundreds of thousands of students nationwide. (Basken, 2020)

Figure 2: Average Adjusted Closing Price By Sector



Upon examination of the financial data on a more granular level, it becomes apparent that all four sectors were impacted in a similar manner. They all exhibit similar trends at similar times, with their lowest point occurring towards the end of March, and all four sectors continue on a positive trend to the present day. The significant decline aligns nearly perfectly with President Trump's declaration of a national emergency in response to the pandemic that took effect on March 1st. (Trump, 2020) For the most part, data behaves as expected. Logic dictates that a global pandemic would have severe repercussions in the stock market. However, the data provides an even more granular look at these four sectors by providing the adjusted closing prices over time of each individual industry within each sector. Visualizations of all industries are included in the appendix of this paper, and some will be highlighted below.

Communication Services

Overall, the Communication Services sector has fared well, by seeing its current average closing price approximately equal to the price before the pandemic took full effect. Within this sector, most industries modeled this same trend, with the exception of a few.

Figure 3: Average Adjusted Closing Price: CommServices - Advertising

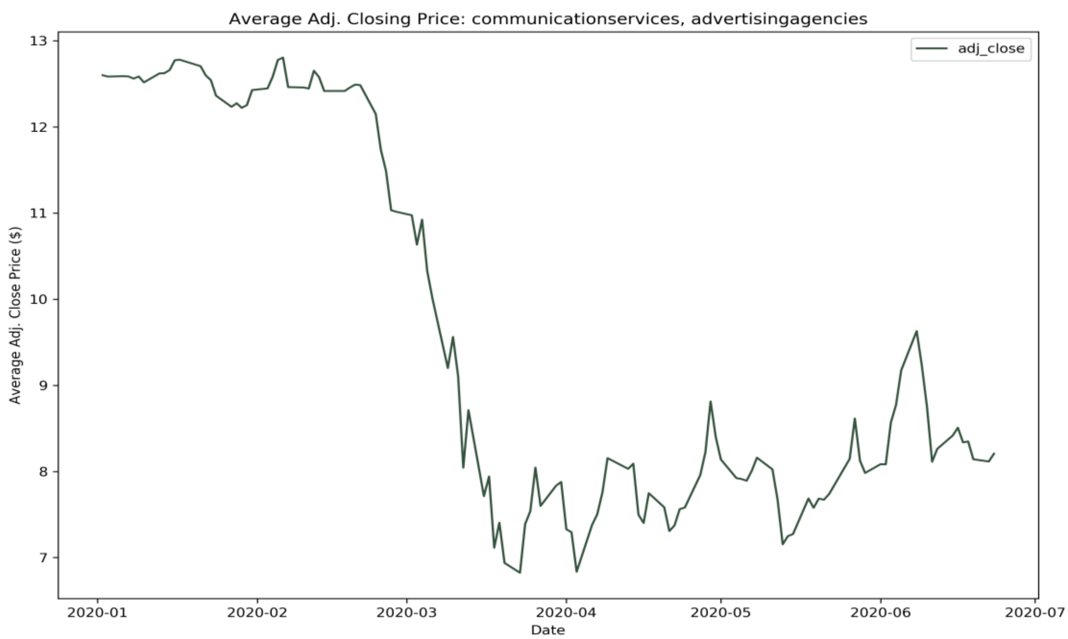
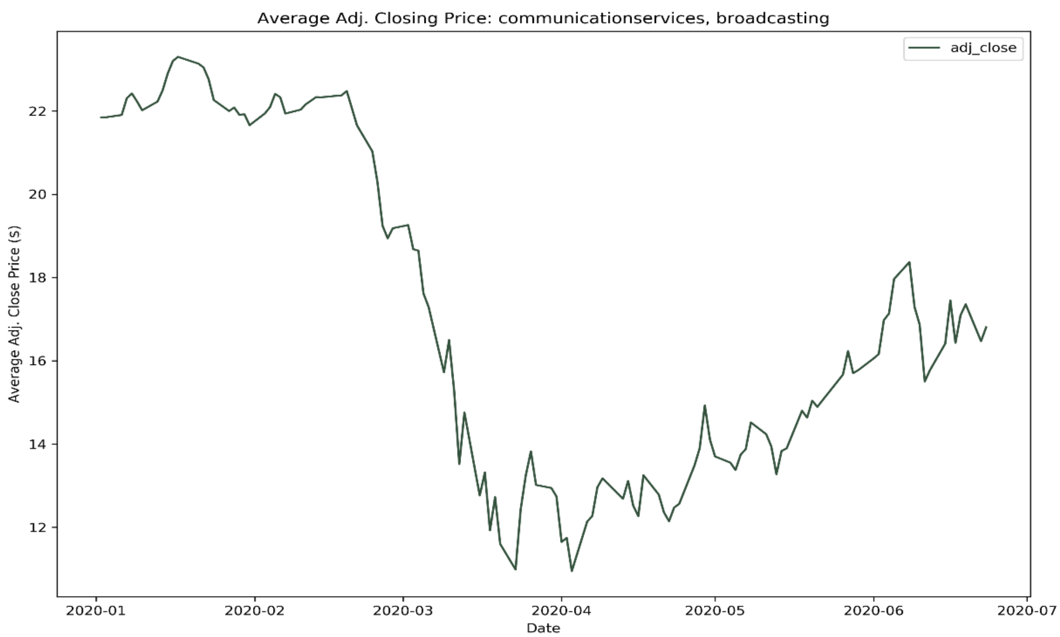
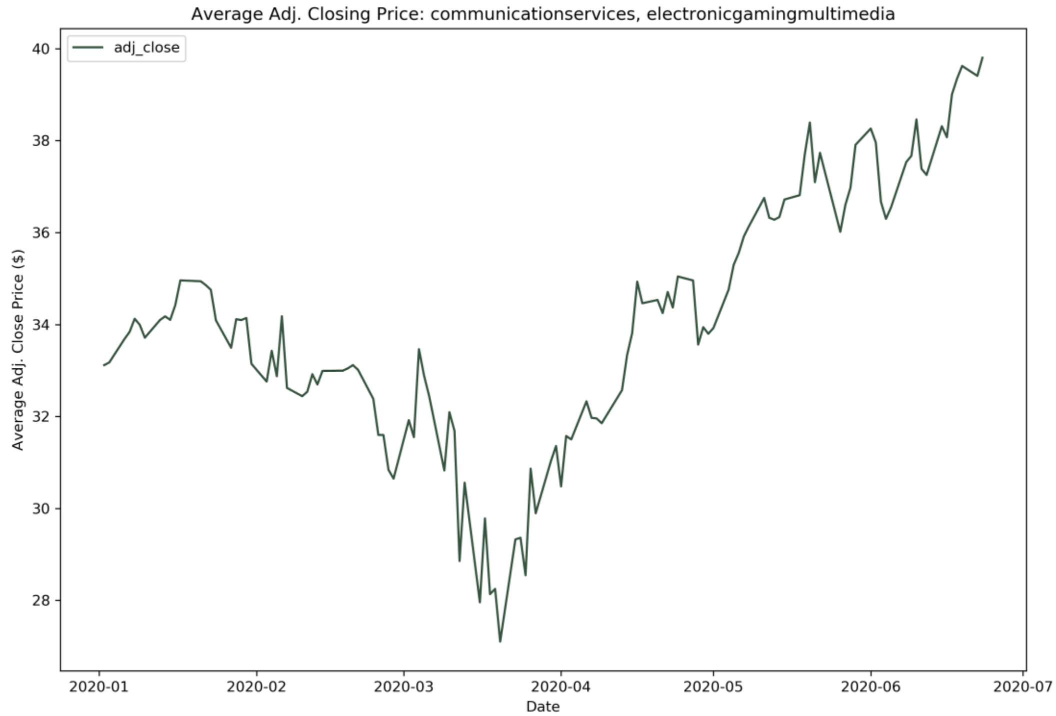


Figure 4: Average Adjusted Closing Price: CommServices - Broadcasting



Advertising agencies and broadcasting were severely negatively impacted, with advertising agencies seeing a negligible recovery and broadcasting seeing approximately 50% recovery. With shelter-in-place orders in effect across most of the country, the demand for out of home advertising would decrease, as travel would decrease. Additionally, the transfer of many companies to a work-from-home model would decrease the number of commuters, decreasing potential interactions with non-online advertising. However, companies that specialize in online advertising would see an increase in traffic, and have likely contributed to the industry's recovery over time. The broadcasting industry includes both radio and television, both of which have been severely impacted by the pandemic. People who listen to the radio during their daily commute who have either been fired or moved to working from home now no longer need to tune in as they are no longer commuting. Television companies that rely on the production of shows or live events were hindered by the cancellation of live events, especially sporting events, as well as large scale productions being hindered by social distancing protocols.

Figure 5: Average Adjusted Closing Price: CommServices - Electronic, Gaming, Multimedia



Of all industries in this sector, electronic gaming and multimedia has seen the best results, as their current adjusted closing price is around $6 higher than it was before the pandemic. With the increase of people spending much more time at home, the usage of video games and electronic entertainment would drastically increase.

Consumer Cyclical

Figure 6: Average Adjusted Closing Price: ConCyc - Department Stores

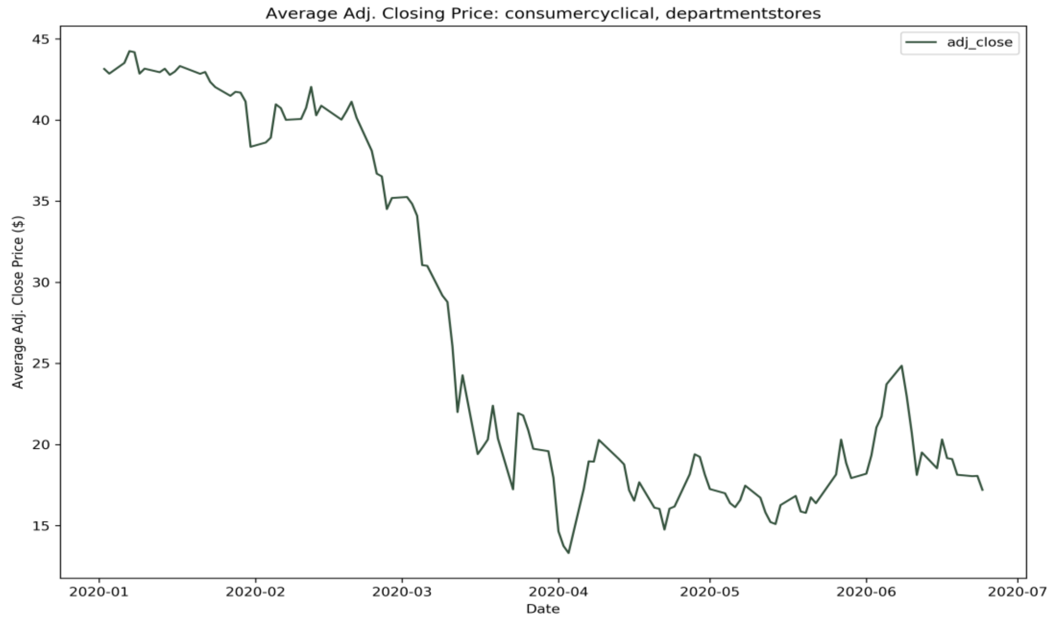
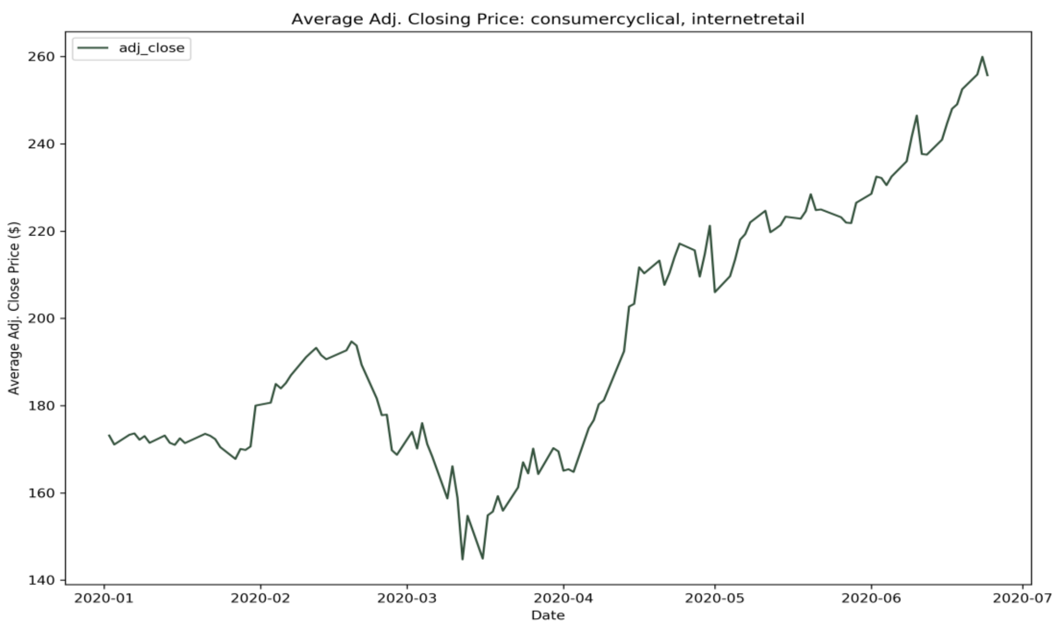


Figure 7: Average Adjusted Closing Price: ConCyc – Internet Retail



Overall, this sector performs similarly to the others, with some more drastic effects. It started with the highest average adjusted closing price and fell to be almost equal with the industrials sector – significantly lower than where it started. There are two industries that could be considered outliers that illustrate the positive and negative economic effects of the pandemic. In fact, these two industries directly compete. With shelter-in-place and social distancing orders in place, shopping at department stores suddenly became more difficult and less practical. As a direct result, online shopping became much more practical, as it requires no travel or interaction with other people. As a result, department stores have suffered the most in this sector, and internet retail has benefited from it.

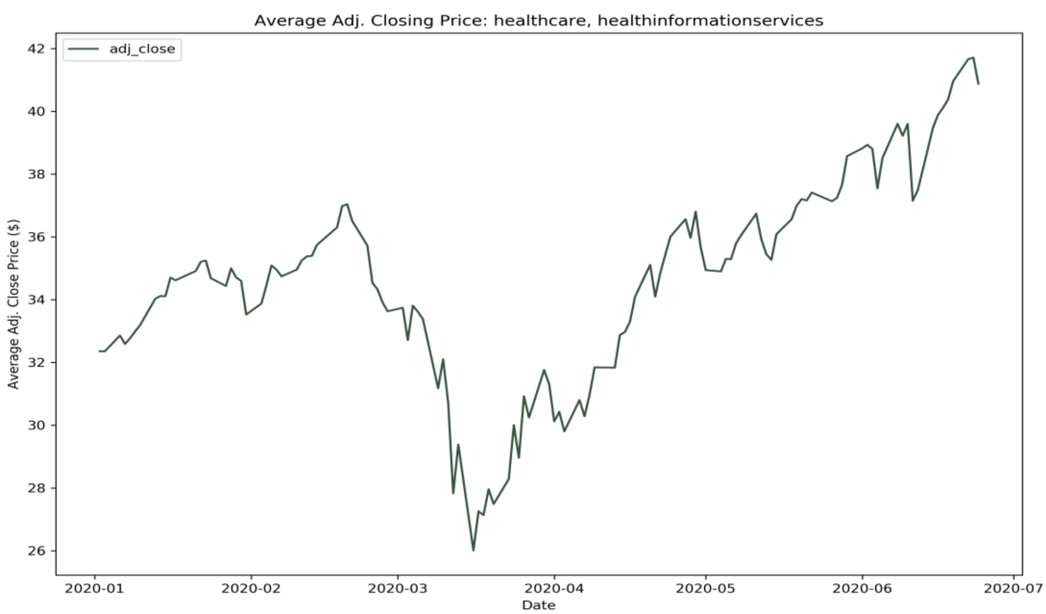
Figure 7: Average Adjusted Closing Price: ConCyc - Restaurants

One recurring question during this pandemic has been about the well-being of restaurants around the country, seeing as it not only requires personal interaction with other people as well as other peoples' interaction with customers' food, but social distancing policies decreased maximum occupancy for many locations. This data shows that the restaurant industry has made a significant, but not full recovery, as many restaurants have adapted by implementing new practices, such as curbside or no-contact delivery, or partnerships with delivery services such as GrubHub. However, it is important to note that this data does not take into account small, local, or mom-and-pop restaurants that are not a part of national chains that are publicly traded on the stock market.

Healthcare

One of the first notable insights gleaned from the original figure showing all four sectors was that the healthcare industry was affected nearly identically to the other sectors. Upon first thought, many would think that with a pandemic, hospitals would see increased traffic, leading to higher revenue, which would potentially lead to increased stock prices. On the contrary, it is important to remember that with a global pandemic, people are less likely to travel to medical facilities for non-COVID-related services out of fear of being infected with the coronavirus. Additionally, people are less likely to participate in elective procedures, which could have a significant negative impact on the revenue of medical facilities.

Figure 8: Average Adjusted Closing Price: Healthcare - Health Info Services



Of the industries within the healthcare sector, the health information services industry has seen the best results. In the midst of a pandemic, the changing conditions of medical facilities would more than likely call for either a restructuring or an increase in IT infrastructure. Companies such as Cerner have provided COVID-19 resource pages detailing their own response to the pandemic as well as how clients can optimize workflow and infrastructure in these conditions. (Cerner, n.d.)

Figure 9: Average Adjusted Closing Price: Healthcare - Med. Care Facilities

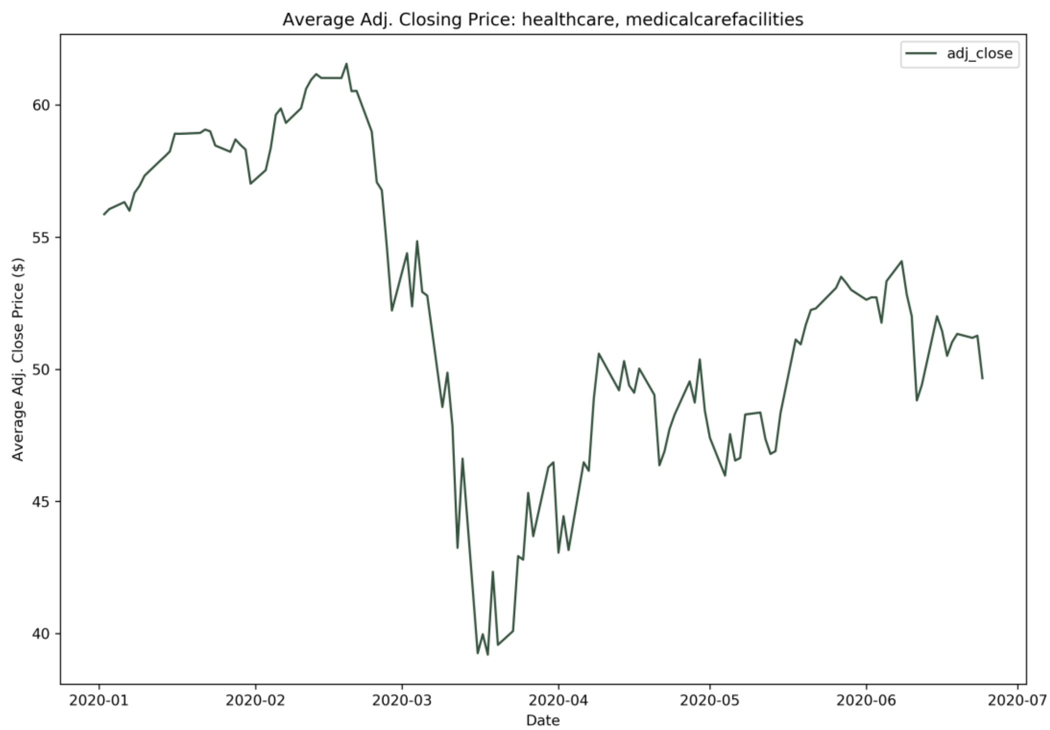
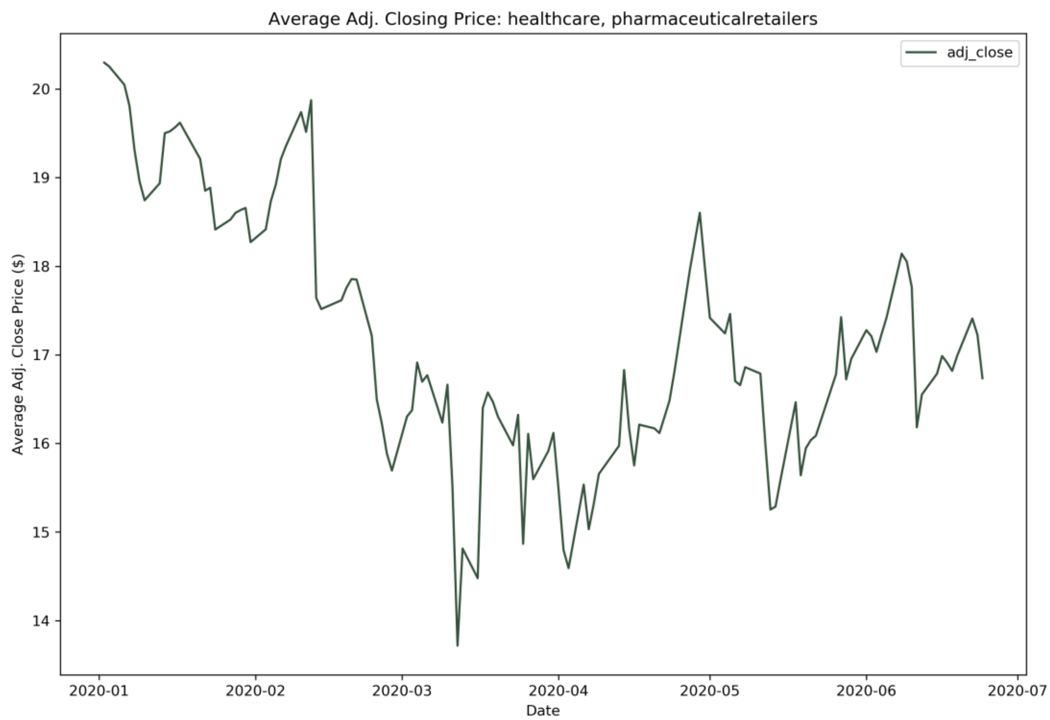


Figure 10: Average Adjusted Closing Price: Healthcare - Pharma Retailers

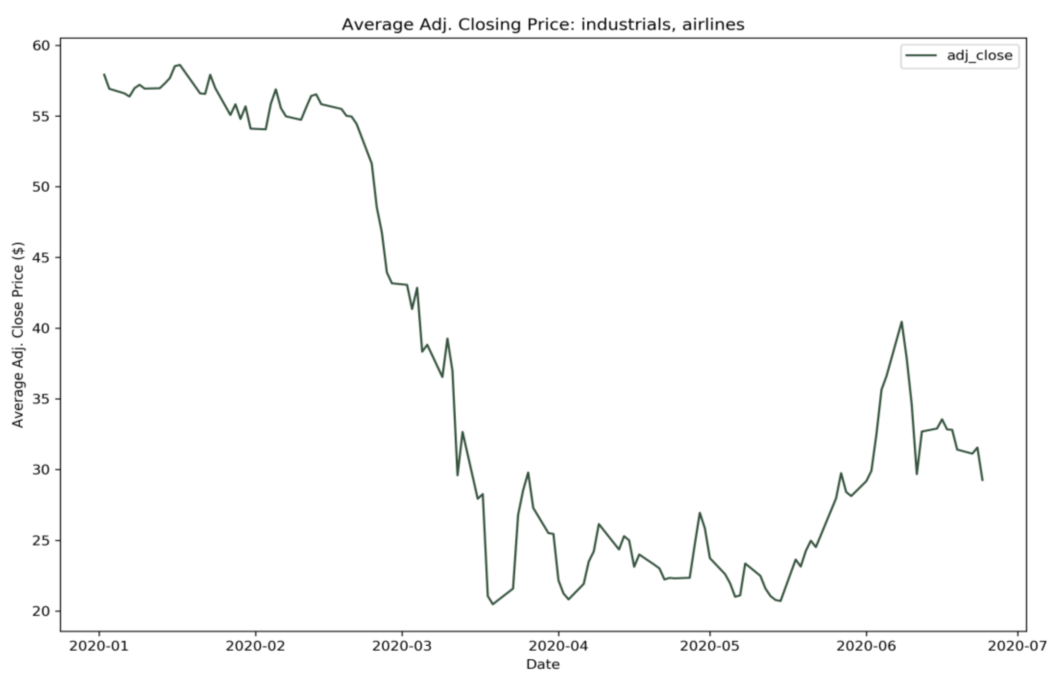


Within the healthcare sector, medical care facilities and pharmaceutical retailers were impacted the hardest. Pharmaceutical retailers such as Walgreens, given that a good portion of their business is done in their physical stores, have more than likely been impacted in a similar way to department stores, as people are less likely to travel due to the restrictions. Medical care facilities have seen a litany of difficulties during this pandemic, including shortages on equipment, such as masks and gowns (FDA, 2020), or worries about exceeding limits on ICU beds. For example, Arizona has seen a steady increase of hospital bed usage, peaking at the beginning of July at 91%. (AZDHS, 2020), and CDC projections from state representatives show that inpatient bed usage will peak at around 80% (CDC, 2020). These difficulties combined with their shift in operations away from routine and elective procedures towards COVID related operations has strained the system, resulting in the low stock recovery.

Industrials

As a whole, the Industrials sector was affected the worst of the four sectors, as after its steep drop, it has yet to fully recover. Each industry within this sector generally follows this same trend. The only industry to have a recovery of 100% or greater is airports and air services. This is partly due to the fact that only two companies represented in this data make up that industry. Of all industries, one of the most talked about is the airlines industry. This is for good reason, as the industry is still suffering from the initial effects of the pandemic with a 23% recovery. Primarily, this can be attributed to consistent discouragement regarding traveling as well as entry restrictions on travelers from certain countries into the US, including China and the United Kingdom. (CDC, 2020) These restrictions and travel discouragement are still going on to this day, preventing the airline industry from fully recovering for the time being.

Figure 11: Average Adjusted Closing Price: Industrials - Airlines

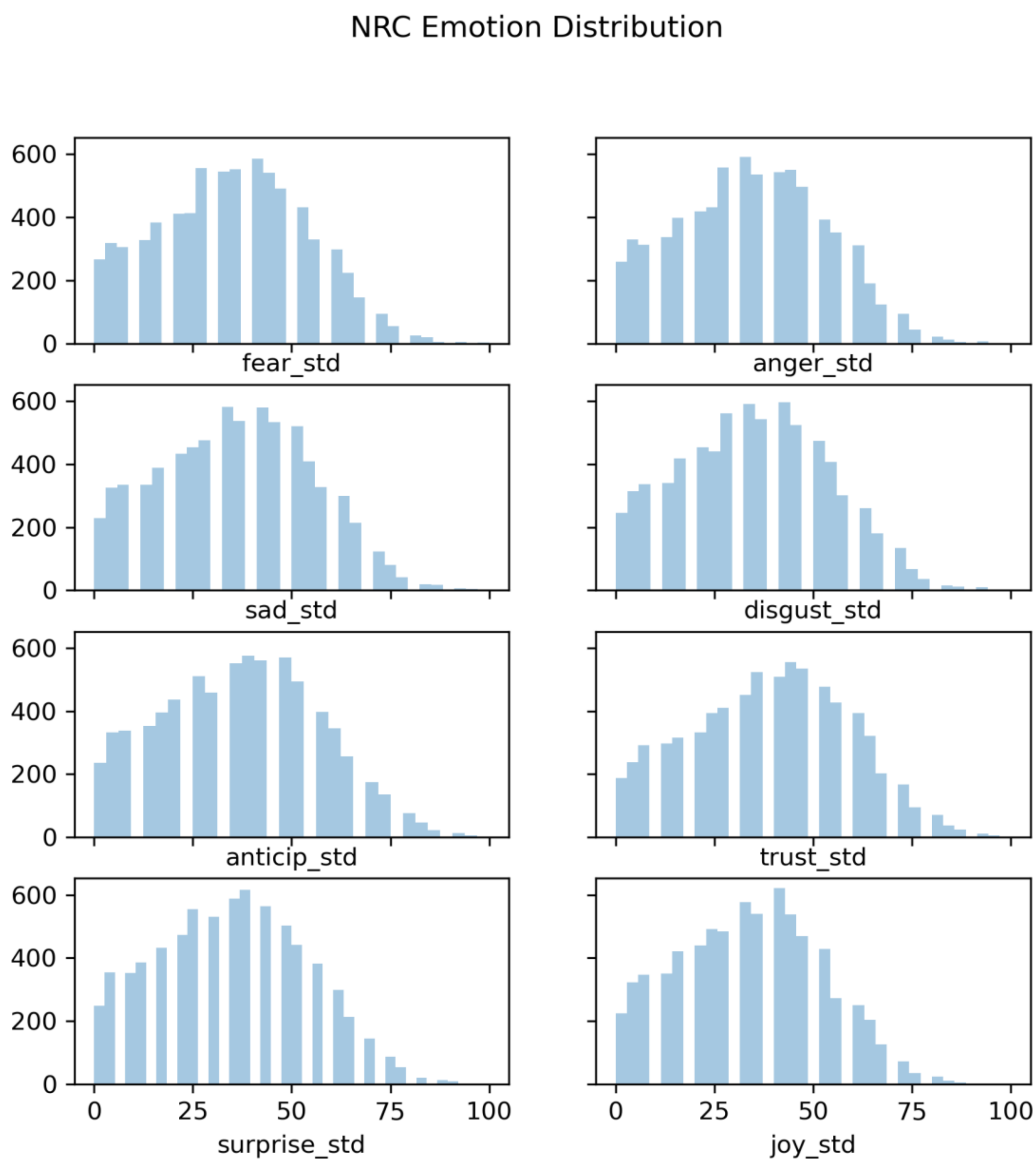


B. Political Tweets

Emotion Analysis with the NRC Lexicon

Of all the lexicons used for this analysis, the NRC lexicon allows for multiple emotions to be tracked, and not just overall positive/negative sentiment. The following visualization shows the distributions of each individual emotion, excluding positive and negative.

Figure 12: Distribution of NRC Emotions



Each emotion exhibits a nearly identical distribution, meaning that there are not any emotions within the NRC lexicon that appears more often than the others. These graphs also show that the data is not significantly charged towards one emotion over the others i.e. the tweets are not visibly angrier than they are joyous.

Sentiment Analysis

As stated earlier, the primary metric used for the sentiment analysis is a combination of metrics across several different lexicons, and will be referred to as the compound sentiment, and is measured on a scale of -100 to 100, representing a spectrum of negative to positive. The following graphs visualize the compound sentiment of the collected Twitter data over time both combined and separated by political party. The scatter plots have been included to better visualize the overall trend within the data, as there is no visible trend within the time series data.

Figure 13: Average Political Sentiment

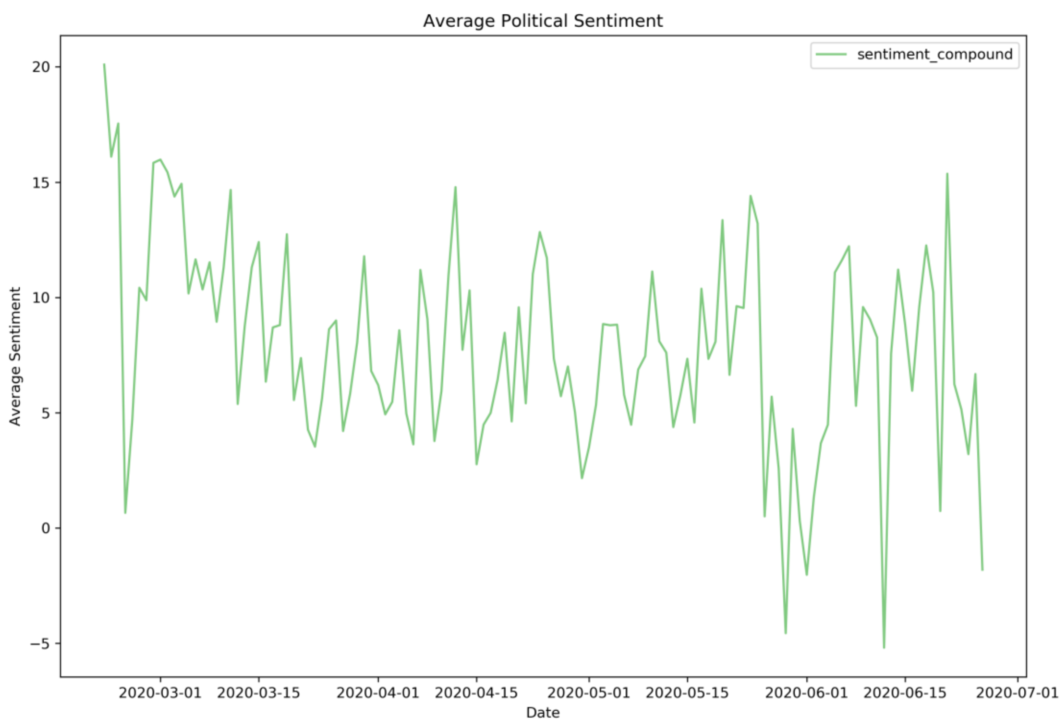


Figure 14: Average Political Sentiment - Trend

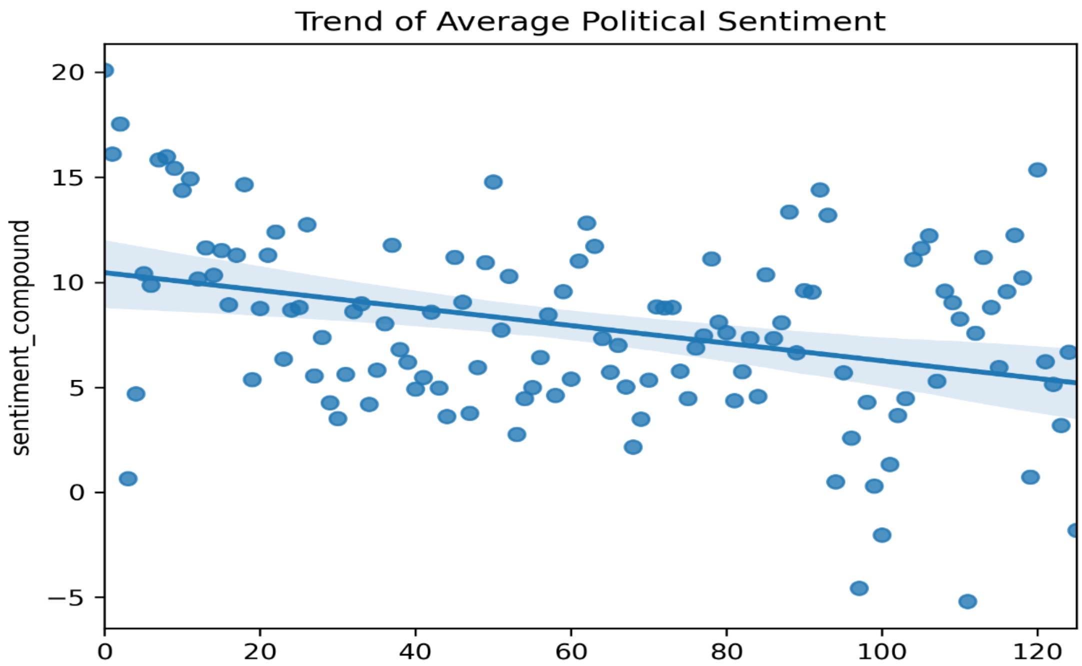


Figure 15: Average Political Sentiment by Party

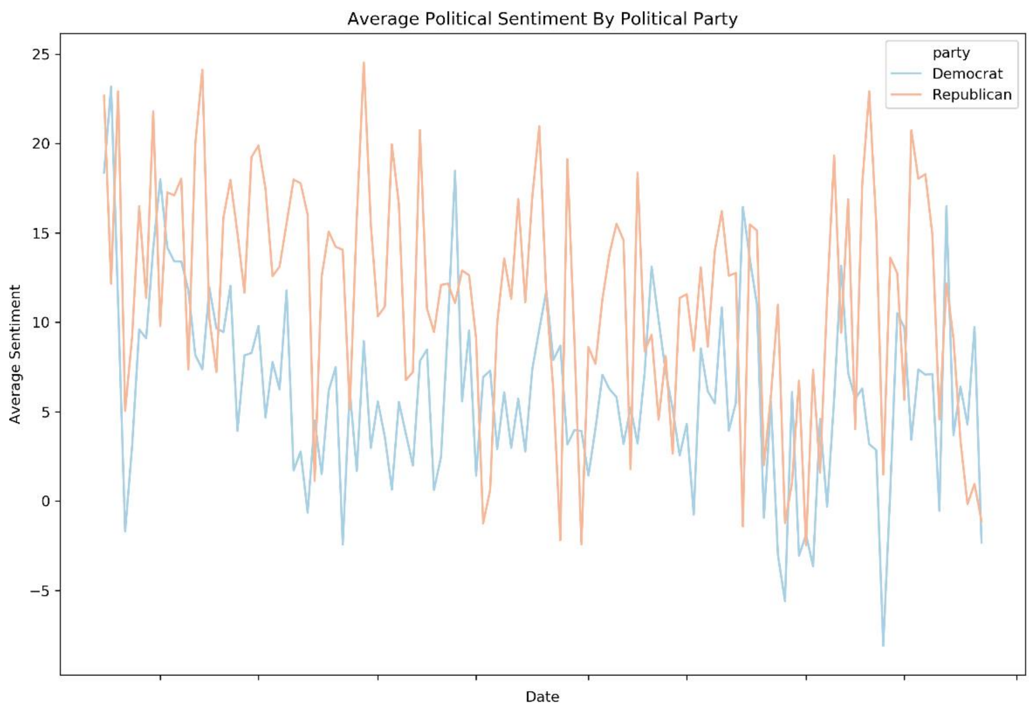
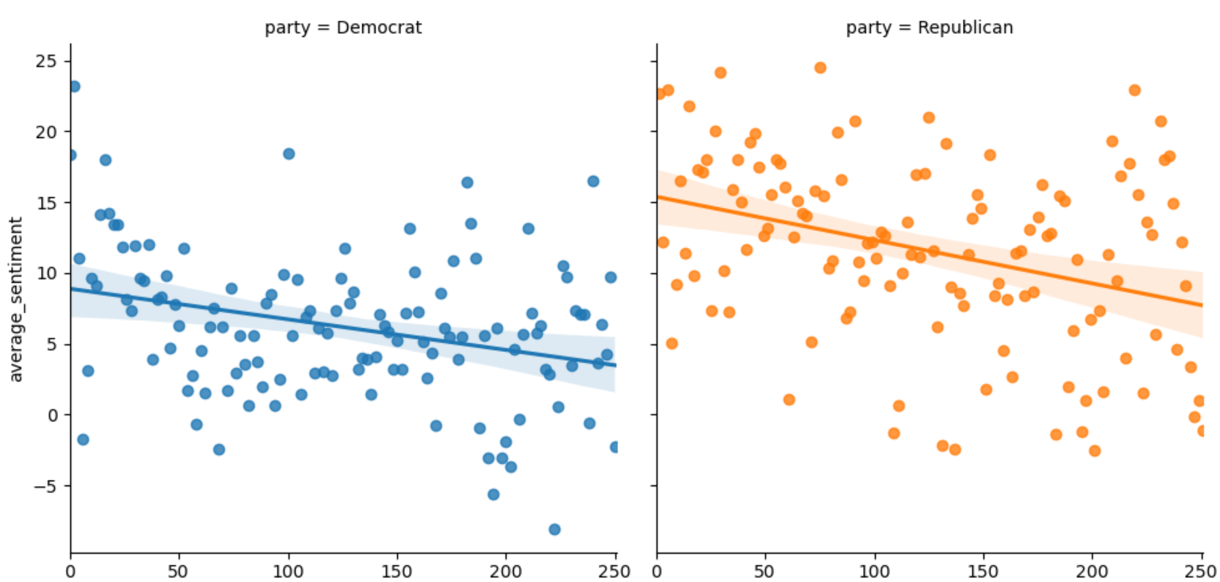


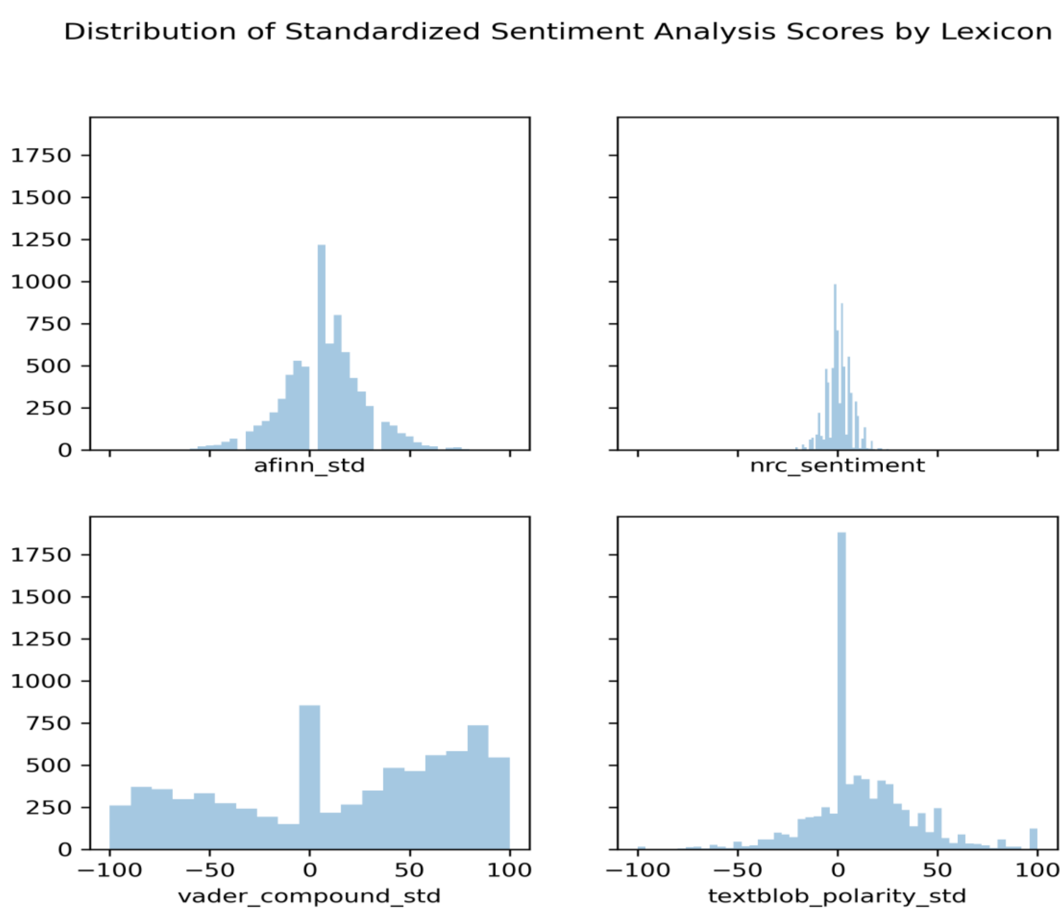
Figure 16: Average Political Sentiment by Party - Trend



Political sentiment has been drifting closer towards zero over time, i.e. it is becoming more negative. This trend is represented by both the overall sentiment, but the same trend is modeled when divided by political party. Of course, over time, the severity of the pandemic has only increased as the number of cases and the death toll has climbed over time. In fact, there are no signs of a positive trend at any points during the data. Despite the improving conditions of the stock market, COVID-related conditions and other social factors led to overall political sentiment declining steadily across all of the observed period.

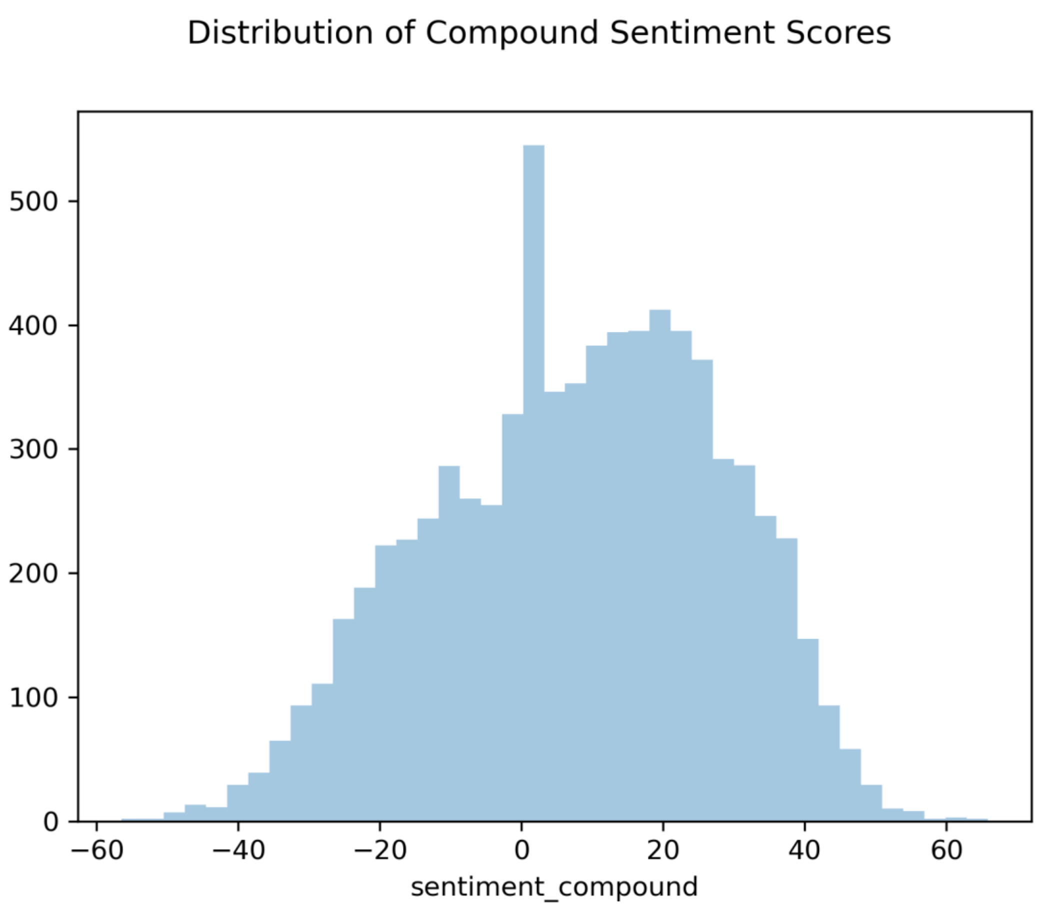
To obtain a more granular view, the following visualizations show the distributions of the individual metrics that make up the compound sentiment: the AFINN score, a calculated NRC score, the compound VADER score, and the textblob polarity value – all standardized to a scale of -100 to 100.

Figure 17: Distribution of Sentiment Scores by Lexicon



The two points of interest regarding the distribution of individual scores lie with the VADER scores and the Textblob scores. VADER is specifically fine-tuned for microblogging, meaning that theoretically, it can pick up on the nuances expressed in Twitter posts. This can potentially explain why, out of all of the lexicons, its distribution contains more extreme values. On the other hand, the Textblob lexicon came from a larger package that acts as an all-in-one tool for working with text in Python. The other lexicons were created specifically for the purpose of sentiment analysis, and Textblob is a similar tool to the NLTK package. The lack of specialization could explain why the vast majority of tweets did not have a polarity value or had a neutral polarity.

Figure 18: Distribution of Compound Sentiment



Based on the distribution of the compound sentiment, it is clear that for the most part, the tweets do not fall on any extreme end. In fact, a good portion of tweets lie around a compound sentiment of 20, meaning that overall, the data tends to be more positive than negative. This phenomenon is backed up by the previous trend lines being positive and remaining positive.

V. Conclusions

The purpose of this paper has been to examine political sentiment as well as stock market behavior during the COVID-19 pandemic. Additionally, this paper is meant to examine any possible correlations between the two. After an initial steep drop, the stock market has been recovering over time. As a whole, the four selected sectors have come close to fully recovering from the drop in average adjusted closing prices. The effect of the pandemic on specific industries has varied, with industries like internet retail and electronic media doing better than ever, and industries like department stores have not recovered nearly as much as others. Without the trend lines, there is no visible trend to political sentiment for either party over this time period. With the trend lines, it becomes apparent that there is a slight negative slope during the pandemic. The beginning of the pandemic had no significant impact on the sentiment expressed in political tweets from the profiles examined here. Both political parties' tweets had positive and negative spikes, but there is no obvious correlation between sentiment and the pandemic nor stock market behavior.

VI. Further Research

This paper is by no means an exhaustive examination into the effects of COVID-19. There is room to further explore every aspect here. The current most pressing issue is that lack of a proper control group of data. The financial data starts approximately two months before the coronavirus seriously impacted the United States, and the Twitter data begins approximately one month before. The starting point of both datasets could be extended to, for example, the beginning of 2012, to examine the changes over time during two different presidential administrations. Combined with a more detailed event analysis, changes in both datasets can be explained in more detail. In addition, the party representation in this data is imbalanced, with 8 democrats and 3 republicans. In future exploration, this could be better balanced.

These events can include significant political, social, and economic events that can potentially have an effect on the data. The Twitter data can be analyzed with more lexicons or different lexicons. e.g. potentially only using the VADER lexicon due to its ability to detect nuance in microblogs. Additionally, a more thorough time series analysis can be conducted on the data to obtain a better idea of trends or seasonality across time.

VII. Works Cited

AZDHS. (2020, July 25). *Data Dashboard*. Retrieved July 26, 2020, from Arizona Department of Health Services: https://www.azdhs.gov/preparedness/epidemiology-disease-control/infectious-disease-epidemiology/covid-19/dashboards/index.php

Basken, P. (2020, March 10). *Coronavirus: classroom closures quicly multiplying in US*. Retrieved July 25, 2020, from The World University Rankings: https://www.timeshighereducation.com/news/coronavirus-classroom-closings-quickly-multiplying-us

BBC. (2020, March 17). *Coronavirus: US Stocks See Worst Fall since 1987*. Retrieved July 20, 2020, from BBC News: https://www.bbc.com/news/business-51903195

CDC. (2020, July 20). *Coronavirus Disease 2019*. Retrieved July 20, 2020, from Centers for Disease Control and Prevention: https://www.cdc.gov/coronavirus/2019-ncov/index.html

CDC. (2020, July 14). *COVID-19 Module Data Dashboard Overview*. Retrieved July 26, 2020, from Centers for Disease control and Prevention: https://www.cdc.gov/nhsn/covid19/report-overview.html

CDC. (2020, June 15). *Travelers Prohibited from Entry to the US*. Retrieved July 27, 2020, from Centers for Disease Control and Prevention: https://www.cdc.gov/coronavirus/2019-ncov/travelers/from-other-countries.html

Cerner. (n.d.). *Cerner COVID-19 Response Center*. Retrieved July 26, 2020, from Cerner: https://www.cerner.com/covid-19/response

Clement, J. (2020, June 19). *Estimated U.S. Social Media Usage Increase Due to Coronavirus Home Isolation 2020*. Retrieved July 21, 2020, from Statista: https://www.statista.com/statistics/1106343/social-usage-increase-due-to-coronavirus-home-usa/

FDA. (2020, June 19). *FAQs on Shortages of Surgical Masks and Gowns During the COVID-19 Pandemic*. Retrieved July 26, 2020, from U.S. Food & Drug Administration: https://www.fda.gov/medical-devices/personal-protective-equipment-infection-control/faqs-shortages-surgical-masks-and-gowns-during-covid-19-pandemic

Hutto, C., & Gilbert, E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. *Eighth International Conference on Weblogs and Social Media.* Ann Arbor, MI.

Long, H. (2020, May 12). *Small Business Used to Define America's Economy. The Pandemic Could Change that Forever*. Retrieved July 21, 2020, from The Washington Post: https://www.washingtonpost.com/business/2020/05/12/small-business-used-define-americas-economy-pandemic-could-end-that-forever/

Ma, J. (2020, March 13). *Coronavirus: China's First Confirmed COVID-19 Case Traced Back to November 17*. Retrieved July 20, 2020, from South China Morning Post: https://www.scmp.com/news/china/society/article/3074991/coronavirus-chinas-first-confirmed-covid-19-case-traced-back

Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence, 29*(3), 436-465.

Nielsen, F. Å. (2011). A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In M. S.-S. Matthew Orwe (Ed.), *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages. Volume 718 in CEUR Workshop Proceedings*, (pp. 93-98).

Ortiz, J. L., & Hauck, G. (2020, March 30). *Coronavirus in the US: How all 50 states are responding - and why eight still refulse to issue stay-at-home orders*. Retrieved July 25, 2020, from USA Today: https://www.usatoday.com/story/news/nation/2020/03/30/coronavirus-stay-home-shelter-in-place-orders-by-state/5092413002/

Rugaber, C. (2020, May 8). *US Unemployment Surges to a Depression-era Level of 14.7%*. Retrieved July 20, 2020, from AP News: https://apnews.com/908d7a004c316baceb916112c0a35ed0

Schumaker, E. (2020, April 23). *Timeline: How Coronavirus Got Started*. Retrieved July 20, 2020, from ABC News: https://abcnews.go.com/Health/timeline-coronavirus-started/story?id=69435165

Trump, D. J. (2020, March 13). *Proclamation on Declaring a National Emergency Concerning the Novel Coronavirus Disease (COVID-19) Outbreak*. Retrieved July 25, 2020, from White House: https://www.whitehouse.gov/presidential-actions/proclamation-declaring-national-emergency-concerning-novel-coronavirus-disease-covid-19-outbreak/#:~:text=1601%20et%20seq.),%2C%20beginning%20March%201%2C%202020.

VIII. Appendix

A. Additional Visualizations

Figure 19: Average Sentiment - Alexandria Ocasio-Cortez

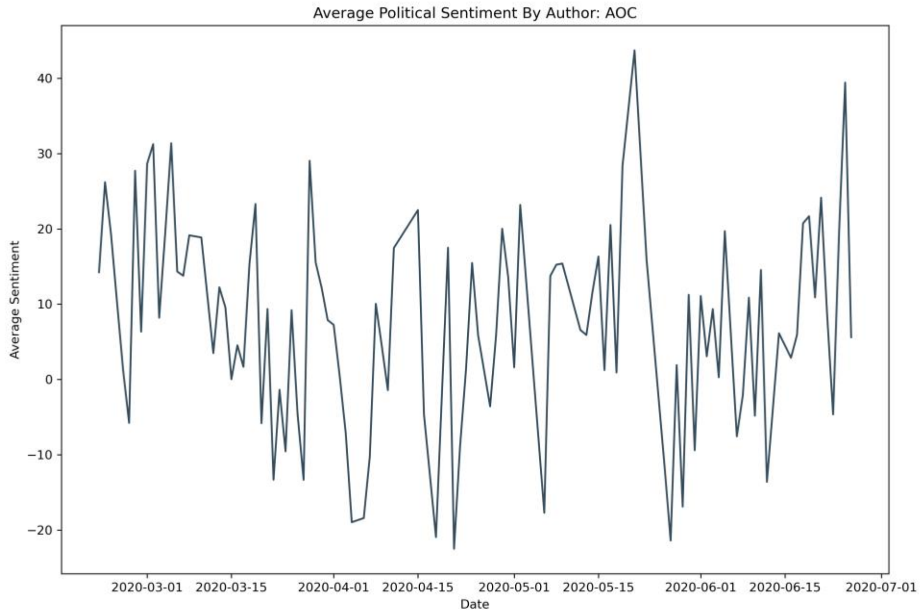


Figure 20: Average Sentiment - Ayanna Pressley

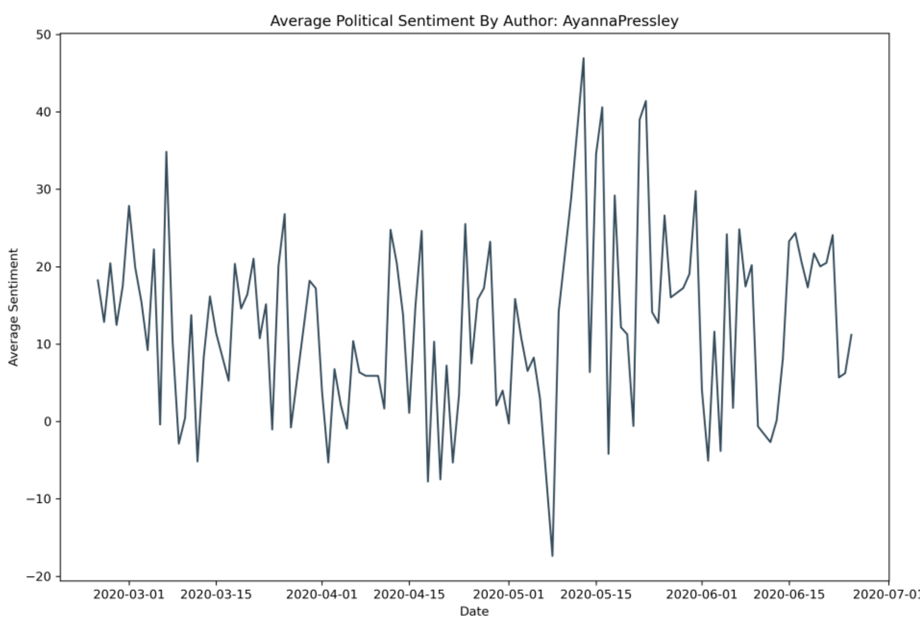


Figure 21: Average Sentiment - Bernie Sanders

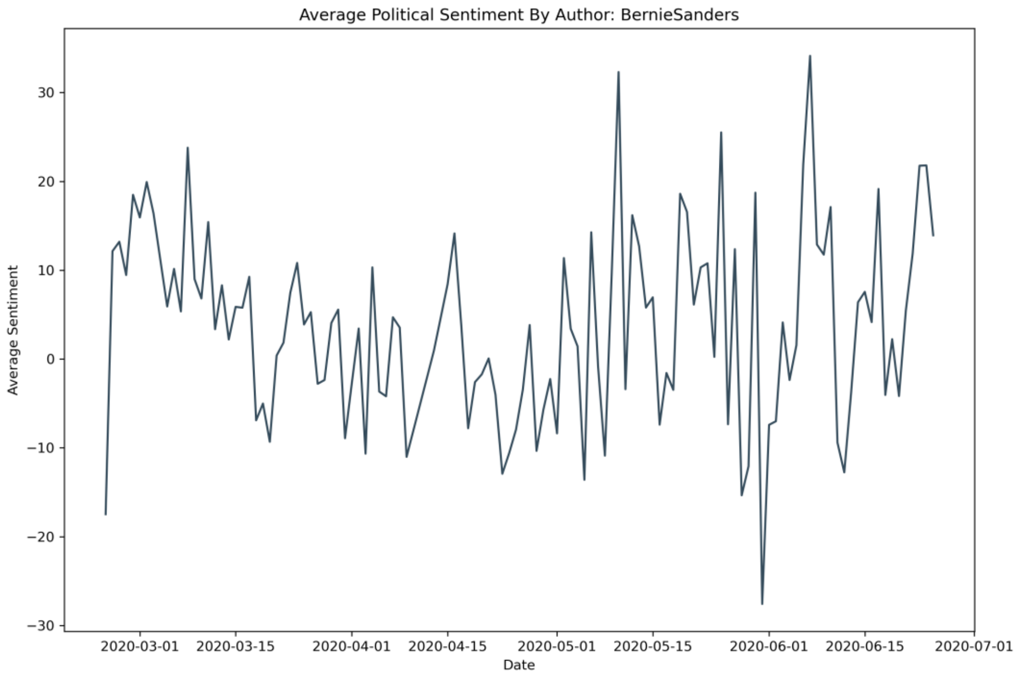


Figure 22: Average Sentiment – Kevin McCarthy

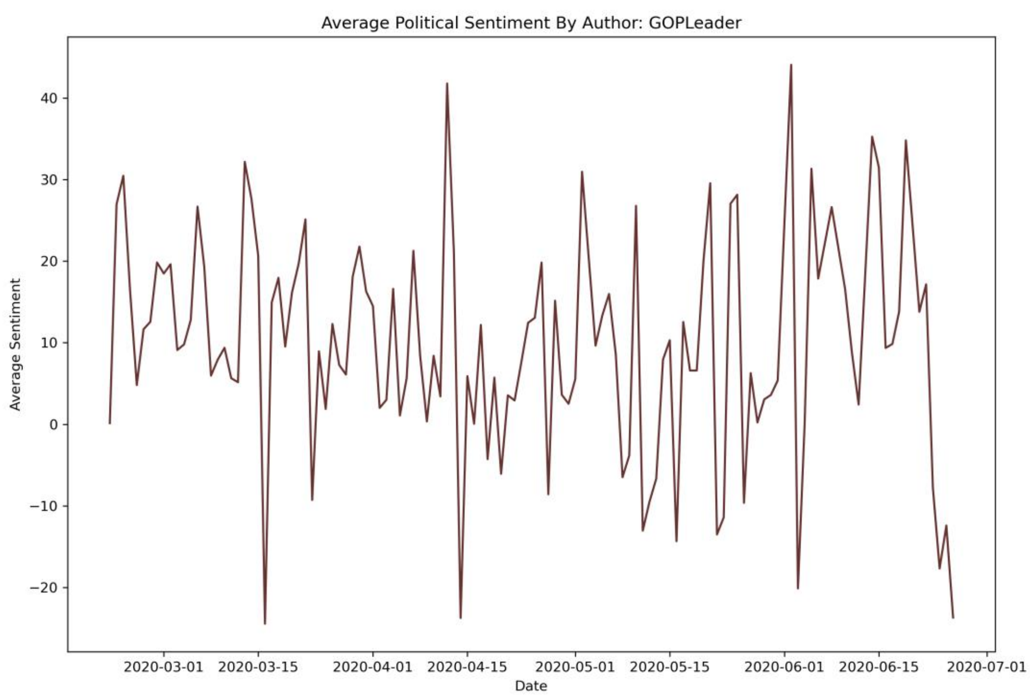


Figure 23: Average Sentiment - Ilhan Omar

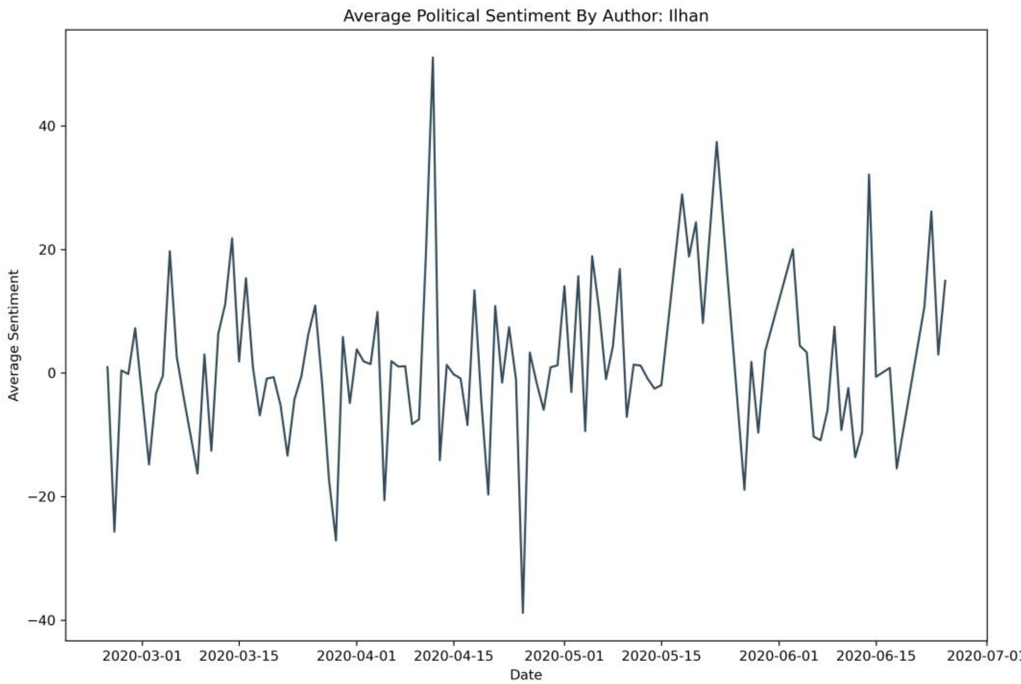


Figure 24: Average Sentiment - Joe Biden

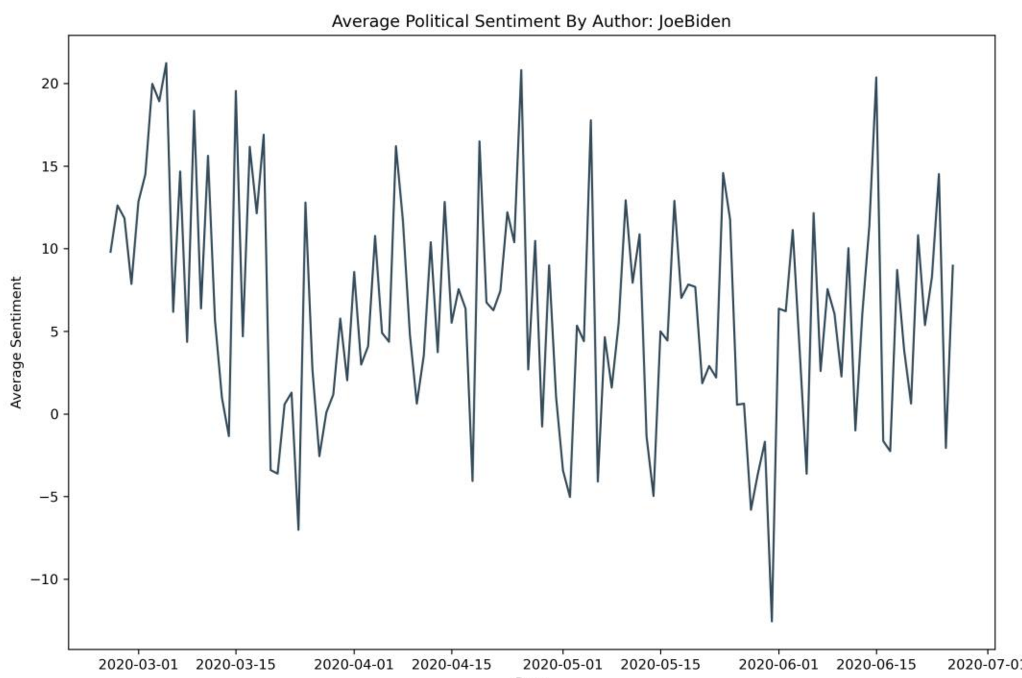


Figure 25: Average Sentiment - Rashida Tlaib

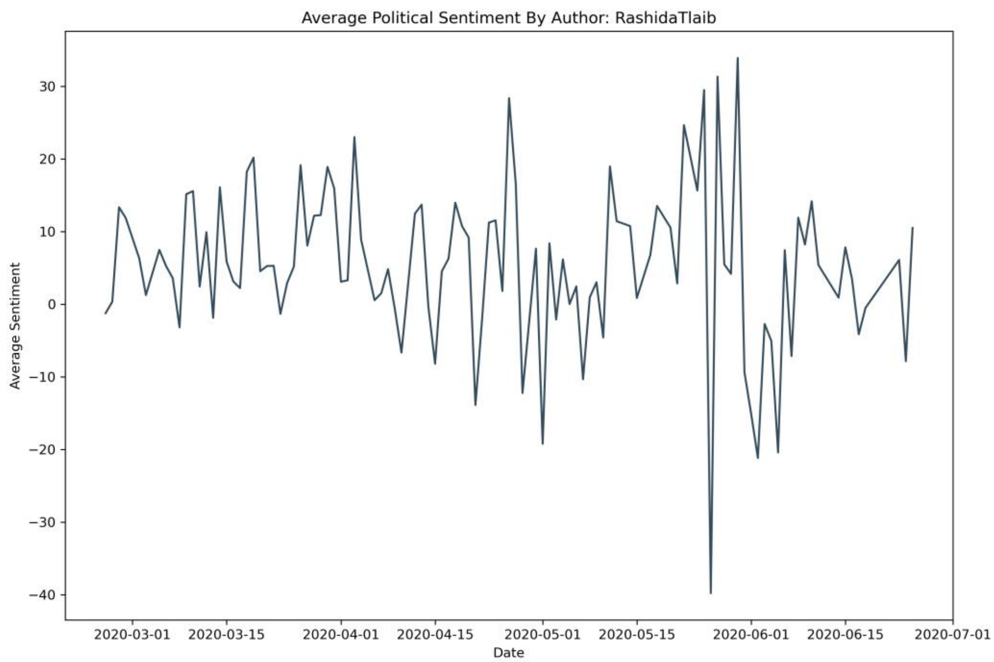


Figure 26: Average Sentiment - Donald Trump

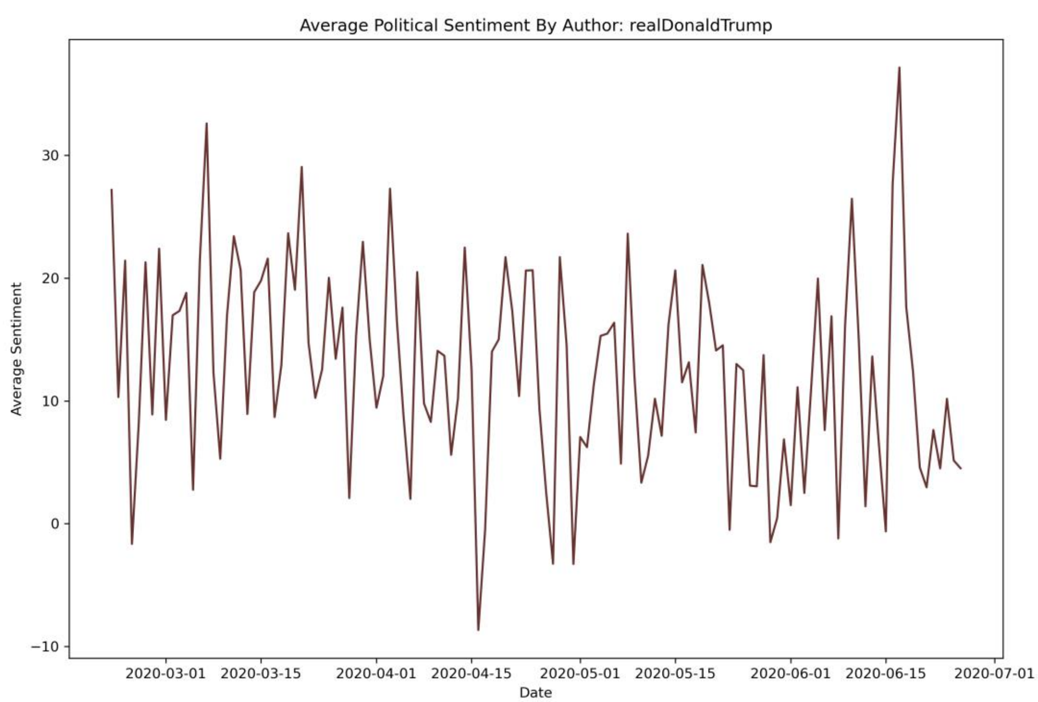


Figure 27: Average Sentiment - Mitch McConnell

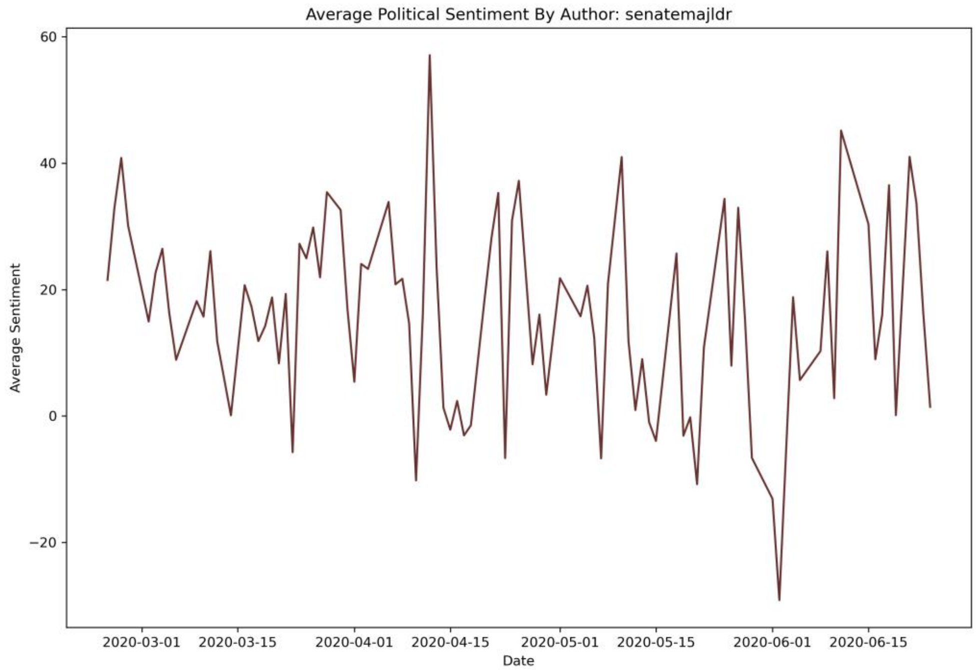


Figure 28: Average Sentiment - Chuck Schumer

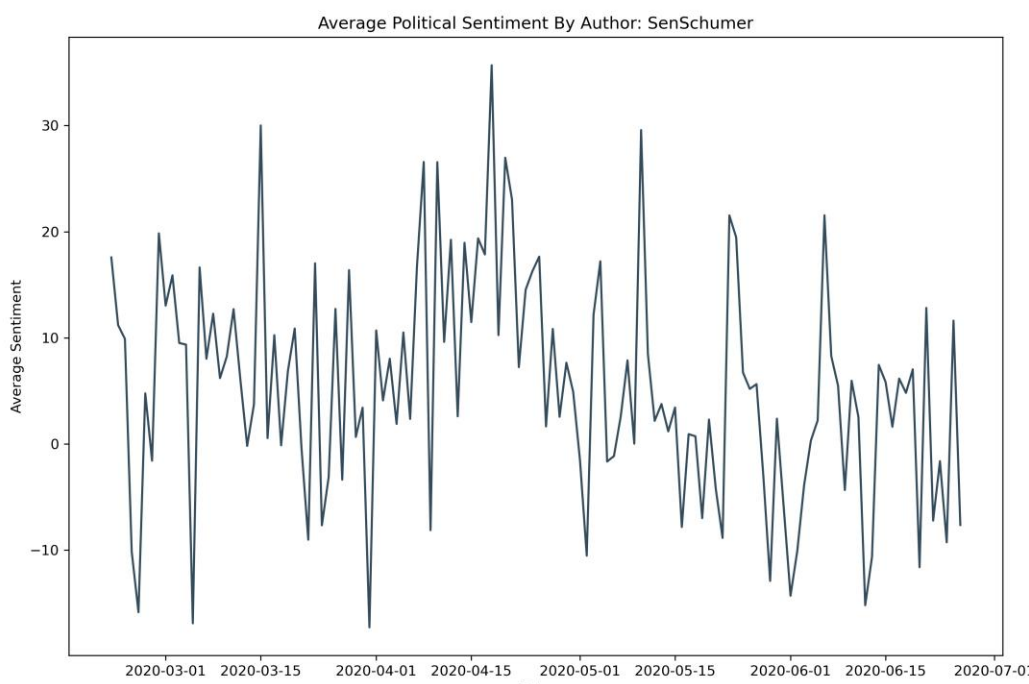


Figure 29: Average Sentiment - Nancy Pelosi

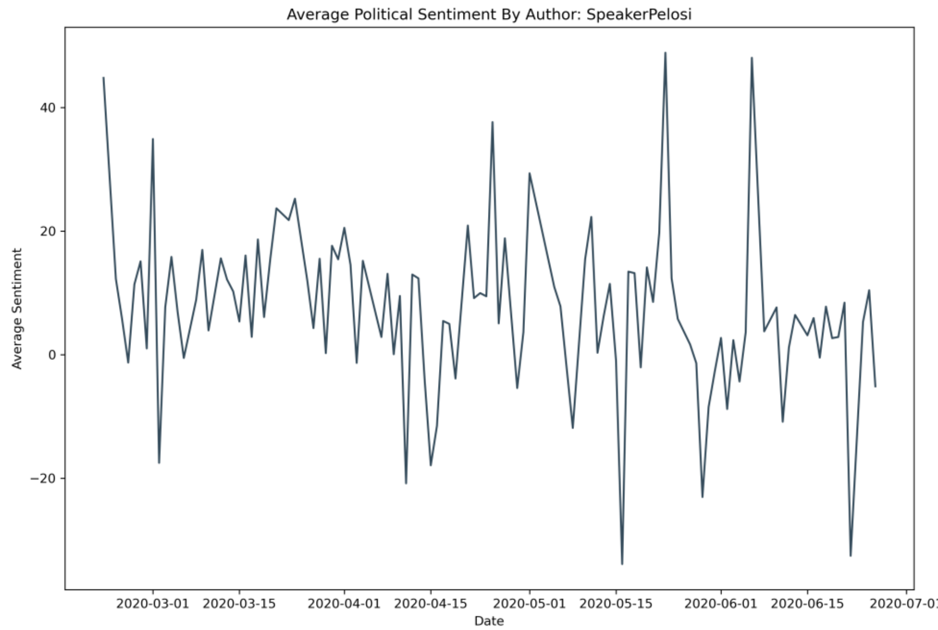


Figure 30: Average Adj. Closing Price: CommServices - Entertainment

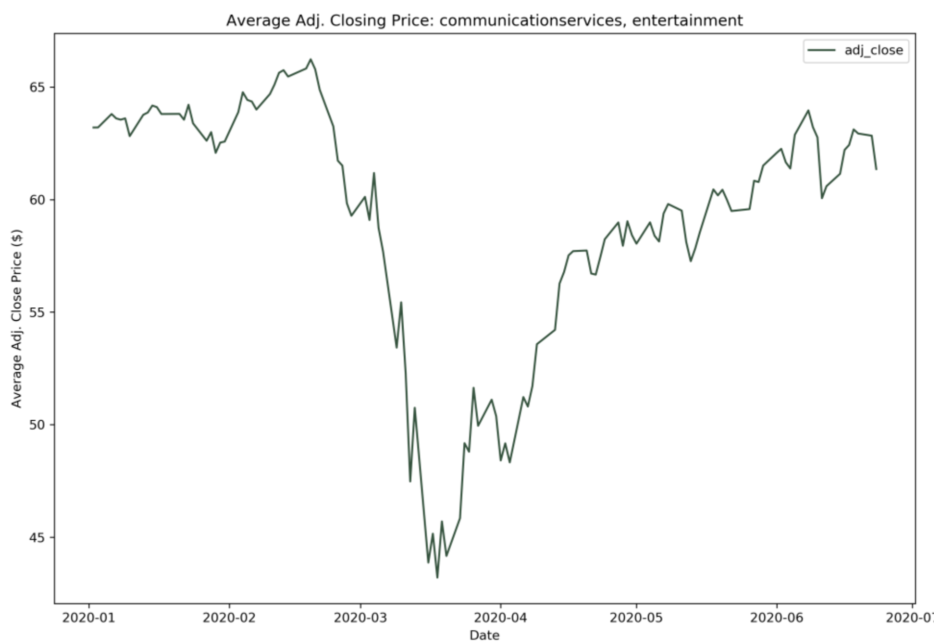


Figure 31: Average Adj. Closing Price: CommServices - Internet Content Info

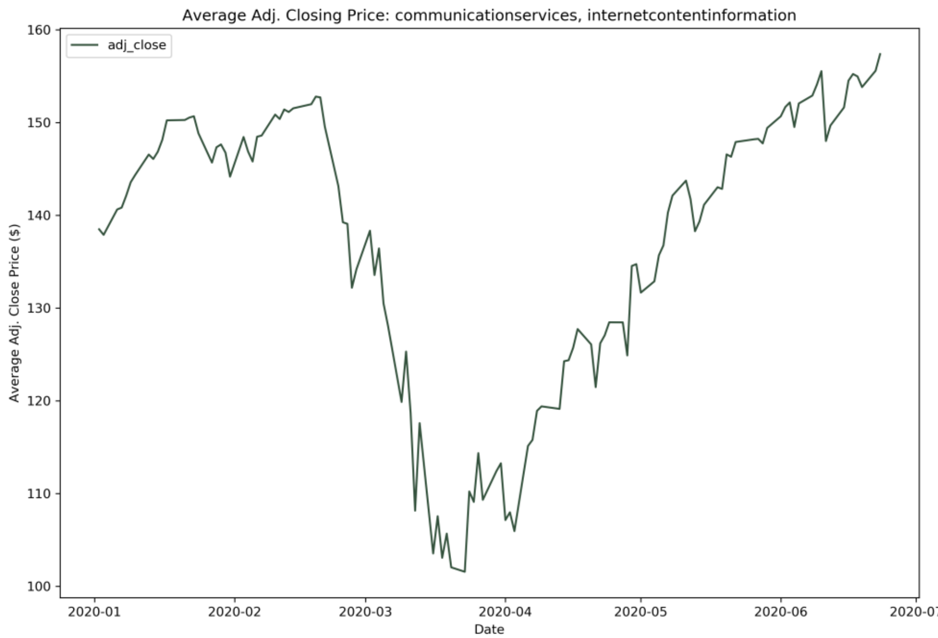


Figure 32: Average Adj. Closing Price: CommServices - Publishing

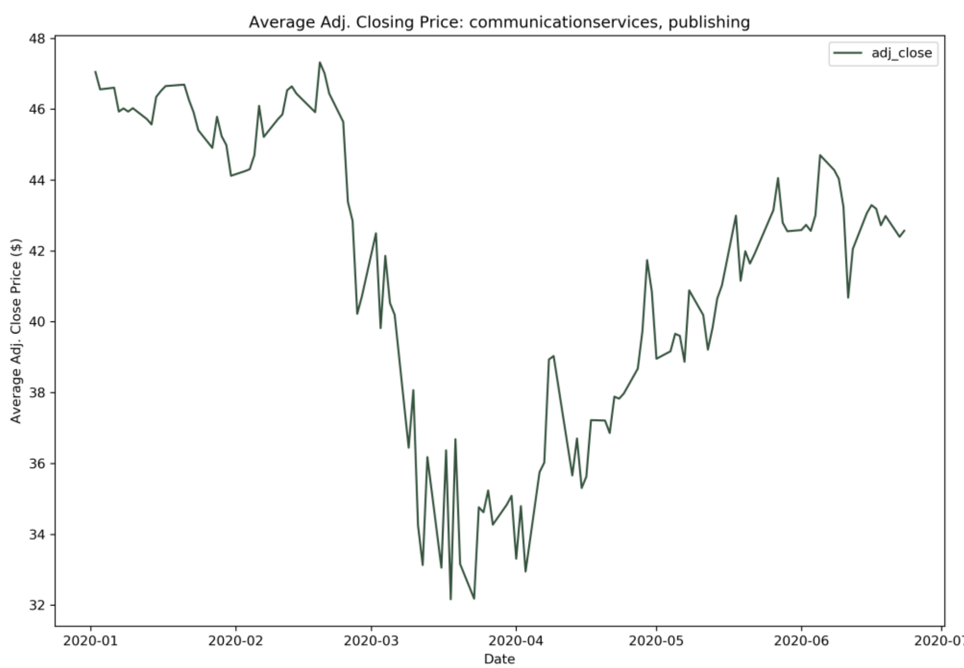


Figure 33: Average Adj. Closing Price: CommServices - TeleComm Services

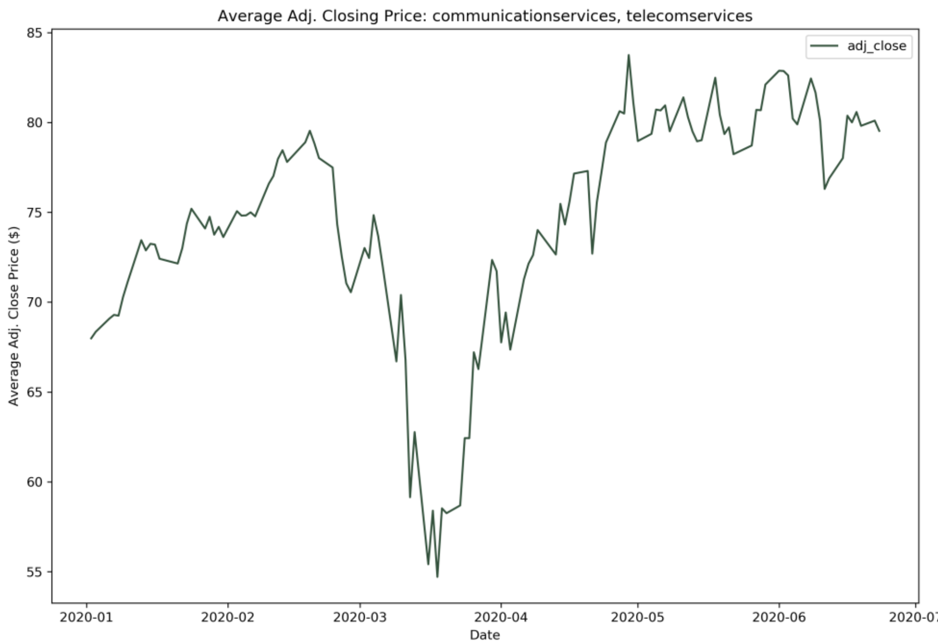


Figure 34: Average Adj. Closing Price: ConCyc - Apparel Manufcaturing

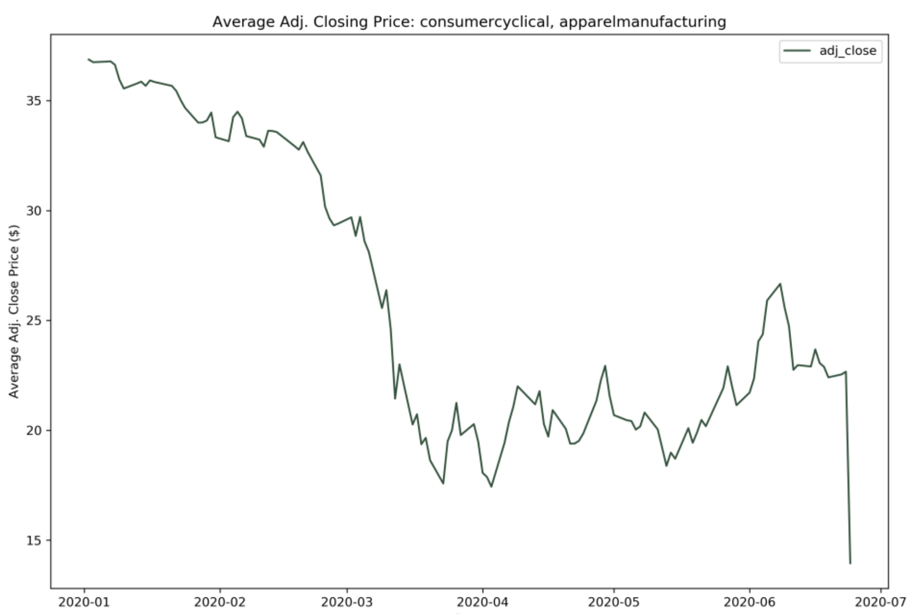


Figure 35Average Adj. Closing Price: ConCyc - Apparel Retail

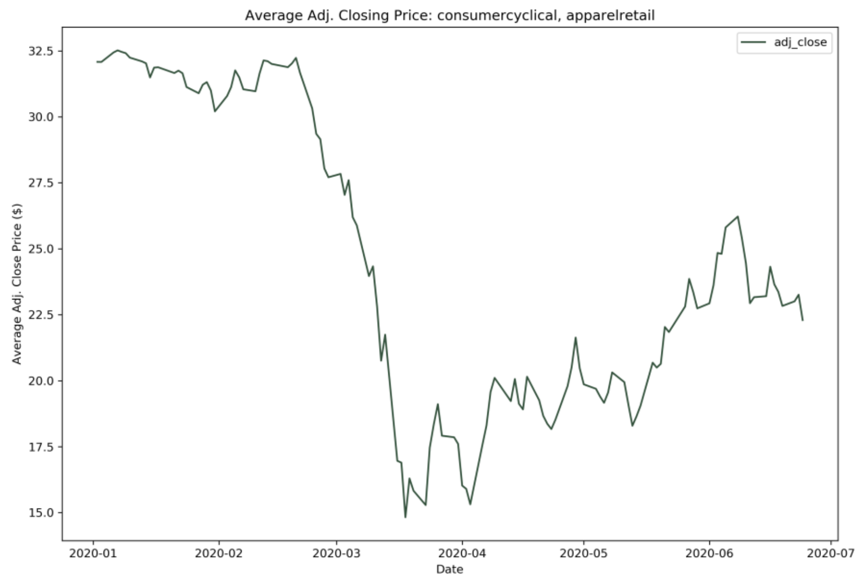


Figure 36: Average Adj. Closing Price: ConCyc - Auto Manufacturers

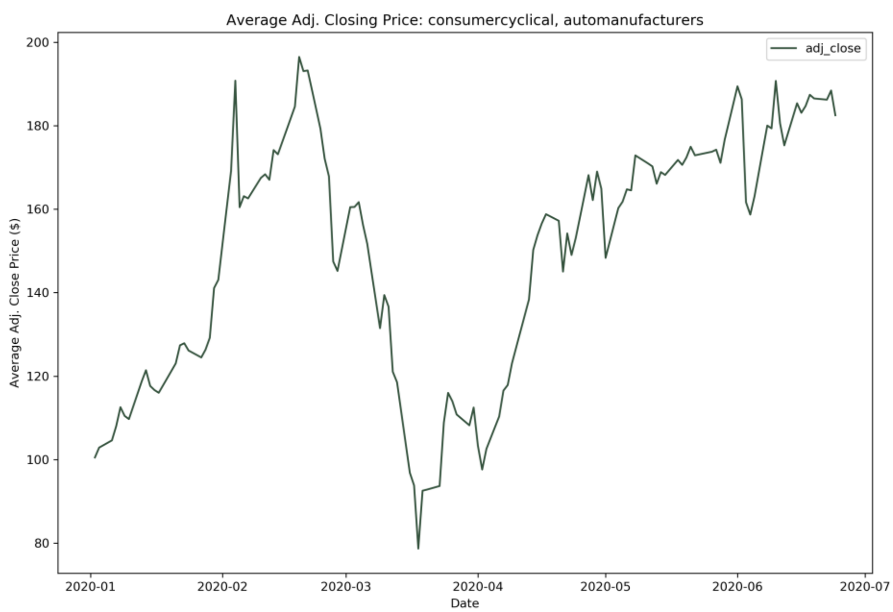


Figure 37: Average Adj. Closing Price: ConCyc - Auto Parts

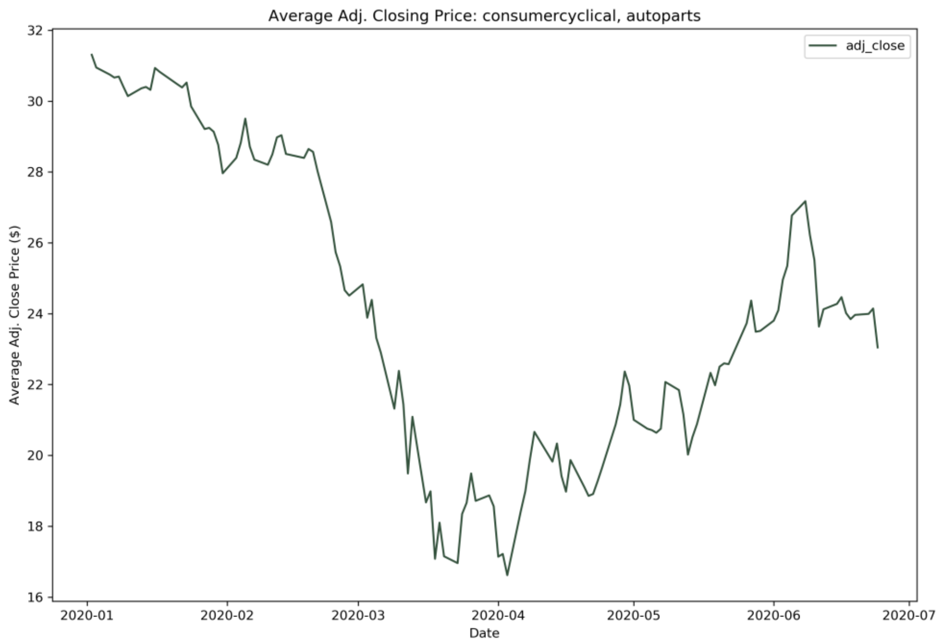


Figure 38: Average Adj. Closing Price: ConCyc -Auto/Truck Dealerships

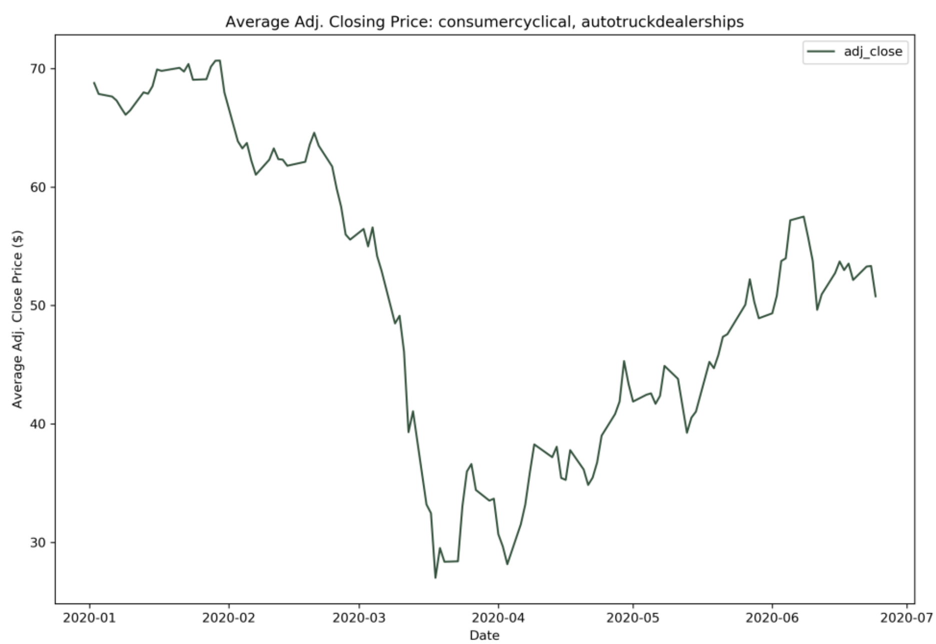


Figure 39: Average Adj. Closing Price: ConCyc - Footwear Accessories

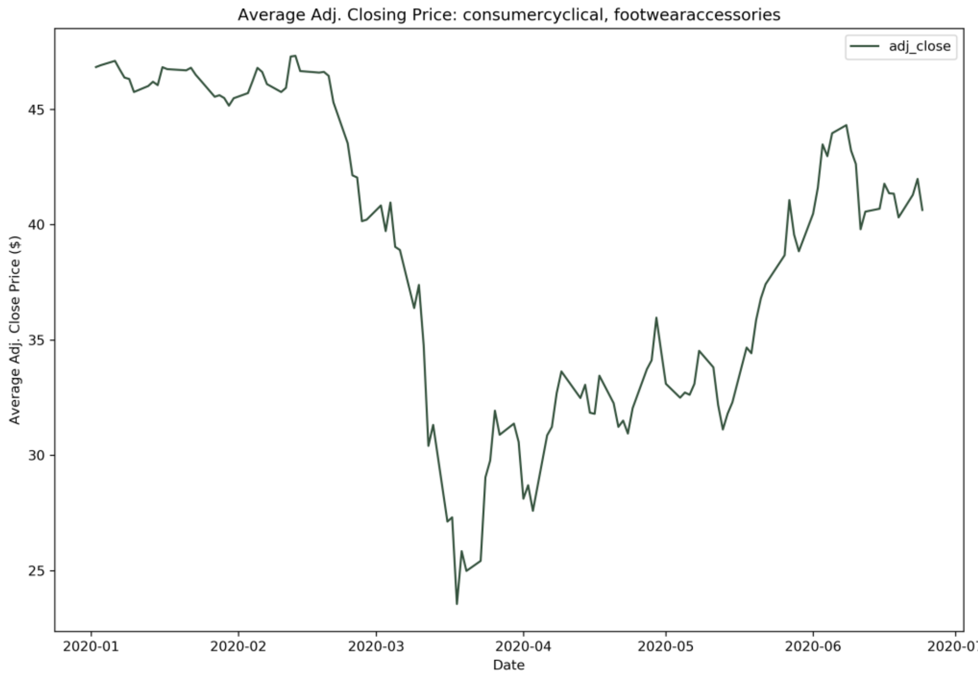


Figure 40: Average Adj. Closing Price: ConCyc - Furnishings, Fixtures, and Appliances

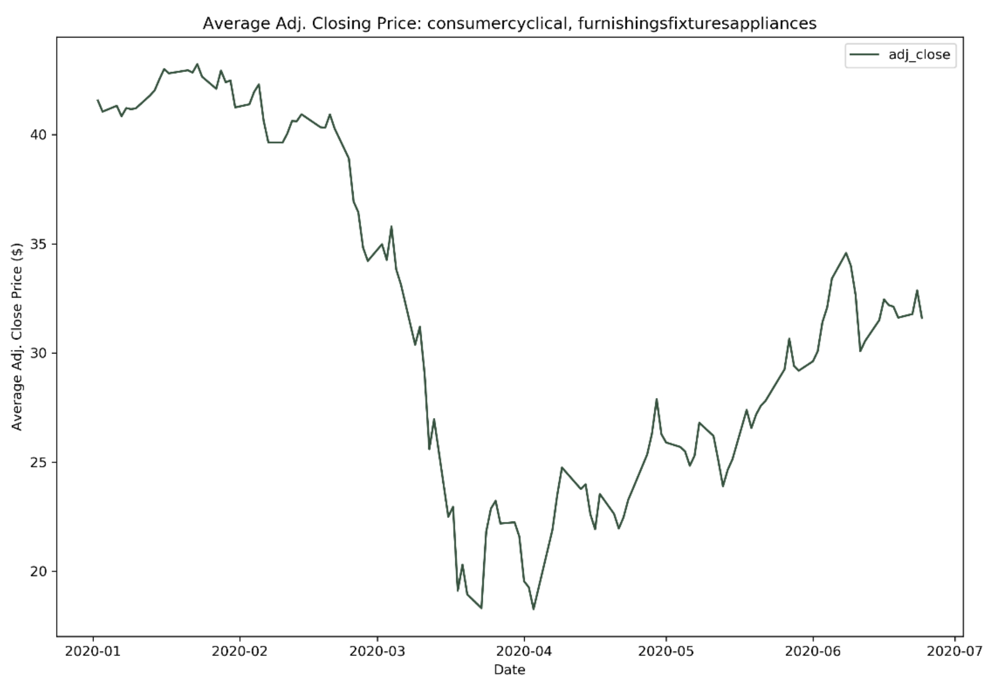


Figure 41: Average Adj. Closing Price: ConCyc - Gambling

Figure 42: Average Adj. Closing Price: ConCyc - Home Improvement Retail

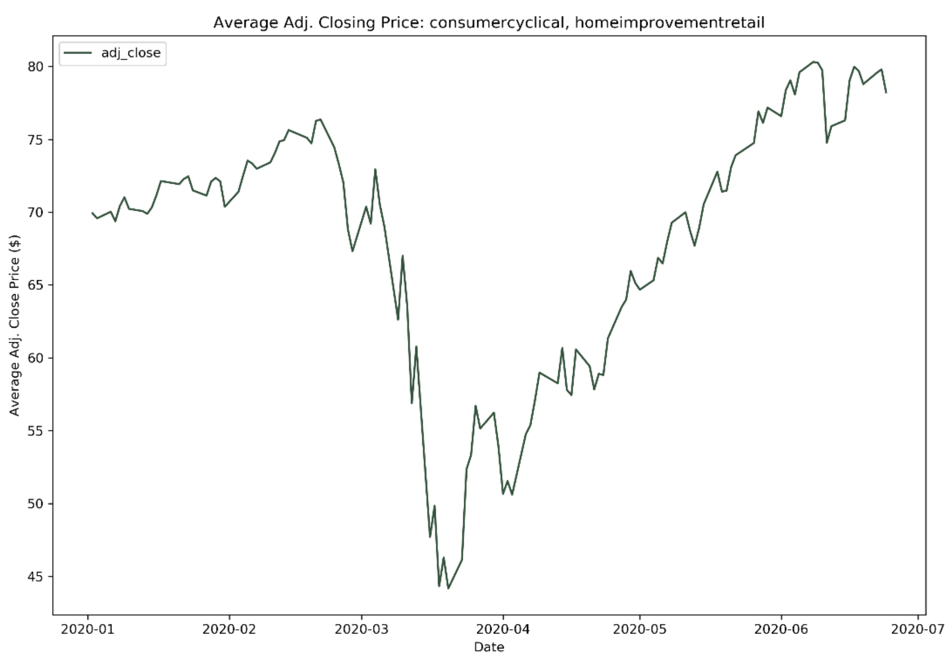


Figure 43: Average Adj. Closing Price: ConCyc - Leisure

Figure 44: Average Adj. Closing Price: ConCyc - Lodging

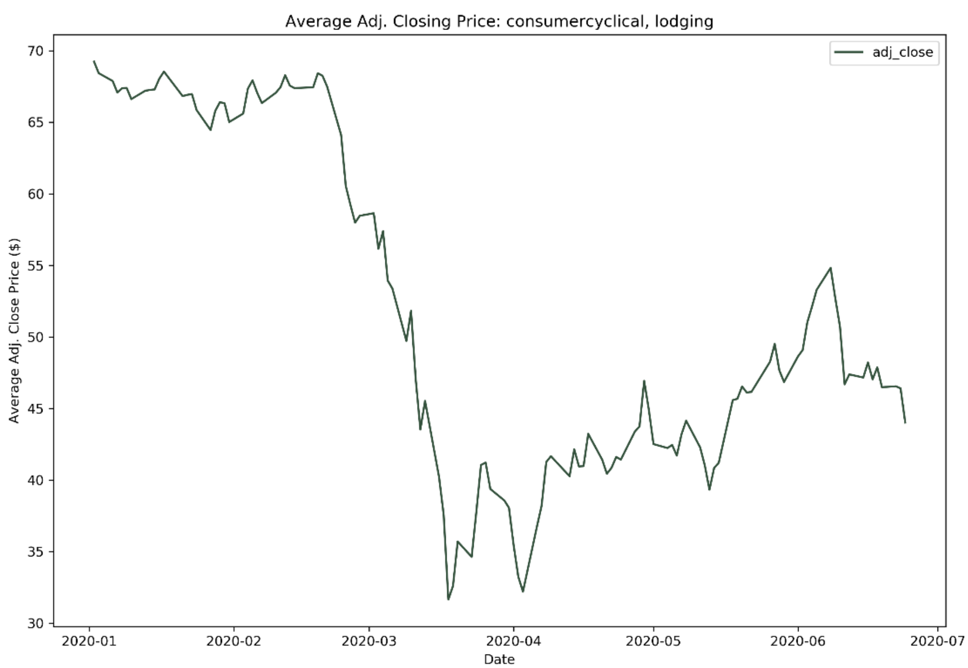


Figure 45: Average Adj. Closing Price: ConCyc - Luxury Goods

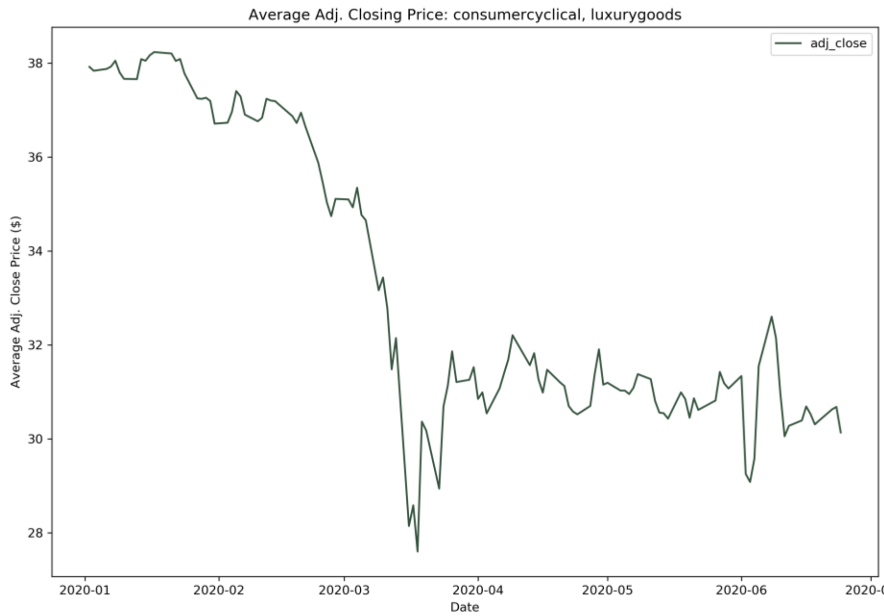


Figure 46: Average Adj. Closing Price: ConCyc - Packaging Containers

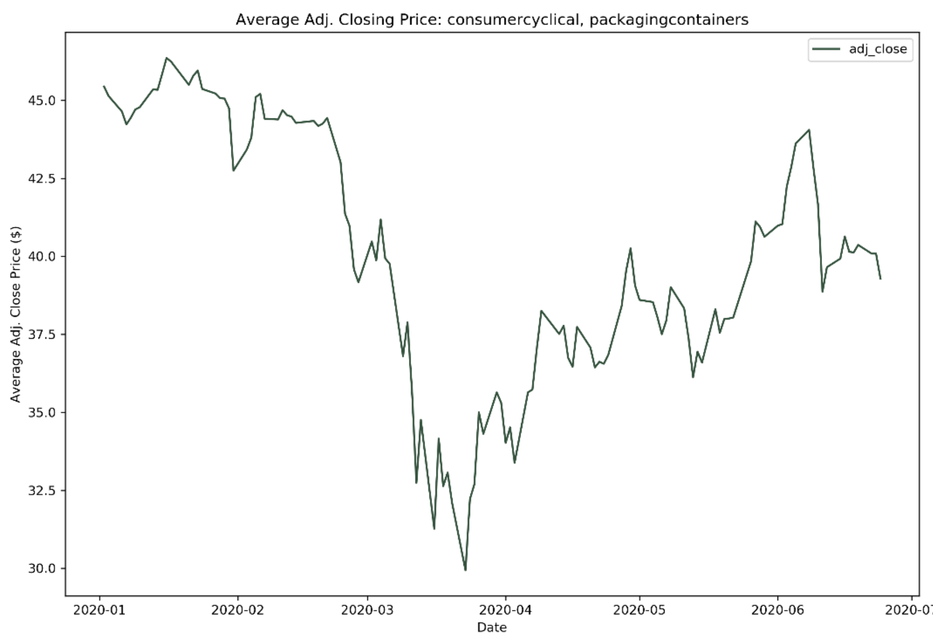


Figure 47: Average Adj. Closing Price: ConCyc - Personal Services

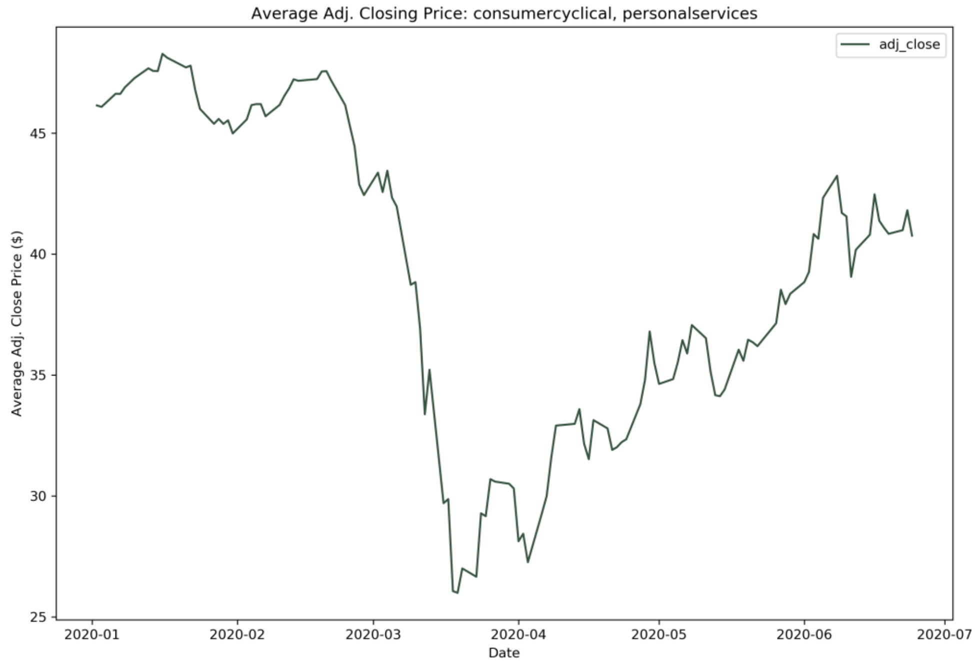


Figure 48: Average Adj. Closing Price: ConCyc - Recreational Vehicles

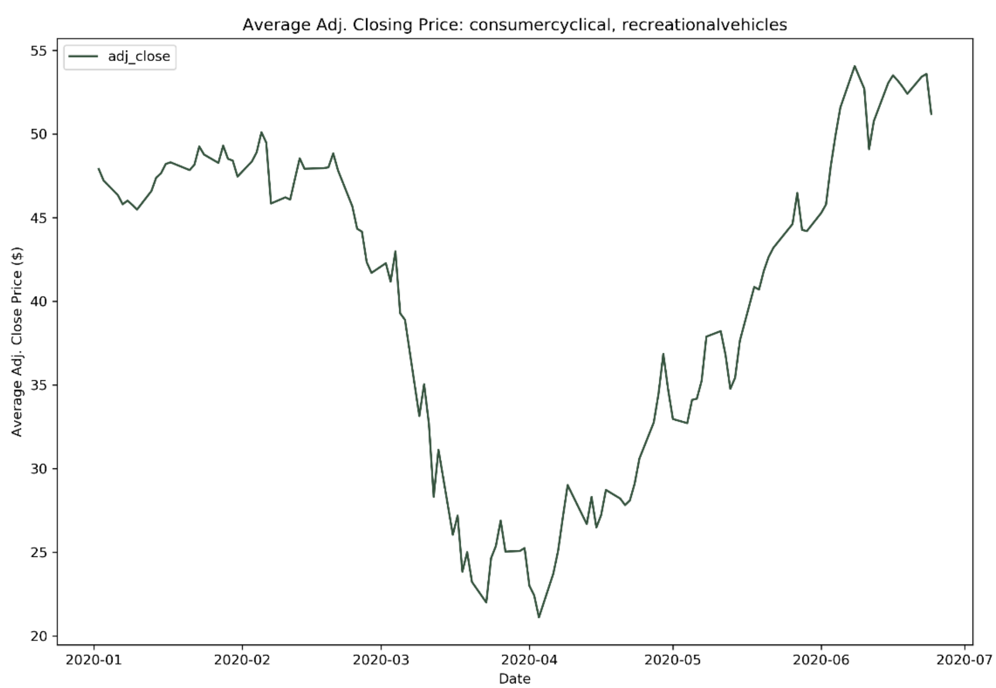


Figure 49: Average Adj. Closing Price: ConCyc - Residential Construction

Figure 50: Average Adj. Closing Price: ConCyc - Resorts and Casinos

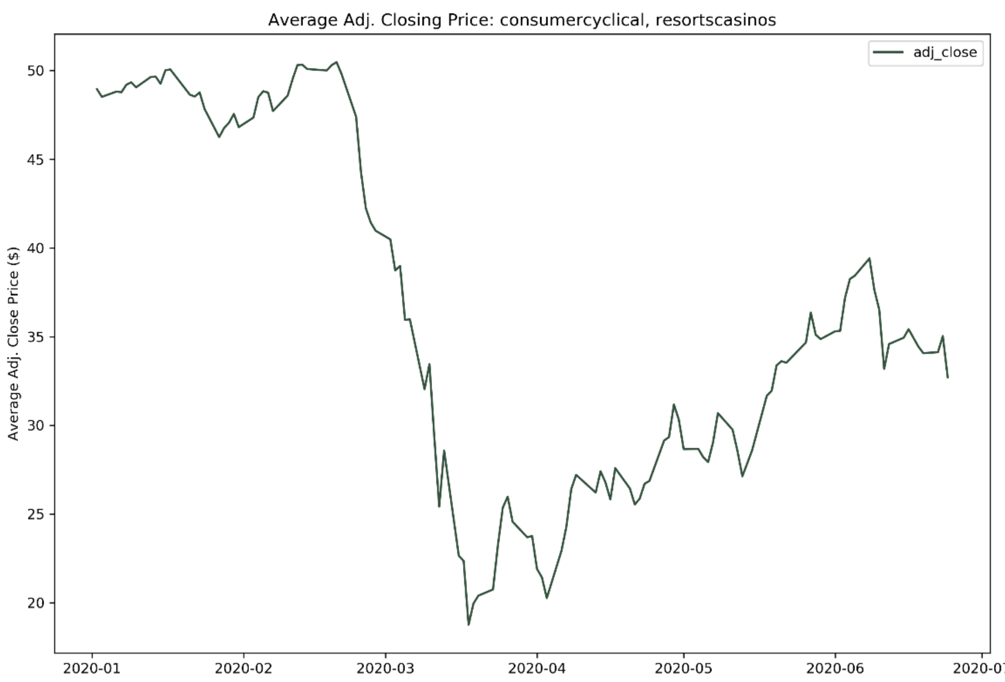


Figure 51: Average Adj. Closing Price: ConCyc - Specialty Retail

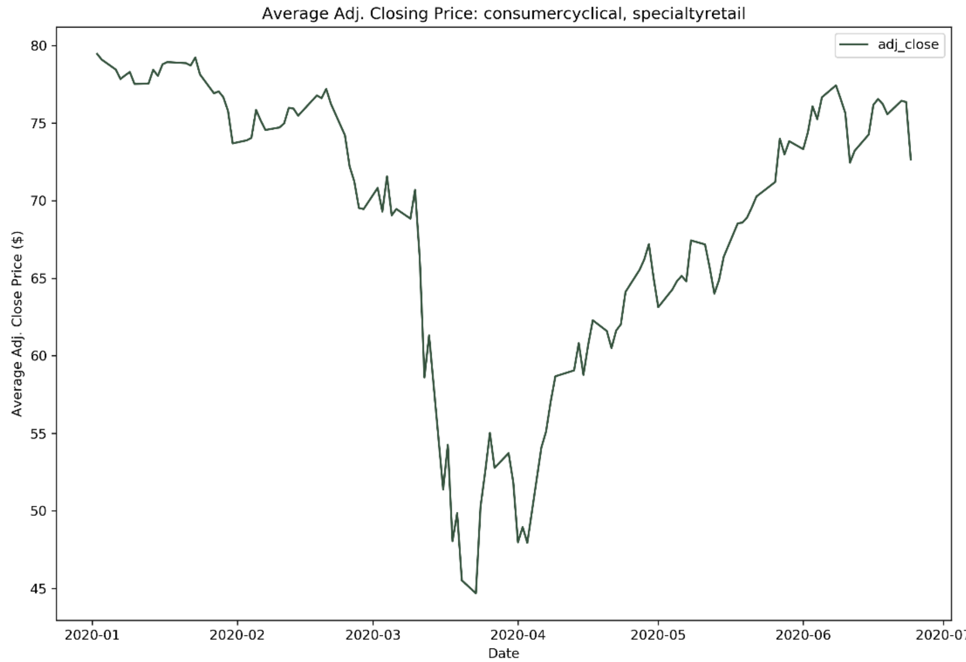


Figure 52: Average Adj. Closing Price: ConCyc - Textile Manufacturing

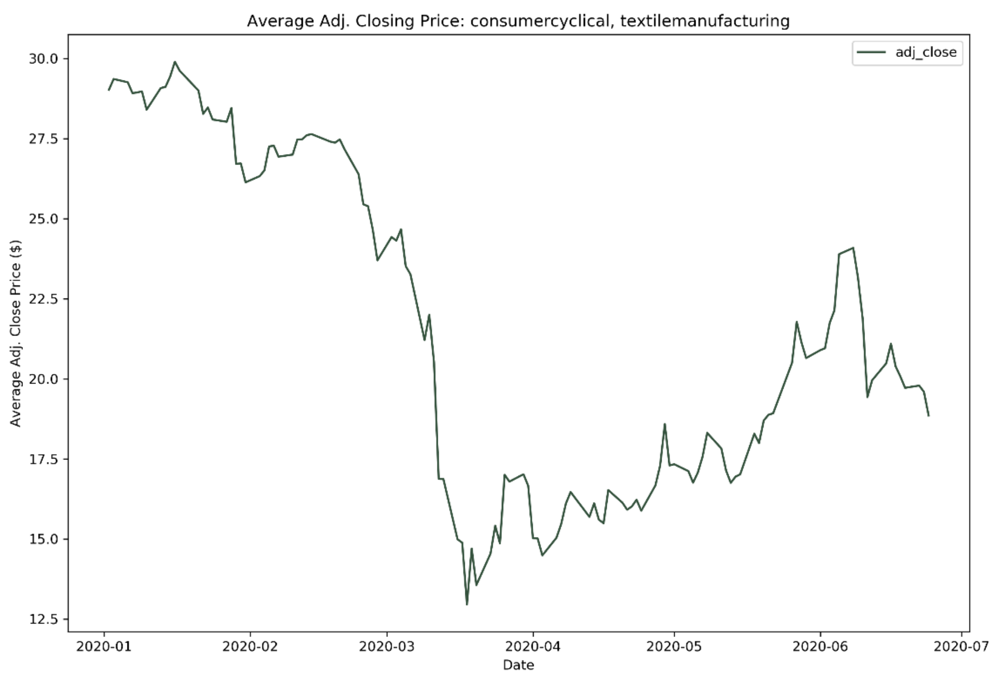


Figure 53: Average Adj. Closing Price: ConCyc - Travel Services

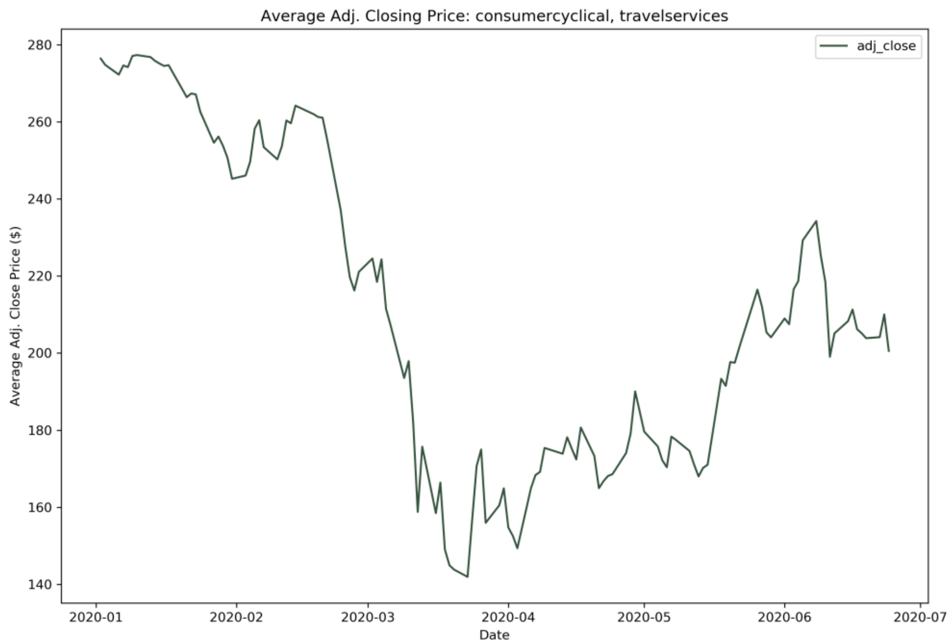


Figure 54: Average Adj. Closing Price: Healthcare - Biotechnology

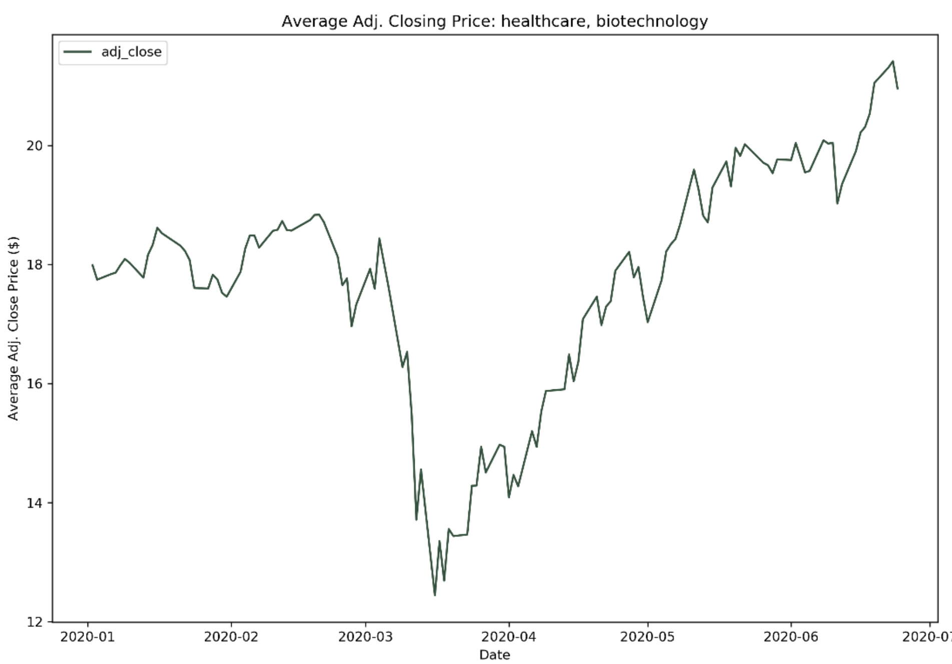


Figure 55: Average Adj. Closing Price: Healthcare - Diagnostics Research

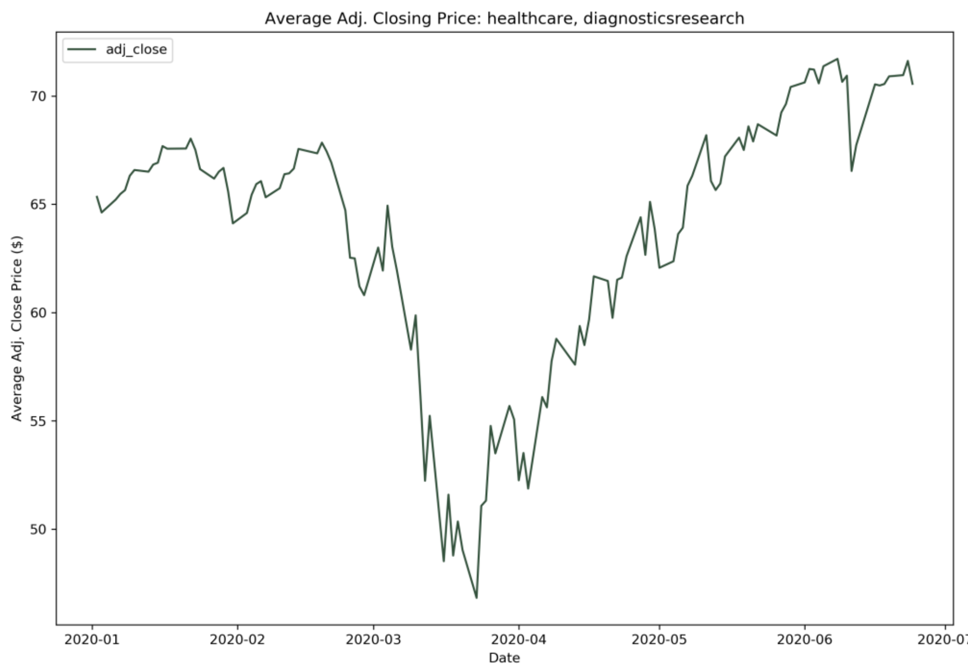


Figure 56: Average Adj. Closing Price: Healthcare - Drug Manufacturers, General

Figure 57: Average Adj. Closing Price: Healthcare - Drug Manufacturers, Specialty/Generic

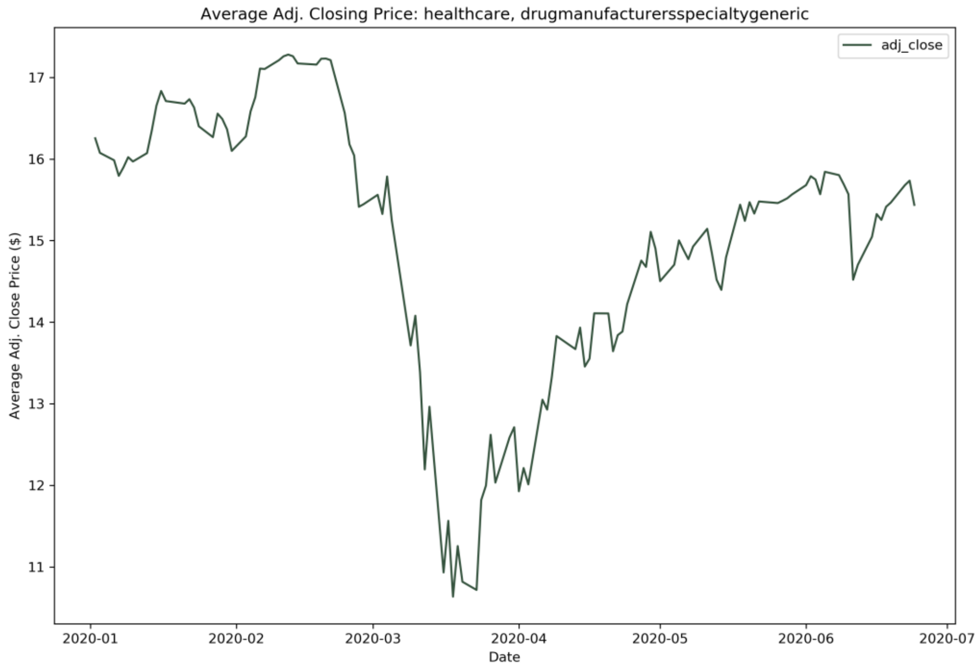


Figure 58: Average Adj. Closing Price: Healthcare - Healthcare Plans

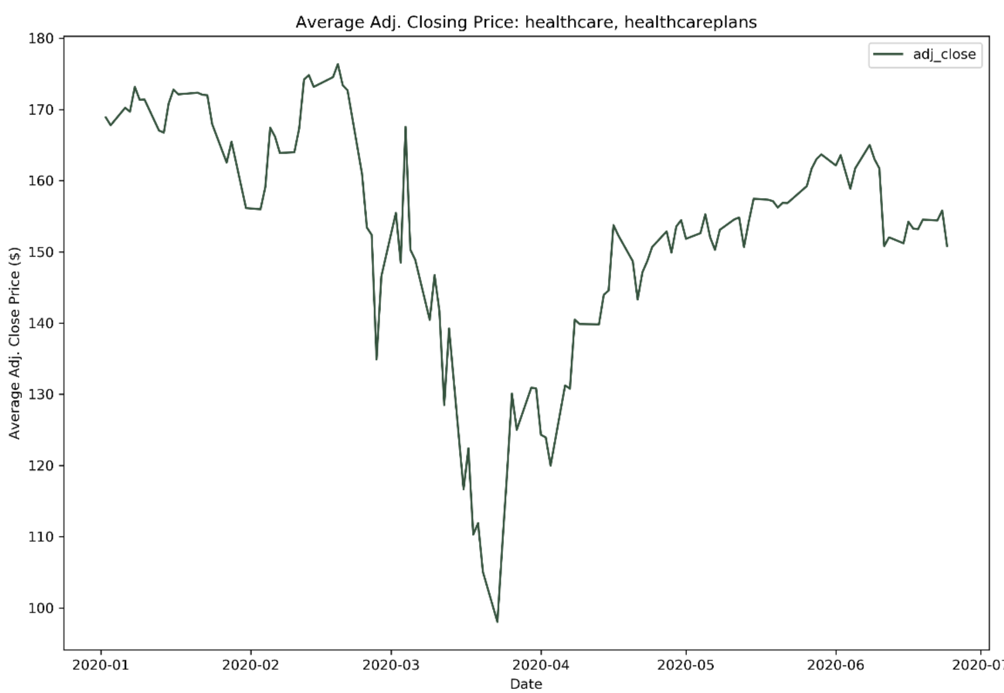


Figure 59: Average Adj. Closing Price: Healthcare - Medical Devices

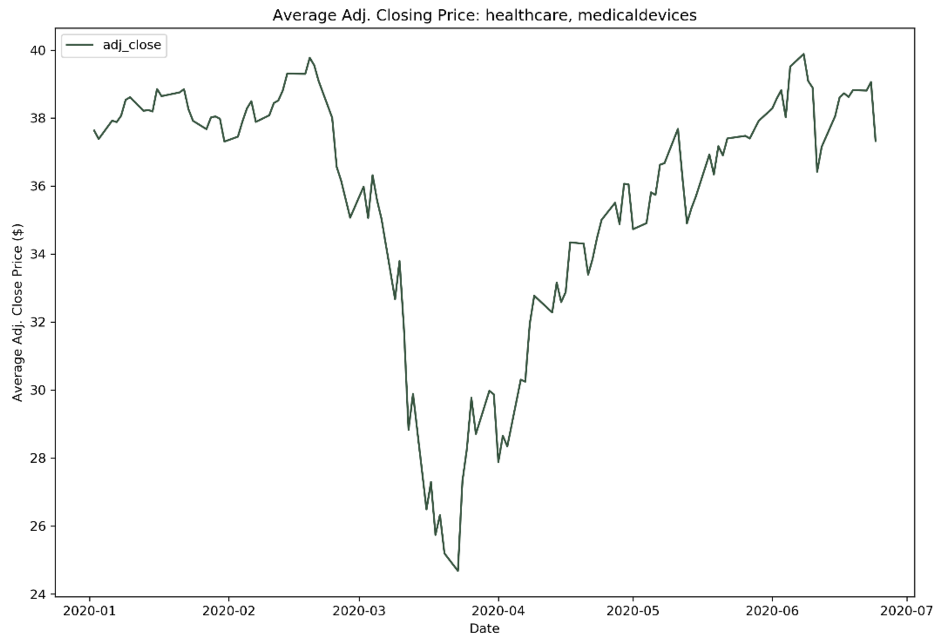


Figure 60: Average Adj. Closing Price: Healthcare - Medical Distribution

Figure 61: Average Adj. Closing Price: Healthcare - Medical Instruments/Supplies

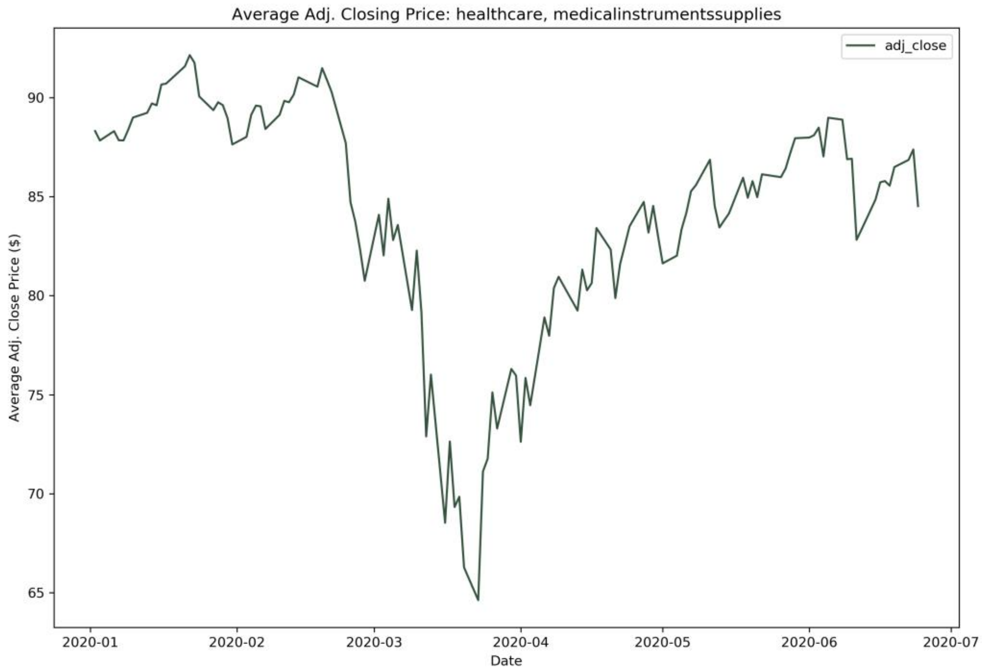


Figure 62: Average Adj. Closing Price: Industrials - Aerospace/Defense

Figure 63: Average Adj. Closing Price: Industrials - Airports & Air Services

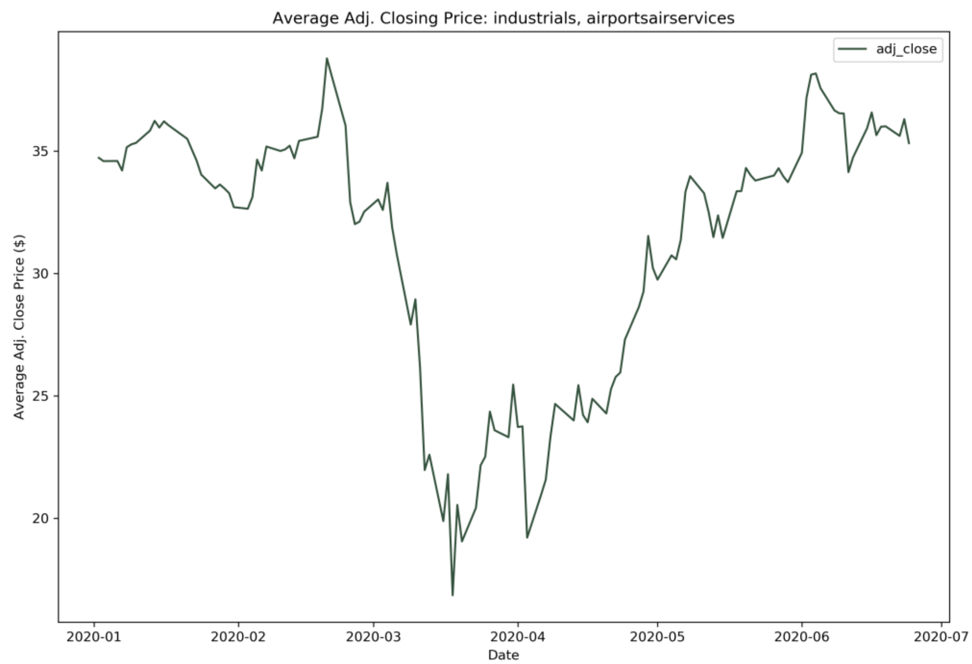


Figure 64: Average Adj. Closing Price: Industrials - Building Products/Equipment

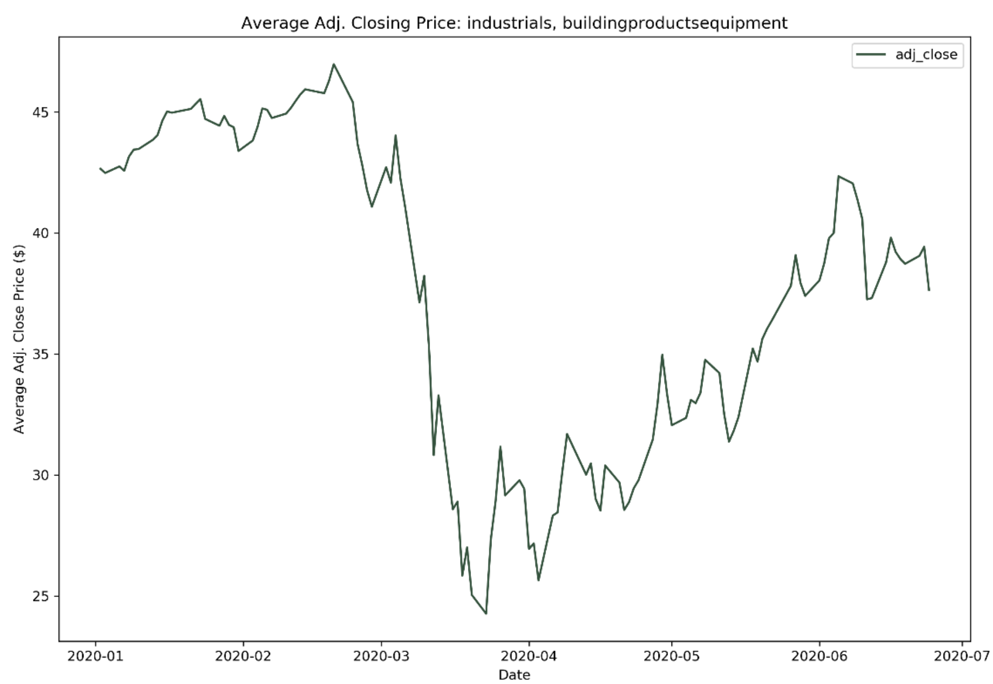


Figure 65: Average Adj. Closing Price: Industrials - Business Equipment/Supplies

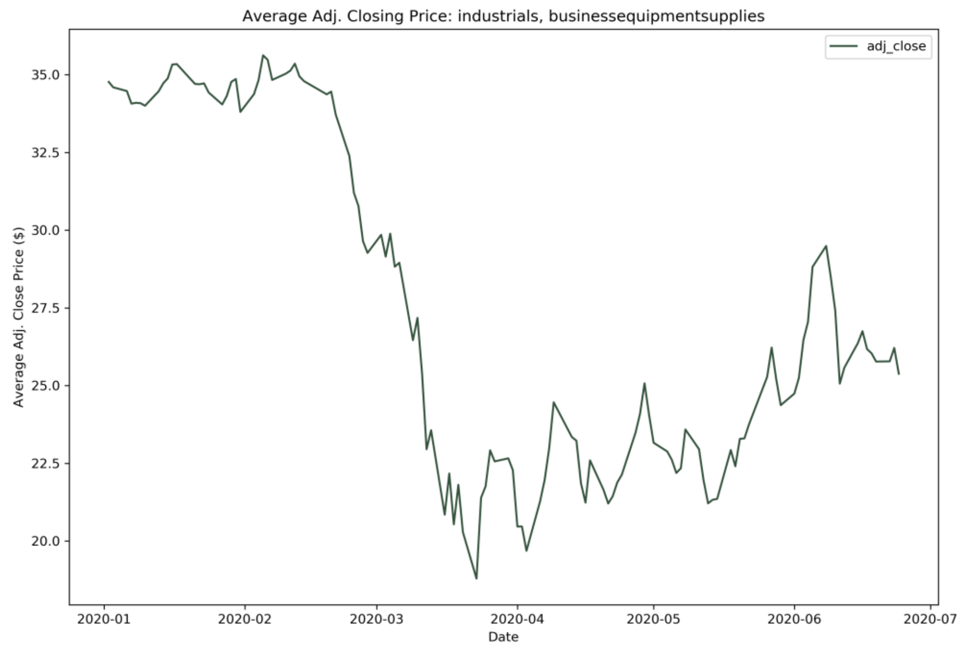


Figure 66: Average Adj. Closing Price: Industrials - Conglomerates

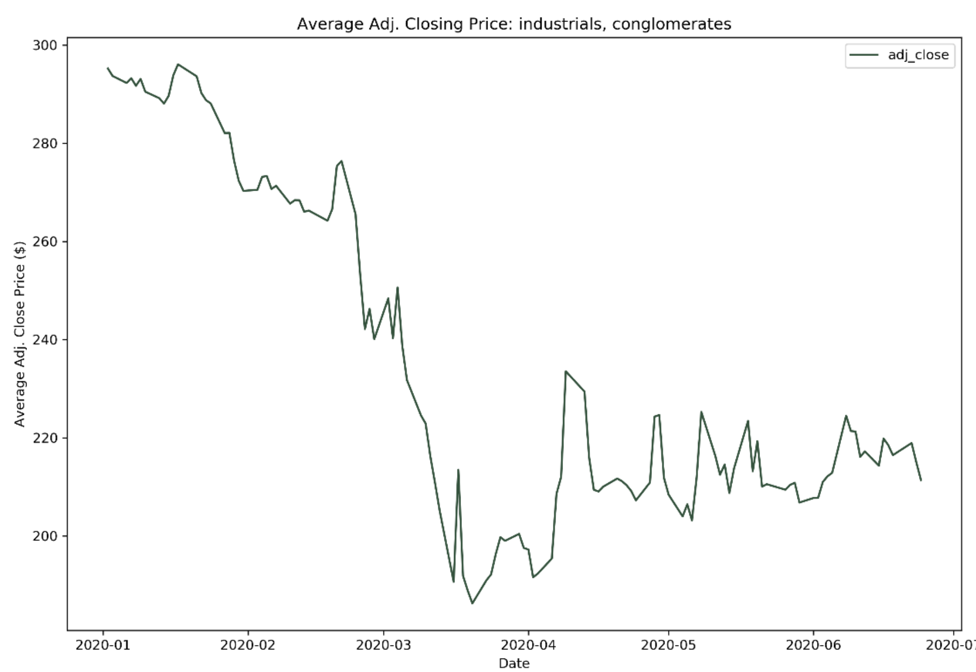


Figure 67: Average Adj. Closing Price: Industrials - Consulting Services

Figure 68: Average Adj. Closing Price: Industrials - Electrical Equipment/Parts

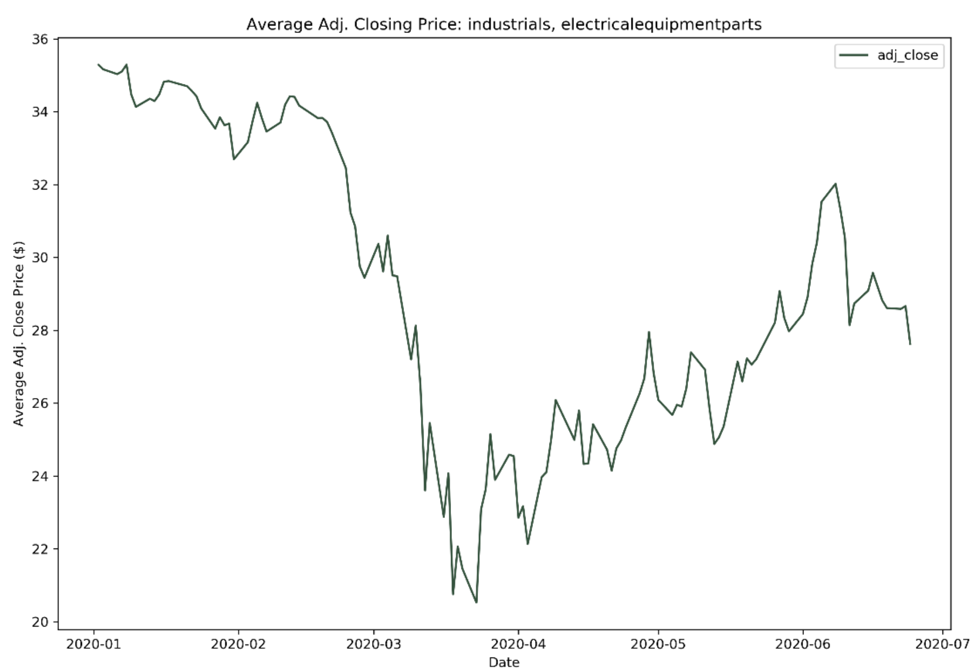


Figure 69: Average Adj. Closing Price: Industrials - Engineering Construction

Figure 70: Average Adj. Closing Price: Industrials - Farm Heavy Construction Machinery

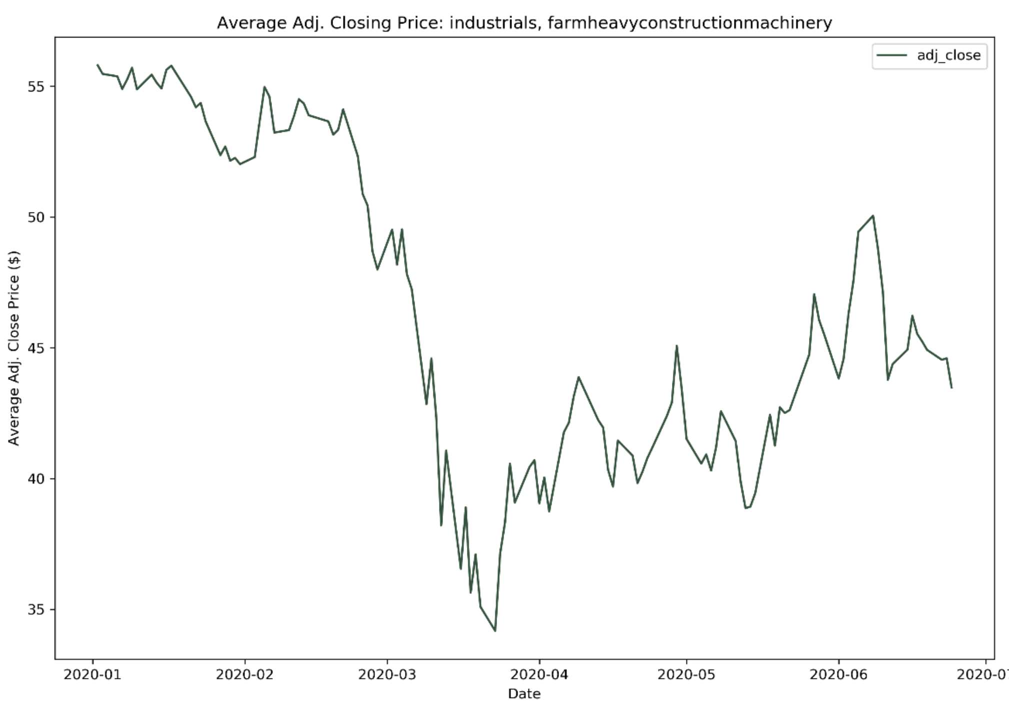


Figure 71: Average Adj. Closing Price: Industrials - Industrial Distribution

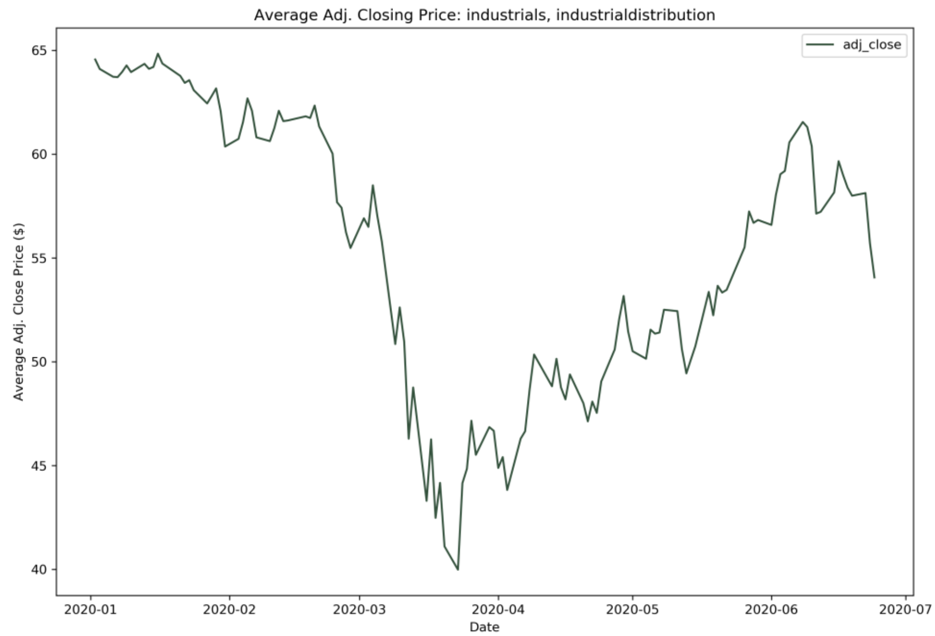


Figure 72: Average Adj. Closing Price: Industrials - Infrastructure Operations

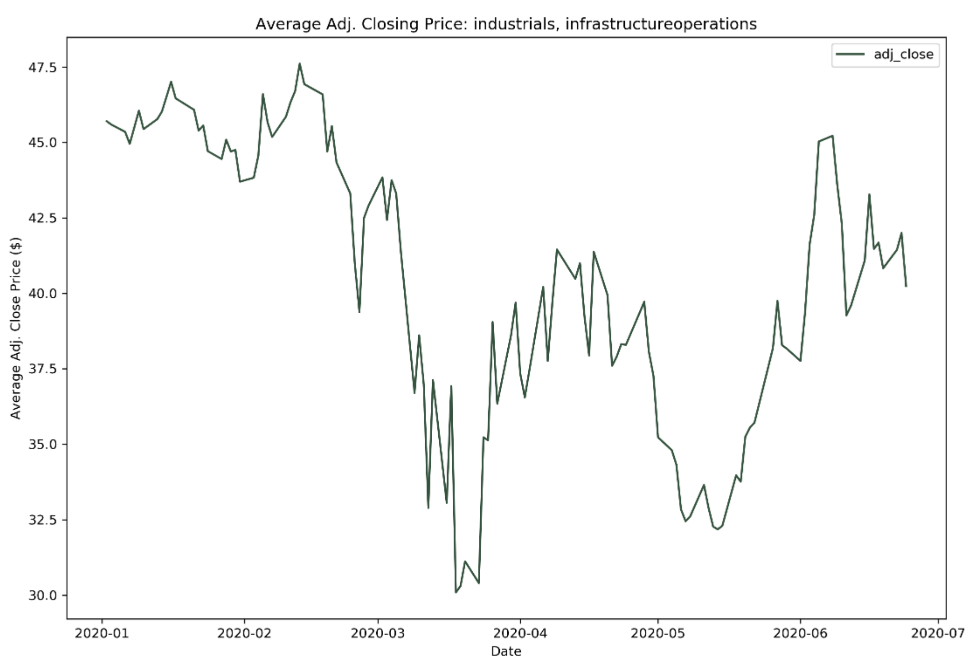


Figure 73: Average Adj. Closing Price: Industrials - Integrated Freight Logistics

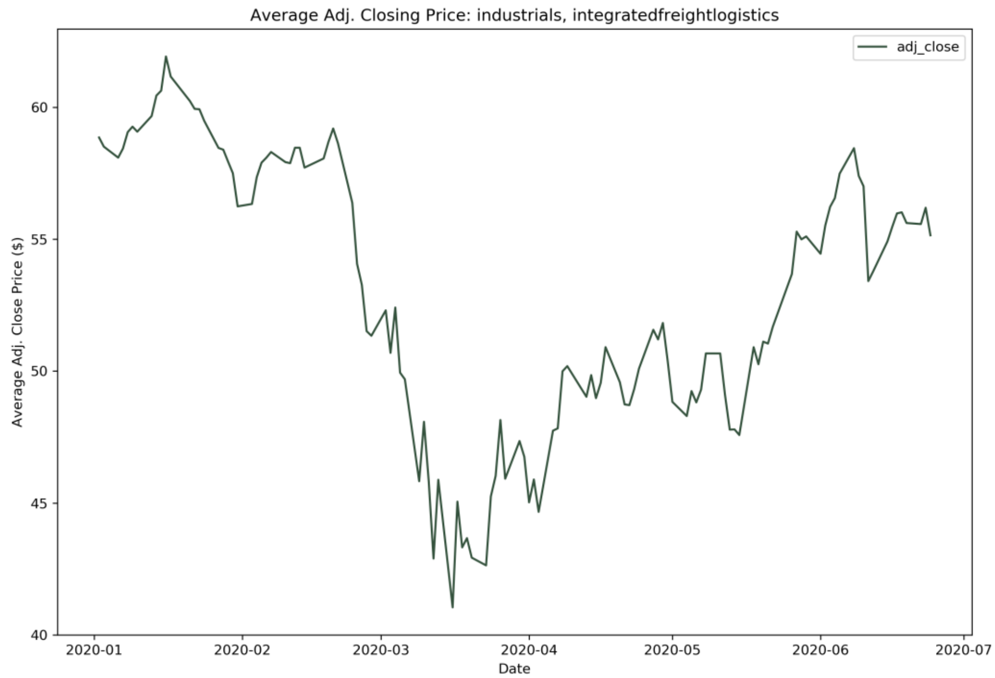


Figure 74: Average Adj. Closing Price: Industrials - Marine Shipping

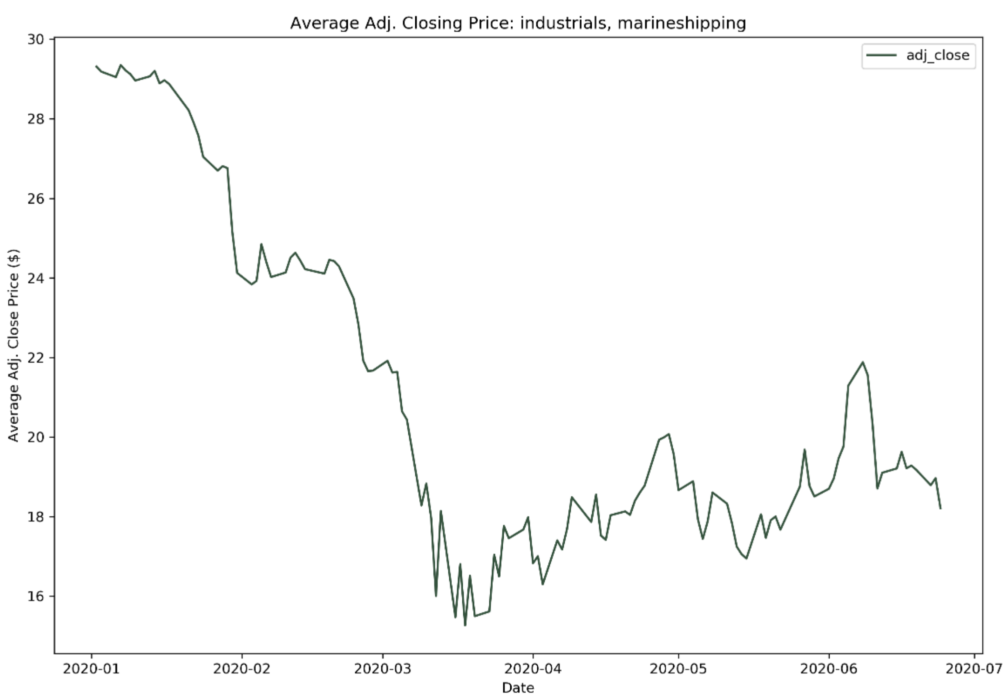


Figure 75: Average Adj. Closing Price: Industrials - Metal Fabrication

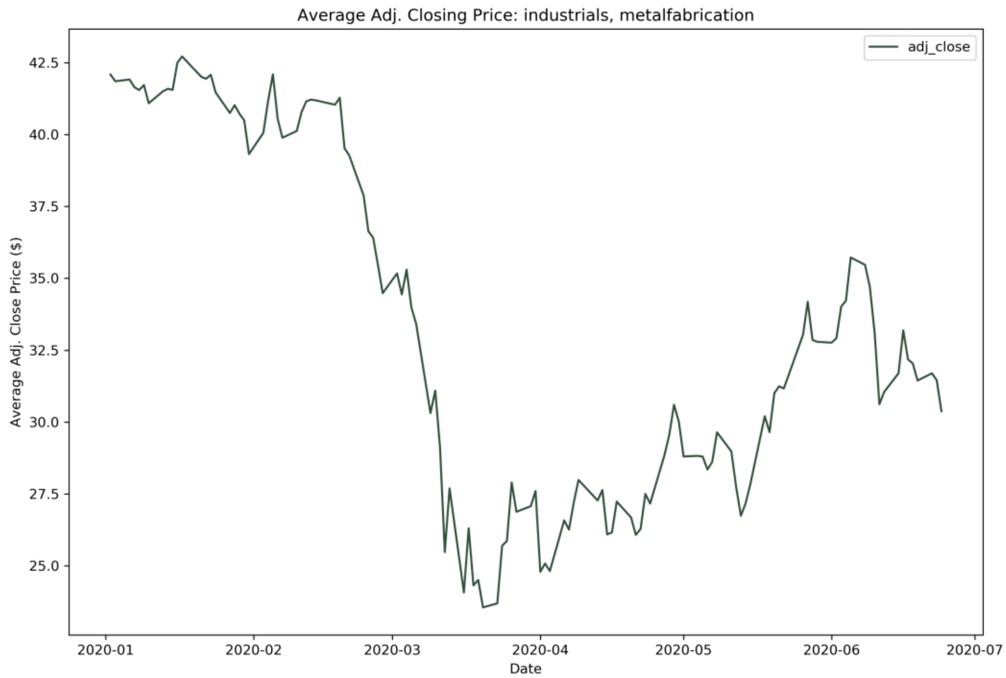


Figure 76: Average Adj. Closing Price: Industrials - Pollution Treatment Controls

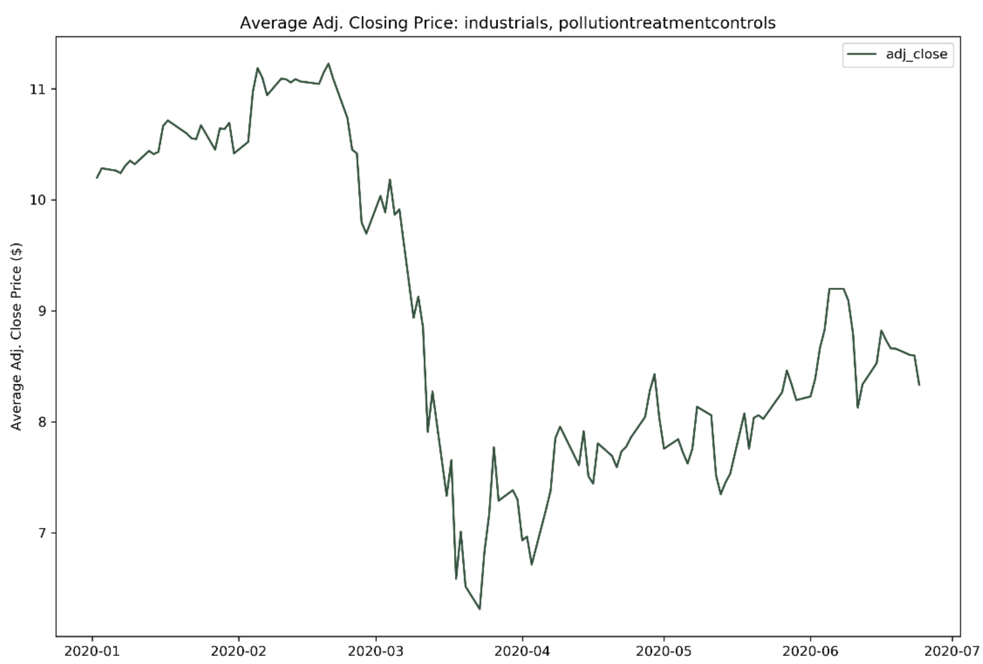


Figure 77: Average Adj. Closing Price: Industrials - Railroads

Figure 78: Average Adj. Closing Price: Industrials - Rental/Leasing Services

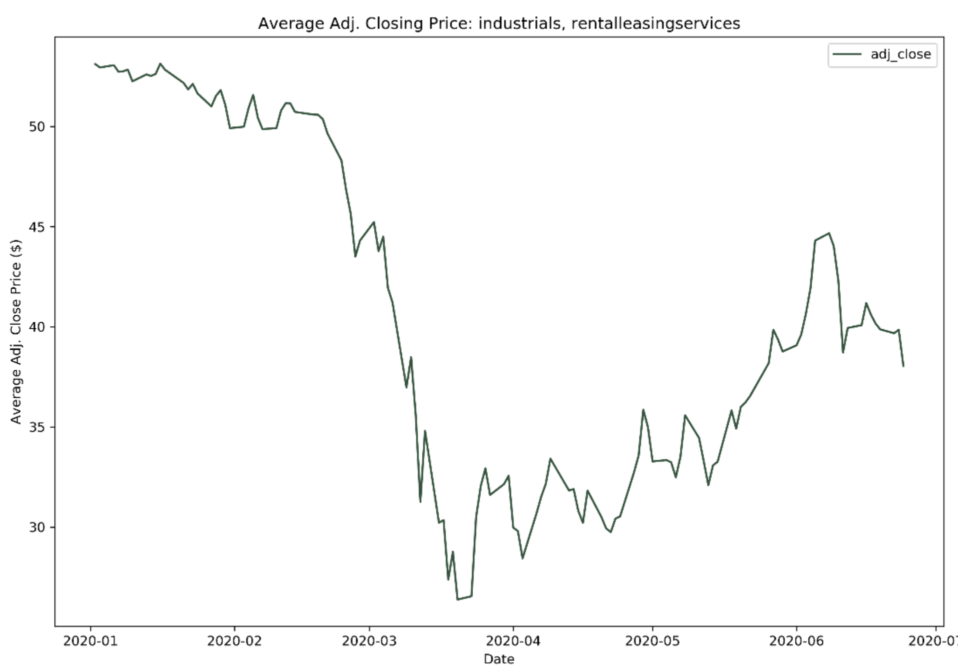


Figure 79: Average Adj. Closing Price: Industrials - Security Protection Services

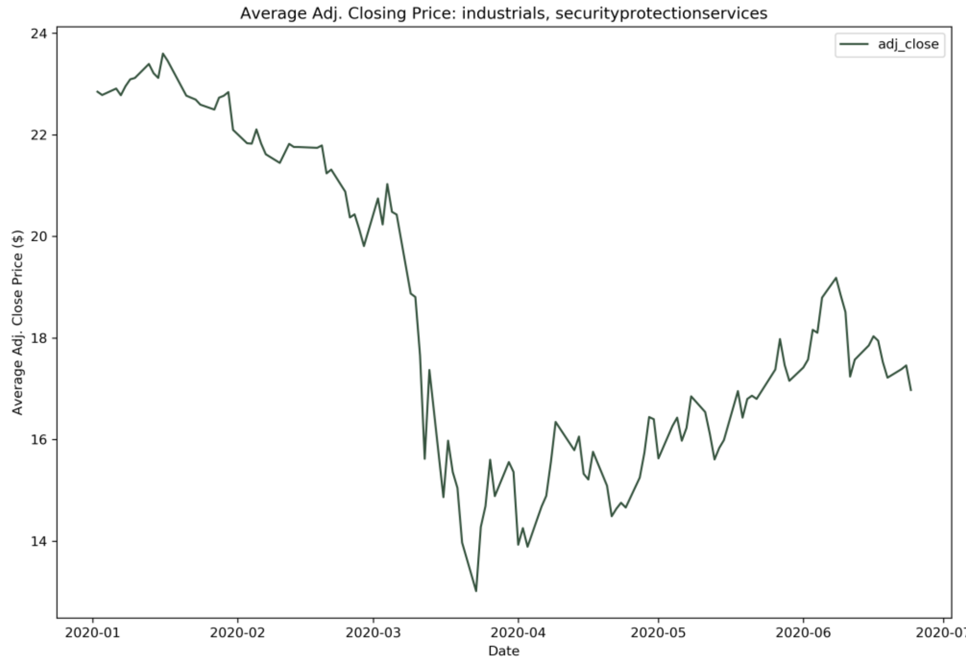


Figure 80: Average Adj. Closing Price: Industrials - Specialty Business Services

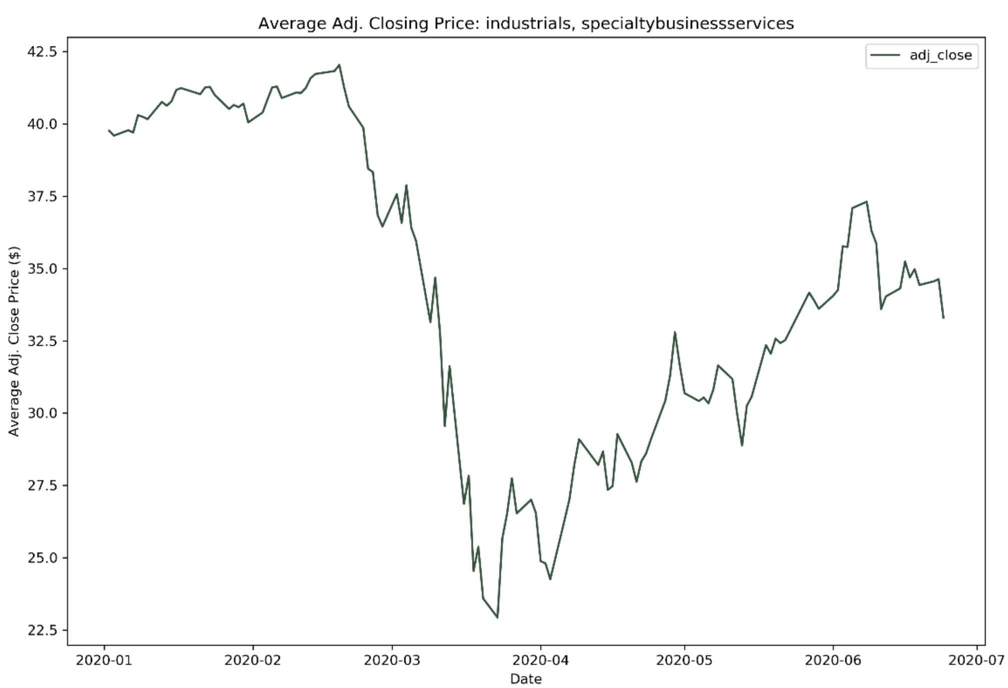


Figure 81: Average Adj. Closing Price: Industrials - Specialty Industrial Machinery

Figure 82: Average Adj. Closing Price: Industrials - Staffing Employment Services

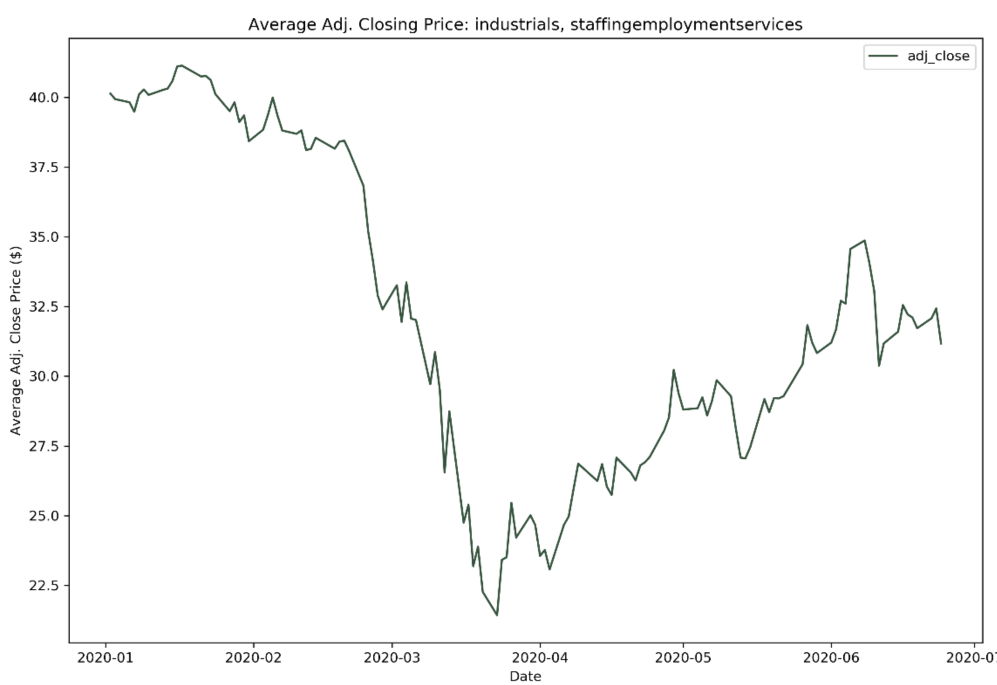


Figure 83: Average Adj. Closing Price: Industrials - Tools/Accessories

Figure 84: Average Adj. Closing Price: Industrials - Trucking

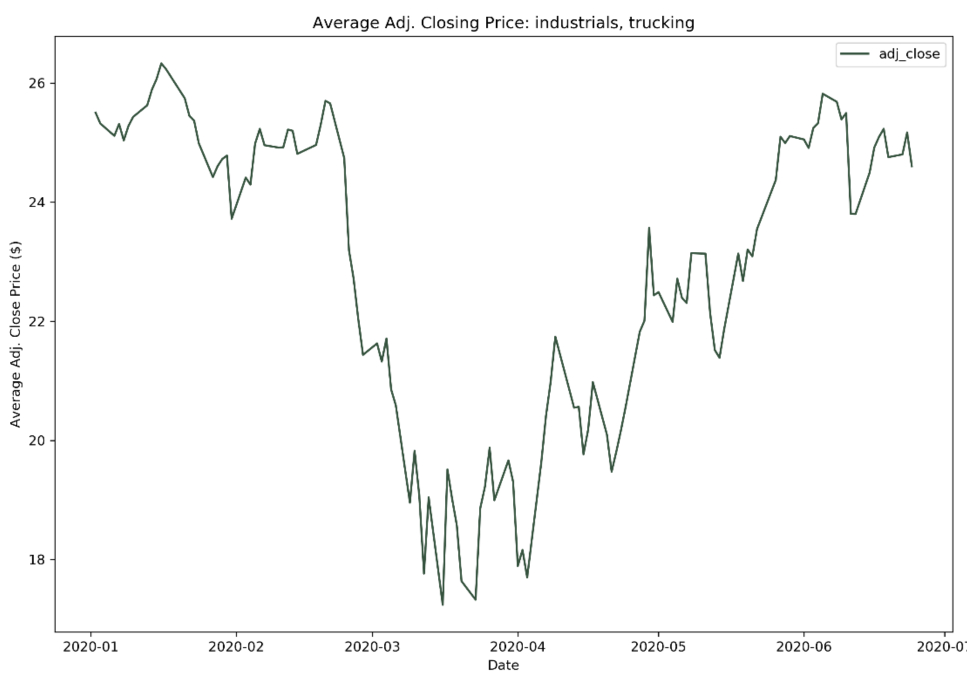


Figure 85: Average Adj. Closing Price: Industrials - Waste Management

B. Source Code

Script_saveonly_AL3.R

```
#install.packages("tidyverse")
#install.packages("rtweet")
#install.packages("tidytext")

#install.packages("SnowballC")
#install.packages("hunspell")

#install.packages("textdata")


library(tidyverse)
library(rtweet)
library(tidytext)

library(SnowballC)
library(hunspell)

library(textdata)

consumer_key <- 'XXXXXXXX'
consumer_key_secret <- 'XXXXXXXX'
appname <- ‘XXXXXXX'

access_token <- ‘XXXXXXXX'
access_token_secret <- 'XXXXXXX'

twitter_token <- create_token(app = appname,
                    consumer_key = consumer_key,
                    consumer_secret = consumer_key_secret,
                    access_token = access_token,
                    access_secret = access_token_secret)

trump_tweets <- search_tweets(q = "(from:realDonaldTrump)", include_rts = FALSE, n = 18000)
save_as_csv(trump_tweets, paste0("trump_", Sys.Date(), ".csv"))

aoc_tweets <- search_tweets(q = "(from:AOC)", include_rts = FALSE, n = 18000)
save_as_csv(aoc_tweets, paste0("aoc_", Sys.Date(), ".csv"))

mcconnell_tweets <- search_tweets(q = "(from:senatemajldr)", include_rts = FALSE, n = 18000)
save_as_csv(mcconnell_tweets, paste0("mcconnell_", Sys.Date(), ".csv"))

mccarthy_tweets <- search_tweets(q = "(from:GOPLeader)", include_rts = FALSE, n = 18000)
save_as_csv(mccarthy_tweets, paste0("mccarthy_", Sys.Date(), ".csv"))

schumer_tweets <- search_tweets(q = "(from:SenSchumer)", include_rts = FALSE, n = 18000)
save_as_csv(schumer_tweets, paste0("schumer_", Sys.Date(), ".csv"))

pelosi_tweets <- search_tweets(q = "(from:SpeakerPelosi)", include_rts = FALSE, n = 18000)
```

```
save_as_csv(pelosi_tweets, paste0("pelosi_", Sys.Date(), ".csv"))

bernie_tweets <- search_tweets(q = "(from:BernieSanders)", include_rts = FALSE, n = 18000)
save_as_csv(bernie_tweets, paste0("bernie_", Sys.Date(), ".csv"))

biden_tweets <- search_tweets(q = "(from:JoeBiden)", include_rts = FALSE, n = 18000)
save_as_csv(biden_tweets, paste0("biden_", Sys.Date(), ".csv"))

tlaib_tweets <- search_tweets(q = "(from:RashidaTlaib)", include_rts = FALSE, n = 18000)
save_as_csv(tlaib_tweets, paste0("tlaib_", Sys.Date(), ".csv"))

omar_tweets <- search_tweets(q = "(from:Ilhan)", include_rts = FALSE, n = 18000)
save_as_csv(omar_tweets, paste0("omar_", Sys.Date(), ".csv"))

ayanna_tweets <- search_tweets(q = "(from:AyannaPressley)", include_rts = FALSE, n = 18000)
save_as_csv(ayanna_tweets, paste0("ayanna_", Sys.Date(), ".csv"))
```

Sent_Analysis.py

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
import math

from afinn import Afinn
from nrclex import NRCLex
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from textblob import TextBlob

from sklearn import preprocessing

#Import separate csv files and compile into one
os.chdir(r"XXXXXXXX")

file_list = os.listdir()

tweets = pd.read_csv(file_list[0])

for i in range(1, len(file_list)):
    tweets = pd.concat([tweets, pd.read_csv(file_list[i])], axis=0, ignore_index=True)

#The script pulled tweets based on time, which included duplicate tweets, this drops them based on id
tweets = tweets.drop_duplicates(subset=['status_id'])

#The Twitter API returns 90 columns, most of which are irrelevant to this project
tweet_df = tweets[['created_at', 'screen_name', 'text', 'favorite_count', 'retweet_count', 'quote_count', 'reply_count', 'hashtags', 'quoted_text', 'retweet_text']]
```

```

#Resets the index after dropping duplicates
tweet_df.index = range(0, len(tweet_df))

#Assign Rep/Dem label
r = ['GOPLeader', 'senatemajldr', 'realDonaldTrump']
d = ['AOC', 'AyannaPressley', 'BernieSanders', 'JoeBiden', 'Ilhan', 'SpeakerPelosi', 'RashidaTlaib', 'SenSchumer']
tweet_df['party'] = ''

for i in range(0, len(tweet_df)):
    if(tweet_df.screen_name[i] in r):
        tweet_df.party[i] = "Republican"
    else:
        tweet_df.party[i] = "Democrat"

#Analysis w/Afinn lexicon
af = Afinn()

af_scores_source = [af.score(tweet) for tweet in tweet_df.text]
af_category_source = ['positive' if score > 0
            else 'negative' if score < 0
            else 'neutral'
            for score in af_scores_source]

#Run analysis on quoted tweets and retweets
tweet_df.quoted_text = tweet_df.quoted_text.fillna('')

af_scores_quote = [af.score(tweet) for tweet in tweet_df.quoted_text]
af_category_quote = ['positive' if score > 0
            else 'negative' if score < 0
            else 'neutral'
            for score in af_scores_quote]

tweet_df.retweet_text = tweet_df.retweet_text.fillna('')

af_scores_retweet = [af.score(tweet) for tweet in tweet_df.retweet_text]
af_category_retweet = ['positive' if score > 0
            else 'negative' if score < 0
            else 'neutral'
            for score in af_scores_retweet]

tweet_df['afinn_source'] = af_scores_source
tweet_df['afinn_source_category'] = af_category_source
tweet_df['afinn_quote'] = af_scores_quote
tweet_df['afinn_quote_category'] = af_category_quote
tweet_df['afinn_retweet'] = af_scores_retweet
```

```
tweet_df['afinn_retweet_category'] = af_category_retweet

#Analysis w/NRC Lexicon
tweet_df['fear_score'] = np.nan
tweet_df['fear_freq'] = np.nan
tweet_df['anger_score'] = np.nan
tweet_df['anger_freq'] = np.nan
tweet_df['anticip_score'] = np.nan
tweet_df['anticip_freq'] = np.nan
tweet_df['trust_score'] = np.nan
tweet_df['trust_freq'] = np.nan
tweet_df['surprise_score'] = np.nan
tweet_df['surprise_freq'] = np.nan
tweet_df['pos_score'] = np.nan
tweet_df['pos_freq'] = np.nan
tweet_df['neg_score'] = np.nan
tweet_df['neg_freq'] = np.nan
tweet_df['sad_score'] = np.nan
tweet_df['sad_freq'] = np.nan
tweet_df['disgust_score'] = np.nan
tweet_df['disgust_freq'] = np.nan
tweet_df['joy_score'] = np.nan
tweet_df['joy_freq'] = np.nan

for i in range(0, len(tweet_df.text)):
    tweet_nrc = NRCLex(tweet_df.text[i].lower())
    scores = tweet_nrc.raw_emotion_scores
    freq = tweet_nrc.affect_frequencies
    try:
        tweet_df.fear_score[i] = scores['fear']
        tweet_df.fear_freq[i] = freq['fear']
        tweet_df.anger_score[i] = scores['anger']
        tweet_df.anger_freq[i] = freq['anger']
        tweet_df.anticip_score[i] = scores['anticip']
        tweet_df.anticip_freq[i] = freq['anticip']
        tweet_df.trust_score[i] = scores['trust']
        tweet_df.trust_freq[i] = freq['trust']
        tweet_df.surprise_score[i] = scores['surprise']
        tweet_df.surprise_freq[i] = freq['surprise']
        tweet_df.pos_score[i] = scores['positive']
        tweet_df.pos_freq[i] = freq['positive']
        tweet_df.neg_score[i] = scores['negative']
        tweet_df.neg_freq[i] = freq['negative']
        tweet_df.sad_score[i] = scores['sadness']
        tweet_df.sad_freq[i] = freq['sadness']
        tweet_df.disgust_score[i] = scores['disgust']
        tweet_df.disgust_freq[i] = freq['disgust']
```

```
        tweet_df.joy_score[i] = scores['joy']
        tweet_df.joy_freq[i] = freq['joy']
    except:
        #Checks for exception if there is no score
        print("An exception occurred")

print("Missing values for fear: {x}".format(x = sum(1 for x in tweet_df['fear_score'] if math.isnan(x))))
print("Missing values for anger: {x}".format(x = sum(1 for x in tweet_df['anger_score'] if math.isnan(x))))
print("Missing values for anticipation: {x}".format(x = sum(1 for x in tweet_df['anticip_score'] if math.isnan(x))))
print("Missing values for trust: {x}".format(x = sum(1 for x in tweet_df['trust_score'] if math.isnan(x))))
print("Missing values for surprise: {x}".format(x = sum(1 for x in tweet_df['surprise_score'] if math.isnan(x))))
print("Missing values for positive: {x}".format(x = sum(1 for x in tweet_df['pos_score'] if math.isnan(x))))
print("Missing values for negative: {x}".format(x = sum(1 for x in tweet_df['neg_score'] if math.isnan(x))))
print("Missing values for sadness: {x}".format(x = sum(1 for x in tweet_df['sad_score'] if math.isnan(x))))
print("Missing values for disgust: {x}".format(x = sum(1 for x in tweet_df['disgust_score'] if math.isnan(x))))
print("Missing values for joy: {x}".format(x = sum(1 for x in tweet_df['joy_score'] if math.isnan(x))))

#Analysis w/VADER lexicon
tweet_df['vader_positive'] = np.nan
tweet_df['vader_neutral'] = np.nan
tweet_df['vader_negative'] = np.nan
tweet_df['vader_compound'] = np.nan

vader_analyzer = SentimentIntensityAnalyzer()
for i in range(0, len(tweet_df.text)):
    scores = vader_analyzer.polarity_scores(tweet_df.text[i])
    tweet_df.vader_positive[i] = scores['pos']
    tweet_df.vader_neutral[i] = scores['neu']
    tweet_df.vader_negative[i] = scores['neg']
    tweet_df.vader_compound[i] = scores['compound']

#Analysis w/Textblob lexicon
tweet_df['textblob_polarity'] = np.nan
for i in range(0, len(tweet_df.text)):
    blob = TextBlob(tweet_df.text[i])
    tweet_df.textblob_polarity[i] = blob.sentiment.polarity
```

```
#Measure virality by the number of interactions
tweet_df['virality'] = tweet_df['favorite_count'].fillna(0) + tweet_df['retweet_count'].fillna(0) + tweet_df['quote_count'].fillna(0) + tweet_df['reply_count'].fillna(0)



os.chdir(r"..\\")
tweet_df.to_csv("tweets_sentiment_scores.csv")


tweet_df['index_col'] = tweet_df.index

afinn_scores = tweet_df[['index_col', 'created_at', 'screen_name', 'text', 'favorite_count', 'retweet_count', 'quote_count', 'reply_count', 'hashtags', 'quoted_text', 'retweet_text', 'afinn_source', 'afinn_source_category', 'afinn_quote', 'afinn_quote_category', 'afinn_retweet', 'afinn_retweet_category']]
afinn_scores.to_csv("tweets_AFINN.csv")

nrc_scores = tweet_df[['index_col', 'created_at', 'screen_name', 'text', 'favorite_count', 'retweet_count', 'quote_count', 'reply_count', 'hashtags', 'quoted_text', 'retweet_text','fear_score', 'fear_freq', 'anger_score', 'anger_freq', 'anticip_score',
    'anticip_freq', 'trust_score', 'trust_freq', 'surprise_score',
    'surprise_freq', 'pos_score', 'pos_freq', 'neg_score', 'neg_freq',
    'sad_score', 'sad_freq', 'disgust_score', 'disgust_freq', 'joy_score',
    'joy_freq']]
nrc_scores.to_csv("tweets_NRC.csv")

vader_scores = tweet_df[['index_col', 'created_at', 'screen_name', 'text', 'favorite_count', 'retweet_count', 'quote_count', 'reply_count', 'hashtags', 'quoted_text', 'retweet_text', 'vader_positive', 'vader_neutral', 'vader_negative', 'vader_compound']]
vader_scores.to_csv("tweets_VADER.csv")

textblob_scores = tweet_df[['index_col', 'created_at', 'screen_name', 'text', 'favorite_count', 'retweet_count', 'quote_count', 'reply_count', 'hashtags', 'quoted_text', 'retweet_text', 'textblob_polarity']]
textblob_scores.to_csv("tweets_TEXTBLOB.csv")

#Notes on missing values:
#Delete all tweets that do not have a score on any lexicon

afinn_null = afinn_scores[(afinn_scores.afinn_source == 0) & (afinn_scores.afinn_quote == 0) & (afinn_scores.afinn_retweet == 0)]
nrc_null = nrc_scores[pd.isnull(nrc_scores.fear_score)]
vader_null = vader_scores[(vader_scores.vader_positive == 0) & (vader_scores.vader_negative == 0)]
textblob_null = textblob_scores[(textblob_scores.textblob_polarity == 0)]
```

```
nrc_null.index = range(0, len(nrc_null))
#Find indexes of null values
null_idx = []
for i in range(0, len(nrc_null)):
    if(nrc_null.index_col[i] in afinn_null.index):
        if(nrc_null.index_col[i] in vader_null.index):
            if(nrc_null.index_col[i] in textblob_null.index):
                null_idx.append(nrc_null.index_col[i])

tweets_NoNull = tweet_df
for idx in null_idx:
    tweets_NoNull = tweets_NoNull.drop(idx)

tweets_NoNull.index = range(0, len(tweets_NoNull))
tweets_NoNull.to_csv("tweets_NoNull.csv")


#List of NRC Emotions:
#Fear
#Anger
#Sadness
#Disgust
#Anticipation
#Trust
#Surprise
#Joy

#Normalizing scores

#AFINN
afinn_source_std = preprocessing.minmax_scale(tweets_NoNull.afinn_source, feature_range=(-100,100))
#Not enough quote/retweet scores to scale

#NRC
fear_score_std = preprocessing.minmax_scale(tweets_NoNull.fear_score, feature_range=(0,100))
anger_score_std = preprocessing.minmax_scale(tweets_NoNull.anger_score, feature_range=(0,100))
sad_score_std = preprocessing.minmax_scale(tweets_NoNull.sad_score, feature_range=(0,100))
disgust_score_std = preprocessing.minmax_scale(tweets_NoNull.disgust_score, feature_range=(0,100))
anticip_score_std = preprocessing.minmax_scale(tweets_NoNull.anticip_score, feature_range=(0,100))
trust_score_std = preprocessing.minmax_scale(tweets_NoNull.trust_score, feature_range=(0,100))
surprise_score_std = preprocessing.minmax_scale(tweets_NoNull.surprise_score, feature_range=(0,100))
```

```
joy_score_std = preprocessing.minmax_scale(tweets_NoNull.joy_score, feature_range=(0,100))
pos_score_std = preprocessing.minmax_scale(tweets_NoNull.pos_score, feature_range=(0, 100))
neg_score_std = preprocessing.minmax_scale(tweets_NoNull.neg_score, feature_range=(0, 100))

nrc_sentiment = pos_score_std - neg_score_std
nrc_PosEmo = anticip_score_std + trust_score_std + surprise_score_std + joy_score_std
nrc_NegEmo = fear_score_std + anger_score_std + sad_score_std + disgust_score_std
nrc_EmoScore = nrc_PosEmo - nrc_NegEmo

#VADER
vader_pos_std = preprocessing.minmax_scale(tweets_NoNull.vader_positive, feature_range=(0, 100))
vader_neg_std = preprocessing.minmax_scale(tweets_NoNull.vader_negative, feature_range=(0, 100))
vader_neutral_std = preprocessing.minmax_scale(tweets_NoNull.vader_neutral, feature_range=(0, 100))
vader_compound_std = preprocessing.minmax_scale(tweets_NoNull.vader_compound, feature_range=(-100, 100))

#Textblob
textblob_polarity_std = preprocessing.minmax_scale(tweets_NoNull.textblob_polarity, feature_range=(-100, 100))

std_df = pd.DataFrame(data = {'afinn_std': afinn_source_std, 'fear_std': fear_score_std, 'anger_std': anger_score_std,
                     'sad_std': sad_score_std, 'disgust_std': disgust_score_std, 'anticip_std': anticip_score_std,
                     'trust_std': trust_score_std, 'surprise_std': surprise_score_std, 'joy_std': joy_score_std,
                     'pos_std': pos_score_std, 'neg_std': neg_score_std, 'nrc_sentiment': nrc_sentiment,
                     'nrc_PosEmo': nrc_PosEmo, 'nrc_NegEmo': nrc_NegEmo, 'nrc_EmoScore': nrc_EmoScore,
                     'vader_pos_std': vader_pos_std, 'vader_neg_std': vader_neg_std, 'vader_neutral_std': vader_neutral_std,
                     'vader_compound_std': vader_compound_std, 'textblob_polarity_std': textblob_polarity_std})

compound_scores = std_df[['afinn_std', 'nrc_sentiment', 'nrc_EmoScore', 'vader_compound_std', 'textblob_polarity_std']].copy()

compound_scores['sentiment_compound'] = compound_scores.mean(axis=1)

final_df = pd.concat([tweets_NoNull, std_df, compound_scores['sentiment_compound']], axis=1)

final_df.to_csv("tweets_final.csv")
```

TickerScrape.py

```
import pandas as pd
import numpy as np
import os

import re

os.chdir(r’XXXXXXX’)

url = "https://finviz.com/screener.ashx"

from selenium import webdriver
from selenium.webdriver.common.keys import Keys

driver = webdriver.Firefox(executable_path=XXXXXX, log_path=XXXXX)
driver.set_page_load_timeout(600)

driver.get(url)

sector_list = driver.find_element_by_id("fs_sec")
options = sector_list.find_elements_by_tag_name("option")

#First entry is "Any", last is "Custom"
options = options[1:-1]
sector_text = []
for option in options:
    sector_text.append(option.get_attribute("value"))

#Adjust list in case of errors
sector_text = sector_text[6:11]

df = pd.DataFrame(columns = ['symbol', 'company_name', 'sector', 'industry'])
for sector in sector_text:
    c_sector = sector
    url = "https://finviz.com/screener.ashx?v=111&f=sec_" + sector
    driver.get(url)
    ind_list = driver.find_element_by_id("fs_ind")
    ind_options = ind_list.find_elements_by_tag_name("option")
    ind_options = ind_options[1:-1]
    ind_text = []
    for industry in ind_options:
        ind_text.append(industry.get_attribute("value"))
    for industry in ind_text:
        c_industry = industry
        r = 1
```

```
        url = "https://finviz.com/screener.ashx?v=111&f=geo_usa,ind_"+industry+",sec_"+sector+"&r="+str(r)
        driver.get(url)
        try:
            pagination = driver.find_element_by_css_selector("#screener-content > table > tbody > tr:nth-child(7)")
        except:
            print("No conent for industry: {}".format(industry))
            continue
        if len(pagination.find_elements_by_tag_name("b")) == 1:
            print("No pagination detected")
            url_num = 1
        elif len(pagination.find_elements_by_tag_name("a")) > 1:
            num_pages = pagination.find_elements_by_tag_name("a")[-2].text
            url_num = (int(num_pages)-1)+(20*int(num_pages)-1)
        while r <= url_num:
            url = "https://finviz.com/screener.ashx?v=111&f=geo_usa,ind_"+industry+",sec_"+sector+"&r="+str(r)
            driver.get(url)
            table_body = driver.find_element_by_css_selector("#screener-content > table:nth-child(1) > tbody:nth-child(1) > tr:nth-child(4) > td:nth-child(1) > table:nth-child(1) > tbody:nth-child(1)")
            table_rows = table_body.find_elements_by_tag_name("tr")[1:]
            for row in table_rows:
                cells = row.find_elements_by_tag_name("td")
                c_symbol = cells[1].text
                c_name = cells[2].text
                new_row = pd.DataFrame(data = {'symbol': c_symbol, 'company_name': c_name, 'sector': c_sector, 'industry': c_industry}, index=[0])
                df = df.append(new_row)
                print("Added Company {} With Symbol {}".format(c_name, c_symbol))
            r += 20

df.to_csv("stocksymbols_2.csv")

```

Yahoo_FinScrape.py

```
import pandas as pd
import numpy as np
import os

import time
from datetime import datetime

os.chdir(XXXX)

```

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys

from bs4 import BeautifulSoup as bs
import requests

symbols = pd.read_csv("symbols.txt", delimiter="\n", header=None, names=['symbol'])


driver = webdriver.Firefox(executable_path=XXXX, log_path=XXXX)
driver.set_page_load_timeout(600)

full_df = pd.DataFrame(columns = ['Date', 'Open', 'High', 'Low',
                        'Close*', 'Adj Close**', 'Volume', 'Symbol'])

#Split the list of symbols into 7 groups to mitigate lost progress by errors
#sym_1 = symbols[0:207]
#sym_1 = sym_1[symbols.index[symbols['symbol'] == "LB"].tolist()[0]:len(sym_1)]
#sym_2 = symbols[207:415]
#sym_2 = symbols[symbols.index[symbols['symbol'] == "NLS"].tolist()[0]:415]
#sym_3 = symbols[415:622]
#sym_4 = symbols[622:829]
#sym_5 = symbols[829:1036]
#sym_6 = symbols[1036:1243]
#sym_7 = symbols[1243:1450]
#sym_8 = symbols[1450:1657]
#sym_9 = symbols[1657:1862]

#Some companies were not found at first, this list ties up loose ends
sym_extra = ['BURL', 'ETH', 'AMZN', 'SIC', 'BBBY', 'CULP', 'VCNX', 'OPK', 'WBT', 'WM']

for symbol in sym_extra:
    url = "https://www.finance.yahoo.com/quote/" + symbol + "/history"
    driver.get(url)
    scroll_to_bottom(driver)
    html = driver.page_source
    soup = bs(html, 'html.parser')
    table = soup.find_all('table')[0]
    df = pd.read_html(str(table))[0][0:-1]
    try:
        df['Date'] = pd.to_datetime(df['Date'])
        df = df[df['Date'] >= '2020-01-01']
    except:
        print("No entry for " + symbol)
        df = pd.DataFrame(columns = ['Date', 'Open', 'High', 'Low',
                        'Close*', 'Adj Close**', 'Volume', 'Symbol'])
    df['Symbol'] = symbol
```

```
    full_df = full_df.append(df)
    print("Added " + symbol)

full_df.to_csv("yahoo_fin_extra.csv")

yf1_2 = pd.read_csv("yahoo_fin_1_2.csv")
yf_3 = pd.read_csv("yahoo_fin_3.csv")
yf_4 = pd.read_csv("yahoo_fin_4.csv")
yf_5 = pd.read_csv("yahoo_fin_5.csv")
yf_6 = pd.read_csv("yahoo_fin_6.csv")
yf_7_8 = pd.read_csv("yahoo_fin_7_8.csv")
yf_9 = pd.read_csv("yahoo_fin_9.csv")
yf_extra = pd.read_csv("yahoo_fin_extra.csv")

yahoo_full = yf1_2.append(yf_3).append(yf_4).append(yf_5).append(yf_6).append(
    yf_7_8).append(yf_9).append(yf_extra)

filter = yahoo_full['Open'].str.contains("Dividend")
yahoo_full = yahoo_full[~filter]

yahoo_full.to_csv("yahoo_fin_full.csv")

#Code for this function obtained from https://stackoverflow.com/questions/32391303/how-to-scroll-to-the-end-of-the-page-using-selenium-in-python
def scroll_to_bottom(driver):

    old_position = 0
    new_position = None

    while new_position != old_position:
        # Get old scroll position
        old_position = driver.execute_script(
                ("return (window.pageYOffset !== undefined) ?"
                 " window.pageYOffset : (document.documentElement ||"
                 " document.body.parentNode || document.body);"))
        # Sleep and Scroll
        time.sleep(1)
        driver.execute_script((
                "var scrollingElement = (document.scrollingElement ||"
                " document.body);scrollingElement.scrollTop ="
                " scrollingElement.scrollHeight;"))
        # Get new position
        new_position = driver.execute_script(
                ("return (window.pageYOffset !== undefined) ?"
                 " window.pageYOffset : (document.documentElement ||"
                 " document.body.parentNode || document.body);"))
```

COVIDVisualizations.py

```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt

import seaborn as sns



from datetime import datetime as dt

os.chdir(r"XXXXX")

tweets = pd.read_csv("tweets_final.csv")
tweets = tweets.drop(columns=['Unnamed: 0'])
tweets.created_at = pd.to_datetime(tweets.created_at)

stocks = pd.read_csv("yahoo_fin_full.csv")
stocks.rename(columns={'Symbol':'symbol'}, inplace=True)
stocks = stocks.drop(columns=['Unnamed: 0', 'Unnamed: 0.1'])

symbols = pd.read_csv("stocksymbols_full.csv")


stock_data = stocks.merge(symbols)
stock_data.to_csv("fin_full.csv")



#Set data types of columns
stock_data.rename(columns={'Date':'date', 'Open':'open', 'High':'high',
                           'Low':'low', 'Close*':'close', 'Adj Close**': 'adj_close',
                           'Volume':'volume'}, inplace=True)

#Companies that have stock splits are causing huge drops in price that are not reflective of market conditions
stock_splits = stock_data[stock_data['open'].astype(str).str.contains('Split')].company_name.unique()

stock_data = stock_data[~stock_data['company_name'].isin(stock_splits)]

stock_data['date'] = pd.to_datetime(stock_data['date'])
stock_data['open'] = pd.to_numeric(stock_data['open'], errors='coerce')
stock_data['high'] = pd.to_numeric(stock_data['high'], errors='coerce')
stock_data['low'] = pd.to_numeric(stock_data['low'], errors='coerce')
stock_data['close'] = pd.to_numeric(stock_data['close'], errors='coerce')
```

```
stock_data['adj_close'] = pd.to_numeric(stock_data['adj_close'], errors='coerce')
stock_data['volume'] = pd.to_numeric(stock_data['volume'], errors='coerce')

tweets['created_at'] = pd.to_datetime(tweets['created_at'].dt.strftime('%Y-%m-%d'))
os.chdir(r".\Visualizations")

sns.set_palette("Accent")
#Overall sentiment over time
avg_sent = tweets[['created_at', 'sentiment_compound']].groupby(['created_at']).mean()
plot = sns.lineplot(data=avg_sent)
plot.set(xlabel='Date', ylabel='Average Sentiment', title='Average Political Sentiment')
#plt.show()
fig = plot.get_figure()
fig.set_size_inches(12, 8)
fig.savefig('overall_sentiment.png', dpi=300)
plt.clf()

#Dot Plot w/trendline
trend_sent = avg_sent
trend_sent = trend_sent.reset_index()

plot = sns.regplot(data=trend_sent, x=test.index, y='sentiment_compound')
plot.set(title='Trend of Average Political Sentiment')
fig = plot.get_figure()
fig.savefig('overall_sentiment_trend.png', dpi=300)
plt.clf()

#By Party
sent = tweets[['created_at', 'sentiment_compound', 'party']]
combo_sent = pd.melt(sent, id_vars=['created_at', 'party'], value_vars=['sentiment_compound']).groupby(['created_at', 'party']).mean().reset_index()
party_plot = sns.lineplot(x='created_at', y='value', data=combo_sent, palette=sns.color_palette("RdBu_r", 2), hue='party')
party_plot.set(xlabel='Date', ylabel='Average Sentiment', title='Average Political Sentiment By Political Party')
party_plot.set_xticklabels(party_plot.get_xticklabels(), rotation=45)
plt.show()
fig = party_plot.get_figure()
fig.set_size_inches(12, 8)
fig.savefig('overall_sentiment_party.png', dpi=300)
plt.clf()

#Party trends
party_trend = combo_sent
party_trend['index1'] = party_trend.index
party_trend.columns = ['created_at', 'party', 'average_sentiment', 'index', 'index1']
plot = sns.lmplot(data=party_trend, x='index1', y='average_sentiment', hue='party', col='party')
```

```
plot.set(xlabel='')
plt.show()
f.savefig("party_trend.png", dpi=300)
plt.clf()

#Individuals
authors = tweets[['created_at', 'screen_name', 'sentiment_compound', 'party']]
for author in authors.screen_name.unique():
    data = authors[authors.screen_name == author]
    if (data.party == 'Republican').sum() > 0:
        sns.set_palette("Reds_d")
    else:
        sns.set_palette("Blues_d")
    plot = sns.lineplot(x='created_at', y='sentiment_compound', data=data, ci=None)
    plot.set(xlabel='Date', ylabel='Average Sentiment', title = 'Average Political Sentiment By Author: '+author)
    fig = plot.get_figure()
    fig.set_size_inches(12, 8)
    fig.savefig('overall_sentiment_'+author+'.png', dpi=300)
    plt.clf()

sns.set_palette("Greens_d")

#Plotting financial data
#Average closing price
stock_copy = stock_data[['date', 'adj_close', 'symbol', 'company_name', 'sector', 'industry']]
avg_close = stock_copy.groupby(['date']).mean()
ax = sns.lineplot(data=avg_close)
ax.set(xlabel='Date', ylabel='Average Adj. Close Price ($)', title='Average Adj. Closing Price Across All Four Sectors')
fig = ax.get_figure()
fig.set_size_inches(12, 8)
fig.savefig('average_stockclose.png', dpi=300)
plt.clf()


#Closing price by sector
for sector in stock_data.sector.unique():
    os.chdir(r".\\"+sector)
    data = stock_data[stock_data.sector==sector][['date', 'adj_close', 'symbol', 'company_name', 'sector', 'industry']]
    sector_data = data.groupby(['date']).mean()
    plot = sns.lineplot(data=sector_data)
    plot.set(xlabel='Date', ylabel='Average Adj. Close Price ($)', title = 'Average Adj. Closing Price By Sector: ' + sector)
    fig = plot.get_figure()
    fig.set_size_inches(12, 8)
```

```
    fig.savefig('avg_adjclose_'+sector+'.png', dpi=300)
    plt.clf()
    for industry in data.industry.unique():
        ind_data = data[data.industry==industry].groupby(['date']).mean()
        plot = sns.lineplot(data=ind_data)
        plot.set(xlabel='Date', ylabel='Average Adj. Close Price ($)', title = 'Average Adj. Closing Price: '+sector+', '+industry)
        plt.show()
        fig = plot.get_figure()
        fig.set_size_inches(12, 8)
        fig.savefig('avg_adjclose_'+sector+"_"+industry+'.png', dpi=300)
        plt.clf()
    os.chdir(r"..\\")

sns.set_palette("hls")
#Line plot w/sectors
plot = sns.lineplot(data = stock_data, x='date', y='adj_close', hue='sector', ci=None)
plot.set(xlabel='Date', ylabel='Average Adj. Close Price ($)', title = 'Average Adj. Closing Price By Sector')
plot.legend(loc='upper center')
#plt.show()
fig = plot.get_figure()
fig.set_size_inches(12, 8)
fig.savefig('avg_adjclose_sector.png', dpi = 300)
plt.clf()

pct_changes = pd.DataFrame(columns=['sector', 'industry', 'start', 'end', 'min', 'pct_change', 'recovery'])
for sec in stock_data.sector.unique():
    data = stock_data[stock_data.sector==sec]
    for ind in data.industry.unique():
        ind_data = data[stock_data.industry==ind].groupby(['date']).mean()
        start = ind_data.adj_close[0]
        end = ind_data.adj_close[len(ind_data)-1]
        min_price = ind_data.adj_close.min()
        pct_change = ((end - start)/start)*100
        recovery = ((end-min_price)/(start-min_price))*100
        new_row = pd.DataFrame.from_records([{'sector': sec, 'industry': ind, 'start':start, 'end':end, 'min':min_price, 'pct_change':pct_change, 'recovery':recovery}])
        pct_changes = pct_changes.append(new_row)

os.chdir(r'..\\')
pct_changes.to_csv("ind_pct_changes.csv")

#Most companies in Apparel manufacturing don't have data for the 24th, causing a significant drop

```

```
#Plotting histograms of all NRC Emotions


nrc_data = tweets[['fear_std', 'anger_std', 'sad_std', 'disgust_std',
            'anticip_std', 'trust_std', 'surprise_std', 'joy_std']]

os.chdir(r".\Visualizations")

f, axes = plt.subplots(4, 2, figsize=(7, 7), sharex=True, sharey=True)
f.suptitle('NRC Emotion Distribution')

sns.distplot(nrc_data.fear_std, ax=axes[0,0], kde=False)
sns.distplot(nrc_data.anger_std, ax=axes[0,1], kde=False)
sns.distplot(nrc_data.sad_std, ax=axes[1,0], kde=False)
sns.distplot(nrc_data.disgust_std, ax=axes[1,1], kde=False)
sns.distplot(nrc_data.anticip_std, ax=axes[2,0], kde=False)
sns.distplot(nrc_data.trust_std, ax=axes[2,1], kde=False)
sns.distplot(nrc_data.surprise_std, ax=axes[3,0], kde=False)
sns.distplot(nrc_data.joy_std, ax=axes[3,1], kde=False)

f.savefig('NRC_Emotions.png', dpi=300)
plt.clf()

#Distribution of Sentiment Analysis Compound Scores
#Afinn_std, nrc_sentiment, vader_compound, textblob_polarity, sentiment_compound
f, axes = plt.subplots(2, 2, figsize=(7, 7), sharex=True, sharey=True)
f.suptitle('Distribution of Standardized Sentiment Analysis Scores by Lexicon')

sns.distplot(tweets.afinn_std, ax=axes[0,0], kde=False)
sns.distplot(tweets.nrc_sentiment, ax=axes[0,1], kde=False)
sns.distplot(tweets.vader_compound_std, ax=axes[1,0], kde=False)
sns.distplot(tweets.textblob_polarity_std, ax=axes[1,1], kde=False)
#plt.show()
f.savefig('Score_Distribution.png', dpi=300)

#Distribution of compound sentiment
plot = sns.distplot(tweets.sentiment_compound, kde=False)
fig = plot.get_figure()
fig.suptitle('Distribution of Compound Sentiment Scores')
#plt.show()
```