Using Sentiment Data from the Global Database for Events, Language and Tone (GDELT) to Predict Short-Term Stock Price Developments

Author: Tibor Jakel
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands

ABSTRACT,
This study is set to investigate whether media sentiment, retrieved from the Global Database of Events, Language and Tone, can be a predictor of stock price. A business process model for the extraction and utilization of sentiment data from GDELT is presented and used for measuring the cross-correlation between average media sentiment and closing stock price of Facebook, Apple, Amazon, Alphabet and Tesla. A total of more than 5 million news article entries from GDELT were analyzed, 58.655 of which are relevant to this research. Alphabet is the only company with a strong positive cross-correlation of average daily media sentiment with adjusted closing stock price on the same day, while Facebook and Tesla exhibit weak negative correlations.

Keywords
Big Data, cross-correlation analysis, GDELT, sentiment analysis, stock market forecasting

1. INTRODUCTION

In the last decade, the amount of information that is available to businesses across all industries, has grown exponentially. Business decision makers all around the globe have started to appreciate the capabilities of Big Data analytics, making their decisions data-driven instead of basing them on intuition. While some companies, most notably those that were born in the digital industry, already utilize their enormous user bases to undermine their day-to-day business activities, the transformation from 'business-as-usual' to the evidence-based approach of data-driven decision making poses a real managerial challenge; one, that many companies fail to adapt to [16]. The reasons for this are manifold and range from a corporate culture, that is resistant to change, to concerns on data security [3]. One of the primary barriers for adopting a data-driven decision-making process based on Big Data analytics, however, is the lack of readily available data to conduct such analyses with in the first place. As a result, this study aims to bring a largely neglected set of publicly available data to the attention of both researchers and executives, as well as to propose a framework on how meaningful metrics can be extracted from the data and used for business decision making. The dataset that is talked about is the Global Database of Events, Language and Tone (GDELT), an online database collecting news articles from all over the world and which is free to access and download for everyone. My study aims at capturing the average sentiment of media coverage on a selection of companies for each day in a two-week time period, and to then correlate the extracted values with each company's respective stock price of the same day. By doing so, I will both unveil the hidden potential that lies in this mostly unknown database, as well as demonstrate with a simple business process model, how this data can be processed in order to make it usable for business decision making.

Classifying the type of knowledge contribution my thesis aims to achieve, as proposed by Gregor & Hevner (2013) [8], it falls into the category of improvement: The goal is to apply a new solution, namely the utilization of sentiment data made available through GDELT, to a known problem, the cross-correlation forecasting of stock prices. Frizzo-Barker et al (2016) [6] have shown that Big Data is an increasingly popular topic in academic research, with an increasing number of studies on the topic being published each year. Nonetheless, little research exists that utilizes publicly available datasets to these means. Even apart from its application for forecasting, the GDELT dataset is largely neglected by the scientific community. My research is aimed at filling this gap, and to encourage academics to further research the hidden capabilities within this underutilized set of data.

Apart from its academic relevance, my research will be of use to businesses that are willing to transform their decision-making process to be more data-driven, but lack either the expertise or data to do so. Naturally, this includes mostly small and medium sized enterprises (SMEs), which have the advantage of being more adaptable to new technologies due to their size, but at the same time lack the funding to implement these practices effectively [19].

In order to assess the predictive power of media sentiment as captured by GDELT and to guide my research, the following research question was formulated:

Does the average daily sentiment of media publications on a given company, retrieved from the Global Database of Events, Language and Tone, significantly correlate with changes in that company's stock price?

First, background information on the topics of stock price prediction and the used data source, GDELT, will be given, followed by a review of literature relating to my work. In the next part, the research design and methodology will be explained, including an in-depth discussion of the technical implementation, presenting the core of this study. The results of the analysis will then be presented and used to answer the research question. Lastly, the limitations will be discussed and suggestions for future research will be given.

2. BACKGROUND

2.1 Stock price forecasting

Attempts at forecasting stock market prices rank among the earliest endeavors in financial analysis. It is not surprising that people would try to make money by 'beating the market' through unveiling hidden trends and cyclical movements in historical stock price data and inferring knowledge about future developments. Since stock market data contains a tremendous amount of noise and is therefore among the most difficult signals to forecast, this approach called time series analysis in the past was seldomly crowned with success, leading many economists to believing in what is known as the efficient market hypothesis (EMH) [1]. While modern approaches to time series analysis, such as support vector machine (SVM), facilitating novel machine learning algorithms, have led to a renaissance for stock price predictions using time series analysis in recent years [11] [9], this paper will focus on forecasting stock movements by employing an exogenous variable: Sentiment in media coverage.

According to EMH, stocks are always traded at their fair value; as investors act rationally the prices reflect all available information [14]. EMH is categorized into three separate forms, a distinction that dates back to Roberts (1967) [18]: In its first variant, generally referred to as the 'weak' form of EMH, only information from the historical sequence of prices are fully incorporated in the present-day prices. The 'semi-weak' form on the other hand asserts that in addition to historical prices, all publicly available information is reflected in current stock prices, while the 'strong' form claims that private information known to market participants is fully reflected, which in turn implies that even with insider information, the market cannot be beaten [14]. If the weak form holds true, it would be impossible to predict future market states by solely relying on technical analyses such as the aforementioned time series methods. Should the semi-weak form apply, the same would be true for stock price prediction attempts that factor external information into the equation, including the approach proposed in this study. Sentiment in news articles is publicly available information after all, and in turn a strong cross-correlation between media sentiment and a respective company's share price would present an indicator that EMH is not universally applicable. The implications of the results of this study for EMH will be shortly discussed in 5.3.

While EMH for decades has been one of the building blocks of research in the study field of economics, more recent findings suggest that the markets are indeed not completely efficient, and that some actors do display irrational behavior [15]. In the end, stock prices like every tradable good are based on supply and demand, and since the supply of shares from a company remains unchanged most of the time, fluctuations in demand are responsible for the majority of changes in price. In the current 'Information Age', it has become common practice for private investors to base their investment decisions on information they receive from the internet, be it investment blogs or from other users like themselves. This makes sentiment in media reports on companies' performances a promising candidate as a predictor for stock price developments.

2.2 The Data Source

The Global Database of Events, Language and Tone is a large database that monitors media activity all around the globe, starting from 1979. The project, created by Kalev Leetaru and Philip Schrodt among others and launched in 2013, covers more than 364 million individual events as of 2016 [23], reflecting the ever-growing media coverage through online publication. Each entry is filed with a timestamp and the URL to its source, and then automatically assessed on a range of criteria, mainly describing the type of event that occurred and the tone in which that event was reported on, along a variety of other factors. The event entries in GDELT are not simply put into one large database, however; rather, they are divided into different sub-sets of data, each focusing on particular types entries and of data, therefore being tailored to specific application areas.

The very foundation of the project and the first service to be launched back in 2013 is GDELT 1.0. This data set is comprised of over 3.5 billion entries, spanning from 1979 to present, and updated on the daily. While selected foreign language articles are available as hand translated content, the majority of entries reflect articles in English language only. In 2015, this original set of data was expanded on tremendously, resulting in the GDELT 2.0 Event Database and the GDELT 2.0 Global Knowledge Graph (GKG). The second iteration of GDELT's core, the Event Database, extends its original version through the addition of machine translations for 65 languages both in real-time and at full volume, while at the same time reducing the update interval from daily to once every 15 minutes, a consequence of the increase of registered data resulting from the inclusion of foreign language articles on a large scale. The introduction of the Global Knowledge Graph on the other hand added a wealth of new features, allowing for data exploration far beyond what was originally possible with the Event Database alone: It complements the Event Database by extracting names of people and organizations, locations, themes and emotions, thereby providing the foundation for any type of endeavor related to sentiment analysis [23].

Apart from these two fundamental building blocks, GDELT is comprised of a variety of more specialized sets of data, such as the Visual Global Knowledge Graph (VGKG). This rather novel addition to the project's tool portfolio utilizes the Google Cloud Vision API to classify images found in the listed news articles, thus further extending GDELT's scope from mere textual to also including visual cues of its entries. While data sets such as this offer interesting possibilities for further study, they do not apply measures for sentiment to their entries, rendering them impractical for my study and redundant for further discussion. Still, the existence and hidden potential of these data sets should not go unmentioned [23].

The type and content of each event are gauged based on the Conflict and Mediation Event Observations (CAMEO) taxonomy, a framework for coding event data. The CAMEO framework is the database's main tool for classifying event data, but since its labels assess events mostly through a geopolitical and not an economic lens, using labels such as 'aggressor' and 'victim', it will not play a role in my research.

3. LITERATURE REVIEW

As aforementioned, there is a lack of research investigating the predictive power of sentiment derived from GDELT to forecast stock market pricings. The use of natural language as a predictor for financial forecasting is an emerging topic in contemporary research, however, being facilitated by recent developments in the field of natural language processing (NLP). There has also been a number of studies utilizing the sentiment metric from the GDELT database for various forecasting purposes, although exclusively with a political science background, so the results are not conclusive for financial data. In this section, I will give an overview of the existing literature and discuss how it relates to my work.

3.1 GDELT in academic literature

When GDELT is utilized in scientific research using regression models, the number of article mentions is most commonly used as a controlling variable. Examples of this include Isotalo et al (2016) [10] and Elshendy et al (2018) [5]. Although both included the metric in their final prediction model, little insight on the validity of the data source can be drawn from this, as they did not utilize any actual content data from GDELT.

A good example of how political science research papers apply GDELT's CAMEO classification to filter for geopolitical events can be found in the work of Galla & Burke (2018) [7]. Here, the authors successfully utilize the database to predict events of social unrest up to one month in the future with probabilities of up to 96.4%.

Qiao et al (2017) [17] conducted a similar study with a different prediction model, outperforming all other tested models. The prediction of social unrest was more precise for countries that exhibited more cases of unrest; the analogy that can be drawn to my research shows, that companies exhibiting high spikes in either positive or negative media representation are favorable to those with a more stable media image.

In his master thesis 'Multivariate Time Series Predictions of EU Sentiment', Dormans [4] conducts a predicative time series analysis of sentiment held towards the European Union, with data acquired from GDELT. The author contrasted five supervised learning algorithms, all of which outperformed the naïve model well, to differing degrees. Although applying more sophisticated analyses and being much broader in scope, this thesis contained helpful input for the methodological part of my research. The sample size covering data for two years, while being unrealistic for my research scope and impractical for the data collection part of my thesis, provides a good reference point for potential follow-up studies.

3.2 Sentiment analysis in Financial Forecasting

In contrast to the literature on utilizing GDELT data for forecasting, the research field of sentiment mining is explored to a far greater extent and academic papers exist that show how public sentiment correlates with and can be used to predict stock prices. In opposition to my research, however, these papers focus almost exclusively on data from social media networks, as these provide an inexhaustible source for emotionally charged texts, which suits the cause of sentiment extraction perfectly. Twitter is one of the most popular sites for doing this, due to its popularity and the comparatively easy access to their data via the Twitter API.

A good example of this is the paper by Kordonis et al (2016) [12], in which the authors use NLP and time series forecasting to predict stock prices of 16 of the largest listed technology companies, achieving an average accuracy across the sample of 86%.

Another notable and in this context often cited publication is that by Tetlock (2007) [22]. Here, the author presents his findings indicating that media pessimism indeed is linked to falling stock prices in the short term, while reverting back to their original state over time. This provides useful insights to compare my results to, but does little in terms of providing a methodological framework, as the time series correlation I will be conducting is not based off of any pre-existing theoretical frameworks, and thus derives its predictive power solely from the given data.

Academic papers such as this contributed in large part to sparking my interest in the usage of human sentiment as a predictor for financial values. It should be noted, though, that the applicability of processes used in the papers utilizing NLP is very limited for my purposes. This is due to the fact that I am not conducting a sentiment analysis on a set of textual data myself, which can, depending on the size of the data sample, easily become resource-intensive both in terms of required time and computational power. Instead I focus on using a data source that everyone has access to and which is usable without the need for further computation besides the data filtering process.

4. METHODOLOGY

4.1 Research design

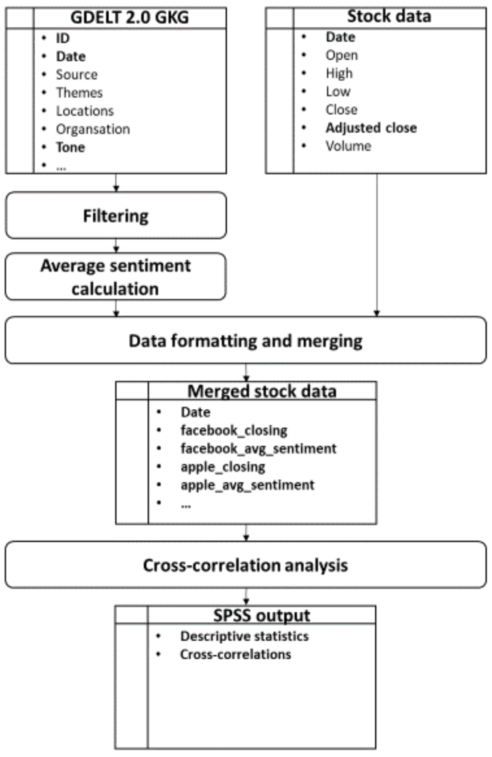


Figure 1: Flow diagram of research design

To investigate the predictive power of media sentiment captured by GDELT, my research was designed as depicted in Figure 1. In the first step, the raw data from the GDELT-GKG needs to be filtered to only contain entries relating to the companies whose stocks are going to be analyzed. The algorithm I used for this at the same time eliminates most of the data in each entry that is unrelated to the task of sentiment analysis, drastically reducing the size of the data sets and enhancing the speed of the subsequent data processing steps. In the next step, the sentiment values of these entries are being averaged for each day in the considered time period, to match the daily format of the stock data, which comes in a format designed for statistical analysis and thus only requires minimal formatting.

In the data formatting and merging step, the two data sets are harmonized and merged on the key variable date, which was previously adjusted to match between the two sources. Resulting from this process is a singular table, summarizing the adjusted closing price and respective average sentiment for each company on each day of the observed time frame. The technical implementation of the steps up until this point will be further elaborated on below.

The merged data table then allows for a cross-correlation analysis between the sentiment value and closing price for each company, to arrive at a conclusion on how strongly the two time series data sets are related.

4.1.1 Company selection

When selecting the six companies to investigate, I took several metrics into account, but ultimately, they were hand-selected. The most important considered aspect is that there needs to be sufficient media coverage on each of the observed days, and that said media coverage is comprised of articles exhibiting sufficient sentiment values to enable a forecast based on this metric. Additionally, to rule out further complications with the time zone formatting, I decided to only choose companies listed on NASDAQ. Furthermore, I prioritized those stocks with a public image reflecting their high profitability, which drew my attention to the 'FAANG' stocks. This acronym describes the collective of Facebook, Apple, Amazon, Netflix and Alphabet (Google), which are some of the best performing stocks in recent years and which have already been studied in contemporary literature covering stock price forecasting [21]. I also added Tesla, due to the eminent presence of its CEO Elon Musk in recent media coverage and the widespread discussions on the value of the Tesla stock that were triggered by some of his utterances on social media.

4.1.2 Sentiment assessment

Capturing the complex phenomenon of human sentiment is a challenging task and contemporary research of the topic covers a plethora of different approaches and algorithms. To make my approach reproducible and keep it in line with that of related studies on sentiment mining using GDELT, I decided to assess it via the 'tone' metric found in the GDELT-GKG. While more recent approaches to sentiment analysis most often utilize machine learning, these algorithms are impractical to use for datasets generated at the volume and velocity of GDELT; thus, the project utilizes a comparatively simple algorithm first presented by Shook et al (2012) [20], which allows for fast computation of a sentiment metric, even for large sets of data. This algorithm is based off of two large dictionaries, one containing words with positive, the other with negative connotations. It then scans the input, in the case of GDELT represented by individual newspaper articles, calculates the percentage of words from the text found in the positive and in the negative dictionary, and subtracts them to arrive at a value which is then displayed in the 'Tone' field. The results can range from +100 – meaning all words in the text are found in the positive dictionary – to -100, but common values range from +10 to -10 [24].

4.2 Data collection

4.2.2 GDELT data

The sentiment data used for the prediction was taken entirely from GDELT. The project's website provides download links for all files ever created, and uploads a new file every 15 minutes. The data is stored in the csv-format, including a multitude of metrics such as the date, CAMEO themes, related people or organizations and tone, which are delimited by semicolons. Since I was unable to automize the download process, acquiring the raw csv-files proved to be a clear bottleneck for my research, and as a result I had to limit the observed time frame to 29 days, starting on the 6th of May 2019 and ending on the 3rd of June 2019. While this number is very low for the type of research I am conducting, it still entailed the manual download of about 2800 individual files. Nonetheless, the short time frame is the most prominent limitation of my study and should be considered when interpreting the results.

As English news articles are collected from different time zones all over the world, the amount of data is relatively consistent across daytime, but differs between weekdays. Common values for the size of the data set for each day average around 2.5GB for weekdays and 1.5GB for weekends, reflecting approximately 220.000 and 130.000 individual entries respectively. While this appears to be a vast figure, the size of the data set diminishes after the application of the filtering algorithm by about 99%; still, the resulting dataset for each company is large enough to give a good estimate of media sentiment about that company on every observed day. Combined with the exclusion of unrelated data from the original GKG files, the size of the intermediate files goes down to a few kB each, making them easily processable even with outdated hardware. The low requirements concerning IT infrastructure are a clear advantage of my approach to other forms of sentiment extraction.

It should be noted, however, that some of the files were corrupted: While being readable with Notepad or Excel, they resulted in a Unicode Error when trying to open them with UTF-8 encoding in Python. As the percentage of corrupted files in the sample was statistically insignificant with about 3% (86 files out of 2784), I did not look further into the technical complications behind this issue and ignored them for the following calculations.

4.2.3 Stock market data

The stock market data was downloaded from Yahoo Finance, which offers fast access to historical data conveniently formatted for statistical use (see Figure 1). As all of the examined stocks are listed at NASDAQ, differences between the individual stocks in time zones and thus in the respective GDELT daytime did not have to be considered in the data collection process.

4.2.4 Business process model

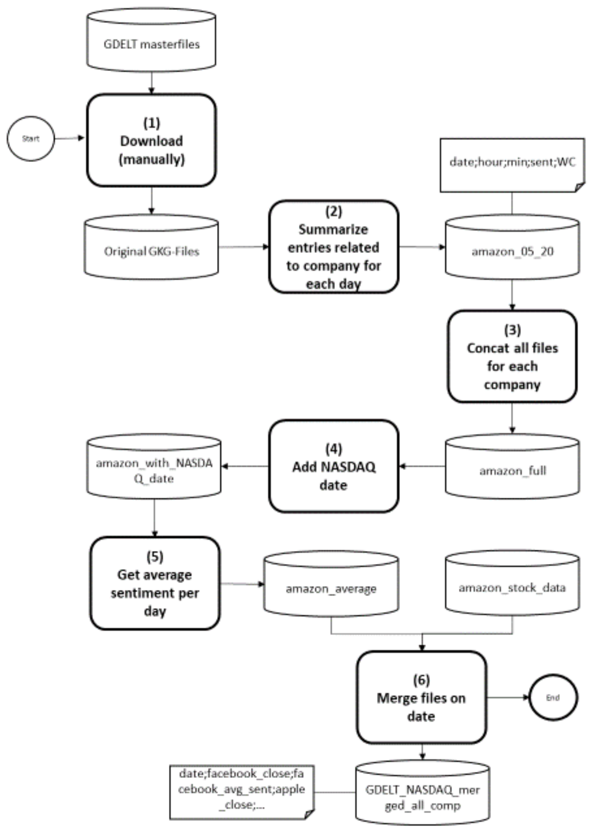


Figure 2. Data processing in BPMN 2.0

4.2.5 Data preprocessing for GDELT

Filtering and preprocessing the data from GDELT to make it usable for a time series forecast is the most integral part of my work. For this stage, I developed the business process model depicted in Figure 2, using the standardized Business Process Model and Notation 2.0 (BPMN 2.0) [2]. In the following, I am going to briefly discuss for each step in the model why it was necessary, how it was implemented and some of the compromises I had to make. The respective Python code used for the implementation can be found in the appendix.

In the first step of the business process model, the raw csv-files need to be downloaded from the GDELT project website. While I did this step manually, due to a lack of HTML-related expertise, in a real business setting this step would preferably be automized. The files come compressed in the ZIP file format and need to be extracted with a decompression tool before being usable.

The second step is the most important one in terms of data collection and also has the biggest impact on the end result. Here, all 96 data files that comprise a day in GDELT are being scanned row by row on whether they relate to one of the companies whose stocks were analyzed. It is also here, that some of the biggest weaknesses of the GDELT database, at least when it comes to filtering for individual companies, come to shine: While the GDELT algorithm based on the CAMEO taxonomy is great for assessing geopolitical events and relevant actors to such, it is not as functional for narrowing down individual companies. As Ward et al (2013) [25] observed, their broad approach to data collection favors capturing a higher number of related articles over preventing false positives, which is sub-optimal for prediction models that require precise data, especially when the sample size is as small as in my study. In practice, this means that my original plan of using the 'Organizations' field as described in the documentation of GDELT results in a large number of entries for each of the stocks, most of which barely have anything to do with the respective company. Tesla, for example, is often listed as a related organization for articles about other car manufacturers. Including these entries in my model would naturally falsify the results. Consequently, I had to make the substantial compromise of filtering the entries based on whether the company name appeared in the URL. This undoubtedly still harms the precision of the results, due to the inclusion of articles that are related to the company, but not in a way meaningful to their stock price development. To give an example, if an article is published on Google's new doodle, calling it 'cute' and 'funny', the resulting GDELT entry will likely assess it with a highly positive sentiment, while it is unlikely that there is any effect on people's buying behavior towards the Alphabet stock. The filtering process was automized in Python by individually accessing each entry in the original GDELT-GKG file, creating a list of each entry whose objects are identified by the delimiting semicolon and accessing the fifth field (containing the URL) in said list to test whether it contains the searched phrase, being the name of the respective company. For Alphabet, I used both the terms 'alphabet' as well as 'google', which might be a contributing factor to the higher number of entries found for the company. If the URL of an article indeed contains the phrase, the program then collects the required data from that entry, most importantly date, time and sentiment, and appends it to a specified output file.

For faster processing, the csv-files containing the sentiment value and timestamp should be concatenated for each company, represented by the third step in the business process model. This can also be done in the previous step, by simply appending the results of each iteration of the filtering algorithm to one document instead of creating a new one for each day, but as I had many problems and setbacks during the data processing steps, it was helpful to be able to look into the individual files from previous steps from time to time to figure out how certain problems with the code came to being.

The fourth step deals with the matching of the time zones between the GDELT servers and NASDAQ: To compute the average sentiment for each day, these days have to be identified first; since GDELT operates in the Coordinated Universal Time zone (UTC) and NASDAQ, being located in New York, in the Eastern Time Zone (ET), there is a four or five hour lag between the time stamps of the data, depending on whether it is daylight saving time (DST). Since the sentiment data from GDELT is continuous, but the stock closing occurs only once a day, the GDELT entries need to be changed so as that they reflect the matching NASDAQ day. The stock exchange closes at 4 pm eastern time, which for my time frame – being in the American DST period – is 8 pm in UTC. Thus, the algorithm needs to change the date of all entries that occur after 8 pm to one day later. As a result, all articles that could influence the closing price are included in the average sentiment calculation for that day, while those that occur over night are incorporated in the next day's average. This part is especially difficult to automize, as calendar dates are unintuitive to code around. This is due to the irregularity of days per month and exceptions like intercalary years, but also due to the aforementioned DST. Since my time frame consisted of only 29 days including one change in months, I did not have to control for most of these contingencies; thus, my solution is not scalable across the entire year. When reproducing and extending my approach for longer periods, I suggest implementing these additional factors one step at a time, working with multiple copies of the data files and manually checking the results of the transformations in-between.

Averaging all sentiment values with the same date identifier in the fifth step of the business process model is comparatively simple to achieve with programming languages, by looping over the files and appending them to a dictionary using the dates as keys and a list containing the respective sentiment as values. If a date is encountered that does not have a corresponding key entry in the dictionary, a new list entry with that date as the key is created. Afterwards, the average value of each date can be easily calculated by dividing the sum of each key's list by its length. This concludes the preprocessing for the data retrieved from GDELT.

4.2.6 Data preprocessing for stock market data and merging of the data sets

The historical stock market data from Yahoo Finance requires far less preprocessing, as it already comes conveniently formatted containing only each day's date, the opening, high, low, closing and adjusted closing price, as well as the volume of shares traded that day, in a comma-delimited csv-file.

One problem with the stock data, however, is that NASDAQ does not open on weekends and public holidays, leading to missing values for those days. There are two ways of dealing with this problem: The one that is closest to reflecting the reality would be to skip the weekends by treating the respective sentiment data of Saturday and Sunday as a 'prolonged Friday night' and to incorporate them in the average calculation for Monday. But since this would disrupt the continuity of the time series data and is difficult to automize, I adopted a simple approach from Kordonis et al (2016) [12], by which each missing value is replaced with the average of the last and the next known value. Thus, a missing value y, with the last known previous value xprevious and the next known value xnext, is:

$$y = (\text{xprevious} + \text{xnext})/2.$$

Naturally, this decreases the quality of the data slightly, but was easily implementable. As only a few values were missing for each stock, I calculated these by hand instead of automizing this process in Python. After this, only the dash characters have to be removed from the date field in the stock csv-files to harmonize the formatting, and the data can be merged with that from GDELT.

Although I originally did not want to use external libraries, in order to make the code more accessible to non-programmers and to not overwhelm myself – being new to programming as well – with more concepts than necessary, I resort to the use of the popular data science library Pandas for the seventh and last step of the business process model, as it simplifies the process of merging the files tremendously. Many of the aforementioned steps can be solved more easily through the use of libraries such as this, but doing so would take away from the understanding of what actually happens with the data, and would make problem analyses more difficult as a result thereof.

Before being able to export the data to SPSS, I actually have to include another final formatting step, by exchanging the periods that are used in the original data set to delimit numbers by commas. Why this is necessary I still do not know, as the SPSS Import Wizard explicitly supports both options when importing csv-files, but for me the former version always results in problems with the loaded data. I thus suspect that this might be a problem exclusive to my installation of the program, so I did not include this last step in the business process model.

4.2.7 Cross-Correlation analysis

To investigate the correlation between the two parallel time series data streams for each company, I use the cross-correlation function in the statistical analysis tool SPSS by IBM. A cross-correlation analysis examines similarity between two time series data sets, as well as their displacement relative to one another. In practice, this means that a significant correlation of average sentiment with adjusted closing stock price at lag 0 or lower indicates that media sentiment to a certain extent predicts the closing price of the respective day or later days (for negative values), while a strong correlation at the lag 1 on the other hand would indicate that a change in closing price predicts media sentiment on the next day.

It should be noted, however, that a significant cross-correlation between the two time series data sets does not imply causation, as there might be a third variable such as financial performance that causes both sentiment and stock price to act a certain way. Rather, a positive result could then be taken as justification to further explore the causation of the correlation and, more generally, the capabilities of the GDELT data source for financial forecasting.

5. RESULTS

For the purpose of reproducibility, the SPSS file resulting from step six of the business process model is accessible to all University of Twente e-mail account holders. Please refer to Appendix 10.2.6.

5.1 Stock price developments

A day's closing stock price is naturally highly correlated with that of previous days. This phenomenon makes stock prices suitable for time series analyses, and it also applies for the six stocks that were observed in this study. As can be seen in Appendix 10.2.4, the autocorrelations, describing the correlation of time series data entries with previous data entries of the same type, ranges between 0.646 for Netflix and 0.882 for Tesla, indicating that Netflix displays the highest variance in stock price over the observed time period, while Tesla displays the least.

When looking at the stock price development graphs for each of the stocks (Appendix 10.2.1), it becomes apparent that they all more or less follow a similar pattern. The movements in market, which affect all of the stocks simultaneously, can also be observed in the table depicting the cross-correlations between the different stocks (Appendix 10.2.3). The stock prices of Netflix and Tesla correlate the least with a value of 0.666. The stock prices of Facebook, Amazon and Google (Alphabet) on the other hand exhibit significant correlations (>= 0.964). In practice, this means that for the observed time frame, the change in Amazon's or Alphabet's stock price are more predicative of today's Facebook stock price than yesterday's Facebook closing price, and vice versa.

5.2 Media sentiment

Table 1. Sentiment statistics

Company	# of entries	Mean	Minimum	Maximum
Facebook	12378	-2.268	-4.453	-0.694
Apple	11481	-0.730	-2.300	0.719
Amazon	10687	0.086	-4.003	1.328
Netflix	6312	0.027	-2.984	2.011
Google	13765	-0.120	-2.399	1.037
Tesla	4023	-0.423	-5.231	0.971

The data collection algorithm as described in 4.2.5 identified and processed a number of entries for each company as depicted in table 1. While the average media sentiment in the observed time period is neutral for most of the stocks with values between -1 and 1, the sentiment towards Facebook is more negatively connotated, with a mean of -2.268.

5.3 Cross-Correlations of average sentiment with closing stock price

To answer the research question, a cross-correlation analysis was conducted for each company to determine the correlation between the average sentiment and adjusted closing stock price. The results of this analysis are found in Appendix 10.2.5. To exclude findings a priori that are clearly not due to actual correlation between the two data streams, but likely the result of chance or external moderating influences, only cross-correlations of time lags between t-4 and t+4 were considered, meaning that the two time series data streams are set apart four days at the maximum.

For Facebook, the average sentiment exhibits a weak negative correlation with the closing stock price of each respective day (-0.191). There is a negative correlation at t+4 of -0.304, indicating that a higher closing price precedes a more negative media coverage of the company four days later, while no other time lag between -4 and 4 shows a notable cross-correlation.

The average sentiment for Apple shows no significant correlation with the closing stock price of the same day (0.051). The only notable correlation is found at t-4 (-0.299), meaning that more positive media coverage precedes a downturn in stock price four days later.

The average sentiment of media coverage for Amazon is not significantly correlated at lag 0, displaying a cross-correlation of 0.047. The strongest correlation between the two time series data sets is found at t+4 with 0.122, indicating a higher adjusted closing price of the stock precedes more positive media coverage on Amazon four days later.

For Netflix, no significant correlation between average sentiment and stock price of the same day is found (0.011). The strongest correlations are found at t-3 with 0.243 and t-2 with 0.200, meaning that a positive media coverage precedes a higher closing price for the company's stock two and three days later respectively.

Alphabet is the only company for which average sentiment has a strong positive correlation with the stock price on the same day of 0.591. This means, that for the observed time period, average media sentiment is a good predictor for the adjusted closing stock price of Alphabet. The correlation is similarly strong for t-1 with 0.405 and t-2 with 0.331 and lessens with each previous day, indicating that the effect is strongest for the next day, but remains present for consecutive days as well. Other significant correlations for Alphabet are found at t+1 with 0.244, indicating that a positive change in stock price also precedes more positive media coverage on the next day, and at t+4 with -0.215, meaning that the opposite is true four days after a change in stock price occurs.

In contrast, the average media sentiment on Tesla exhibits a weak negative correlation with the closing stock price of the same day of -0.150. Other notably strong correlations are found at t-1 (0.270), t-2 (-0.241), t-3 (-0.302) and t-4 (-0.394). Overall, Tesla's stock price appears to react diametrically opposed to the average sentiment of preceding media coverage for the observed time period.

To summarize, the results obtained from the cross-correlation analysis are non-conclusive. While all of the stocks are highly correlated with previous values of themselves, as demonstrated in the autocorrelation analysis, as well as with movements of the other stocks, accounted for in the cross-correlation analysis between different stocks' closing prices, they react contrarily to changes in media sentiment. The only stock that reflects the respective day's average media sentiment in its closing price is that of Alphabet. It should be noted, however, that due to the small sample size and the resulting standard error of 0.186 at time lag 0, these results are not statistically significant for longer periods of time by any means. The probability that the correlations are the result of random noise in the data and not due to actual correlation or even causation is high and this should be kept in mind when interpreting the outcome of the analysis.

The results do not allow for answering the research question blanketly; instead, it must be distinguished between the six companies: While the cross-correlation between average daily sentiment and stock price in the observed time frame is statistically significant for Alphabet, the contrary holds true for the remaining companies. As my study did not investigate any causation, but only correlation effects, it is difficult to judge why this is the case; assessing this and reproducing the results for a larger set of data could be taken as a starting point for future research. The results of my research so far indicate that of the six selected companies, Alphabet is the only one that yields promising results for stock price predictions using sentiment data obtained from GDELT.

Consecutively, the implications for EMH, as introduced in 2.1, are of very similar nature: While there is no indication that the stock prices of Facebook, Apple, Amazon, Netflix and Tesla do not fully reflect information in media sentiment on the respective companies, there is some evidence suggesting that said sentiment can be a predictor of the stock price for Alphabet. Ignoring the discussed statistical uncertainties, it would follow that at least for the share price of this company, the semi-strong as well as strong form of EMH do not apply.

6. LIMITATIONS & RECOMMENDATIONS FOR FUTURE RESEARCH

The research conducted in this study exhibits numerous limitations that have a potential negative impact on the validity of the results and which could be improved on in future research. These limitations encompass decisions that were made in terms of research design, compromises made in the technical implementation, as well as such stemming from the database structure of GDELT, which in its essence is not designed for application in financial forecasting.

The first and foremost limitation is that of sample size; conducting a time series analysis on 29 individual data points will not yield statistically significant results, reflected in the high standard error of the cross-correlation results. Limiting the time period to four weeks was a necessity due to the size of the database and the large number of individual files which needed to be processed. Future research could eliminate this problem by automizing the download process. The chosen time frame and the selection of companies may also present a problem, due to the strong external effects that impact all of the observed stocks and which are captured in the cross-correlations between the individual closing stock prices. The former would be taken care of by expanding the observed time period as suggested above, while the second poses intriguing possibilities for future research by itself: By further automizing the data collection and processing, a larger number of companies of different sizes and from different industries could be analyzed, which might yield interesting results in terms of which companies are suitable for stock price forecasting using media sentiment.

Compromises are also made in the selection of articles on GDELT that are relevant to the chosen companies, as the algorithm used in the data collection selects entries based on whether or not the name of a company is mentioned in the URL of the website where it was released. While GDELT offers more advanced options for selections based on fields such as 'Related Organizations', the associations captured by the database are very broad and unintuitive to use for narrowing down individual subjects. This was already observed by Ward et al (2013) [25], making GDELT's implemented selection features unsuitable for the article selection step. While certainly not perfect, spot checks show that selecting entries based on the URL is an adequate measure of the association of an article to a company, since most news outlets use the title of their articles in the respective URLs.

Whether an article that is associated to a company also bares any connection to that company's stock price development is an aspect that is highly difficult to narrow down and not adequately measurable by only relying on GDELT data. The next step in this direction could be to only rely on data from specific news outlets, such as financial sites. As GDELT is mainly focused on capturing data on the content of the listed articles and not on their source, except for providing the name of the publisher and weblink to the article, this would require cross-checking the entries with external data. The same constraint applies when trying to assign weights to the sentiment values of different entries based on the influence that article might have: A New York Times article prognosing the impending doom of Tesla will certainly have a bigger influence than that of some unknown news outlet with only a couple hundreds of readers. This effect is not accounted for at all in the calculation of average news sentiment, since it is calculated as a function that is divided by the number of articles, irrespective of the number of readers these articles have. Besides expanding the time frame, this limitation should be prioritized in future research, as it can be alleviated by cross-checking the news outlet for each entry, and assigning a weight to the sentiment value based on the number of monthly readers of the respective publisher or a similar metric.

A more fundamental problem is found in the sentiment assessment algorithm used by GDELT: While dictionary-based methods have the advantage of being fast to process even for large amounts of data, making them predestined for application in databases such as GDELT, they have the disadvantage of being less precise than more modern approaches; as Kumar & Sebastian (2012) [13] have demonstrated, the classification is not domain specific, meaning that a word is always assessed as either positive or negative without taking into account the context in which it is used. Usually, the effect can be minimized by using a dictionary constructed for dealing with a particular type of text, but the issue becomes especially severe when dealing with data from a broad array of different topics, as is the case with GDELT. This raises the question of whether GDELT is even capable of adequately assessing the tone of the listed articles, which in itself opens up new and intriguing possibilities for further study.

While not necessarily a limitation, it should be noted that GDELT differentiates between entries based on the URL and not based on the source. While this might sound trivial, it plays a distinct role for news outlets such as iHeartRadio: An article that is published via iHeart.com is often reuploaded immediately on many of the radio channel websites that are subscribed to the service. In practice, this leads to some articles being listed multiple times, in extreme cases up to 20 times. Since the target audience can be assumed to differ and the upload of these articles ordinarily ensues in recent succession, thus falling on the same date, multiple entries listing the same article were not filtered out for the calculation of the average sentiment per day.

7. CONCLUSION

Sentiment of textual data is a concept that is difficult to define and even more difficult to capture correctly. The Global Database of Events, Language and Tone is an openly available dataset which automatically catalogues news reports from all over the world and assesses them on different metrics, among them sentiment. This study aimed at creating a link between sentiment data, as presented by GDELT, and stock market movements. In particular, the stocks of Facebook, Apple, Amazon, Netflix, Alphabet and Tesla were analyzed.

To filter the data from GDELT and make it usable for a cross-correlation analysis in the statistical analysis software SPSS, a business process model was created and subsequently implemented through use of the programming language Python. The original GDELT files were scanned and entries relating to one of the aforementioned companies were extracted. The entries were then transformed to match the date with the closing time of NASDAQ, and a daily average of the sentiment values was calculated for each of the companies.

The stock market data for each of the six stocks was retrieved from Yahoo Finance. Missing days, such as weekends and public holidays, were accounted for using a simple statistical function. After an additional step of harmonizing the format of the two datasets, they were merged using the Pandas data science library in Python, before conducting a cross-correlation analysis on the two time series data streams of each company using SPSS.

Unfortunately, Alphabet was the only company displaying a significant correlation between average daily media sentiment and the stock price. Significant correlations that were exhibited between the datasets of the other companies are likely a result of statistical error, due to the small sample size, or of external influences not accounted for by the research design. Nonetheless, the correlation and potential causation between media sentiment on Alphabet and the company’s stock price should be further investigated in future research. The lack of correlations between the other observed company’s media coverage sentiment and their stock prices may be a result of the numerous limitations that were discussed in this study, and should not be interpreted as proof that the GDELT dataset is not suited for financial analysis. However, the design of the database clearly favors geopolitical analyses and should be regarded as such when being used in future research.

8. ACKNOLEDGEMENTS

I would like to thank the members of my bachelor circle as well as my supervisor Fons Wijnhoven for their valuable insights, which were indispensable for conducting my research and writing on the thesis. Furthermore, I would like to thank my parents for their abundant dedication and support, as well as Carsten Mersch for helping me out when I was stuck with the Python coding.

9. REFERENCES

[1] Abu-Mostafa, Y. S., & Atiya, A. F. (1996). Introduction to financial forecasting. Applied Intelligence, 6(3), 205-213.

[2] BPMN 2.0 by example. Technical report. dtc/2010-06-02, Object Management Group, June 2010

[3] Coleman, S., Göb, R., Manco, G., Pievatolo, A., Tort-Martorell, X., & Reis, M. S. (2016). How can SMEs benefit from big data? Challenges and a path forward. Quality and Reliability Engineering International, 32(6), 2151-2164.

[4] Dormans, S. Multivariate Time Series Predictions of EU Sentiment.

[5] Elshendy, M., Colladon, A. F., Battistoni, E., & Gloor, P. A. (2018). Using four different online media sources to forecast the crude oil price. Journal of Information Science, 44(3), 408-421.

[6] Frizzo-Barker, J., Chow-White, P. A., Mozafari, M., & Ha, D. (2016). An empirical study of the rise of big data in business scholarship. International Journal of Information Management, 36(3), 403-413.

[7] Galla, D., & Burke, J. (2018, July). Predicting Social Unrest Using GDELT. In International Conference on Machine Learning and Data Mining in Pattern Recognition (pp. 103-116). Springer, Cham.

[8] Gregor, S., & Hevner, A. R. (2013). Positioning and presenting design science research for maximum impact. MIS quarterly, 337-355.

[9] Huang, W., Nakamori, Y., & Wang, S. Y. (2005). Forecasting stock market movement direction with support vector machine. Computers & Operations Research, 32(10), 2513-2522.

[10] Isotalo, V., Saari, P., Paasivaara, M., Steineker, A., & Gloor, P. A. (2016). Predicting 2016 US Presidential Election Polls with Online and Media Variables. In Designing Networks for Innovation and Improvisation (pp. 45-53). Springer, Cham.

[11] Kim, K. J. (2003). Financial time series forecasting using support vector machines. Neurocomputing, 55(1-2), 307-319.

[12] Kordonis, J., Symeonidis, S., & Arampatzis, A. (2016, November). Stock price forecasting via sentiment analysis on Twitter. In Proceedings of the 20th Pan-Hellenic Conference on Informatics (p. 36). ACM.

[13] Kumar, A., & Sebastian, T. M. (2012). Sentiment analysis: A perspective on its past, present and future. International Journal of Intelligent Systems and Applications, 4(10), 1-14.

[14] Malkiel, B. G. (1989). Efficient market hypothesis. In Finance (pp. 127-134). Palgrave Macmillan, London.

[15] Malkiel, B. G. (2003). The efficient market hypothesis and its critics. Journal of economic perspectives, 17(1), 59-82.

[16] McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. J., & Barton, D. (2012). Big data: the management revolution. Harvard business review, 90(10), 60-68.

[17] Qiao, F., Li, P., Zhang, X., Ding, Z., Cheng, J., & Wang, H. (2017). Predicting social unrest events with hidden Markov models using GDELT. Discrete Dynamics in Nature and Society, 2017.

[18] Roberts, H. 1967. Statistical versus clinical prediction of the stock market. Unpublished manuscript, CRSP, Chicago: University of Chicago. May.

[19] Sen, D., Ozturk, M., & Vayvay, O. (2016). An overview of big data for growth in SMEs. Procedia-Social and Behavioral Sciences, 235, 159-167.

[20] Shook, E., Leetaru, K., Cao, G., Padmanabhan, A., & Wang, S. (2012, October). Happy or not: Generating topic-based emotional heatmaps for Culturomics using CyberGIS. In 2012 IEEE 8th International Conference on E-Science (pp. 1-6). IEEE.

[21] Skehin, T., Crane, M., & Bezbradica, M. (2018, December). Day ahead forecasting of FAANG stocks using ARIMA, LSTM networks and wavelets. CEUR Workshop Proceedings.

[22] Tetlock, P. C. (2007). Giving content to investor sentiment: The role of media in the stock market. The Journal of finance, 62(3), 1139-1168.

[23] The Datasets Of GDELT As Of February 2016. (2016, March 13). Retrieved from https://blog.gdeltproject.org/the-datasets-of-gdelt-as-of-february-2016/

[24] The GDELT Global Knowledge Graph (GKG) Data Format Codebook V2.1 (2015). Retrieved from http://data.gdeltproject.org/documentation/GDELT-Global_Knowledge_Graph_Codebook-V2.1.pdf

[25] Ward, M. D., Beger, A., Cutler, J., Dickenson, M., Dorff, C., & Radford, B. (2013). Comparing GDELT and ICEWS event data. Analysis, 21(1), 267-97

10. APPENDIX

10.1 Python code

(Numbers represent steps in business process model)

(2) Summarize entries related to company for each day

```
output_amazon = open("C:/Users/Tibor/Desktop/GDELT_DC_amazon_06_02.csv", "a", encoding = "UTF-8")
output_apple = open("C:/Users/Tibor/Desktop/GDELT_DC_apple_06_02.csv", "a", encoding = "UTF-8")
output_netflix = open("C:/Users/Tibor/Desktop/GDELT_DC_netflix_06_02.csv", "a", encoding = "UTF-8")
output_google = open("C:/Users/Tibor/Desktop/GDELT_DC_google_06_02.csv", "a", encoding = "UTF-8")
output_tesla = open("C:/Users/Tibor/Desktop/GDELT_DC_tesla_06_02.csv", "a", encoding = "UTF-8")
output_facebook = open("C:/Users/Tibor/Desktop/GDELT_DC_facebook_06_02.csv", "a", encoding = "UTF-8")
counthour = '00'
chtemp = 0
countminute = '00'
pthtemp = counthour + countminute
errorcount = 0

while int(pthtemp) <= 2345:
    fntemp = 'C:/Users/Tibor/Desktop/GDELT Data/unzipped/02_06/20190602' + pthtemp + '00.gkg'
    filename = "%s.csv" % fntemp
    print(filename)
    try:
        temp1 = open(filename, encoding = "UTF-8")
        content = temp1.read()
        temp1.close()
        content_split = content.split(chr(10))  # ASCII character 10 = end of line

        for line in content_split:
            var = line.split(chr(9))
            try:
                if var[4].find("facebook") > -1:
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]   # %positive - %negative words
                    wc = tone[6]
                    output_facebook.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" +
wc + chr(10))
                if var[4].find("amazon") > -1:
                    var = line.split(chr(9))
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]
                    wc = tone[6]
                    output_amazon.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" + wc
+ chr(10))
```

```
                if var[4].find("apple") > -1:
                    var = line.split(chr(9))
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]
                    wc = tone[6]
                    output_apple.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" + wc
+ chr(10))
                if var[4].find("netflix") > -1:
                    var = line.split(chr(9))
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]
                    wc = tone[6]
                    output_netflix.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" +
wc + chr(10))
                if var[4].find("google") > -1 or var[4].find("alphabet") > -1:
                    var = line.split(chr(9))
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]
                    wc = tone[6]
                    output_google.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" + wc
+ chr(10))
                if var[4].find("tesla") > -1:
                    var = line.split(chr(9))
                    date = var[0][:8]
                    hour = var[0][8:10]
                    minute = var[0][10:12]
                    tone = var[15].split(",")
                    sentiment = tone[0]
                    wc = tone[6]
                    output_tesla.write(date + ";" + hour + ";" + minute + ";" + sentiment + ";" + wc
+ chr(10))

            except IndexError:
                print('IndexError occured')
        if countminute == '45':
            chtemp = int(counthour) + 1
            if chtemp <= 9:
                counthour = '0' + str(chtemp)
            else:
                counthour = str(chtemp)
        if countminute == '00':
            countminute = '15'
        elif countminute == '15':
            countminute = '30'
        elif countminute == '30':
            countminute = '45'
        else:
            countminute = '00'
        pthtemp = counthour + countminute

    except UnicodeError:
        if countminute == '45':
            chtemp = int(counthour) + 1

            if chtemp <= 9:
                counthour = '0' + str(chtemp)
            else:
                counthour = str(chtemp)
        if countminute == '00':
            countminute = '15'
        elif countminute == '15':
            countminute = '30'
        elif countminute == '30':
            countminute = '45'
        else:
            countminute = '00'
        pthtemp = counthour + countminute
        errorcount += 1
```

```
output_facebook.close()
output_amazon.close()
output_apple.close()
output_netflix.close()
output_google.close()
output_tesla.close()
print('UnicodeError occurences: '+ str(errorcount))
```

(3) Concat all files for each company

```
temp1=open("C:/Users/Tibor/Desktop/GDELT_concat_tesla.csv","a")

for num in range(1,29):
    for line in open("C:/Users/Tibor/Desktop/GDELT
Data/results_by_comp/tesla/GDELT_DC_tesla_day"+str(num)+".csv"):
         temp1.write(line)

temp1.close()
```

(4.1) Add NASDAQ date

```
output = open("C:/Users/Tibor/Desktop/tesla_correct_time.csv", "a", encoding = "UTF-8")
read_input = open("C:/Users/Tibor/Desktop/GDELT_tesla_NASDAQ_time.csv", "r", encoding = "UTF-8")
content = read_input.read()
content_split = content.split(chr(10))
count = 0

for line in content_split:
    if count < len(content_split) - 1:
        count += 1
        var = line.split(';')
        string1 = var[1] + ';' + var[3] + ';' + var[4] + chr(10)
        if int(var[0][8:10]) <= 31:
            output.write(line + chr(10))
        else:
            output.write('2019-06-01;' + var[1] + ';' + var[2] + ';' + var[3] + ';' + var[4] + ';' +
var[5] + chr(10))
```

(4.2) Solve problem with previous algorithm resulting in 32nd of May instead of 1st of June

```
output = open("C:/Users/Tibor/Desktop/tesla_correct_time.csv", "a", encoding = "UTF-8")
read_input = open("C:/Users/Tibor/Desktop/GDELT_tesla_NASDAQ_time.csv", "r", encoding = "UTF-8")
content = read_input.read()
content_split = content.split(chr(10))
count = 0

for line in content_split:
    if count < len(content_split) - 1:
        count += 1
        var = line.split(';')
        string1 = var[1] + ';' + var[3] + ';' + var[4] + chr(10)
        if int(var[0][8:10]) <= 31:
            output.write(line + chr(10))
        else:
            output.write('2019-06-01;' + var[1] + ';' + var[2] + ';' + var[3] + ';' + var[4] + ';' +
var[5] + chr(10))
```

(5) Get average sentiment per day

```
output = open("C:/Users/Tibor/Desktop/GDELT_tesla_avg.csv", "a", encoding = "UTF-8")
read_input = open("C:/Users/Tibor/Desktop/tesla_correct_time.csv", "r", encoding = "UTF-8")
content = read_input.read()
content_split = content.split(chr(10))
count = 0
days_dictionary = {}

for line in content_split:
    if count < len(content_split) - 1:
        var = line.split(';')
        count += 1
        templistname = var[0][:4] + var[0][5:7] + var[0][8:10]
        if templistname not in days_dictionary:
            days_dictionary[templistname] = []
        days_dictionary[templistname].append(float(var[4]))
```

```
days_keys = list(days_dictionary.keys())
count2 = 0

for i in days_dictionary:
    days_avg = sum(days_dictionary[i]) / len(days_dictionary[i])
    #print(days_avg)
    #print(days_keys[count2])
    output.write(days_keys[count2] + ';' + str(days_avg) + chr(10))
    count2 += 1
```

(6.1) Yahoo Finance data transformation

```
output = open("C:/Users/Tibor/Desktop/transformed_AMZN.csv", "a", encoding = "UTF-8")
read_input= open("C:/Users/Tibor/Desktop/AMZN.csv", "r", encoding = "UTF-8")
content = read_input.read()
content_split = content.split(chr(10))
count = 0

for line in content_split:
    if count < len(content_split) - 1:
        var = line.split(',')
        count += 1
        tempdate = str(var[0][:4]) + str(var[0][5:7]) + str(var[0][8:10])
        if var[0] != 'Date':
            output.write(tempdate + ';' + str(var[5]) + chr(10))#var[5] = adjusted close
```

(6.2) Merge files on date

```
import pandas as pd

fbnasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_facebook_we.csv", sep=';', header = None,
names=['date', 'fb_closing'])
fbgdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_facebook_avg.csv", sep=';', header = None,
names=['date', 'fb_avg_sent'])
fbmerged = fbnasdaq.merge(fbgdelt, on='date')

applenasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_apple_we.csv", sep=';', header = None,
names=['date', 'ap_closing'])
applegdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_apple_avg.csv", sep=';', header = None,
names=['date', 'ap_avg_sent'])
applemerged = applenasdaq.merge(applegdelt, on='date')

amazonnasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_amazon_we.csv", sep=';', header = None,
names=['date', 'am_closing'])
amazongdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_amazon_avg.csv", sep=';', header = None,
names=['date', 'am_avg_sent'])
amazonmerged = amazonnasdaq.merge(amazongdelt, on='date')

netflixnasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_netflix_we.csv", sep=';', header = None,
names=['date', 'nf_closing'])
netflixgdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_netflix_avg.csv", sep=';', header = None,
names=['date', 'nf_avg_sent'])
netflixmerged = netflixnasdaq.merge(netflixgdelt, on='date')

googlenasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_google_we.csv", sep=';', header = None,
names=['date', 'g_closing'])
googlegdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_google_avg.csv", sep=';', header = None,
names=['date', 'g_avg_sent'])
googlemerged = googlenasdaq.merge(googlegdelt, on='date')

teslanasdaq = pd.read_csv("C:/Users/Tibor/Desktop/transf_tesla_we.csv", sep=';', header = None,
names=['date', 't_closing'])
teslagdelt = pd.read_csv("C:/Users/Tibor/Desktop/GDELT_tesla_avg.csv", sep=';', header = None,
names=['date', 't_avg_sent'])
teslamerged = teslanasdaq.merge(teslagdelt, on='date')

total_merger =
fbmerged.merge(applemerged.merge(amazonmerged.merge(netflixmerged.merge(googlemerged.merge(teslamerge
d, on='date'), on='date'), on='date'), on='date'), on='date')
print(total_merger)
total_merger.to_csv("C:/Users/Tibor/Desktop/total_merged.csv", index=False, sep=';')
```

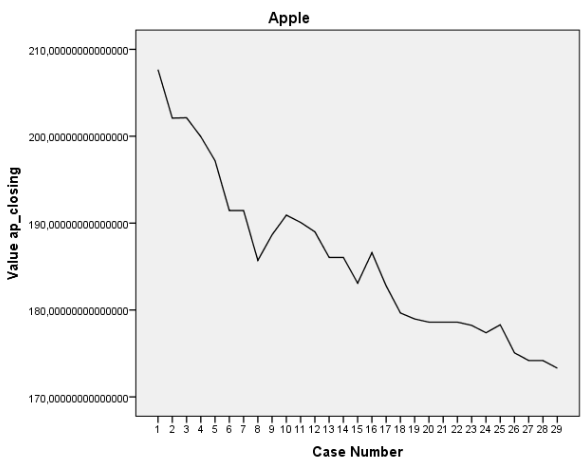
Convert period to comma

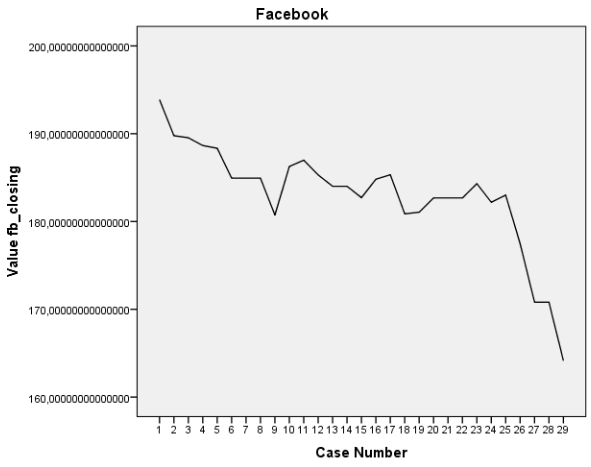
```
temp1 = open("C:/Users/Tibor/Desktop/total_merged.csv", encoding = "UTF-8")
transformed = open("C:/Users/Tibor/Desktop/transformed_total_merged.csv", "w", encoding = "UTF-8")
```

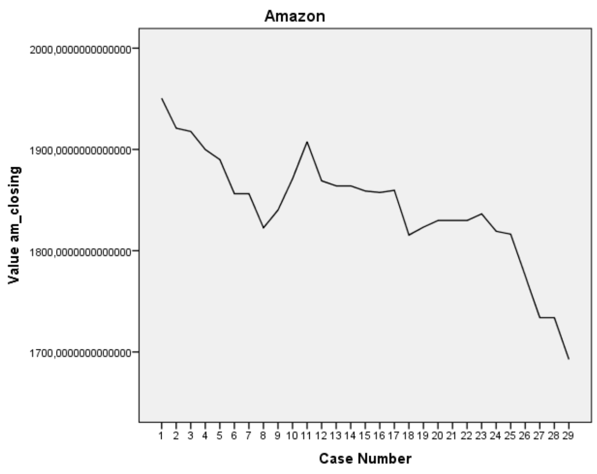
```
content = temp1.read()
temp1.close()
content_split = content.split(chr(10))
count = 0

for line in content_split:
    count += 1
    if count < len(content_split):
        transformed.write(line.replace('.', ',') + chr(10))
```

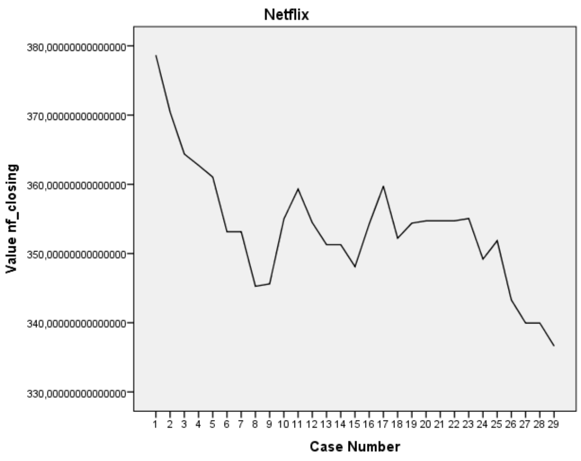
10.2 SPSS output

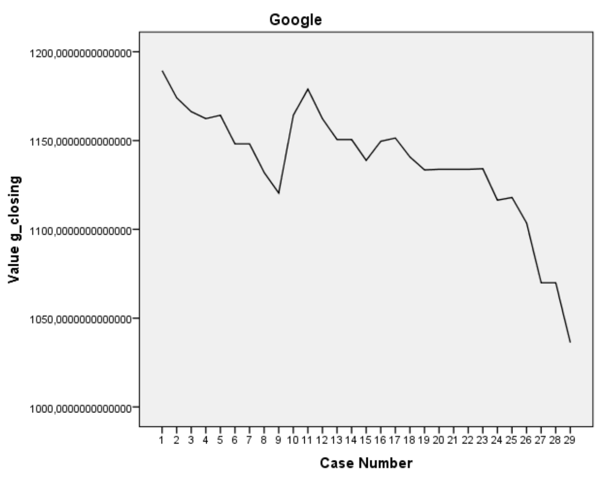
10.2.1 Stock price developments

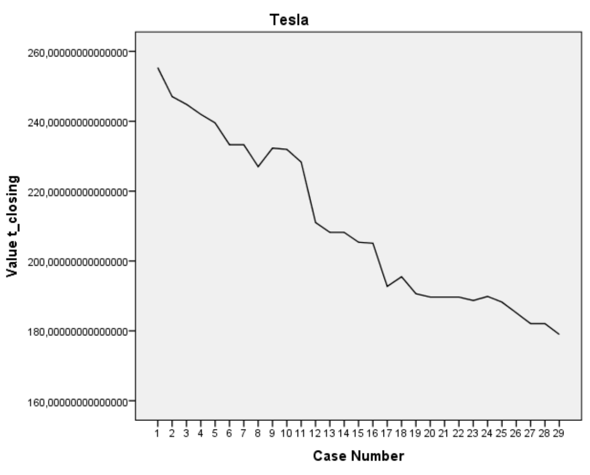




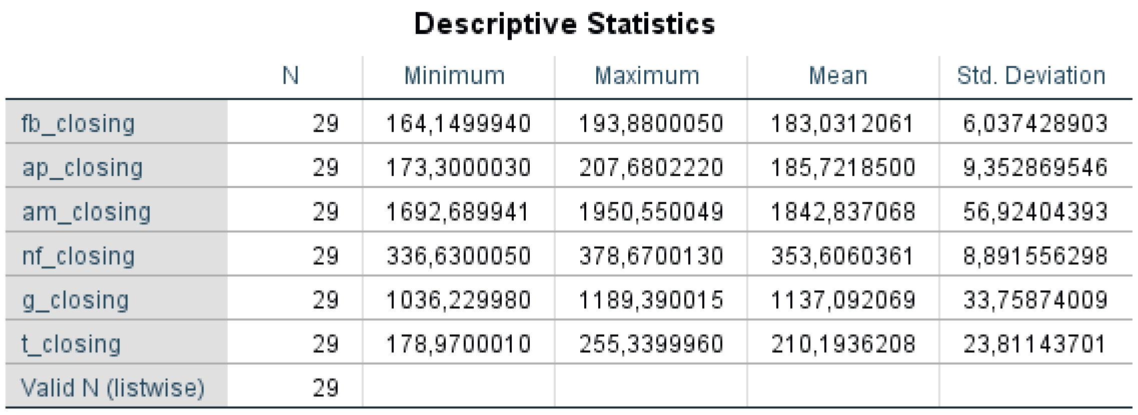




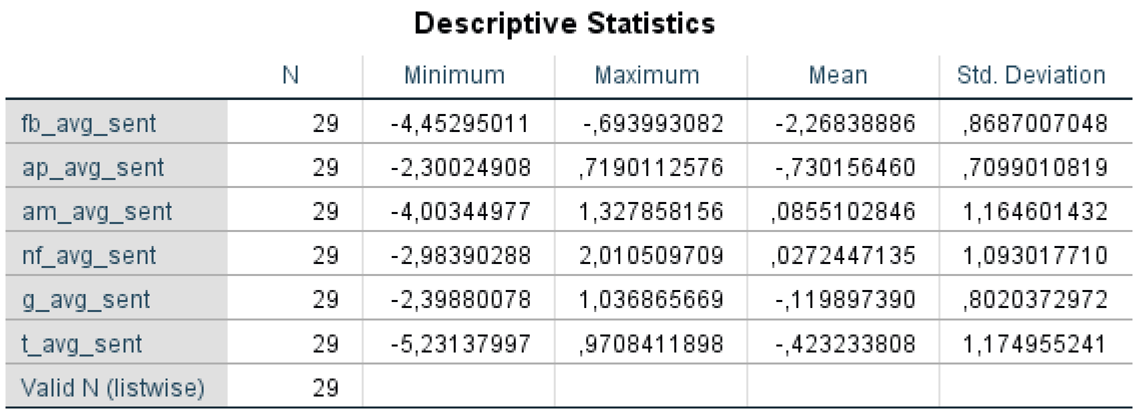




10.2.2 Descriptive statistics


Descriptive Statistics

	N	Minimum	Maximum	Mean	Std. Deviation
fb_closing	29	164,1499940	193,8800050	183,0312061	6,037428903
ap_closing	29	173,3000030	207,6802220	185,7218500	9,352869546
am_closing	29	1692,689941	1950,550049	1842,837068	56,92404393
nf_closing	29	336,6300050	378,6700130	353,6060361	8,891556298
g_closing	29	1036,229980	1189,390015	1137,092069	33,75874009
t_closing	29	178,9700010	255,3399960	210,1936208	23,81143701
Valid N (listwise)	29				


Descriptive Statistics

	N	Minimum	Maximum	Mean	Std. Deviation
fb_avg_sent	29	-4,45295011	-,693993082	-2,26838886	,8687007048
ap_avg_sent	29	-2,30024908	,7190112576	-,730156460	,7099010819
am_avg_sent	29	-4,00344977	1,327858156	,0855102846	1,164601432
nf_avg_sent	29	-2,98390288	2,010509709	,0272447135	1,093017710
g_avg_sent	29	-2,39880078	1,036865669	-,119897390	,8020372972
t_avg_sent	29	-5,23137997	,9708411898	-,423233808	1,174955241
Valid N (listwise)	29				

10.2.3 Cross-Correlation between stocks at lag 0

Lag = 0	**Facebook**	**Apple**	**Amazon**	**Netflix**	**Google**
Facebook					
Apple	0.794				
Amazon	0.964	0.868			
Netflix	0.869	0.805	0.887		
Google	0.965	0.784	0.970	0.852	
Tesla	0.735	0.961	0.799	0.666	0.723

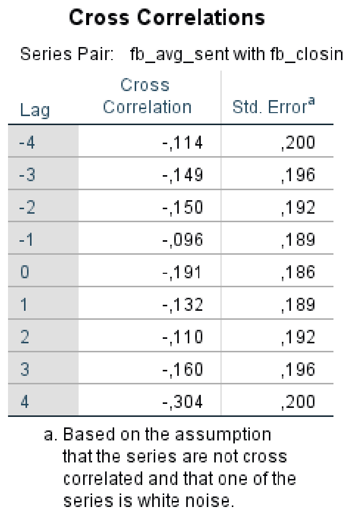
Std. Error = 0.186

10.2.4 Autocorrelation for each stock

Lag	Facebook	Apple	Amazon	Netflix	Google	Tesla
1	0.651	0.831	0.728	0.646	0.692	0.882
2	0.462	0.714	0.538	0.384	0.492	0.787
3	0.245	0.575	0.321	0.203	0.308	0.686

10.2.5 Cross-Correlation between average sentiment and stock price for each company

Facebook

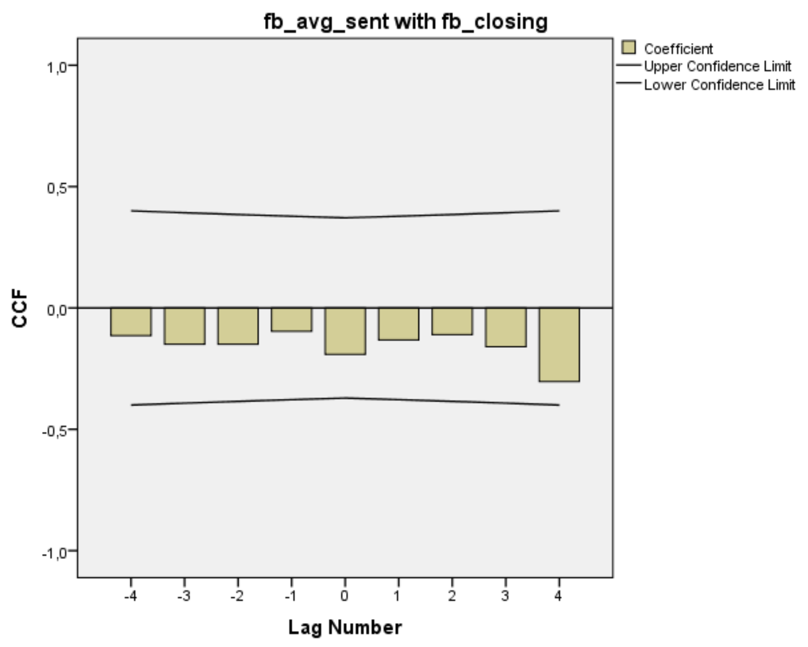


Cross Correlations

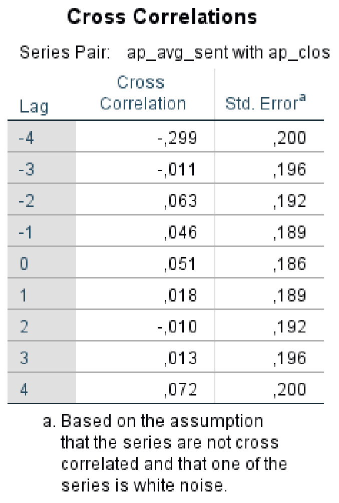
Series Pair: fb_avg_sent with fb_closin

Lag	Cross Correlation	Std. Error[a]
-4	-,114	,200
-3	-,149	,196
-2	-,150	,192
-1	-,096	,189
0	-,191	,186
1	-,132	,189
2	-,110	,192
3	-,160	,196
4	-,304	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



Apple

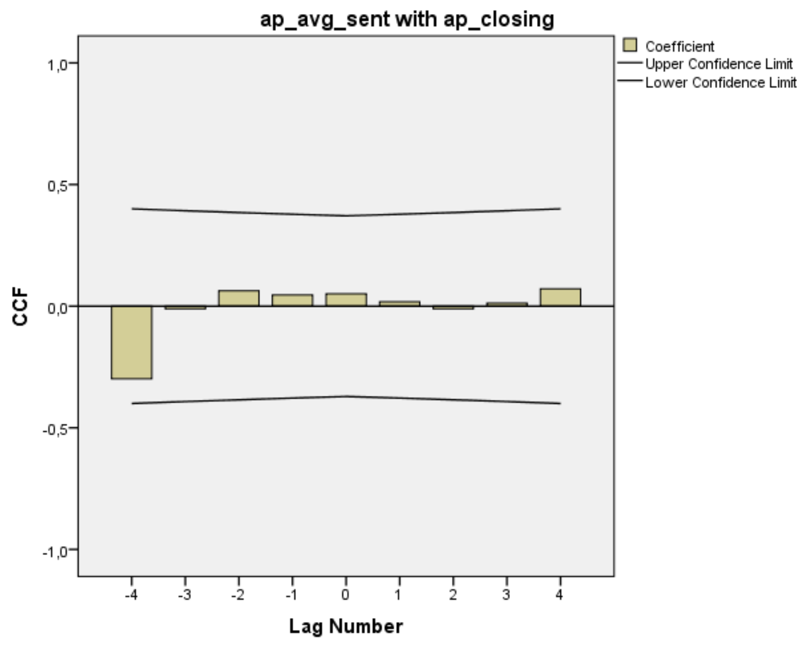


Cross Correlations

Series Pair: ap_avg_sent with ap_clos

Lag	Cross Correlation	Std. Error[a]
-4	-,299	,200
-3	-,011	,196
-2	,063	,192
-1	,046	,189
0	,051	,186
1	,018	,189
2	-,010	,192
3	,013	,196
4	,072	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



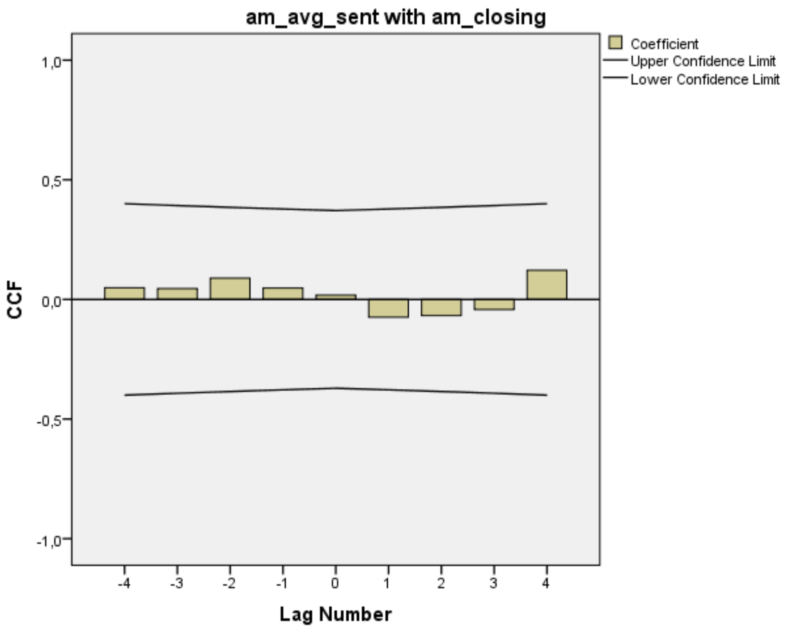
Amazon

Cross Correlations

Series Pair: am_avg_sent with am_clo

Lag	Cross Correlation	Std. Error[a]
-4	,049	,200
-3	,045	,196
-2	,089	,192
-1	,047	,189
0	,018	,186
1	-,075	,189
2	-,067	,192
3	-,042	,196
4	,122	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



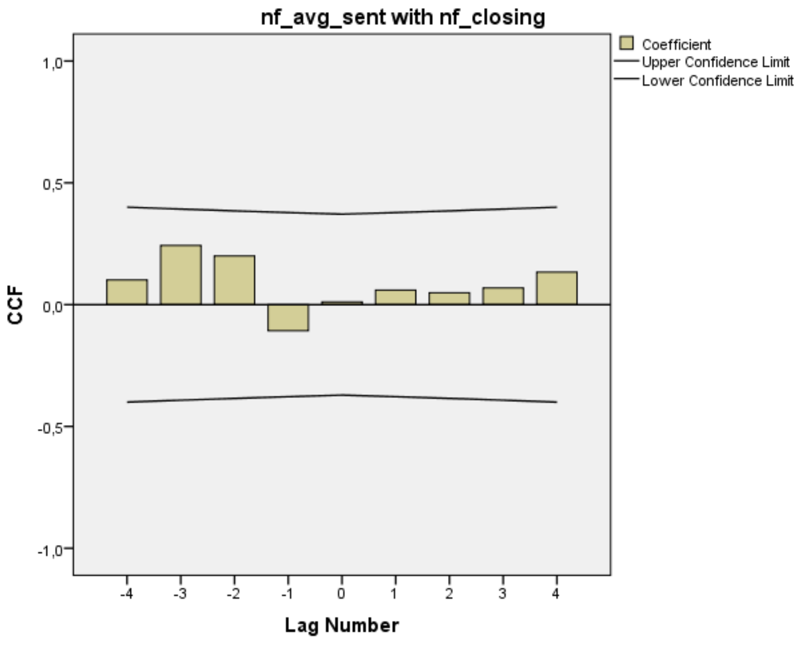
Netflix

Cross Correlations

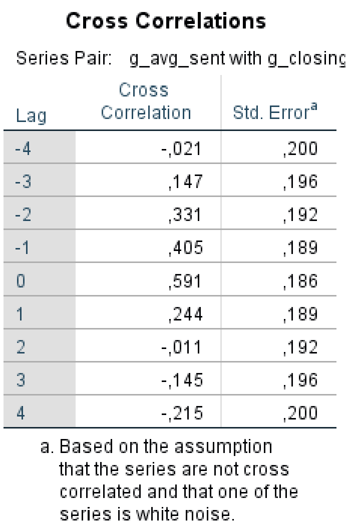
Series Pair: nf_avg_sent with nf_closin

Lag	Cross Correlation	Std. Error[a]
-4	,101	,200
-3	,243	,196
-2	,200	,192
-1	-,107	,189
0	,011	,186
1	,060	,189
2	,049	,192
3	,069	,196
4	,133	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



Google

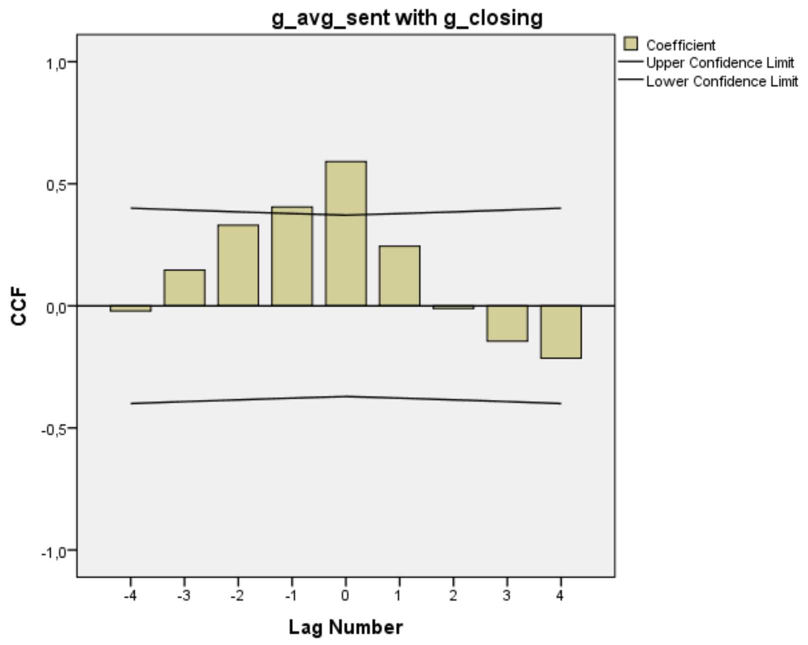


Cross Correlations

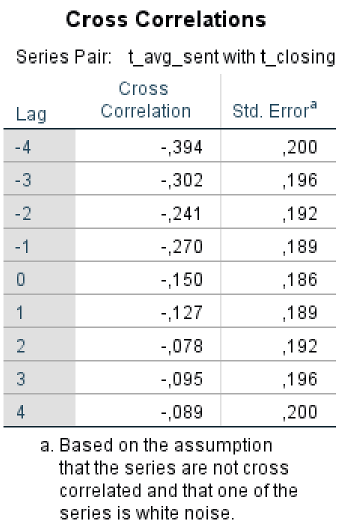
Series Pair: g_avg_sent with g_closing

Lag	Cross Correlation	Std. Error[a]
-4	-,021	,200
-3	,147	,196
-2	,331	,192
-1	,405	,189
0	,591	,186
1	,244	,189
2	-,011	,192
3	-,145	,196
4	-,215	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



Tesla

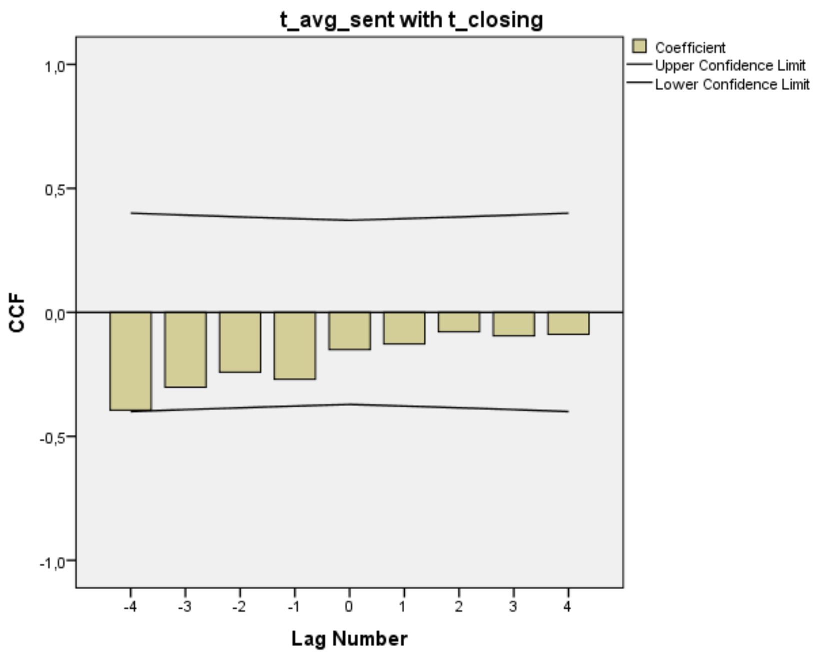


Cross Correlations

Series Pair: t_avg_sent with t_closing

Lag	Cross Correlation	Std. Error[a]
-4	-,394	,200
-3	-,302	,196
-2	-,241	,192
-1	-,270	,189
0	-,150	,186
1	-,127	,189
2	-,078	,192
3	-,095	,196
4	-,089	,200

a. Based on the assumption that the series are not cross correlated and that one of the series is white noise.



10.2.6 SPSS file containing merged GDELT and NASDAQ data

The result of step (6) in the business process model can be found here:

https://drive.google.com/file/d/1fkMKe1dUffGTtm7cqNsSs7CqvrXe6LfI/view?usp=sharing

Note that access is only permitted to University of Twente e-mail accounts and running the file requires a working (licensed) version of SPSS. The file was created in SPSS 24, so compatibility might be limited when using a different version.