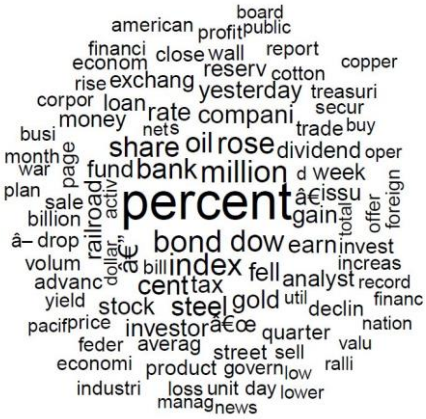
Does financial news influence the market?

Analyzing news impact on market returns with Supervised Latent Semantic Indexing



Erasmus School of Economics

Rotterdam, 09 of March 2017

Subject: Master Thesis

Name: Valentin Sigrist

Student Number: 427452vs

Supervisor and First Reader: Dr. Vitalie Spinu

Second Reader: Dr. Guido Baltussen

Content

1 Introduction

During the preparations of this analysis in late 2016, political events presented ample opportunity to observe how media may influence decision making in a democratic society. Influencing the opinion of US voters led to an unprecedented power shift in their country. After the US presidential elections in September 2016, the political landscape shifted from conventional to non-conventional and from predictable to unpredictable. Of course, not the media alone caused this political shift, but media echoed and multiplied populistic messages and thereby influenced election outcomes. Media are an excellent channel to report on past and present issues. However, media may also be used to predict future events. Predictability might enable democracies to foresee and in some cases, prevent specific political events in the future.

More than anywhere else, predictability is a high valued asset in the financial world. Yet, the financial markets dynamics, too, are both reported and influenced by the media. Hence, media reporting could be a predictor to financial markets, considering its obvious impact elsewhere. Irrationality in markets has already been empirically shown in the behavioral finance literature. The notion of bringing news media into the predictor universe is therefore not that far-fetched. The question is: *Does news text data have the potential to drive market returns?*

Retrieving information from text into quantifiable variables is likewise a barrier and opportunity to be mastered. The fact that information retrieval is quite imprecise is certainly a barrier because it is words and not numbers to be dealt with. Reducing imprecision could therefore enhance the opportunity that comes with the vast and ever growing amount of data stored in text.

An important step to improve precision is to filter what is important. That is, ignoring those parts of the text that are irrelevant to what should be predicted. For any topic, large parts of a text are irrelevant. For example, when reading any text, only a fraction of what is written is of actual interest to a specific reader in a given moment. This fraction depends on the readers´ interest, one time we look for a quick overview and another time for detailed insights. Ignoring the rest of the text is easy for most humans but most tricky for algorithms. Now consider market returns: Which fraction of a news article is most prone to influence market participants and drive market returns? In an attempt to incorporate this fraction thinking into a text

processing model, a suitable approach is *supervision*. *Supervision* is the process of creating a connection between the independent variable and words, before text categorization happens. By doing so, irrelevant words are filtered out right from the beginning.

For information retrieval, also known as text mining, a range of tools, broadly described as lexicon based and machine learning based approaches, can be used. Both approaches categorize words into topics. The latter does so by statistical indexing and the former with the help of predefined lexicons.

The method chosen in this paper is a *supervised machine learning* approach. This tool, called *Supervised Latent Semantic Indexing* will be used to retrieve information from 35 thousand news articles from The New York Times, reporting on over 100 years of financial markets history. The resulting quantitative information will serve as a potential predictor for the Dow Jones Industrial Average. The aim of predicting the stock index is as ambitious as difficult because an entire range of other variables such as macroeconomics, momentum and sector metrics play a role.

The remainder of this analysis will be structured as follows. First, related scientific work will be examined. The literature review will touch upon the topics of behavioral finance and the connection between finance, market sentiment and text mining. Machine learning based text mining and the basis for a supervision algorithm will be explored. In a subsequent methodology outline, the combination of supervision, automatic indexing and a predictive time series model will be set out. That section will refrain from technical details. Thereafter, the technical and programmatic side of the analysis is presented. It includes text pre-processing and econometric peculiarities of time series modelling. Finally, results, alternative models and limitations will be presented. The conclusion section will conclude.

2 Literature Review

The traditional approach to information retrieval is lexicon based text mining. Researchers in this domain are often linguists or psychologists, trying to disentangle text with sophisticated topic queries. A good example for such work is *Lexicon-Based Methods for Sentiment Analysis* (Taboada & et al., 2011). The authors describe a sentiment lexicon called *Semantic Orientation Calculator* (SO-CAL). Their method provides the polarity and strength of certain semantic patterns, mainly positive and negative polarization. In their paper, they describe a variety of steps and linguistic peculiarities for the creation of SO-CAL.

To make their semantic lexicon as extensive as possible, the authors suggest two approaches. The first is to create the lexicon with seed words for a semantic direction and later take synonyms and antonyms for these seed words. The General Inquirer (Stone , Dunphy, Smith, & Ogilvie, 1966) and WordNet (Hu and Liu, 2004) are popular examples for lexicons that are created this way. The second approach is using automatic classifiers, where the classification algorithm is fed with seed words just like above and then searches for correlating words in a training set. This way, words that have similar frequency patterns across text documents are classified as synonyms. Reversely, if their frequency diverges, they are classified as antonyms. As this involves classification and training data, it is also known as machine learning approach to lexicon creation.

Subsequently, intensification and negation are introduced to SO-CAL, as they pose a tricky part of lexicon based text processing. Both can have nested effects which can turn around the polarization of a text fragment. Finally, Taboada & et al. briefly comment on weighing schemes, which are important to the relative quantity of words used in text. Words that appear often are inflated when it comes to their impact on a semantic score. For this reason, they must be weighted down, otherwise they deteriorate model outcome. An important weighing scheme is called Term Frequency – Inverse Document Frequency (tf-idf) which will be discussed in depth in section 5.1.5 of this analysis.

In text mining, the authors usually try to read out a variety of polarizations and semantic directions. The General Inquirer (Stone , Dunphy, Smith, & Ogilvie, 1966), for example has 77 predetermined categories.

In this analysis, the direction of text mining is pointed towards financial markets and asset returns. Authors who deal with text mining with regard to asset returns, usually seek two directions: positive and negative. They refer to it as sentiment in the market.

Sentiment can be broadly defined as collective buying or selling pressure which can emerge in short- and medium term horizons. Sentiment is mostly due to movements in macroeconomic fundamentals, which is a rather rational market reaction. However short term sentiment is often due to irrational and short sighted buying or selling behavior. This short horizon "chatter" about markets in the moment is assumed to be reflected in the news on a higher frequency than fundamentals. To provide a conceptual basis for sentiment in general, papers by Wurgler & Baker (2007) and Shleifer and Vishny (1998) are considered.

Wurgler & Baker (2007) deal with sentiment in a top-down approach. They identify macroeconomic consequences and market dynamics resulting from irrational investor behavior and provide insights to quantifying the concept of sentiment. They start examining investor surveys and mood, then continue with trading quantities and capital flows. More specific figures are IPO- and insider trades as well as option implied volatility. Next, they construct an index out of the most reliable and available variables. These are trading volume, dividend premium, closed-end fund discounts, IPO data and new issues equity shares. Sentiment betas are calculated as coefficients of the sentiment index with stock returns. The stock picks range from safe and predictable to speculative and uncertain stocks. A central proposition is, that the latter are more likely to be subject to sentiment due to their nature. Because of a lack of information, uncertain stocks are prone to subjectivity in their valuation which leads to speculative and irrational buying behavior (DeLong, Shleifer, Summers and Waldmann, 1990). Finally, the proposition of high sentiment in uncertain stocks is confirmed empirically. Furthermore, high sentiment predicts relatively lower returns for speculative stocks compared to safe stocks Wurgler & Baker (2007).

Shleifer and Vishny (1998) provide more insights on noise traders and arbitrageurs. Their work is important because it relativizes perfect arbitrage as defined in classic finance models. If perfect arbitrage was assumed in markets, no short-term sentiment would be present because it would be arbitraged away immediately. Shleifer and Vishny (1998) however describe situations where inflated security prices do not return to their fundamental value. During this period, arbitrage traders experience opportunity costs for an alternative investment or even

losses in case of a further price deterioration. Arbitrage traders face the pressure of performance based fund allocation. If their fund incurs temporary losses, investors might eventually withdraw money. Potentially profitable positions are then liquidated with a loss before they pay out. At the same time, only high volume arbitrage trades can bring asset prices back to their fundamental value. These trades are less likely to happen when the respective market participants have less funds available. Arbitrage trades in stocks whose fundamental value is difficult to estimate are even less likely to happen. These are the same stocks which are prone to sentiment trades in the first place (DeLong, Shleifer, Summers and Waldmann, 1990). Arbitrage dynamics therefore plays a crucial role in market sentiment, making it much longer lasting than traditional models would suggest.

Tetlock (2007) presents an analysis which links text mining to sentiment in stock markets. He analyses a column by The Wall Street Journal (WSJ) called *Abreast of the Market*. The methodology employed could be called a hybrid approach to text mining. This is because a lexicon based approach is used with the support of machine learning techniques. Text mining can be tackled in these two distinct ways. Tetlock uses The General Inquirer (GI) (Stone , Dunphy, Smith, & Ogilvie, 1966), the popular lexicon introduced above. GI software counts words and classifies them into 77 psychosocial categories. Each daily WSJ column over the period 1984 until 1999 has frequencies on each of these categories. Out of these variables, a media pessimism factor is constructed as follows. The extracted underlying media pessimism factor is a linear combination of the 77 GI categories. In order to capture the entire range of captured variation, Tetlock performs a Principal Component Analysis (PCA) on the term frequencies of all categories. With a PCA, several principal components are calculated, all are uncorrelated to each other by definition. Taking the first few strongest principal components from the PCA results in a lower-dimensional representation of the GI data. Here the author does not look for precise document representation but rather for a low-dimensional source of sentiment variation. In Tetlocks' terminology, dimensions are the entire range of unique words appearing in all WSJ documents. These dimensions are in fact reduced twofold, first by categorizing them with the General Inquirer (lexicon based) and second by performing the Principal Component Analysis (machine learning based). Finally, the author uses only one principal component: the one that captures the highest variation in the underlying GI categories. This principal component turns out to have high loadings the GI categories *pessimism* and *negativism*, which is why it is called the media pessimism factor.

The pessimism factor enters a time series model predicting returns of the Dow Jones Industrial Average (DJIA), not disregarding a range of control variables such as past volatility, autocorrelation and a SMB variable. Small stocks are identified as being particularly prone to sentiment, having high loadings on the pessimism factor. This is because their ownership structure is more concentrated in individual investors, which are in turn more prone to irrational investment behavior. Tetlock finds that his pessimism factor indeed has a bigger impact on small stocks than on the market overall. This supports the statements about noise traders in Wurgler & Baker (2007). Tetlock´s sentiment index reasonably predicts market returns and could be subject to a trading strategy yielding 7.3% annual excess return. The same index is however not suitable to predict market volatility and the trading strategy does not consider trading costs.

A second analysis on sentiment and media text is a plain lexicon based approach. Garcia (2012) uses a lexicon based text mining approach to extract positive and negative semantics from media text data. His working hypothesis is that the market will be more sensitive to news during recessions. The hypothesis rests upon research on subjects´ decision making abilities in different types of moods. It turns out that uncertainty and fear are collectively felt by traders during recessions and that this type of mood makes traders more prone to be influenced by media (Gino, Wood and Schweitzer, 2009). Garcia develops a sentiment index which he incorporates into a time series model predicting returns of DJIA. The text data used are two New York Times (NYT) columns which appear daily. Compared to other analyses, Garcia uses a relatively easy lexicon based approach, counting positive and negative words with a lexicon specified on finance jargon (McDonald, 2011). His sentiment index is the fraction of positive or negative words over all words of a column. The findings confirm the proposition that during recessions, investors buy more irrationally, therefore causing sentiment in the market which is measurable in news media text.

The Garcia paper is important to this analysis because his dataset and the same stock index are used, however with an entirely different methodology.

While lexicons and psychosomatic queries such as the General Inquirer can retrieve information on a specific topic, it is more difficult to perform a topic-unrelated information retrieval. A lexicon will return term frequencies on topics, but it will disregard how the same words can have different meanings in different phrase contexts. Creators of sophisticated

lexicons place great concern on this issue by working out things like negations, polarity flips, shift negation and intensifications (Taboada et al 2011). The core problem behind this issue in linguistics is called *polysemy* and *synonymy. Polysemy* means that in most languages many words have more than one meaning. *Synonymy*, as one might guess, is the fact that one topic can be described in myriads of different word combinations. These two linguistic issues are a major cause for poor precision in lexicon based text mining. Deerwester (1990) claims that *Latent Semantic Indexing (LSI)* is suitable to overcome these issues.

Rather than manually sorting word groups into categories, in LSI, words are arranged into word clouds based on their occurrence patterns over documents. A document can include a number of words for a topic. Another document deals with the same topic but the author uses different wording to describe it. When many documents revolve around the same topic, however, specific words occur together often. These word clouds are referred to as latent text structure. To detect latent text structure, the authors employ a method called two mode factor analysis, which has the advantage that it creates representations of both the documents and the words in a text. Two mode factor analysis is also known as *Singular Value Decomposition* (SVD), almost the same dimensionality reduction technique as PCA (used in Tetlock, 2007). Deerwester describes it in the context of LSI as a method to derive uncorrelated (orthogonal) concepts which reasonably represent the underlying variables. While the original corpus can easily have several thousand dimensions, the number dimensions representing it after the SVD can be a few hundred or less. The author recommends to employ a *higher dimensional representation* of several hundred concepts to enhance precision. He goes on testing the LSI method with two text datasets against a lexicon and achieves better or equal performance on information retrieval. In these tests, an *impoverished* version of LSI is being used for demonstrational purposes. *Impoverished* in the sense that common text mining instruments such as stemming, sparse terms deletion and stop word deletion are not yet used. Drawbacks of LSI are that it typically needs large amounts of data to yield generalizable outcomes. Furthermore, it does not perform very well on noisy text data.

LSI can be used to make a lower dimensional representation of text. It can also be used to look for specific word clouds and their impact on an *external* variable. However, in its classical

application, always the entire text corpus is taken into account by LSI, while only parts of it might be of interest regarding an external variable.

An extension for a more channeled use of LSI is introduced by Bair et al. (2006). The authors come from the field of DNA biology and seek to predict survival rates of cancer patients based on characteristics in their DNA. They use gene expression measurements from DNA microarrays. More generally, the authors deal with the problem of predicting a variable with predictors whose quantity is much larger than the number of observations of it. Bair et al. also assume that many underlying gene characteristics are uncorrelated with survival rates. A large part of their variable set is thus noise which deteriorates meaningful conclusions from predicted characteristics. These DNA biologists face the same noise problem described above: In text mining, one can run into document quantity limitations while having a large set of terms and redundant text structure.

Bair et al. introduce an augmented form of Principal Component Analysis (PCA) which they call *Supervised Principal Component Analysis* (SPCA). It has useful implications for the exploration of latent semantic structures, too, specifically with the aim of prediction. The underlying procedure is simple. Before performing PCA, the matrix of independent variables is reduced to only those variables that are correlated with the dependent variable. Any variables whose absolute linear univariate coefficient with the dependent variable is less than some cutoff value are left out. PCA is then performed only with the reduced set of independent variables. Bair fits only the largest principal component into a regression model. Compared to other, more sophisticated methods, the authors find SPCA to be performing equally well or better, given its relative simplicity.

3 Model Outline

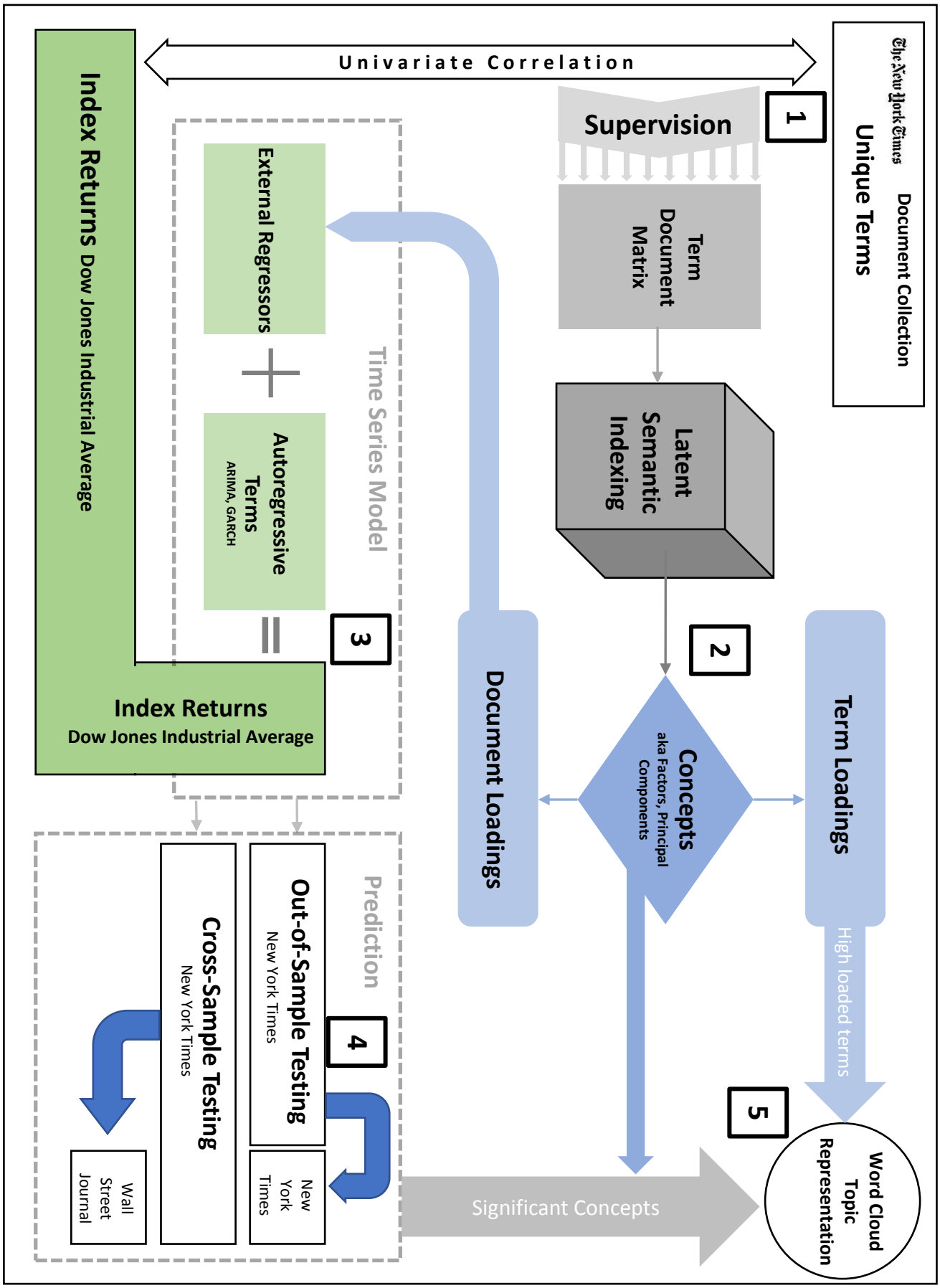
The following section provides an overview to the entire methodology to illustrate the step-wise approach that was taken. Not one part of the methodology is most important, but the process and combination as a whole. This overview is also important to embed more specific research hypotheses into the methodological context. These hypotheses will be formulated after this section. Two directions of research questions motivated this analysis. Not only the general relationships between news media and index returns are explored but also the benefit of *supervision,* which will be explained directly after this introduction.

RQ 1: *Does news text data consist of detectable latent semantic structures which are potential predictors for the market?*

RQ 2: *Does supervision enhance the detection of relevant latent semantic structures which are potential predictors for the market?*

The following methodology assembles several concepts from the literature into one framework, which is best described as *Supervised Latent Semantic Indexing* in combination with a time series regression model. Figure A provides a graphical guidance through five steps of this framework with each step explained separately in the following paragraphs.

Figure A: Flow- Chart of Supervised Latent Semantic Indexing and Time Series Modelling



Step 1: Document Collection, Supervision and Cross – Validation

The first step taken is the quantification of a large collection of financial news text documents. The occurrence or frequency of every unique term in any of the text documents is counted. The result is called term frequency per document. Since the news documents appear daily, every term is turned into a daily variable through its term frequency. Market index returns are also reported daily, which enables matching news document to every return observation. Every term is then related to the index returns by calculating a linear univariate regression coefficient between its frequency and the market returns. Terms with absolute coefficients close to zero have low correlation with the index returns, implying that their frequency is *irrelevant* to the market returns. These terms are left out of further analysis, providing a restriction to the text information at an early stage. This restriction is called *supervision*.

Supervision happens in ten stages, also known as ten-fold cross-validation. On every stage, the term coefficient threshold for the term to enter further analysis is set higher. This is because it is unknown how high the threshold should be for an optimal filtering of relevant terms. At every cross-validation stage, a specific number of terms enters the following steps of the model. These following steps are thus calculated ten times in a parallel fashion, resulting in ten sub-models in any step. The ten "filtering" stages are also referred to *supervision strength* because restriction through supervision gets stronger with every cross-validation stage. The effectiveness of supervision in text analysis and prediction is uncertain. For this reason, it is subject to a separate research question and hypothesis.

Any methodological steps taken regarding text data, term frequencies, cross-validation and supervision strength are derived in detail in sections 4, 5.1 and 5.2.

Step 2: Latent Semantic Indexing

Subsequently, text data is reorganized into the form of a *term-document matrix.* The collection of text documents is displayed in a matrix where documents are in the row names, terms are in the column names and the cells contain the corresponding term frequencies per document. This matrix only contains those terms in the columns that are not filtered in during supervision.

Latent Semantic Indexing (LSI) is then applied to the *term-document matrix.* As touched upon earlier, *Latent Semantic Indexing* is a two-mode factor analysis, designed to reduce the dimensionality of the text data. For the *term-document matrix*, the document dimensions are the terms because every document has a number of terms it is made of. LSI creates alternative dimensions whose quantity is smaller than the number of terms and documents, respectively. The result is a lower-dimensional representation of the *term-document matrix.* These dimensions are called *concepts* and they represent mentioned latent text structures with concentrated information out of the underlying text. The concepts might contain channeled and concentrated information from news media. This information might drive market index returns when it is relevant to buying and selling behavior.

Documents have loadings on concepts, indicating how much of the corresponding informational channel is present in each document. Document loadings are processed in the following Step 3. A detailed derivation of the *term-document matrix, Latent Semantic Indexing* and a discussion about concepts is presented in section 5.1, 5.2 and 5.3.

Step 3 and 4: Prediction through Document Loadings as Independent Variables

After performing SLSI (Steps 1 and 2), document loadings are used as independent variables in a linear time series regression model to predict index returns. On any given day, the index returns might be related to some concept, or informational channel, being prevalent in that day´s news text. Theoretically, some concepts could be associated with particularly high or low returns and could thus be used to predict them. The document loadings enter the linear time series regression together with autoregressive terms from the index returns, serving as correction variables. The quality of this model is judged by letting it predict index returns out of sample. This test set consists documents that have not entered SLSI. Index return prediction with out-of-sample documents is evaluated by how far predicted returns deviate from the actual returns. Mean prediction errors give indication about the model´s accuracy. Note, that every model is calculated ten times, one time for each supervision strength stage (Step 1). This implies multiple prediction errors whose distribution depends on the supervision strength. The set of prediction errors is subject to the second research question which puts them in relation to supervision strength.

Furthermore, out-of-sample testing happens in two different text samples. This is of minor importance at this point but will later serve as robustness test. If concepts are extracted from a different text sample than they are tested on, the outcome of such testing gives information on the generalizability of these concepts. Generalizable concepts imply sample-independent text structure. Similar prediction errors on two different test samples point toward generalizable concepts.

The time series model and testing schemes are derived section 5.2 and 5.3.

Step 5: Concept representation by Word Clouds

Apart from predictability, a qualitative method called *word clouds* is used to illustrate the informational topic of chosen concepts. Word clouds are graphs that print words of higher numerical importance bigger. An example is printed on the cover if this paper. Concepts, whose document loadings are significant in time series models with low prediction errors contain significant concentrated information. Terms that occur in these concepts carry their information and are therefore displayed in a word cloud. Section 6.6 provides more detail.

Research Hypotheses

Four research hypotheses are formally tested with the described methodology:

H1: There is a detectable correlation between document concept loadings from news text media and market index returns.

H2: Document concept loadings from news text media can serve as predictors for market index returns.

H3: Supervision improves model prediction when tested on the same text sample source.

H4: Supervision improves model prediction when tested on a different text sample source.

4 Data

4.1 Text Data

The news media text dataset is a collection of columns from the New York Times (NYT) and the Wall Street Journal (WSJ). These columns are available in the newspapers archives all the way back to its origins in 1851 for the NYT and until mid-1980 for the WSJ.

For the NYT, two financial columns were published in the daily print continuously until the present. These two columns, called “Financial Markets” and “Topics in Wall Street” have long been viewed as the most read financial columns in the world. Both columns are around 700-900 words long and have the purpose of summarizing the market dynamics on that day with a clear focus on finance, industry and macroeconomics. Compared to industry earnings announcements or market specific news, these daily briefings are more likely to contain semantic structures. The former will always be based on quantitative indicators wrapped in text. Columns however, package the market dynamics in words which also contain collective opinion about the present and the future. Collective, because the journalists that wrote the columns are not professional market analysts. They rather reproduce what the professionals tell them on a daily basis and from a wide range of opinions (Garcia, 2012). Most of the time span, these columns are only available as scanned pictures. *Optical character recognition* (OCR) was used to digitalize scanned text which is crucial to enable word counts. Before 1905, the quality of the scans was too poor to recognize characters correctly which is why the dataset starts in that year.

The NYT dataset was collected originally by Garcia (2012) just like described above. It was taken over digitalized and time-stamped. It starts on the 4th of January 1905 and ends on the 23rd of December 2015 which amounts to 35492 unique documents on trading days. For Garcia´s work, this large amount of data was certainly helpful but assumingly not crucial. Clearly, to *this* analysis it is. Due to the law of big numbers, automatic indexing such as SLSI works most reliably with large amounts of data, which makes the size of this dataset an enabler for this analysis.

The Wall Street Journal (WSJ) dataset is digitalized available on Factiva (Dowjones, 2017) and includes several columns that have the same purpose as the NYT columns and approximately

the same length. The ones covering most trading days are called “Abreast of the market” and “Business Finance”. The dataset starts on the 1st of January 1984 and ends on the 3rd of October 2016, amounting to 8486 unique documents on trading days. It is therefore too short to be used in SLSI. Here, it serves as cross-sample testing set, enabling a longer part of the NYT dataset to be used for training and providing test data from a different sample source.

4.2 Market Index Data

The default index used is the Dow Jones Industrial Average (DJIA) retrieved from the Center for Research in Security Prices (CRSP, 2016) during the last century. This index was the oldest stock index on the American market and the first worldwide. It is today comprised of the 30 largest US companies. The DJIA´s components are not determined by performance metrics but rather by their public reputation, sustained growth and investor interest in a qualitative sense. The index does not include dividend payments and thus reflects only price movements based on buying and selling pressure (S&P Global, 2017). In many financial analyses, the authors use an index that represents a proxy to the market portfolio. The market portfolio in theory reflects all stocks that are traded in the market, which is considered diversified up until pure market risk. For such a portfolio, the S&P 500 is usually considered because it reflects a much larger percentage of the market compared to the DJIA. However, not the market portfolio is needed here, but rather the one that tracks the largest and most popular stocks. For this, the DJIA more suitable.

It is also attractive because of its long-term availability. In fact, when using a supervised model together with a time series model the index must be available throughout the entire text data coverage. This is because the index (as of now being referred to as market index) is incorporated twice in this analysis. First, it is used for supervision by identifying correlations with term frequencies and second it is the dependent variable on which a time series model is fitted. The Dow Jones is the only financial index that exists this long.

5 Methodology

Some details have intentionally been omitted in above´s outline. These are less important for an overview but vital to the specificity of this analysis. The following section will examine these details. First, the preprocessing of the New York Times and Wall Street Journal text dataset will be examined. Then, a brief, but more mathematical explanation to *Supervised Latent Semantic Indexing* will be given. Third, market index data will be explained including standard time series corrections. This section will conclude with a brief discussion of the number of concepts entering the time series regression model, as well as testing schemes of the same.

5.1.1 Data Sorting

Before being able to use the NYT dataset, it had to be cleaned and sorted. This is often taken for granted but it was very time consuming and therefore deserves a paragraph.

The first challenge is, that the dataset sometimes shows multiple entries per day. Theoretically it should have one to three articles per trading day, depending on the time. Unfortunately, some other NYT columns got mixed into the dataset and were not clearly tagged. On top of that, most trading days in the earlier part of the dataset have more than three entries. From this, it is concluded that there might be copies of the same articles on the same trading day.

In order to identify copies, the R-package *stringdist* (van der Loo, 2016) is used to calculate the text distance of two documents. The metric is based on how many single character edits would have to be made to change one text string to the other. This method is also known as Levenshtein-Method (Levenshtein, 1965) and gives a good indication whether two strings are equal. A straightforward indicator per two strings is calculated with the string distance divided by the sum of the character length of the two strings. Equal strings turn out to have an indicator of below 0.1. Unequal strings have an indicator of around 0.5 because it takes almost all characters to change an unequal string into another, which is around half of the sum of both if they have approximately the same length. With this procedure, around 3000 identical strings, thus copies were identified and deleted. After all, the above method is obsolete once a properly tagged dataset is available. Nevertheless, it is a handy approach to deal with messy text datasets and might add value to the text mining in general.

5.1.2 Preprocessing

After the clear allocation to calendar dates and text columns, some preprocessing is being done, which is standard for any text-mining related work. Most of the preprocessing is an attempt to reduce noise in the data, which later deteriorates the actual outcome from it. The R-package *tm* (Feinerer and Hornik, 2015) for basic text mining does a good job for these general modifications. As of now, the collection of text documents is referred to as *text corpus*.

First, stopwords are removed. Stopwords are non-indicative words such as *what*, *she* and *yes*. These words are unlikely to add semantic indication value to the data but are likely to add noise to the data. For this reason, stopwords are not desired and deleted. In order to define the words that are deleted, I use a list of stopwords from the *SMART Information Retrieval System* (Buckley, 1985). This list also includes negation words such as *not*, and *neither*. However, negation words add to a more pronounced polarization of a text, which is why I manually remove them from the original stopword list. Removing them from the stopword list means that they remain in the text (Appendix 1).

Furthermore, punctuation, numbers and extra whitespace are removed and all words are set to lowercase. Doing so avoids that an uppercase word is counted separately from a lowercase word while being the same.

Subsequently, Porter´s stemming algorithm (Porter, 1980) is applied to the data. Stemming reduces words to their word stem. It enables summarizing of two words that have the same meaning but different word application. For example, the words *cancellation* and *cancel* are very similar in terms of their linguistic meaning. The stemming algorithm reduces both words to *cancel*, enabling them to be counted as one word.

5.1.3 Negation Tagging and Part-of-Speech

It is generally possible to extend preprocessing by tagging, thereby categorizing certain words in various topics at this stage. However, doing so goes into the domain of lexicon based text mining. Therefore, only two, very straightforward tagging schemes are introduced here.

The first is to make more active use of negations and negative words. These words provide the option to tag other words as negated or negatively polarized words, based on their distance to a negation- or negative word in the text. In case *economy* appears one or two words after the word *bad,* it is likely that *economy* actually has a negative polarization. In order to acknowledge this polarization, its occurrence is measured separately from another, 'non-negative' *economy* in the text. Here, this is being done by tagging it with the tag *_neg* at the end of the word. Tagging negatively polarized words artificially augments the number of unique words in the text corpus. Adding negative tagging constitutes an alternative corpus and model, whose relative quality will be judged in the result section.

The second optional preprocessing stage is part-of-speech (POS) tagging. POS tagging classifies words into their grammatical meaning such as nouns, articles and verbs. POS-tagging, too could create a more nuanced text representation. It is possible to weigh term frequencies differently, depending on the terms´ POS-tag. The R-package *NLP* (Hornik, 2016) offers useful tools to do so, but it is computationally too slow to be implemented on the current corpus in this analysis.

Having mentioned these steps, it is important to note that the Base Model of this analysis does not include them. Only an alternative model is tested with negation tagging.

5.1.4 Word Count

The occurrence of a term in the text, called term frequency is the quantitative metric this analysis is based on. First, the number of *unique* terms per document are counted ($tf_{absolute}$) as well as the total number of occurrences of all terms in the clean document ($length_{doc}$). Term frequencies in the absolute form are biased for the length of the document they occur in. The more words a document contains overall, the higher is the probability for occurrence per word. To make term frequencies comparable across documents, absolute term frequencies must be converted into the ratio of term frequency over document length. Conversion is done by (1) on the document level.

$$\boldsymbol{tf} = \frac{\boldsymbol{tf_{absolute}}}{\boldsymbol{length_{doc}}} \times \mathbf{100} \qquad (1)$$

The next variables are on a global basis, the number of documents in the corpus ($\boldsymbol{n\_docs}$), the global term frequency ($\boldsymbol{tf\_glob}$) which is the overall occurrence of that term in the corpus and the global proportion of a word in the entire corpus ($\boldsymbol{tf\_prob_{glob}}$).

The latter ($\boldsymbol{tf\_prob_{glob}}$) is an important variable helping to identify whether a term is a term. The algorithm does not know how a term looks like. It just counts the number of unique combinations of letters. In most cases these are indeed words but in sometimes they can be random combinations of letters such as *stknfh*. Random noise words can occur when a word has not been read properly by OCR or for other reasons. Unlike actual words, random noise words are unlikely to occur the same way very often. The variable $\boldsymbol{tf\_prob_{glob}}$ gives reasonable information about the nature of a word because if its value is very low, the word rarely occurs in the overall corpus. The benchmark value for a minimum $\boldsymbol{tf\_prob_{glob}}$ required is set at 0.0025. Any words occurring in less than 0.25% of all documents will be left out. This value was chosen after realizing that the number of unique terms increases rapidly for lower thresholds.

5.1.5 Term Frequency Inverted Document Frequency

The plain term frequency ratio $\boldsymbol{tf}$ represents a biased picture about the importance of a word in the overall text. Take for example the word *York* (of The New York Times). Since it is part of the name of the text source, it is likely to occur in the header of every document. Its global (overall) term frequency is thus quite high, which would give *York* a high importance in the model. *York*, however has no added value in terms of semantics because it occurs too often: Newspaper readers pay no attention to the newspaper name printed on every page.

A word occurring in a large part of a document cannot serve as a semantic discriminator. It should be given a lower weight on its term frequency than one that occurs in less documents (Zhang et al., 2011). This results from the assumption that words generally have so-called eliteness, which represents the relevance of the term any given topics (Robertson, 2004). The word *York* does not add to any topic except the title of the newspaper, so it has low eliteness. Another example could be the word *economy* which does add to the topic of economy; quite relevant for financial markets? Not really, because the known content of the columns at hand is financial markets. The word *economy* is thus likely to occur more often in such content and

has low eliteness, too. Suppose the text data was about contemporary art. There, *economy* might have a higher eliteness because contemporary art journalists care less about the economy and use the word less frequently.

To correct for this phenomenon, words with a low eliteness are downgraded with a weighting scheme, which also upgrades words with a high eliteness on a relative basis. This weighting scheme is called *term frequency inverse document frequency* (*tf_idf*). It weights the term frequency based on the global occurrence of the word. The *tf_idf* weighting scheme is formally represented below.

$$\boldsymbol{tf\_idf} = \boldsymbol{tf} \times \mathbf{log}(\frac{\boldsymbol{n\_docs}}{\boldsymbol{tf\_glob}}). \quad (2)$$

In words, it means that the term frequency per document is multiplied with the logarithm of the ratio between the number of documents in the corpus and the global term frequency. $\boldsymbol{Tf\_idf}$ is a standard weighting scheme for both lexicon based and machine learning based text mining (Zhang et al., 2011).

5.1.6 Data Cast

As mentioned, the above steps including data cleaning, preprocessing, word count and the weighing scheme are just setting the stage for the actual core of the analysis. The following variables will enter the actual SLSI. *Date* matches a word occurrence with its document date point. *Term* are the words in that document and $\boldsymbol{tf\_idf}$ are the weighted term frequencies used as quantitative variable. From the long form data format the data will be casted into wide form, which is a more compact, less detailed, matrix format of data representation. This wide form is called *term-document matrix* ($\boldsymbol{TDM}$). $\boldsymbol{TDM}$ has terms as column names and documents as row names. The cells are filled with $\boldsymbol{tf\_idf}$. Any further steps are taken with $\boldsymbol{TDM}$.

5.2 Supervised Latent Semantic Indexing

The following section derives what was briefly introduced in the framework outline. The term *Supervised Latent Semantic Indexing (SLSI)* has not been established in the literature and is used here merely to summarize a concept in a name. Parts of it were theoretically examined in the literature review. The two approaches *Supervised Principal Component Analysis* (SPCA) and *Latent Semantic Analysis* (LSA) are combined based on their common root which is a *Singular Value Decomposition* (SVD). The document corpus is involved in the form $\boldsymbol{TDM}$ from preprocessing.

5.2.1 Supervision Strength

Supervision means the process of filtering words whose $\boldsymbol{tf\_idf}$ have no correlation with the independent variable index returns. It thereby channels words and excludes irrelevant or insignificant words in the LSA training stage. The process can be described as guiding or supervising the indexing algorithm, which would otherwise take every word into account.

For this purpose, linear univariate regression coefficients between the index return data and each term of $\boldsymbol{TDM}$ (each term´s weighted frequency in the corpus) are calculated. The linear univariate regression formula has the following form:

$$\boldsymbol{R_{DJIA} = \beta \times tf\_idf_{term} + \varepsilon} \qquad (3)$$

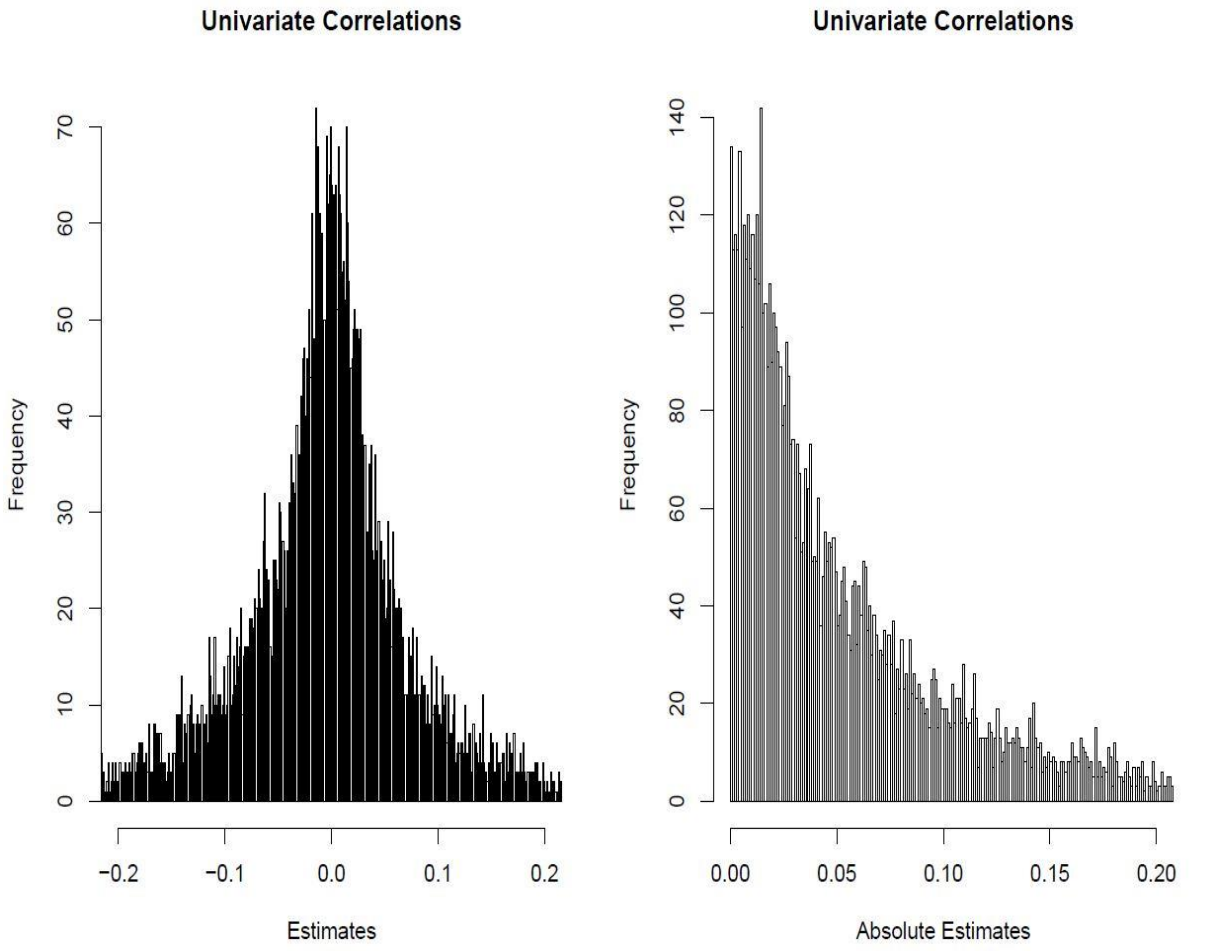
The coefficient $\boldsymbol{\beta}$ gives information if a term´s weighted term frequency has correlation with the index data. The coefficients are taken in its absolute form $|\boldsymbol{\beta}|$ because the direction of the correlation is irrelevant[1] at this point. Based on $|\boldsymbol{\beta}|$, it is determined whether that particular term in $\boldsymbol{TDM}$ stays, or is omitted. To determine what a minimum $|\boldsymbol{\beta}|$ must be, for the term not to be omitted, the threshold $\boldsymbol{\theta}$ is introduced. The distribution of $|\boldsymbol{\beta}|$over the word corpus can be examined as a density plot (Figure B).

A ten-fold cross validation is employed which means that the model is entirely calculated with ten different thresholds $\boldsymbol{\theta}$. Then, the model with the best prediction value is considered to

[1] Making the estimate direction relevant (i.e. acknowledging positive and negative values) is a possible alternative model as described in section 5.7

have the best balance between quantity and relevance of words. The ten values of $\boldsymbol{\theta}$ are a function of the distribution of $|\boldsymbol{\beta}|$, representing the deciles of the sample. Using deciles gradually and equally decreases the number of unique words with every higher $\boldsymbol{\theta}$ threshold. With the highest $\boldsymbol{\theta}$, the least words and with the lowest $\boldsymbol{\theta}$, all words enter the training stage LSI. This span of benchmarks is referred to as *supervision strength*. High $\boldsymbol{\theta}$ means high strength and low $\boldsymbol{\theta}$ means low strength. $\boldsymbol{TDM}$ under supervision turns into $\boldsymbol{TDM^{\theta}}$. The case of $\boldsymbol{\theta = 0}$ implies $\boldsymbol{TDM^{\theta} = TDM}$. The higher $\boldsymbol{\theta}$ gets, the fewer columns $\boldsymbol{TDM^{\theta}}$ has.

Figure B: Distribution of univariate linear coefficients of all unique terms in the corpus



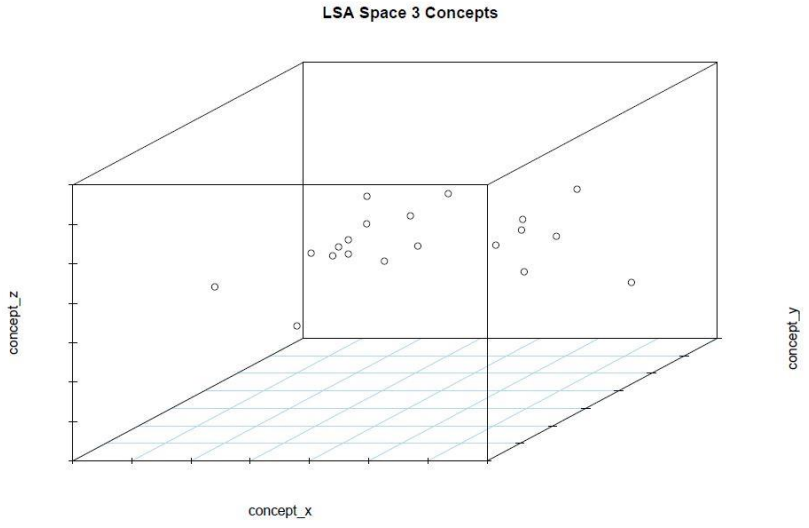
5.2.2 Latent Semantic Indexing

After supervised dimensionality reduction, a *Singular Value Decomposition* (SVD) is performed. SVD is a generalized matrix dimensionality reduction technique. *Latent Semantic Indexing* is the process of applying SVD specifically to a matrix of the form *term-document matrix* representing the text corpus, here on $\boldsymbol{TDM^{\theta}}$.

The text corpus is represented in the matrix $\boldsymbol{TDM^{\theta}}$ of dimensions $\boldsymbol{t} \times \boldsymbol{d}$, with $\boldsymbol{t}$ and $\boldsymbol{d}$ equal to the number of unique terms and documents, correspondingly. The number of word dimensions are equal the rank of $\boldsymbol{TDM^{\theta}}$. The rank is defined as the maximum number of columns or rows of which no nonzero combination equals the zero vector 0. These columns are considered independent. Since the values at hand are term frequencies it can be assumed that independence holds. As a result, the rank $\boldsymbol{r}$ of $\boldsymbol{TDM^{\theta}}$ equals the number of columns $\boldsymbol{t}$. In case the SVD was performed on a document-term matrix $\boldsymbol{DTM^{\theta}}$, or $(\boldsymbol{TDM^{\theta}})'$, r would equal $\boldsymbol{d}$, with the same implications.

In SVD, $\boldsymbol{TDM^{\theta}}$ is displayed in an $\boldsymbol{r} \ll \boldsymbol{t}$ dimensional coordinate system either as a document- or term space (hence two-mode factor analysis). The SVD rotates around the dimensions, so that the dimensions better fit the data and overall more variation is captured by them. This process is also known as clustering because groups of observations are clustered on new dimensions. The resulting new dimensions are referred to as c*oncepts* for the remainder of this text and the coordinate system as *concept space*.

Figure C: Three-dimensional representation of concept space with three concepts



The *concept space* is represented in the decomposition matrices $\boldsymbol{U^{\theta}}, \boldsymbol{\Sigma^{\theta}}$ and $\boldsymbol{V^{\theta}}$ and is an approximation of $\boldsymbol{TDM^{\theta}}$ of the following form:

$$\boldsymbol{TDM^{\theta}} = \boldsymbol{U^{\theta}} \times \boldsymbol{\Sigma^{\theta}} \times (\mathbf{V^{\theta}})' \quad (4)$$

$\boldsymbol{U^{\theta}}$ is a $\boldsymbol{t} \times \boldsymbol{r}$ *column-orthonormal* matrix and relates terms to concepts. It is the matrix representation of the term space: Cells contain coordinates per term in the document dimensions. $\boldsymbol{V^{\theta}}$ is a $\boldsymbol{d} \times \boldsymbol{r}$ *column-orthonormal* matrix. It relates documents to concepts. It is the matrix representation of the document space and the rows are the coordinates of each document in the document space with term dimensions. The matrix $\mathbf{V^{\theta}}$is always used transposed, hence $(\mathbf{V^{\theta}})'$. Finally, $\boldsymbol{\Sigma^{\theta}}$ is a $\boldsymbol{r} \times \boldsymbol{r}$ diagonal matrix with singular values on its diagonal and 0´s everywhere else. The singular values indicate the strength of the underlying concepts. Multiplication of the form (4) returns the original matrix $\boldsymbol{TDM^{\theta}}$, or its approximation.

An example for the document space is provided in Figure C where graphically, only three dimensions and 20 documents are displayed. In the document space, documents are displayed as coordinates in their concept dimensions. The axes of Figure C correspond to the concept dimensions. The dots, or coordinates represent 20 documents with loadings on all of the three displayed dimensions.

At this point, the number of concepts is still equal to the number of terms $\boldsymbol{t}$. However, the concepts have more diverged variation strengths, which is the whole point of rotating. Concepts with low strength, indicated by a low singular value can be dropped until the concept space equals $\boldsymbol{k}$ concepts.

This results into the term space $\boldsymbol{U_k^{\theta}}$ of size $\boldsymbol{t} \times \boldsymbol{k}$, the diagonal matrix $\boldsymbol{\Sigma_k^{\theta}}$ of size $\boldsymbol{k} \times \boldsymbol{k}$ and the document space $\mathbf{V_k^{\theta}}$ of size $\boldsymbol{d} \times \boldsymbol{k}$. The approximation of $\boldsymbol{TDM^{\theta}}, \boldsymbol{TDM_k^{\theta *}}$ is calculated with (4a).

$$\boldsymbol{TDM_k^{\theta *}} = \boldsymbol{U_k^{\theta}} \times \boldsymbol{\Sigma_k^{\theta}} \times (\mathbf{V_k^{\theta}})' \quad (4a)$$

As mentioned, setting low singular values to zero is the process of dimensionality reduction. Concepts with high singular values are retained, being important to the concept space. They are called concepts of high rank for the remainder of this text.

The R-package *lsa* (Wild, 2015) offers convenient functions to perform an SVD in the context of LSI. The benchmark for dimensionality reduction is set to k beforehand. A plot of the

diagonal vector of Σ_k^θ illustrates the distribution of concept strengths for the k retained concepts. The function takes the shape of a convex decreasing curve as marginal variation explained by an additional concept decreases (Figure D). Calculating an SVD for a matrix of several thousand rows and columns can challenge a computers capacity. That is because most packages calculate all concepts first and kick all but k out later. The R-package *irlba* (Baglama, Reichel, Lewis, 2016) calculates only the k concepts needed which dramatically decreases calculation time.

5.2.3 Number of Concepts

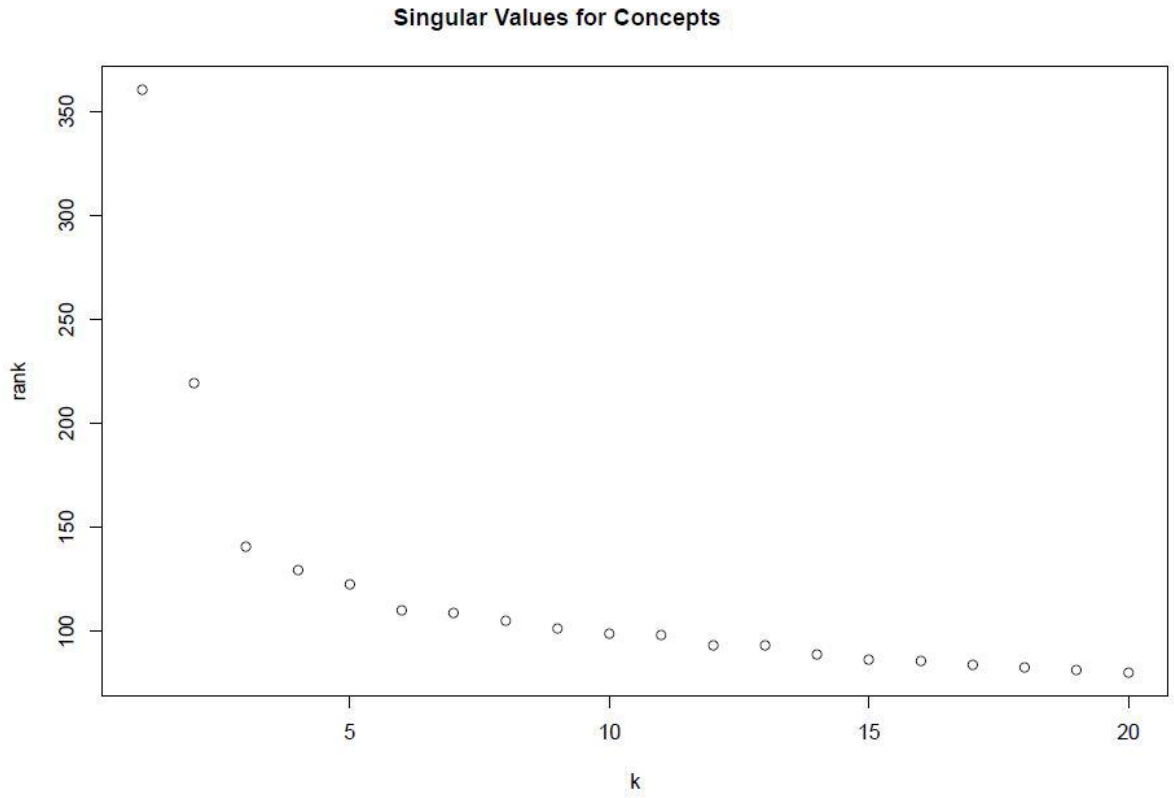
The parameter $\boldsymbol{k}$ is subject to different approaches in the literature. The purpose of LSI for any analysis must be acknowledged. Using all concepts ($\boldsymbol{k = t}$) would exactly represent the original matrix which is precise, but no dimensionality reduction would happen. Using fewer dimensions means less precision. In Derweester et al (1990) the purpose of LSI is information retrieval. The authors want a lower-dimension of text but still seek minimal loss of information. Then, precision is important while the number concepts can be a few hundreds. In Bair et al. (2006) the authors seek only the most important of underlying dimensions which means that precision becomes less important. In fact, they will not make an approximation of the data and thus use only the first concept with the highest singular value. Clearly, some balance between dimensionality reduction and precision must be found. The first objective in this analysis is generalizable prediction, which calls for fewer concepts. On the other hand, a thorough exploration of latent semantics requires more concepts.

The parameter $\boldsymbol{k}$ could, just like $\boldsymbol{\theta}$ be determined by cross-validation. Another 10-fold cross-validation for $\boldsymbol{k}$ would however result in a 100-fold cross-validation combined with the one for $\boldsymbol{\theta}$. This requires large calculation time with little added value since $\boldsymbol{k}$ can also be evaluated with the plot of singular values. The first few singular values in Figure D are clearly the strongest and after the tenth singular value the curve becomes much flatter. For this reason, $\boldsymbol{k}$ is set to 10. Note, that in the result section of this analysis, concepts will be addressed as their rank in the singular value ranking. For example, the concept with the second highest singular value is called “rank 2 concept” for better identification.

As introduced in the outline, the data of interest is the loadings of documents on the concepts, or in other words the matrix $\mathbf{V}_{10}^{\boldsymbol{\theta}}$. The matrix $\mathbf{V}_{10}^{\boldsymbol{\theta}}$ is finally the matrix of independent variables, entering the linear time series regression.

Of similar importance are the implications of $\boldsymbol{U}_{10}^{\boldsymbol{\theta}}$. Being the other mode of this two-mode factor analysis, $\boldsymbol{U}_{10}^{\boldsymbol{\theta}}$ is shows the strength of terms on the concepts. There is no quantitative use for $\boldsymbol{U}_{10}^{\boldsymbol{\theta}}$ in time series modelling but it is essential to make literal sense of the concepts. This can be done by ranking the terms based in their loading on each concept separately and represent them in a word cloud as presented in the result section.

Figure D: Ranked Singular Values in a k=20 Concept Space



5.2.4 Training and Test Data

Automatic indexing such as LSI can provide more useful results if the calculated concepts are generalizable. For the daily news media, this means recognition of consistent latent structures over time. Without generalizable concepts LSI would merely be a temporary snapshot of random data patterns. Generalization, however, is only possible if sufficient training data is available for machine learning. Here, 35492 documents and roughly 350 million word

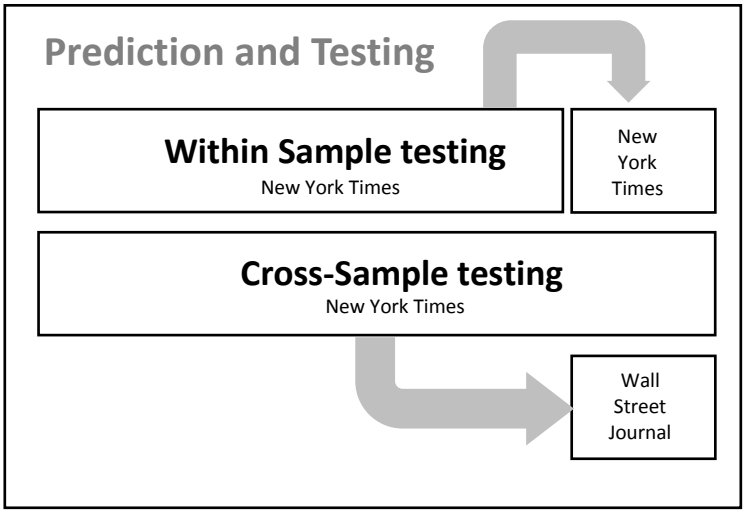
occurrences are available for training. This amount is just enough given the heterogeneity of the data. The English language, news reporting and the economy constantly changed over the century. Therefore, any training data available should be used.

Furthermore, model predictions must be tested on a test data set. For this analysis, two test approaches are considered. The first is out-of-sample testing with the same sample as used for training. Thereby, the dataset is split and the first part is used as training set and the second as test set. Same sample testing implies less data available for training and ignoring things that happen in the test set. This also means ignoring important time effects. For example, if the model is trained before 1980 and tested after, anything after 1980 is ignored, quite a loss of information. Of course, the model cannot be trained and tested on the same data because it would not truthfully represent the model´s *predictive* power. The advantage of within-sample testing is that the training and test set come from the same sample distribution.

The second approach is cross-sample testing. It allows training on the entire dataset. The testing happens on another dataset which is assumed to have a similar sample distribution as the training sample. This could be another financial newspaper than the New York Times but of similar impact and domain which is The Wall Street Journal. As explained in the data section, characteristics of these two datasets are quite alike because The Wall Street Journal provides similar daily columns for the same topic. The drawback is that training and testing does not happen on the same sample. After all, the major difference between the two newspapers is, that there are two different authors writing the text who choose different wording. However, this is the case also within one newspaper sample. Over the course of one century many different authors wrote the NYT columns.

A graphical illustration of testing schemes can be found in Figure E. The base model of this analysis will use an out-of-sample testing on the same sample. Proportion of the training set is approximately 80%, whereas the test set proportion is approximately 20% of the entire sample. Cross-sample testing will be implemented in an alternative model, where the proportions between training and test set are similar, however the sample size is bigger. It implies that the entire NYT sample is used for training and the WSJ sample is used for testing. The WSJ sample is approximately 20% of the size of the NYT sample.

Figure E: Testing Schemes

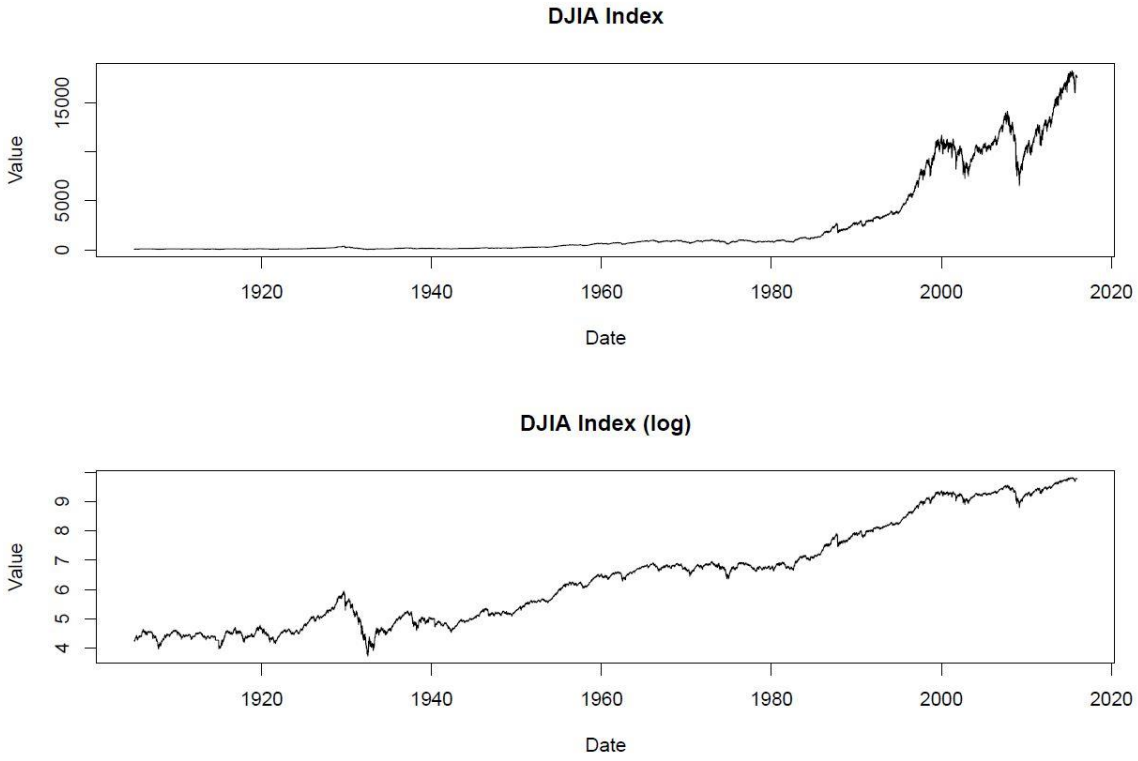


5.3 Market Index Returns

Now that the independent variables, the matrix $\mathbf{V}_{\mathbf{10}}^{\boldsymbol{\theta}}$ is derived, the dependent variable of the time series model must be examined. The supervised nature of this analysis requires a model that includes the dependent variable right before the calculation of the concepts. In the restricted model, only text frequencies that correlate with the dependent variable are considered in LSI. This dependent variable is market index returns from the Dow Jones Industrial Average. The following steps are standard for time series modelling, it is the basic treatment for any financial return data[2].

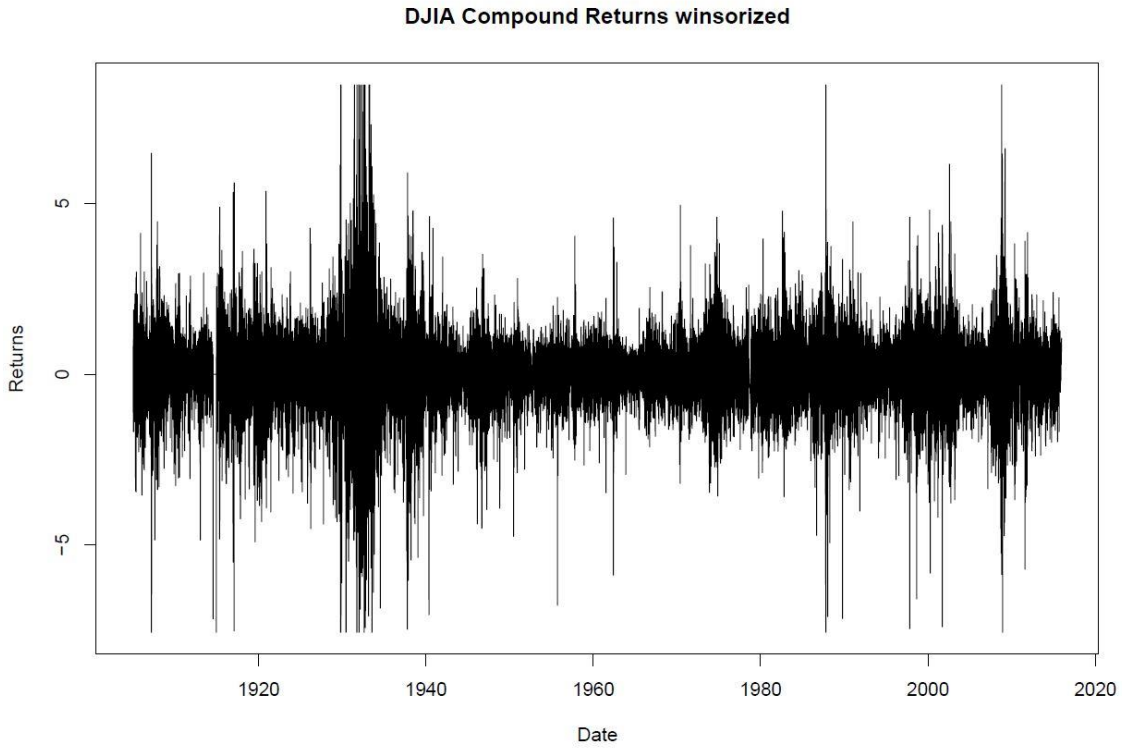
[2] To be precise, the supervision part of SLSI must be done with the corrected return variable which comes out of the steps below. And so, programmatically, the below steps happen before supervision and thus before any text processing.

Figure F: Dow Jones Industrial Average absolute values



$$\boldsymbol{R_{DJIA}^{comp} = \Delta \log(DJIA) \times 100} \tag{5}$$

Figure G: Dow Jones Industrial Average Compound Returns winsorized



5.3.1 Compound Returns and Outliers

The given time series of the index absolute value and the log index absolute value are presented in Figure F). Compound returns of the form (5) are then calculated and used for further time series modelling (Figure G).

The market returns clearly show outliers in the data. Outliers in compound returns are extreme returns on some trading days which cannot be explained. To tackle them, a winsorizing algorithm is applied. Winsorizing returns is a common approach to deal with outliers especially in financial return data. The algorithm considers any value beyond a percentile limit in the sample distribution as outlier and resets it to the percentile value. In return data, extreme observations are not uncommon and neither can they be considered measurement errors. They are rather the result of external shocks, whose probability to happen is low. These tail events must be considered by the model as such, because they could indeed be echoed in the text data. By allocating percentile limits, outliers are incorporated with a lower weight, preventing them to create a skewness bias and maintaining a normal return distribution which is required for linear regression analysis (Barber & Lyon, 1997). In the case of DJIA compound returns, a rather low fraction of the distribution is being winsorized, namely 0.05% on both tails of the return distribution.

5.3.2 Data Merging

Not all 35 thousand text documents fall exactly on one perfectly matching trading day. Some of the columns are printed on weekends or other non-trading days. Therefore, the datasets must be matched, which is done by allocating the *Date* variables in the document identifiers and the index data. The *Date*s are merged in a rolling fashion. If there is, for example a Sunday publishing of a newspaper column, the index is not traded that day. The rolling merging function allocates a *Date* point from the document identifier to the next available trading day from the index returns because this is also the trading day on which the newspaper content actually impacts the market. This implies that Monday observations occur three times because they account for the weekend days, too. This way valuable information from weekends, flowing into trading day activity is captured.

5.4 Time Series model

Compound returns come with two econometric conveniences: Stationarity and non-exponentiality. Stationarity[3] is achieved by differencing and is an important assumption for any time series modelling. It is formally tested below. The plain absolute index values (Figure F) show a pattern of exponentiality[4]. Linearity (that is non-exponentiality) is, also an assumption for linear regression analysis and so the data is made linear by taking the log.

Still, compound returns are formally suspect to non-stationarity, autocorrelation[5] and heteroskedasticity[6] which is tested in the next paragraphs. Any tests below were executed separately for the Base Model and alternative models if they imply different sample sizes (see section 5.2.4). The statistical test outcomes were equal throughout. Autoregressive models, however, have slightly different outcomes for the alternative model, whose metrics are reported in the footnotes.

5.4.1 Augmented Dickey-Fuller Test

In some cases, variables can still be non-stationary after taking the first difference. Therefore, stationarity must be formally tested with the Augmented Dickey-Fuller (ADF) stationarity test, performed on the compound index returns (Dickey & Fuller, 1979). The ADF test formally tests the null hypothesis of the presence of a unit root, implying non-stationarity. Non-stationarity tested in an autoregressive model of the form (6).

$$\boldsymbol{R_t = \rho R_{t-1} + u_t} \qquad (6)$$

The null hypothesis translates into $\boldsymbol{\rho = 1}$, an indication for the presence of a unit root. If the null hypothesis is rejected, it means that there is no unit root present in the sample and the variable is stationary. Here, the null hypothesis can be rejected with a p-value of nearly zero[7], thus confirming stationarity for the compound returns.

[3] Appendix 2
[4] Appendix 2
[5] Appendix 2
[6] Appendix 2
[7] Appendix 3

5.4.2 Autoregressive Integrated Moving Average (ARIMA)

To investigate autocorrelation, the autocorrelation function (ACF) and the partial autocorrelation function (PACF) of the compound returns are presented[8]. The ACF suggests an AR(p) process: An autocorrelation process with lag p because of several significant lags. The PACF also shows significant lags, suggesting a MA(q) process: A moving average process with lag q. Together, and formally also including non-stationarity this can be corrected with an ARIMA (p, d, q) model, a catchall autoregressive model for time series, capturing any components of first or second order autocorrelation.

An automated ARIMA model provided by the R-package *forecast (Hyndman, 2016)* calculates the best model by iteratively fitting lags (AR), integrations (I) and moving averages (MA) to the return data. The ARIMA model should also be represented with as few parameters as possible, which is why the model minimizes the Akaike Information Criterion (AIC), indicating the best tradeoff between goodness-of-fit and complexity of the model.

The result is an ARIMA(5, 0, 3), or ARMA(5, 3) model[9]. It implies an AR(5) process, corrected with 5 return lags. The I(0) process is expected: No additional differencing has to be done as formally described with the ADF test. The MA(3) implies a 3-lags return moving average process.

5.4.3 Autoregressive Conditional Heteroskedasticity (ARCH)

Another issue is heteroskedasticity or non-constant volatility in the compound index returns. This can be observed through clustered volatility in Figure E, also known as the ARCH effect. The ACF and PACF of the squared residuals of the ARMA(5, 3) [10]hint towards the presence of the ARCH effect if they display significant lags.

To check for the ARCH effect, the Ljung-Box test (Ljung & Box, 1978) is performed[11] (R-package *tseries,* Trapletti & Hornik, 2016*).* The LB test formally checks for the null hypothesis of independence of the residuals from the earlier estimated ARMA (5, 3) model. The null

[8] Appendix 3
[9] The outcome for the alternative model is an ARMA(5, 2), see Appendix 4
[10] Appendix 4
[11] Appendix 3

hypothesis translates into *no ARCH effect* and would make fitting an GARCH model obsolete if not rejected. Here, the LB test shows a p-value close to zero, with which the null hypothesis must be rejected. The outcome confirms the suspicion about non-constant return volatility as formulated above, which requires the fitting of an GARCH(1, 1) model of the general form below (7).

$$\boldsymbol{\sigma_t^2 = \omega + \alpha \varepsilon_{t-1}^2 + \beta \sigma_{t-1}^2} \quad (7)$$

In words this means that the variance of $\boldsymbol{R_t}$, $\boldsymbol{\sigma_t^2}$ is a function of the lag of the squared residuals from the above estimated ARMA(5, 3), $\boldsymbol{\varepsilon_{t-1}^2}$ and the lag of the variance of $\boldsymbol{R_t}$, $\boldsymbol{\sigma_{t-1}^2}$. Specifics such as imposed standard deviation, the empirical density of standardized residuals and the norm _QQ plot as well as formal output can be found in Appendix 5.

5.4.4 Time Series Equation

Summarizing the ARMA(5, 3) and GARCH(1, 1) autoregressive terms in the vector $\mathbf{A_t}$, it is possible to set up preliminary regression formulae of autoregressive terms and the intercept (8).

$$\mathbf{A}_t = \boldsymbol{\mu + \phi_1 R_{t-1} + \phi_2 R_{t-2} + \phi_3 R_{t-3} + \phi_4 R_{t-4} + \phi_5 R_{t-5} + \gamma_1 \varphi_{t-1} + \gamma_2 \varphi_{t-2} + \gamma_3 \varphi_{t-3} + \theta_1 \varepsilon_{t-1} + \delta \varepsilon_{t-1}^2} \quad (8)$$

Note, that ϕ corresponds to the AR process, γ to the MA process, and θ and δ to the GARCH process. Latent concepts from SLSI are stored in $\mathrm{V}_{10t}^{\prime\theta}$ (9) which leads to the final model (10).

$$\mathbf{V}^{\prime\boldsymbol{\theta}}_{\mathbf{10}t} = \boldsymbol{\beta_1 v_{1t}^{\prime\theta} + \beta_2 v_{2t}^{\prime\theta} + \beta_3 v_{3t}^{\prime\theta} + \beta_4 v_{4t}^{\prime\theta} + \beta_5 v_{5t}^{\prime\theta} + \beta_6 v_{6t}^{\prime\theta} + \beta_7 v_{7t}^{\prime\theta} + \beta_8 v_{8t}^{\prime\theta} + \beta_9 v_{9t}^{\prime\theta} + \beta_{10} v_{10t}^{\prime\theta}} \quad (9)$$

$$\boldsymbol{R_t = \beta V_{10t}^{\prime\theta} + \gamma A_t + \epsilon_t} \quad (10)$$

The independent variable R_t is the compound index return on day t. The linear coefficient β indicates the relations between the daily compound returns R and a vector of document concept loadings $\mathrm{V}_{10}^{\prime\theta}$ on day t, calculated under supervision benchmark θ. The vector A_t consists of autoregressive terms of R_t.

Equation (10) formally translated into a GARCHX model, which is a GARCH(1, 1) based on ARMA(5, 3) residuals augmented with external regressors. In many cases, however such a GARCHX model does not converge which results in invalid outcome, especially invalid p-values

for coefficients. For this reason, the GARHX model is not used for results, but is merely reported in the appendix. The GARCH(1, 1) without external regressors is still used as prediction benchmark.

5.4.5 Prediction

To evaluate the model quality, out of sample testing is performed. Thereby the model is predicted with out-of-sample values of all independent variables. An out-of-sample $\mathrm{V}_{10}^{\prime\theta}$ matrix is a concept representation of documents, that were not used during the indexing process. These documents are *folded* into the document space. They do not influence the calculation of the dimensions, but are merely mapped into the existing concept space.

It is done formally by multiplying the vector representation q of each document with the low dimensional representation of the term space U_{10}^{θ}. The equation of the form (11) results in the supervised low dimensional representation of q^{θ} in the document space.

$$\mathbf{V\_fold}_{\mathbf{10}}^{\boldsymbol{t\theta}} = \boldsymbol{q}^{\boldsymbol{\theta}} \times \boldsymbol{U}_{\mathbf{10}}^{\boldsymbol{\theta}} \tag{11}$$

Supervision still holds because for new documents only filtered terms enter q^{θ}.

With these independent variables, $\hat{R}_t$ is calculated. Any deviation is measured with the Mean Absolute Error (MAE) displayed in (12).

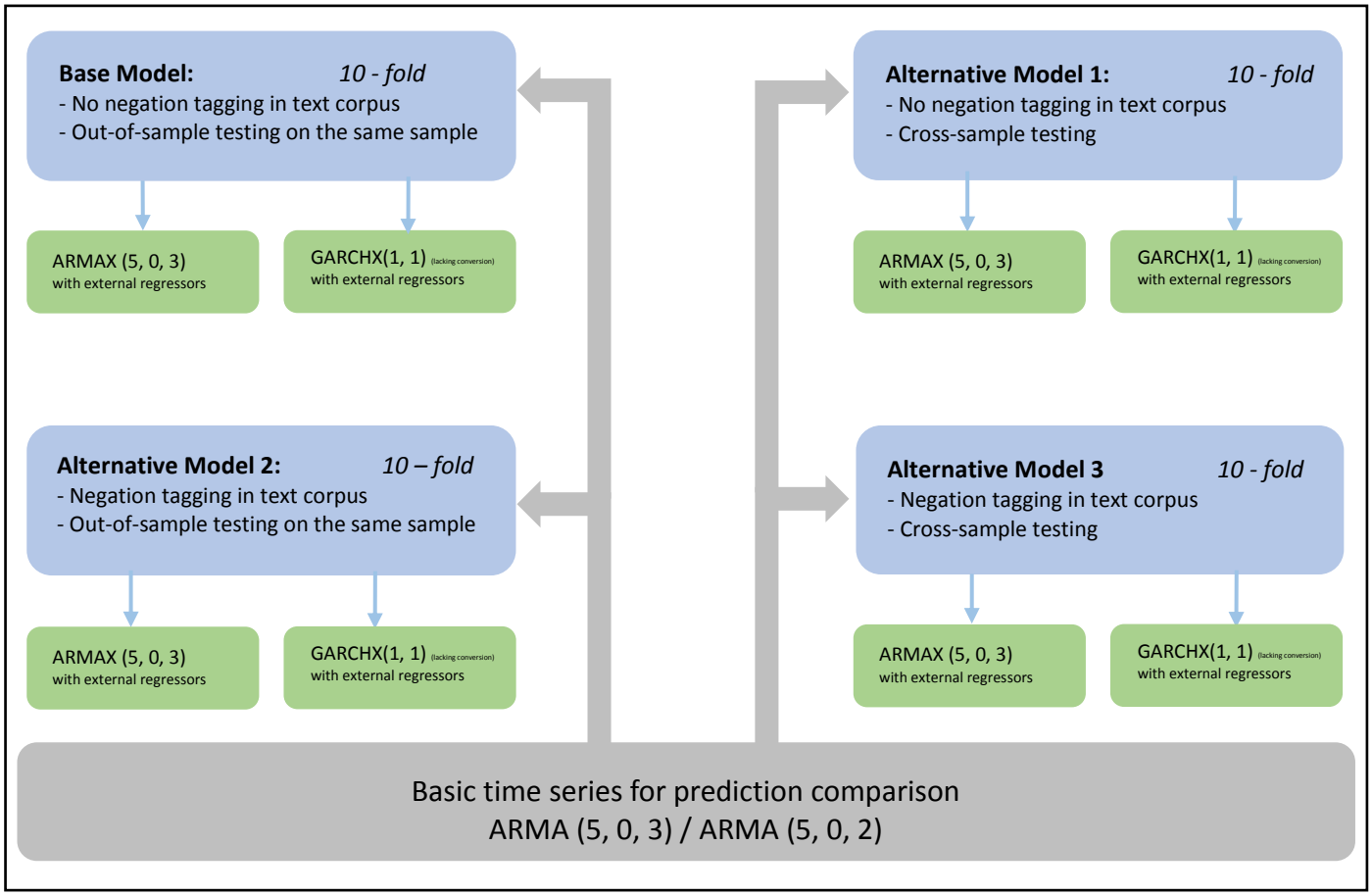
$$\boldsymbol{MAE} = \frac{\mathbf{1}}{\boldsymbol{n}} \sum_{\boldsymbol{t}=\mathbf{1}}^{\boldsymbol{n}} \left| \widehat{\boldsymbol{R}}_{\boldsymbol{t}} - \boldsymbol{R}_{\boldsymbol{t}} \right| \tag{12}$$

These prediction errors are aimed to be minimized and should be lower than a comparing autoregressive model which does not involve external variables. Any model with external variables should perform better than a plain autoregressive model if the external variables are assumed to serve as predictors. With these indicators, a truthful statement about model quality is presented in the following result section.

6 Results

Apart from the base model, three alternative models were calculated, which is graphically presented in Figure H.

Figure H: Model Specifics



Alternative Model 2 and 3 differ from the *Base Model* and *Alternative Model 1* in their allowance for *negations tagging* (neg tagging) in the text corpus. As described in section 5.1.4, words can be tagged as negatively nuanced or not. Where *negations tagging* is switched on, more unique words in the corpus can be expected, compared to the model where it is switched off. This results in two different corpora with which the entire methodology is calculated.

Base Model and *Alternative Model 2* differ from *Alternative Model 1 and 3* in their testing schemes *out-of-sample testing* and *cross sample testing*. They are distinct through the size of their training sets and the fact that in *cross sample testing*, the generalizability of concepts is tested for robustness by letting them predict on a new sample.

Any calculated models model the error terms from GARCH(1, 1) and ARMA(5, 3) (ARMA(5, 2)), also called ARMAX, and GARCHX models. However, GARCHX tends to not converge into meaningful outcomes, which is a technical problem. For this reason, the prediction errors are reported for ARMAX only. All models were calculated 10-fold, because of the cross-validation scheme of supervision strength. For comparison, ARMA(5, 3) GARCH(1, 1) were calculated without external regressors.

Table A: Coefficients for the Base Model, lowest MAE out of Supervision Stage

MAE: 0.7015		**GARCH MAE: 0.6998**			Supervision Stage: 4					
ARMA Terms	AR1	AR2	AR3	AR4	AR5	MA1	MA2	MA3		
Coefficients	0.05	0.93***	-0.18***	-0.09***	0.09***	0.28***	-0.78***	-0.15**		
Variables	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Coefficients	1.07	3.53	0.12	5.2***	0.01	-2.35	-1.52	-0.68	-2.82**	-0.53

*Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively*

6.1 Correlation and Prediction (H1, H2)

To present prediction performance, the lowest MAE out of the 10 folds of the Base Model ARMAX is taken. It equals 0.7015 as compared to an MAE of 0.6998 for the GARCH model. This implies, that a plain autoregressive model predicts index returns marginally better, although the difference is very small. Table A presents the coefficients for this sub model. Equivalent tables for all alternative models can be found in Appendix 6.

In the Base Model and all alternative models, the lowest prediction errors coincide with at least one significant concept expressed in its document concept loadings. These variables are marked in Table A.

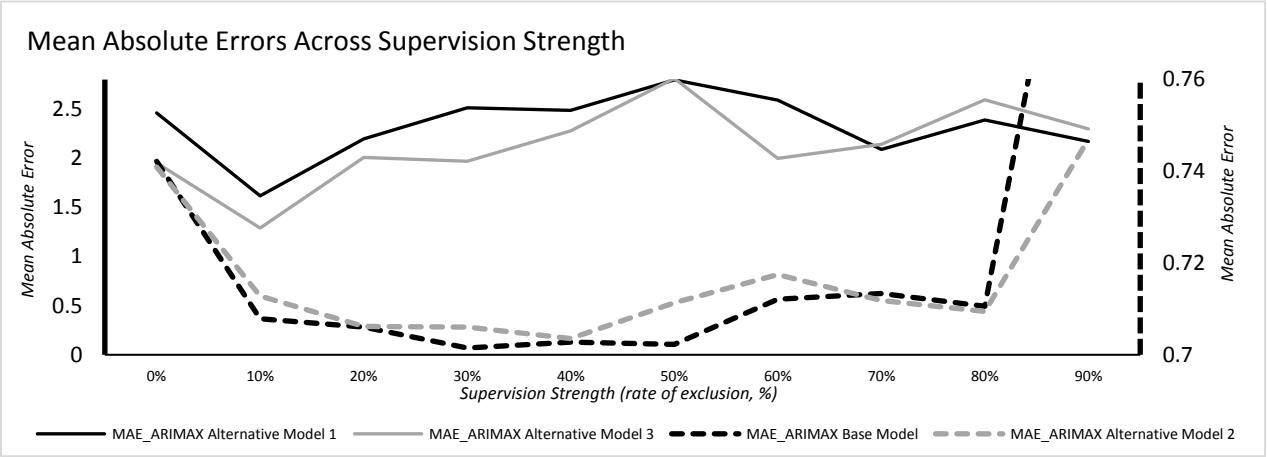
Research hypothesis H1 tests whether there is a *detectable correlation between document concept loadings from news text media and market index returns.* With significant concepts in the Base Model, *H1 is not rejected*. Nevertheless, it is important to note that neither the Base Model nor one of the alternative models produce an MAE lower than its corresponding ARMA benchmark. This means that a plain autoregressive model is always predicting market returns better than any sub model. Research hypothesis H2 tests whether *document concept loadings from news text media can serve as predictors for market index returns.* Following this, *H2 must*

be rejected. Latent semantic concepts as detected with this methodology do show correlation with index returns, but they do not serve as predictors.

6.2 Supervision Strength (H3, H4)

Research hypothesis H3 tests whether *supervision improves model prediction when tested on the same text sample source.* To evaluate the outcomes, MAE's across cross-validation stages and for all alternative models are presented in Figure K.

Figure K: Mean Absolute Errors



Note that supervision strength is measured in the deciles of the distribution of the univariate coefficients. The higher the threshold gets, the bigger share of terms is excluded from the model, indicated by the percentages. This leads to gradually excluding parts of the sample, i.e. the terms. Excluding 0% of the sample equivalents to *no supervision*, which is the starting point for evaluating supervision.

The development of MAE´s in the Base Model is initially decreasing, indicating enhanced prediction compared to no supervision. Towards strong supervision and 80% exclusion of the term sample, the effect reverses and prediction gets worse. With this outcome, *H3 is not rejected*, as channeled exclusion of terms from processing does enhance prediction over cross-validation. This effect, however cannot be confirmed for Alternative Model 1 and 3. These models are tested on different samples as they were trained. Compared to no supervision, MAE´s tend to increase towards higher supervision which is counterintuitive. Research hypothesis H4 tests whether *supervision improves model prediction when tested on a different*

text sample source. This outcome *rejects H4* because benefits of supervision on a cross-sample testing scheme cannot be shown.

6.3 Negations Tagging

The neg-model and the non-neg-model behave very similarly within their testing schemes. The lowest prediction errors occur in the same supervision stages and so are significant concept coefficients. The effect of tagging negative pronounced words on prediction errors is therefore neglectable. Nevertheless, tagging does not result in deterioration of concept correlations. Despite an increased number of unique terms through tagging, Latent Semantic Indexing results in the same outcomes as for no negations tagging.

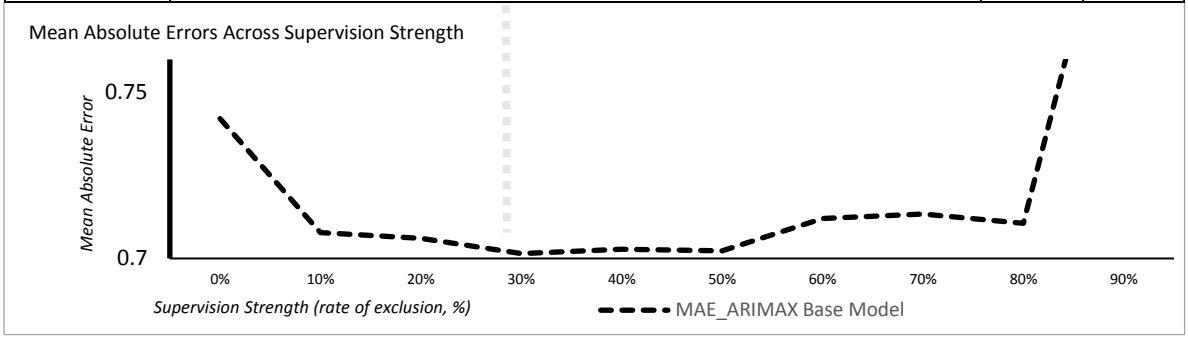
6.4 Concepts Correlation and Topics

The Base Model and all alternative models show at least one significant linear coefficient between document concept loadings and index returns. The coefficients and underlying concepts are subject to closer examination in the following paragraphs.

As depicted in Figure K, the prediction errors for the Base Model are similar from the second to the sixth supervision stage with the fourth having the smallest MAE. In order to evaluate consistencies of significant concepts, Table B presents the distribution of significant concepts over supervision in the base model.

Table B: Distribution of Significant Concept Coefficients (Base Model)

Supervision	SV 1	SV 2	SV 3	SV 4	SV 5	SV 6	SV 7	SV 8	SV 9	SV 10
Concept 1	**						**	***	*	***
Concept 2	*				**	**				***
Concept 3		***	**			**	***		***	***
Concept 4	***		*	***	***	***	**	***	**	**
Concept 5								***		**
Concept 6	***						**	*	***	
Concept 7	***		***				**		***	
Concept 8	***	*					**	**		
Concept 9				**	**			**		
Concept 10	***	**	**							***



Coinciding with low prediction errors is the continuous significance of the rank 4 concept (marked in Table B, combined with parts of Figure K). Rank 4 concept means that its singular value is the fourth highest among the other concepts.

Intuitively, it could be concluded that these rank 4 concepts represent the same information over the course of supervision as they have the same rank and show a pattern of consistency in significance. However, across supervision stages, terms drop out of the overall corpus. This is likely to change concepts even of the same rank. In addition, the direction of the coefficients of same ranked concepts change from one supervision stage to another (see extensive tables in Appendix 7). This points toward the fact that there is different information in the same ranked concepts from one supervision stage to another.

Therefore, it is necessary to examine the information that these concepts contain, thereby examining the concept space from a different angle. Up until here, only document loadings have been considered. By ranking the term loadings in a concept, it is possible to see what terms have high loadings on concepts. The best way to do so are word clouds. In the following paragraph, four concepts from the Base Model are chosen for closer examination.

6.5 Word Clouds

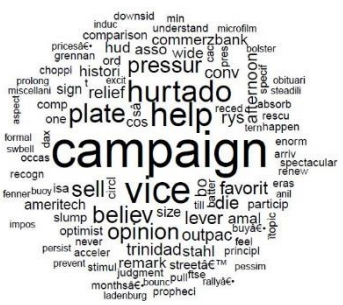
The first two concepts chosen for word cloud representation are both rank 4 concepts but at supervision stages 4 and 8. They are tagged in Table B. Both have significant coefficients and the MAE´s of their supervision stage are among the lowest. However, supervision in stage 4 is much lower than in stage 8. Their word clouds are presented in Figure L.

Figure L: Word Clouds of the rank 4 concepts from the fourth and eighth supervision stage (Base Model)

The first concept contains words like *economy*, *company*, *merger* and *policy* which are all from the topic real economy. The second concept contains the words *speculation, dealer, and shipment* which is a much more specific and operational topic. Most important is, that these two concepts contain entirely different words although having the same rank. They clearly should be treated as distinct informational variables.

The third concept presented is a rank 10 concept from the ninth supervision stage with an insignificant coefficient. The concept is presented to illustrate information has no correlation with index returns (Figure M). The concept contains terms related to politics with words like *campaign, vice, opinion* and *favorite*.

Figure M: Word Cloud of the rank 10 concept from the ninth supervision stage (Base Model)



The last presented concept is a rank 1 concept from the first supervision stage. It is provided to examine an unsupervised concept. The first supervision stage corresponds to no supervision at all (Figure N). This concept, being significant at the 5% level, shows a clear focus on finance with words like *bond, million, bank* and *investor*. The rank 1 concepts generally have the highest term loadings of all, given their highest singular value. It is not surprising, that this concept contains finance words which are the most used in a financial markets column.

Figure N: Word Cloud of the rank 1 concept from the first supervision stage (Base Model)

6.6 Result Summary

Out of the four research hypotheses, two important ones had to be rejected. The question if news text data consists of detectable latent semantic structures or concepts can be answered with yes, however the precision of these concepts for prediction of index returns is in doubt. Clearly, document concept loadings cannot serve as predictors for index returns at this stage. A similar answer can be given on the question if supervision enhances model outcomes. It does, indicated by lower prediction errors, but it does not lower prediction errors to levels below those of plain autoregressive models. Neither does supervision work at this stage on cross-sample testing schemes. *As a result, H2 and H4 must be rejected, whereas H1 and H3 are not rejected.*

6.7 Limitations

Several limitations to this analysis must be acknowledged.

A possible alternation concerns supervision. When the thresholds are set based on the deciles of the $|\beta|$ distribution over words, their absolute value is taken. Allowing for negative coefficients and splitting words in sub models accordingly, could disentangle the two directions of the market and enable better prediction as well as better word clouds.

In terms of model parameters, the number of concepts, here set to k=10 could also be estimated by cross-validation. This requires more calculation time but promises better variable selection.

In terms of data, the observation can be aggregated to more than one day. This is a more complex approach because supervision and the time series would be computed in an aggregated fashion, but LSI should remain in a daily fashion in order to maintain high frequented data for machine learning. Additionally, time effects could be examined by leads and lags of document loadings. It is also possible to perform sub-sample time series, to capture time effects over the course of the century.

Finally, the Dow Jones Industrial Average is not the most meaningful market index. The *S&P500,* momentum and fear indices, sector specific indices or particular stocks have the potential of representing markets much better. These are indices that allow for trading strategies. The problem is, that supervision cannot be executed the way it is here, because no other index is available this long. A solution to this comes from data. Modern news is available not daily, but secondly. Nowadays, a much richer text corpus can be gathered quickly with intraday news data from a variety of sources. All indices have intraday reporting, too. Consequently, this analysis could be performed on a more specific domain more precisely, with the help of modern data density.

7 Conclusion

Supervised Latent Semantic Indexing combines into a novel approach to text mining and forecasting. The results presented in the preceding paragraphs paint a mixed picture.

The Base Model and alternative models show significant coefficients between concepts and index returns. Nevertheless, a plain autoregressive model can predict market returns better than the concept based time series model. The ability to predict market returns from text data could, at least at this stage not be proven.

Supervision does enhance model prediction when the model is tested out-of-sample with the same text dataset. However, when tested on a different text sample, supervision increases prediction errors. Supervision proves to be an asset to the Base Model, improving both concept significance and word clouds. A small step towards hybrid text mining by negative words tagging has shown no improvements, yet not a deterioration of the outcomes.

Word cloud analysis yields interesting insights to the concept content and unveils actual topics out of the statistically retrieved concepts. Furthermore, a clear topic distinction across examined concepts could be observed. This outcome qualitatively backs the hypothesis, that Supervised Latent Semantic Indexing is suitable to uncover distinct latent text structures.

The implementation of provided extensions might reach the goal of prediction at some point. Until then, it is unclear, how much sentiment in the market is intrinsically predictable through news text.

8 References

Bair, E. T. (2006). Prediction by supervised principal components. *Journal of the American Statistical Association 101(473)*, pp. 119-137.

Barber, B. M. (1997). Detecting long-run abnormal stock returns: The empirical power and specification of test statistics. *Journal of financial economics 43, no. 3*, pp. 341-372.

Buckley, C. (1985). Implementation of the SMART Information Retrieval System. *Computer Science Technical Reports*.

Deerwester, S. (n.d.). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science 1990*.

Dickey, D. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association 74, no. 366a*, pp. 427-431.

Factiva. (2017). *Factiva - Premium business news and content*. Retrieved from https://www.dowjones.com/products/factiva

García, D. (2012). Sentiment during recesions. *Journal of Finance*.

Gino, F. A. (2009). How anxiety increases advice-taking (even when the advice is bad). *working paper, Wharton School of the University of Pennsylvania*.

Global, S. (2017). *S&P Global* . Retrieved from https://www.djindexes.com/averages/

Global, S. (n.d.). *Dow Jones Index* . Retrieved from https://www.djindexes.com/

J. Bradford de Long, A. S. (1990, June). Positive Feedback Investment Strategies and Destabilizing Rational Speculation. *The Journal of Finance Vol. 45, No. 2*, pp. 379-395.

Levenshtein, 1. (1965). Binary codes with correction for deletions and insertions of the symbol 1. *Problemy Peredaci Informacii*, p. 12.

McDonald, l. a. (2011). When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. *Journal of Finance*, pp. 35-65.

McKay, A. (Director). (2015). *The Big Short* [Motion Picture].

Minqing Hu, B. L. (2004). Mining and Summarizing Customer Reviews. *KDD '04 Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 168-177.

Nicholas Barberis, A. S. (1998, September). A model of investor sentiment. *Journal of Financial Economics Volume 49, Issue 3*, pp. 307–343.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program 14.3*, pp. 130-137.

Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation 60, no. 5* , pp. 503-520.

Stone , P., Dunphy, D., Smith, M., & Ogilvie, D. (1966). The General Inquirer: A Computer Approach to Content Analysis. *MIT Press, Cambridge, MA*.

Taboada, M., & et al. (2011). Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*.

Tetlock, P. (2007). Giving Content to Investor Sentiment: The Role of Media in the Stock Market. *The Journal of Finance Vol. 62, No. 3*, pp. 1139-1168.

Wang, K. Y. (1997). A new look at the Monday effect. *The Journal of Finance 52, no. 5* , pp. 2171-2186.

Wurgler, J., & Baker , M. (2007). Investor Sentiment in the Stock Market. *Journal of Economic Perspectives*, pp. 129 - 151.

Zhang, W. T. (2011). A comparative study of TF* IDF, LSI and multi-words for text classification. *Expert Systems with Applications 38, no. 3*, pp. 2758-2765.

9 Software References

R Core Team (2016). R: A language and environment for statistical computing.
R Foundation for
Statistical Computing, Vienna, Austria. URL
https://www.R-project.org/.

Adrian Trapletti and Kurt Hornik (2016). tseries: Time Series Analysis and Computational Finance.
R package version 0.10-35.

Hyndman RJ (2016). _forecast: Forecasting functions for time series and linear models_.
R package version7.3,
<URL: http://github.com/robjhyndman/forecast>.

Spencer Graves (2014). FinTS: Companion to Tsay (2005) Analysis of Financial Time Series.
R package version 0.4-5.
https://CRAN.R-project.org/package=FinTS

Alexios Ghalanos (2015). rugarch: Univariate GARCH models.
R package version 1.3-6.

Andreas Alfons (2016). robustHD: Robust Methods for High-Dimensional Data.
R package version 0.5.1.
https://CRAN.R-project.org/package=robustHD

Achim Zeileis, Torsten Hothorn (2002). Diagnostic Checking in Regression Relationships.
R News 2(3),
7-10. URL http://CRAN.R-project.org/doc/Rnews/

Thomas Lumley using Fortran code by Alan Miller (2009). leaps: regression subset selection.
R package version 2.9.
https://CRAN.R-project.org/package=leaps

Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition.
Springer, New York. ISBN 0-387-95457-0

Jerome Friedman, Trevor Hastie, Robert Tibshirani (2010).
Regularization Paths for Generalized Linear Models via Coordinate Descent.
Journal of Statistical Software, 33(1), 1-22.
URL http://www.jstatsoft.org/v33/i01/.

M Dowle, A Srinivasan, T Short, S Lianoglou with contributions from R Saporta and E Antonyan (2015).
data.table: Extension of Data.frame.
R package version 1.9.6.
https://CRAN.R-project.org/package=data.table

Ingo Feinerer and Kurt Hornik (2015). tm: Text Mining Package.
R package version 0.6-2.
https://CRAN.R-project.org/package=tm

Ingo Feinerer, Kurt Hornik, and David Meyer (2008).
Text Mining Infrastructure in R.
Journal of Statistical Software 25(5): 1-54.
URL: http://www.jstatsoft.org/v25/i05/.

Milan Bouchet-Valat (2014). SnowballC: Snowball stemmers based on the C libstemmer UTF-8 library.
R package version 0.5.1.
https://CRAN.R-project.org/package=SnowballC

Fridolin Wild (2015). lsa: Latent Semantic Analysis.
R package version 0.73.1.
https://CRAN.R-project.org/package=lsa

Gagolewski M., Tartanus B. (2016). R package stringi: Character string processing facilities.
http://www.gagolewski.com/software/stringi/.
DOI:10.5281/zenodo.32557

Yixuan Qiu, Jiali Mei and authors of the ARPACK library. See file AUTHORS for details. (2016).
rARPACK: Solvers for Large Scale Eigenvalue and SVD Problems.
R package version 0.11-0.
https://CRAN.R-project.org/package=rARPACK

Eric Bair and R. Tibshirani (2012). superpc: Supervised principal components.
R package version 1.09.
https://CRAN.R-project.org/package=superpc

Duncan Temple Lang and the CRAN team (2016). RCurl: General Network (HTTP/FTP/...) Client Interface for
R. R package version 1.95-4.8.
https://CRAN.R-project.org/package=RCurl

Jim Lemon and Philippe Grosjean (2015). prettyR: Pretty Descriptive Stats.
R package version 2.2.
https://CRAN.R-project.org/package=prettyR

H. Wickham. Reshaping data with the reshape package.
Journal of Statistical Software, 21(12), 2007.

Hadley Wickham (2007). Reshaping Data with the reshape Package.
Journal of Statistical Software, 21(12), 1-20.
URL http://www.jstatsoft.org/v21/i12/.

Hadley Wickham (2016). stringr: Simple, Consistent Wrappers for Common String Operations.
R package version 1.1.0.
https://CRAN.R-project.org/package=stringr

Kurt Hornik (2016). NLP: Natural Language Processing Infrastructure.
R package version 0.1-9.
https://CRAN.R-project.org/package=NLP

Simon Urbanek (2016). rJava: Low-Level R to Java Interface.
R package version 0.9-8.
https://CRAN.R-project.org/package=rJava

van der Loo M (2014). "The stringdist package for approximate string matching.
The R Journal_, *6*, pp. 111-122.
URL: https://CRAN.R-project.org/package=stringdist.

Garrett Grolemund, Hadley Wickham (2011). Dates and Times Made Easy with lubridate.
Journal of Statistical Software, 40(3), 1-25.
URL http://www.jstatsoft.org/v40/i03/.

Hadley Wickham (2016). dtplyr: Data Table Back-End for 'dplyr'.
R package version 0.0.1.
https://CRAN.R-project.org/package=dtplyr

Rinker, T. W. (2013). qdap: Quantitative Discourse Analysis Package. 2.2.5.
University at Buffalo. Buffalo, New York.
http://github.com/trinker/qdap

Jim Baglama, Lothar Reichel and B. W. Lewis (2016).
irlba: Fast Truncated SVD, PCA and Symmetric Eigendecomposition for Large Dense and Sparse Matrices.
R package version 2.1.2.
https://CRAN.R-project.org/package=irlba

Rahul Premraj (2015). mailR: A Utility to Send Emails from R.
R package version 0.4.1.
https://CRAN.R-project.org/package=mailR

10 Appendix

Appendix 1: stopwords (Buckley, 1985) amended

```
> stopwords_en_pos
  [1] "a"          "about"      "above"      "across"     "after"      "again"
  [7] "against"    "all"        "almost"     "alone"      "along"      "already"
 [13] "also"       "although"   "always"     "among"      "an"         "and"
 [19] "another"    "any"        "anybody"    "anyone"     "anything"   "anywhere"
 [25] "are"        "area"       "areas"      "around"     "as"         "ask"
 [31] "asked"      "asking"     "asks"       "at"         "away"       "b"
 [37] "back"       "backed"     "backing"    "backs"      "be"         "became"
 [43] "because"    "become"     "becomes"    "been"       "before"     "began"
 [49] "behind"     "being"      "beings"     "best"       "better"     "between"
 [55] "big"        "both"       "but"        "by"         "c"          "came"
 [61] "can"        "cannot"     "case"       "cases"      "certain"    "certainly"
 [67] "clear"      "clearly"    "come"       "could"      "d"          "did"
 [73] "differ"     "different"  "differently" "do"         "does"       "done"
 [79] "down"       "downed"     "downing"    "downs"      "during"     "e"
 [85] "each"       "early"      "either"     "end"        "ended"      "ending"
 [91] "ends"       "enough"     "even"       "evenly"     "ever"       "every"
 [97] "everybody"  "everyone"   "everything" "everywhere" "f"          "face"
[103] "faces"      "fact"       "facts"      "far"        "felt"       "few"
[109] "find"       "finds"      "first"      "for"        "four"       "from"
[115] "full"       "fully"      "further"    "furthered"  "furthering" "furthers"
[121] "g"          "gave"       "general"    "generally"  "get"        "gets"
[127] "give"       "given"      "gives"      "go"         "going"      "good"
[133] "goods"      "got"        "great"      "greater"    "greatest"   "group"
[139] "grouped"    "grouping"   "groups"     "h"          "had"        "has"
[145] "have"       "having"     "he"         "her"        "here"       "herself"
[151] "high"       "higher"     "highest"    "him"        "himself"    "his"
[157] "how"        "however"    "i"          "if"         "important"  "in"
[163] "interest"   "interested" "interesting" "interests"  "into"       "is"
[169] "it"         "its"        "itself"     "j"          "just"       "k"
[175] "keep"       "keeps"      "kind"       "knew"       "know"       "known"
[181] "knows"      "l"          "large"      "largely"    "last"       "later"
[187] "latest"     "least"      "less"       "let"        "lets"       "like"
[193] "likely"     "long"       "longer"     "longest"    "m"          "made"
[199] "make"       "making"     "man"        "many"       "may"        "me"
[205] "member"     "members"    "men"        "might"      "more"       "most"
[211] "mostly"     "mr"         "mrs"        "much"       "must"       "my"
[217] "myself"     "n"          "necessary"  "need"       "needed"     "needing"
[223] "needs"      "new"        "newer"      "newest"     "next"       "non"
[229] "noone"      "nothing"    "now"        "nowhere"    "number"     "numbers"
[235] "o"          "of"         "off"        "often"      "old"        "older"
[241] "oldest"     "on"         "once"       "one"        "only"       "open"
[247] "opened"     "opening"    "opens"      "or"         "order"      "ordered"
[253] "ordering"   "orders"     "other"      "others"     "our"        "out"
[259] "over"       "p"          "part"       "parted"     "parting"    "parts"
[265] "per"        "perhaps"    "place"      "places"     "point"      "pointed"
[271] "pointing"   "points"     "possible"   "present"    "presented"  "presenting"
[277] "presents"   "put"        "puts"       "q"          "quite"      "r"
[283] "rather"     "really"     "right"      "right"      "room"       "rooms"
[289] "s"          "said"       "same"       "saw"        "say"        "says"
[295] "second"     "seconds"    "see"        "seem"       "seemed"     "seeming"
[301] "seems"      "sees"       "several"    "shall"      "she"        "should"
[307] "show"       "showed"     "showing"    "shows"      "side"       "sides"
[313] "since"      "small"      "smaller"    "smallest"   "so"         "some"
[319] "somebody"   "someone"    "something"  "somewhere"  "state"      "states"
[325] "still"      "such"       "sure"       "t"          "take"       "taken"
[331] "than"       "that"       "the"        "their"      "them"       "then"
[337] "there"      "therefore"  "these"      "they"       "thing"      "things"
[343] "think"      "thinks"     "this"       "those"      "though"     "thought"
[349] "thoughts"   "three"      "through"    "thus"       "to"         "today"
[355] "together"   "too"        "took"       "toward"     "turn"       "turned"
[361] "turning"    "turns"      "two"        "u"          "under"      "until"
[367] "up"         "upon"       "us"         "use"        "used"       "uses"
[373] "v"          "very"       "w"          "want"       "wanted"     "wanting"
[379] "wants"      "was"        "way"        "ways"       "we"         "well"
[385] "wells"      "went"       "were"       "what"       "when"       "where"
[391] "whether"    "which"      "while"      "who"        "whole"      "whose"
[397] "why"        "will"       "with"       "within"     "without"    "work"
[403] "worked"     "working"    "works"      "would"      "x"          "y"
[409] "year"       "years"      "yet"        "you"        "young"      "younger"
[415] "youngest"   "your"       "yours"      "z"
```

Appendix 2 (Didactical Appendix)

Stationarity: In case of stationarity, the probability distribution of a time series variable is constant over time. If it were not, it would be non-stationary, which means that the variable follows a unit root. Including non-stationary variables in a regression model can result in a spurious regression, detecting correlation between two variables that are not causally related.

Exponentiality in market data: Exponentiality yields from the nature of stock market prices. A market index is based on economic growth, which shows by nature an exponential growth pattern. Even if the economy would grow linearly (for example in a socialist economy) then the underlying population still grows exponentially. For this reason, any market data can be expected to show exponentiality.

Autocorrelation: For the example of index returns, autocorrelation means that high returns are followed by high returns and low returns are followed by low returns, therefore ultimately putting the independence of observations in question.

Non-constant volatility: For the example of index returns, heteroskedasticity means that high volatility is followed by high volatility and low volatility is followed by low volatility. More generally, the volatility in the data is not constant; it shows heteroskedasticity. This would violate the assumption that the residuals of a model are i.i.d (independently and identically distributed).

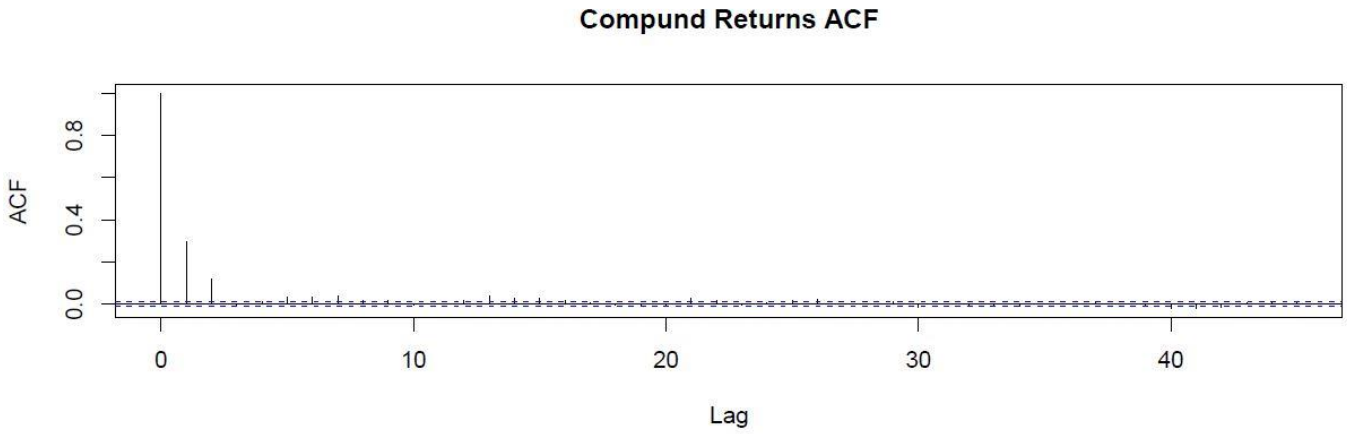
Abreviations:

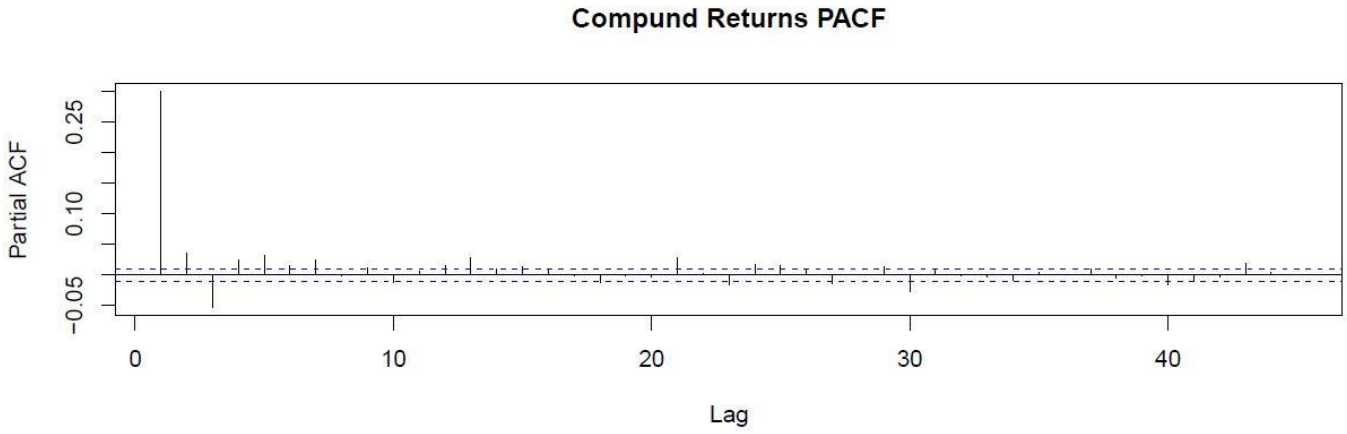
Wall Street Journal (WSJ)
New York times (NYT)
Dow Jones Industrial Average (DJIA)
Semantic Orientation Calculator (SO-CAL)
Performance-Based Arbitrage (PBA)
The General Inquirer (GI)
Principal Component Analysis (PCA)
Supervised Principal Component Analysis (SPCA)
Latent Semantic Indexing (LSI)
Supervised Latent Semantic Indexing (SLSI)
Singular Value Decomposition (SVD)
Optical character recognition (OCR)
Within sample testing (ws)
Cross sample testing (ws)
Term frequency (tf)
Term frequency – inverse document frequency (tf_idf)
Term-document matrix (tdm)
Augmented Dickey-Fuller (ADF)
Autocorrelation function (ACF)
Partial autocorrelation function (PACF)
Akaike Information Criterion (AIC)
Autoregressive integrated moving average (ARIMA)
Autoregressive Conditional Heteroskedasticity (ARCH)
The Ljung-Box test (LB)
Independently and identically distributed (i.i.d.)
Part-of-speech (POS)
Mean Square Error (RMSE)
Mean Average Error (MAE)

Appendix 3, 4, 5: Test Outcomes and Autoregressive Models

Auto ARIMA, GARCH Same-Sample testing

```
> print(adf.test(ind_nyt[ , cprt_win]))

Augmented Dickey-Fuller Test

data:  ind_nyt[, cprt_win]
Dickey-Fuller = -30.374, Lag order = 32, p-value = 0.01
alternative hypothesis: stationary
```






```
>ts_II = auto.arima(returns,
                    d = 0, D = 0, stationary = TRUE, seasonal = TRUE,
                    trace = TRUE, ic = "aic")

Iterations:

ARIMA(2,0,2) with non-zero mean : 84844.19
ARIMA(0,0,0) with non-zero mean : 88593.82
ARIMA(1,0,0) with non-zero mean : 85027.9
ARIMA(0,0,1) with non-zero mean : 85647.07
ARIMA(0,0,0) with zero mean     : 88594.04
ARIMA(1,0,2) with non-zero mean : 84841.32
ARIMA(1,0,1) with non-zero mean : 85005.78
ARIMA(1,0,3) with non-zero mean : 84843.43
ARIMA(2,0,3) with non-zero mean : 84790.81
ARIMA(2,0,3) with zero mean     : 84789.55
ARIMA(1,0,3) with zero mean     : 84842.55
```

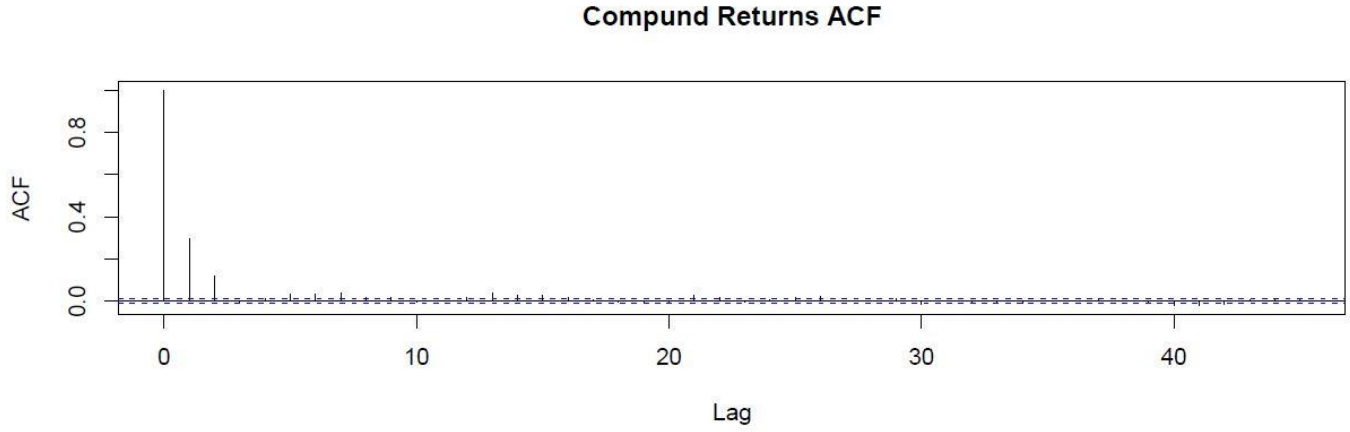
```
ARIMA(3,0,3) with zero mean     : 84778.59
ARIMA(3,0,2) with zero mean     : 84842.77
ARIMA(3,0,4) with zero mean     : 84787.17
ARIMA(2,0,2) with zero mean     : 84843.32
ARIMA(4,0,4) with zero mean     : 84780.41
ARIMA(3,0,3) with non-zero mean : 84779.71
ARIMA(4,0,3) with zero mean     : 84773.56
ARIMA(4,0,2) with zero mean     : 84777.6
ARIMA(5,0,4) with zero mean     : 84754.23
ARIMA(5,0,4) with non-zero mean : 84755.23
ARIMA(5,0,3) with zero mean     : 84742.79
ARIMA(5,0,3) with non-zero mean : 84744.06
ARIMA(5,0,2) with zero mean     : 84744.9

 Best model: ARIMA(5,0,3) with zero mean

> ts_II
Series: returns
ARIMA(5,0,3) with zero mean

Coefficients:
         ar1     ar2      ar3      ar4     ar5     ma1      ma2      ma3
      0.0662  0.9263  -0.1945  -0.0833  0.0908  0.2687  -0.7831  -0.1320
s.e.  0.0589  0.0441   0.0652   0.0230  0.0067  0.0591   0.0448   0.0657

sigma^2 estimated as 1.158:  log likelihood=-42360.68
AIC=84739.36   AICc=84739.36   BIC=84813.64

> ts_IV

*---------------------------------*
*          GARCH Model Fit        *
*---------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(5,0,3)
Distribution      : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.023060    0.006658    3.46335 0.000533
ar1    -0.038615    0.047408   -0.81452 0.415346
ar2     0.790830    0.002879  274.67379 0.000000
ar3    -0.029083    0.138675   -0.20972 0.833888
ar4    -0.093607    0.043559   -2.14900 0.031635
ar5     0.057740    0.009771    5.90918 0.000000
ma1     0.380971    0.047197    8.07195 0.000000
ma2    -0.634322    0.002066 -307.04946 0.000000
ma3    -0.234411    0.137615   -1.70338 0.088496
omega   0.002850    0.000001 2020.72902 0.000000
alpha1  0.073849    0.000854   86.47812 0.000000
beta1   0.925151    0.000494 1872.50058 0.000000

Robust Standard Errors:
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.023060    0.008909    2.58833 0.009644
ar1    -0.038615    0.036099   -1.06968 0.284763
ar2     0.790830    0.004438  178.19521 0.000000
ar3    -0.029083    0.264344   -0.11002 0.912395
ar4    -0.093607    0.083077   -1.12676 0.259845
ar5     0.057740    0.017487    3.30190 0.000960
ma1     0.380971    0.035651   10.68617 0.000000
ma2    -0.634322    0.004892 -129.65223 0.000000
ma3    -0.234411    0.263132   -0.89085 0.373011
omega   0.002850    0.000001 5008.39676 0.000000
```

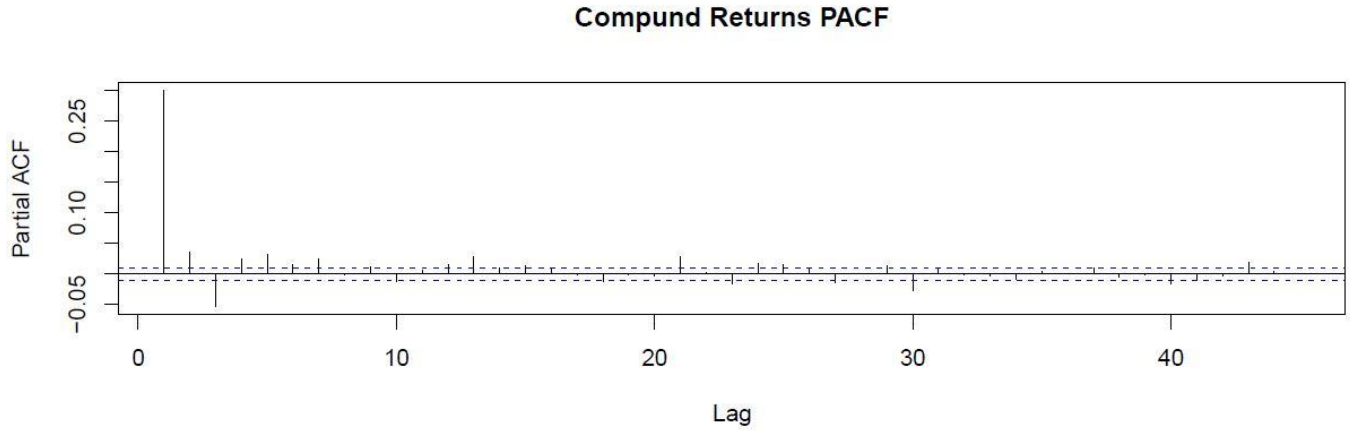
```
alpha1  0.073849    0.001014   72.83879 0.000000
beta1   0.925151    0.000365 2532.30272 0.000000

LogLikelihood : -35338.87

Information Criteria
------------------------------------

Akaike       2.4904
Bayes        2.4939
Shibata      2.4904
Hannan-Quinn 2.4915

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                         statistic   p-value
Lag[1]                       18.16 2.034e-05
Lag[2*(p+q)+(p+q)-1][23]     54.32 0.000e+00
Lag[4*(p+q)+(p+q)-1][39]     64.21 0.000e+00
d.o.f=8
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      1.830  0.1762
Lag[2*(p+q)+(p+q)-1][5]     2.250  0.5606
Lag[4*(p+q)+(p+q)-1][9]     2.447  0.8455
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]   0.01941 0.500 2.000  0.8892
ARCH Lag[5]   0.18162 1.440 1.667  0.9698
ARCH Lag[7]   0.25017 2.315 1.543  0.9951

Nyblom stability test
------------------------------------
Joint Statistic:  24.2393
Individual Statistics:
mu     0.1385
ar1    0.5284
ar2    2.8621
ar3    0.1544
ar4    0.1522
ar5    0.1596
ma1    0.7567
ma2    3.3906
ma3    0.2251
omega  1.8229
alpha1 3.3922
beta1  1.1609

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:         2.69 2.96 3.51
Individual Statistic:    0.35 0.47 0.75

Sign Bias Test
------------------------------------
                   t-value    prob sig
Sign Bias           1.1518 0.24941
Negative Sign Bias  1.5065 0.13195
Positive Sign Bias  0.5343 0.59316
Joint Effect        7.0149 0.07143   *


Adjusted Pearson Goodness-of-Fit Test:
```

```
-------------------------------------
  group statistic p-value(g-1)
1    20     494.0    9.955e-93
2    30     576.1   5.718e-103
3    40     584.0    2.441e-98
4    50     655.6   4.823e-107


Elapsed time : 15.7289
```

Auto ARIMA, GARCH Cross-Sample testing

```
> print(adf.test(ind_nyt[ , cprt_win]))

Augmented Dickey-Fuller Test

data:  ind_nyt[, cprt_win]
Dickey-Fuller = -30.374, Lag order = 32, p-value = 0.01
alternative hypothesis: stationary
```





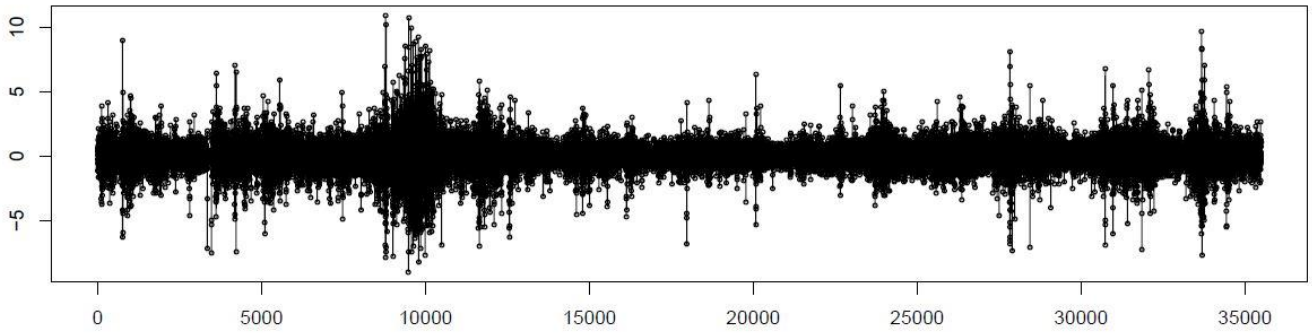

```
>ts_I = auto.arima(returns,
                   d = 0, D = 0, stationary = TRUE, seasonal = TRUE,
                   trace = TRUE, ic = "aic")

Iterations:

 ARIMA(2,0,2) with non-zero mean : 105885.8
 ARIMA(0,0,0) with non-zero mean : 109375.7
 ARIMA(1,0,0) with non-zero mean : 106044.7
 ARIMA(0,0,1) with non-zero mean : 106544.4
```

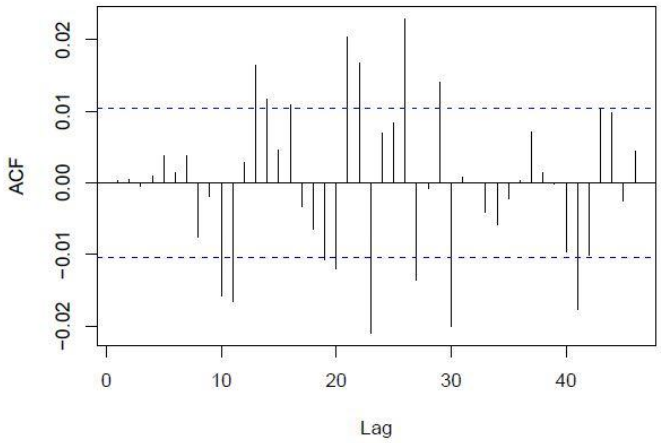
```
 ARIMA(0,0,0) with zero mean     : 109373.8
 ARIMA(1,0,2) with non-zero mean : 105883.9
 ARIMA(1,0,1) with non-zero mean : 106014
 ARIMA(1,0,3) with non-zero mean : 105885.7
 ARIMA(2,0,3) with non-zero mean : 105858.2
 ARIMA(2,0,3) with zero mean     : 105856.5
 ARIMA(1,0,3) with zero mean     : 105883.7
 ARIMA(3,0,3) with zero mean     : 105825.2
 ARIMA(3,0,2) with zero mean     : 105882.9
 ARIMA(3,0,4) with zero mean     : 105833.3
 ARIMA(2,0,2) with zero mean     : 105883.9
 ARIMA(4,0,4) with zero mean     : 105825.9
 ARIMA(3,0,3) with non-zero mean : 105828.6
 ARIMA(4,0,3) with zero mean     : 105820.5
 ARIMA(4,0,2) with zero mean     : 105823.2
 ARIMA(5,0,4) with zero mean     : 105794.5
 ARIMA(5,0,4) with non-zero mean : 105796.6
 ARIMA(5,0,3) with zero mean     : 105794.3
 ARIMA(5,0,3) with non-zero mean : 105796.3
 ARIMA(5,0,2) with zero mean     : 105792.5
 ARIMA(4,0,1) with zero mean     : 105821
 ARIMA(5,0,2) with non-zero mean : 105794.5
 ARIMA(5,0,1) with zero mean     : 105823.9

 Best model: ARIMA(5,0,2) with zero mean

> ts_IV

*----------------------------------*
*          GARCH Model Fit         *
*----------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model : sGARCH(1,1)
Mean Model  : ARFIMA(5,0,3)
Distribution      : norm

Optimal Parameters
------------------------------------
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.023060    0.006658    3.46335 0.000533
ar1    -0.038615    0.047408   -0.81452 0.415346
ar2     0.790830    0.002879  274.67379 0.000000
ar3    -0.029083    0.138675   -0.20972 0.833888
ar4    -0.093607    0.043559   -2.14900 0.031635
ar5     0.057740    0.009771    5.90918 0.000000
ma1     0.380971    0.047197    8.07195 0.000000
ma2    -0.634322    0.002066 -307.04946 0.000000
ma3    -0.234411    0.137615   -1.70338 0.088496
omega   0.002850    0.000001 2020.72902 0.000000
alpha1  0.073849    0.000854   86.47812 0.000000
beta1   0.925151    0.000494 1872.50058 0.000000

Robust Standard Errors:
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.023060    0.008909    2.58833 0.009644
ar1    -0.038615    0.036099   -1.06968 0.284763
ar2     0.790830    0.004438  178.19521 0.000000
ar3    -0.029083    0.264344   -0.11002 0.912395
ar4    -0.093607    0.083077   -1.12676 0.259845
ar5     0.057740    0.017487    3.30190 0.000960
ma1     0.380971    0.035651   10.68617 0.000000
ma2    -0.634322    0.004892 -129.65223 0.000000
ma3    -0.234411    0.263132   -0.89085 0.373011
omega   0.002850    0.000001 5008.39676 0.000000
alpha1  0.073849    0.001014   72.83879 0.000000
beta1   0.925151    0.000365 2532.30272 0.000000
```

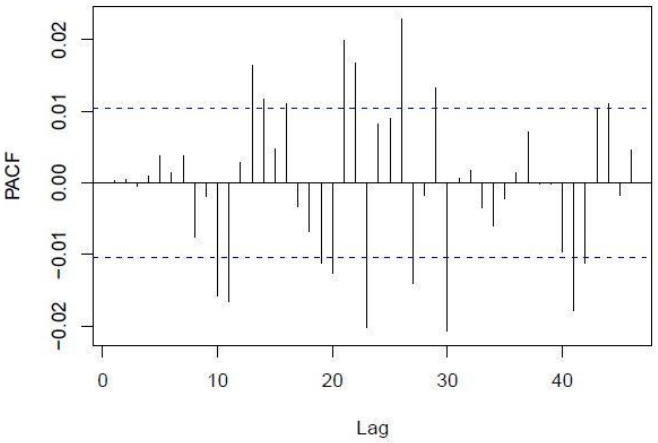
```
LogLikelihood : -35338.87

Information Criteria
------------------------------------

Akaike       2.4904
Bayes        2.4939
Shibata      2.4904
Hannan-Quinn 2.4915

Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic   p-value
Lag[1]                      18.16 2.034e-05
Lag[2*(p+q)+(p+q)-1][23]    54.32 0.000e+00
Lag[4*(p+q)+(p+q)-1][39]    64.21 0.000e+00
d.o.f=8
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      1.830  0.1762
Lag[2*(p+q)+(p+q)-1][5]     2.250  0.5606
Lag[4*(p+q)+(p+q)-1][9]     2.447  0.8455
d.o.f=2

Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]   0.01941 0.500 2.000  0.8892
ARCH Lag[5]   0.18162 1.440 1.667  0.9698
ARCH Lag[7]   0.25017 2.315 1.543  0.9951

Nyblom stability test
------------------------------------
Joint Statistic:  24.2393
Individual Statistics:
mu     0.1385
ar1    0.5284
ar2    2.8621
ar3    0.1544
ar4    0.1522
ar5    0.1596
ma1    0.7567
ma2    3.3906
ma3    0.2251
omega  1.8229
alpha1 3.3922
beta1  1.1609

Asymptotic Critical Values (10% 5% 1%)
Joint Statistic:         2.69 2.96 3.51
Individual Statistic:    0.35 0.47 0.75

Sign Bias Test
------------------------------------
                   t-value    prob sig
Sign Bias           1.1518 0.24941
Negative Sign Bias  1.5065 0.13195
Positive Sign Bias  0.5343 0.59316
Joint Effect        7.0149 0.07143   *


Adjusted Pearson Goodness-of-Fit Test:
------------------------------------
  group statistic p-value(g-1)
```

```
1    20     494.0    9.955e-93
2    30     576.1   5.718e-103
3    40     584.0    2.441e-98
4    50     655.6   4.823e-107


Elapsed time : 15.7289
```

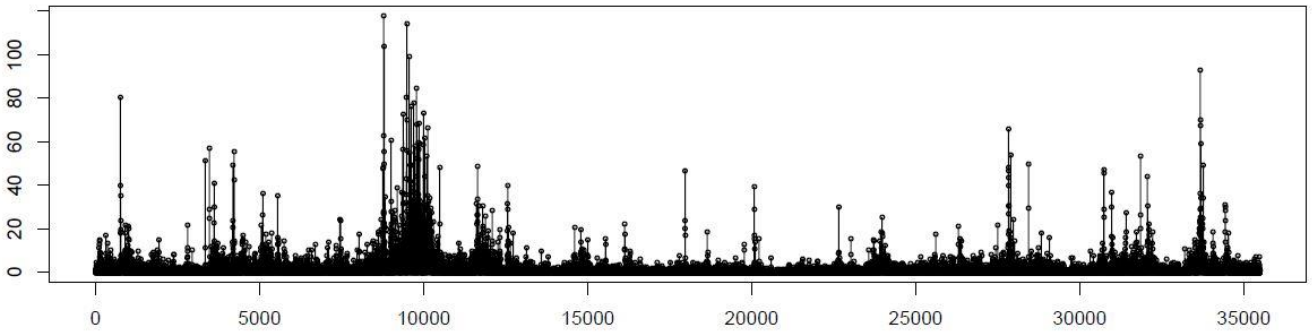
Residuals ARIMA (5,0,2)

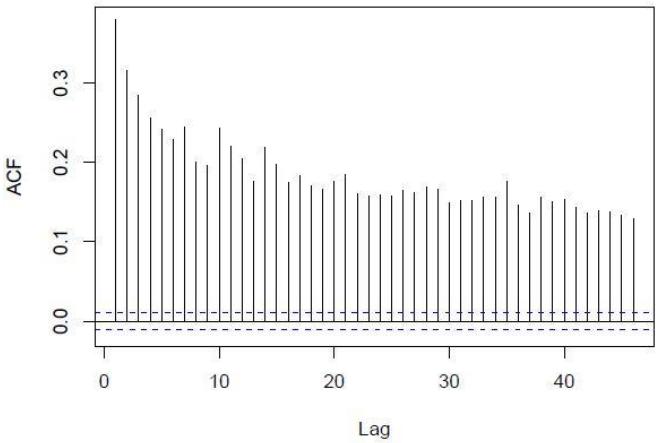


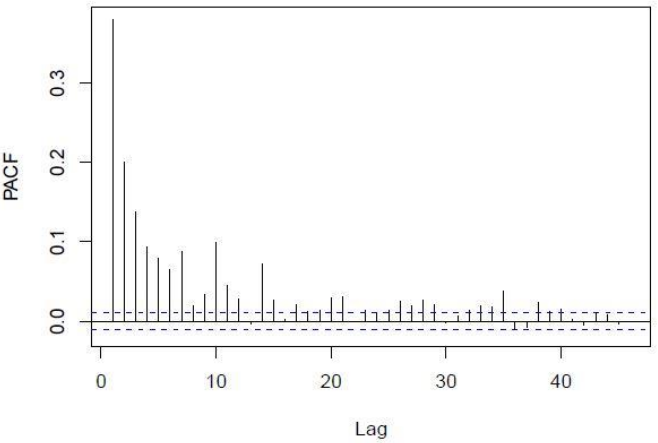


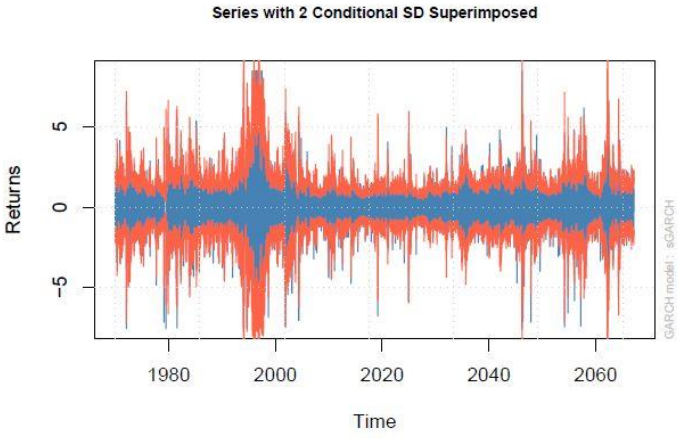


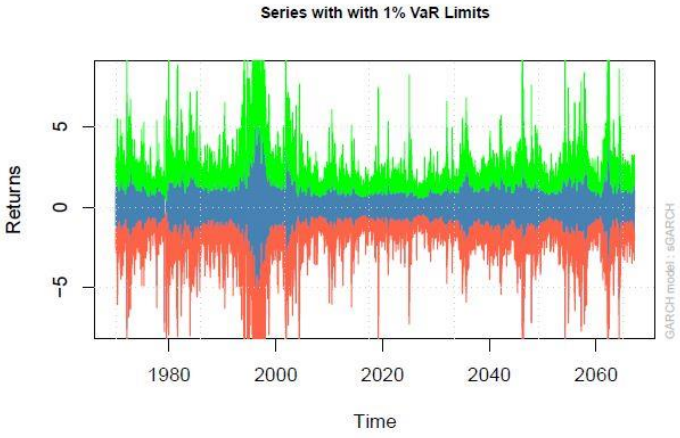
Squared Residuals ARIMA (5,0,2)

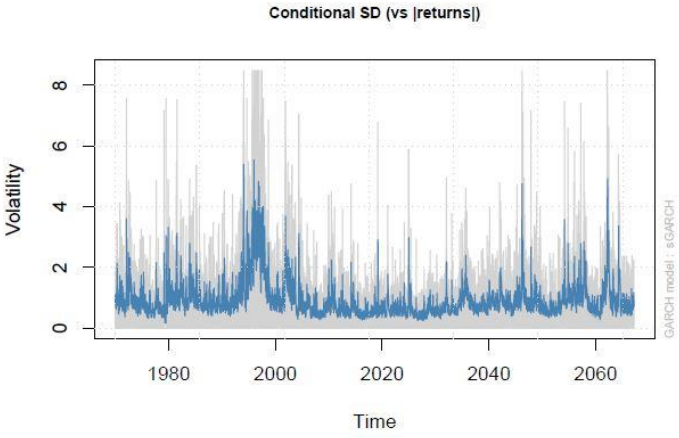


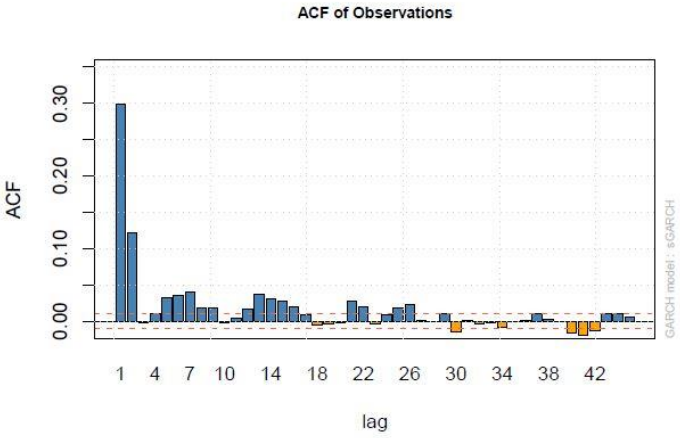


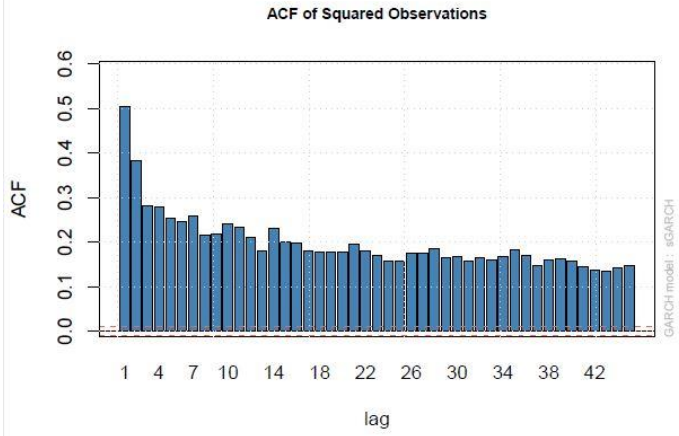


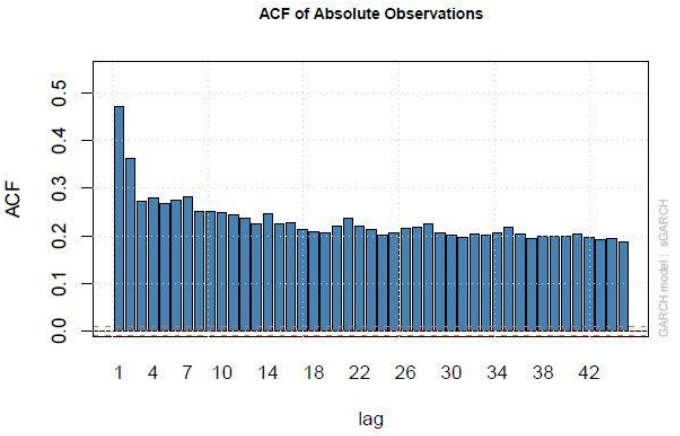

Series with 2 Conditional SD Superimposed
Returns
Time
GARCH model : sGARCH

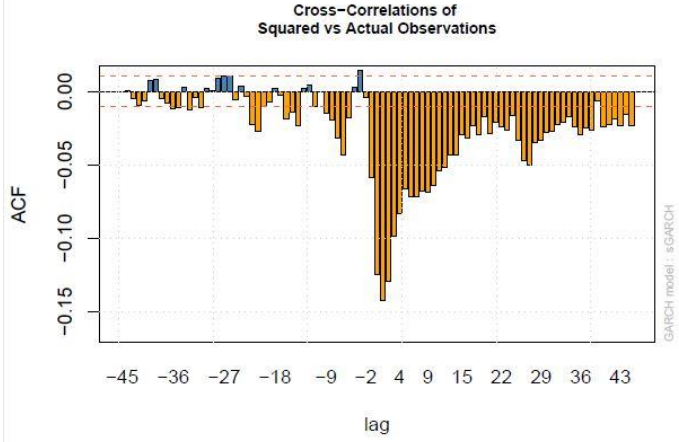

Series with with 1% VaR Limits
Returns
Time
GARCH model : sGARCH

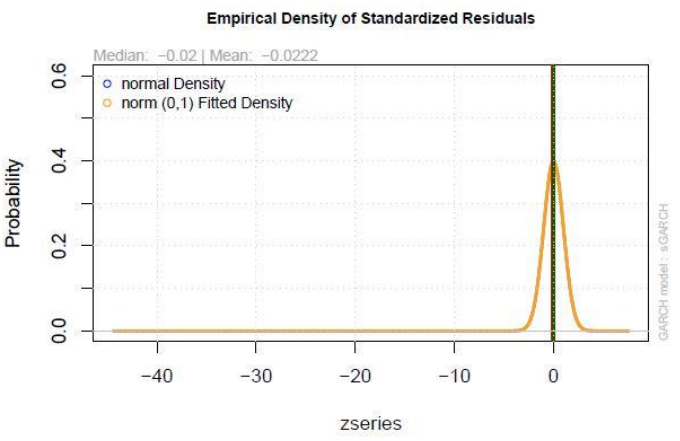

Conditional SD (vs |returns|)
Volatility
Time
GARCH model : sGARCH

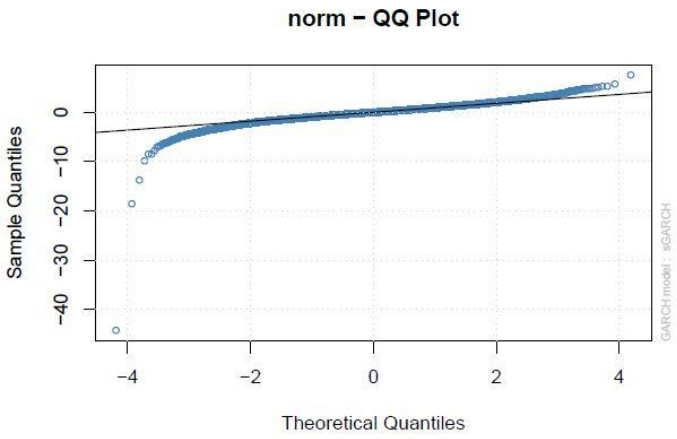

ACF of Observations
ACF
lag
GARCH model : sGARCH

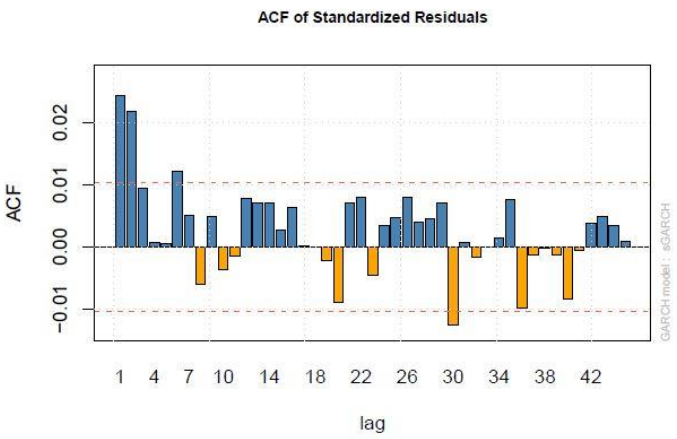

ACF of Squared Observations
ACF
lag
GARCH model : sGARCH

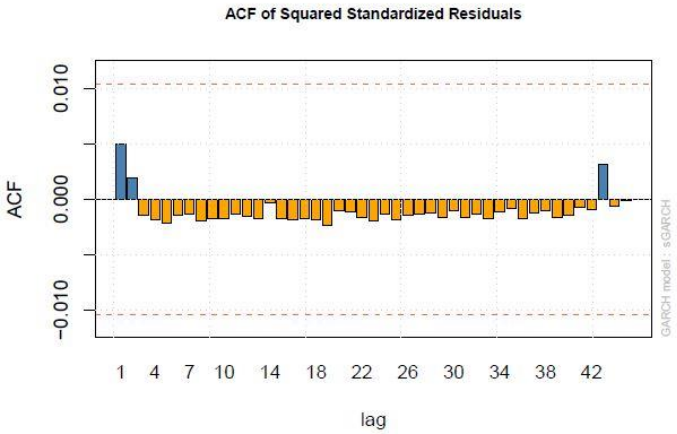

ACF of Absolute Observations
ACF
lag
GARCH model : sGARCH

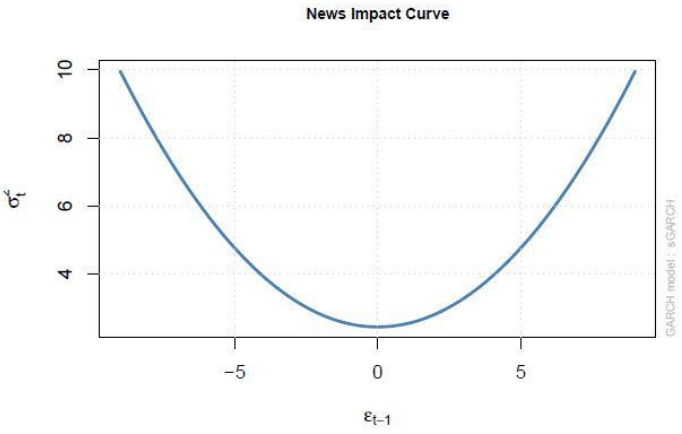

Cross−Correlations of
Squared vs Actual Observations
ACF
lag
GARCH model : sGARCH


Empirical Density of Standardized Residuals
Median: −0.02 | Mean: −0.0222
normal Density
norm (0,1) Fitted Density
Probability
zseries
GARCH model : sGARCH


norm − QQ Plot
Sample Quantiles
Theoretical Quantiles
GARCH model : sGARCH


ACF of Standardized Residuals
ACF
lag
GARCH model : sGARCH


ACF of Squared Standardized Residuals
ACF
lag
GARCH model : sGARCH


News Impact Curve
GARCH model : sGARCH

Appendix 6: Alternative Model Outcome

Table A: Coefficients for the Base Model, lowest MAE out of cross-validation fold

MAE: 0.7015		**GARCH MAE: 0.6998**			Cross-validation fold: 4					
ARMA Terms	AR1	AR2	AR3	AR4	AR5	MA1	MA2	MA3		
Coefficients	0.05	0.93***	-0.18***	-0.09***	0.09***	0.28***	-0.78***	-0.15**		
Variables	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**
Coefficients	**1.07**	**3.53**	**0.12**	**5.2*** **	**0.01**	**-2.35**	**-1.52**	**-0.68**	**-2.82****	**-0.53**

*Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively*

Table C: Coefficients for Alternative Model 1, lowest MAE out of cross-validation fold

MAE: 1.6169		**GARCH MAE: 0.7241**			Cross-validation fold: 2					
ARMA Terms	AR1	AR2	AR3	AR4	AR5	MA1	MA2			
Coefficients	0.15***	0.93***	-0.29***	-0.04***	0.07***	0.14***	-0.84***			
Variables	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Coefficients	3.96***	-1.97	8.08***	2.89***	0.77	-1.44	0.62	-2.55**	-1.55	-2.8***

*Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively*

Table D: Coefficients for Alternative Model 2, lowest MAE out of cross-validation fold

MAE: 0.7060		**GARCH MAE: 0.7002**			Cross-validation fold: 4					
ARMA Terms	AR1	AR2	AR3	AR4	AR5	MA1	MA2	MA3		
Coefficients	0.06	0.93***	-0.18***	-0.09***	0.09***	0.28***	-0.78***	-0.14**		
Variables	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Coefficients	3.38	0.9	3.28*	-4.97***	3.9	1.69	-1.43	-2.49**	2	1.31

*Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively*

Table E: Coefficients for Alternative Model 3, lowest MAE out of cross-validation fold

MAE: 1.2985		**GARCH MAE: 0.7241**			Cross-validation fold: 2					
ARMA Terms	AR1	AR2	AR3	AR4	AR5	MA1	MA2			
Coefficients	0.15***	0.92***	-0.29***	-0.04***	0.07***	0.14***	-0.83***			
Variables	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
Coefficients	-4.63***	1.59	-2.56**	-4.13***	1.55	-1.43	3.04***	-4.81***	4.16***	1.6

*Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively*

Appendix 7: Model Outcome, extensive: Coefficients are denoted for significance at the 10%, 5% and 1% level with *, **, and ***, respectively

Table F: Prediction Errors and Coefficients Base Model across all submodels										
Supervision	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MAE_I_LM	0.70	0.70	0.70	0.71	0.71	0.71	0.71	0.71	0.71	0.71
MAE_II_ARIMA	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
MAE_III_ARIMAX	0.74	0.71	0.71	0.70	0.70	0.70	0.71	0.71	0.71	0.83
MAE_IV_GARCH	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
MAE_V_GARCHX	0.71	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.71	0.78
Estimates ARIMAX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
AR1	0.06	0.05	0.06	0.05	0.05	0.06	0.05	0.05	0.05	0.04
AR2	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.94***
AR3	-0.19***	-0.18***	-0.19***	-0.18***	-0.18***	-0.19***	-0.18***	-0.18***	-0.19***	-0.17***
AR4	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***
AR5	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***
MA1	0.27***	0.28***	0.28***	0.28***	0.28***	0.28***	0.28***	0.28***	0.28***	0.29***
MA2	-0.78***	-0.78***	-0.78***	-0.78***	-0.78***	-0.78***	-0.78***	-0.79***	-0.79***	-0.79***
MA3	-0.14**	-0.15**	-0.14**	-0.15**	-0.14**	-0.14**	-0.15**	-0.14**	-0.14**	-0.15**
Concept 1	-6.45**	3.15	3.12	1.07	-0.77	-1.22	4.81**	-5.37***	3.77*	20.2***
Concept 2	5.49*	-1.27	-1.13	3.53	5.82**	5.66**	-2.07	1.58	-5	-49.08***
Concept 3	0.25	4.83***	-3.96**	0.12	1.08	-3.36**	6.69***	-0.48	-8.35***	9.57***
Concept 4	3.82***	-2.72	3.36*	5.2***	-4.07***	5.44***	-4.06**	6.91***	-4.22**	2.7**
Concept 5	-1.59	1.1	1.28	0.01	-4.19	-1.21	-0.26	6.92***	3.05	2.26**
Concept 6	-9.53***	-1.72	0.05	-2.35	-1.66	-0.07	-4.72**	-3.52*	6.71***	-0.92
Concept 7	9.49***	0.31	4.4***	-1.52	1.77	2.03	3.44**	-0.39	3.14***	-1.91
Concept 8	13.15***	2.63*	-0.13	-0.68	-0.04	-1.32	6.45**	-4.79**	-0.87	-1.64
Concept 9	-11.57	-1.93	-2.02	-2.82**	3.14**	0.59	0.1	5.56**	1.05	-1.86
Concept 10	-4.64***	3.34**	-2.76**	-0.53	0.55	0.28	-0.81	-0.76	-0.89	-4.02***
Estimates GARCHX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MU	0	0***	0.02	-0.05***	0.02	0.02	0.04**	0.04***	0.02***	0.03***
AR1	0.02	0.03***	0.02	0.04***	0.02	0.04	0.04	0.11***	0.12***	0.05
AR2	0.54***	0.55***	0.54***	0.59***	0.54***	0.55***	0.55***	0.64***	0.76***	0.55***
AR3	-0.37***	-0.36***	-0.37***	-0.36***	-0.38***	-0.37***	-0.36***	-0.35***	-0.15***	-0.38***
AR4	0.05	0.04***	0.05	0.04***	0.05	0.05**	0.05*	0.02***	-0.06***	0.05***
AR5	0.06***	0.07***	0.06***	0.07***	0.06***	0.07***	0.07***	0.07***	0.06***	0.07***
MA1	0.32***	0.31***	0.32***	0.31***	0.32***	0.3***	0.3***	0.22***	0.22***	0.29***
MA2	-0.4***	-0.42***	-0.4***	-0.45***	-0.41***	-0.42***	-0.42***	-0.54***	-0.65***	-0.42***
MA3	0.2	0.17***	0.2	0.16***	0.2	0.18***	0.17**	0.12***	-0.12***	0.19***
Concept 1	3.88	-2.89***	-1.3	-14.42***	3.23	0.88	4.16	-5.21***	0.78***	15.15***
Concept 2	5.12**	-1.55***	-2.69	1.06***	0.12	1.04	-4.41***	3.44***	-6.46***	-37.88***
Concept 3	0.56	1.51***	-1.8	0.69***	0.89	-2.84***	1.63	-1.07***	-5.01***	3.83***
Concept 4	2.22**	0.22***	-0.94	1.79***	-0.18	1.25	-1.25	2.23***	-1.84***	-0.6
Concept 5	-1.32*	-0.01***	0.84	-0.49***	-4.78***	-0.05	0.13	2.25***	-1.64***	1.54*
Concept 6	-5.37**	-1.46***	-0.31	-1.52***	-2.19**	0.67	-1.77	-0.96***	2.03***	1.01
Concept 7	4.17*	-0.35***	1.05	0.26***	1.96**	1.36	1.53	0.29***	1.31***	0.56
Concept 8	5.24*	1.11***	-0.39	-0.12***	0.14	-0.51	3.32*	-2.44***	0.59***	-1.59***
Concept 9	-7.38*	-1.33***	-1.81**	-1.77	0.48	0.63	-0.74	2.06***	-0.57***	-1.38*
Concept 10	-0.75	0.82***	-1.51**	0.84***	1.09	1.65*	1.14	-1.31***	-0.77***	-1.73*
OMEGA	0.02***	0***	0.02***	0***	0.02***	0***	0***	0***	0***	0***
ALPHA 1	0.1***	0.07***	0.1***	0.07***	0.1***	0.07***	0.07***	0.07***	0.07***	0.07***
BETA 1	0.88***	0.93***	0.88***	0.93***	0.88***	0.93***	0.93***	0.93***	0.92***	0.93***

Table G: Prediction Errors and Coefficients Alternative Model 1 across all submodels										
Supervision	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MAE_I_LM	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73
MAE_II_ARIMA	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
MAE_III_ARIMAX	2.46	1.62	2.20	2.51	2.49	2.80	2.59	2.09	2.39	2.17
MAE_IV_GARCH	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
MAE_V_GARCHX	39.21	1.15	13.10	33.77	4.01	25.39	19.63	7.82	4.18	1.19
Estimates ARIMAX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
AR1	0.16***	0.15***	0.16***	0.16***	0.16***	0.16***	0.16***	0.16***	0.17***	0.16***
AR2	0.93***	0.93***	0.93***	0.92***	0.92***	0.93***	0.92***	0.93***	0.93***	0.92***
AR3	-0.3***	-0.29***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***
AR4	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***
AR5	0.08***	0.07***	0.08***	0.07***	0.07***	0.07***	0.07***	0.07***	0.07***	0.08***
MA1	0.13***	0.14***	0.13***	0.13***	0.13***	0.13***	0.13***	0.13***	0.13***	0.13***
MA2	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.85***	-0.84***
Concept 1	0.32	3.96***	-0.15	-0.01	0.06	-0.11	-0.47	2.11	-3.08*	1.94
Concept 2	-0.75	-1.97	0.69	0.6	2.11	4.1***	4.19***	0.82	-1.77	-0.31
Concept 3	1.13	8.08***	2.43**	-1.83	-1.46	-0.26	-1.56	6.47***	-1.2	-7.19***
Concept 4	1.04	2.89***	-3.21**	3.9***	4.81***	-4.02***	6.06***	-4.57***	6.65***	-4.35***
Concept 5	-0.5	0.77	1.63	1.9*	-0.61	-3.21***	-2.46**	-2.09*	5.96***	2.75**
Concept 6	-1.07	-1.44	-2.79**	-0.22	-2.91**	-0.63	-0.66	-5.11***	-3.22***	4.49***
Concept 7	0.85	0.62	0.21	4.29***	-1.26	1.3	1.24	1.61	-0.6	2.87***
Concept 8	2.44*	-2.55**	1.96*	0.03	-0.85	-0.61	-0.08	1.28	-3.37***	0.01
Concept 9	1.7	-1.55	-1.61	-0.75	-2.32**	2.8**	-0.37	-0.77	5.09***	1.73
Concept 10	-3.1***	-2.8***	1.86*	-2.73**	-0.18	0.39	0.16	0.11	-0.28	-2.03*
Estimates GARCHX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MU	-0.03	0.05***	0.01	-0.05***	0.03	-0.04***	-0.05***	0.07***	0.06***	0.02***
AR1	-0.08***	-0.07***	-0.08***	-0.06***	-0.08***	-0.07***	0.34***	-0.07***	-0.07***	0.26***
AR2	0.76***	0.76***	0.76***	0.73***	0.76***	0.73***	0.52***	0.76***	0.76***	0.88***
AR3	-0.21***	-0.22***	-0.21***	-0.21***	-0.21***	-0.21***	-0.18***	-0.22***	-0.22***	-0.28***
AR4	-0.02***	-0.02***	-0.02***	-0.01***	-0.02***	-0.02***	-0.03***	-0.02***	-0.02***	-0.03***
AR5	0.04***	0.05***	0.05***	0.05***	0.04***	0.05***	0.06***	0.05***	0.05***	0.05***
MA1	0.37***	0.36***	0.37***	0.36***	0.37***	0.37***	-0.02***	0.36***	0.36***	0.03***
MA2	-0.62***	-0.63***	-0.62***	-0.6***	-0.62***	-0.59***	-0.53***	-0.63***	-0.63***	-0.84***
Concept 1	12.49**	2.39*	-4.85	-13.78***	-2.05	13.91***	13.21***	5.97*	-4.59*	0.77***
Concept 2	1.25	-1.19	-0.37	-2.08***	-0.44	1.52	2.33***	-3.34***	2.55**	-0.62***
Concept 3	0.97	2.13**	0.44	-1.36***	0.65	0.67	-2.87***	1.27	-1.48*	-2.98***
Concept 4	0.83	0.06	-0.3	0.83***	1.3	-0.89	1.83***	-0.91	2.04**	-1.48***
Concept 5	-0.7	0.44	0.36	1.42***	-0.56	-3.48***	-0.41***	-0.62	1.81**	0.93***
Concept 6	-0.53	0.6	-2.65***	-0.92***	-2.21***	-2.36***	1.26***	-0.83	-1.42*	1.58***
Concept 7	-0.51	2.05***	0.27	1.27***	0.59	1.83**	2.21***	-0.37	0.76	1.45***
Concept 8	0.01	-2.7***	0.42	-0.78***	-0.91	-1.05	-0.05***	-0.4	-1.3*	1.03***
Concept 9	0.8	-1.02	-1.45**	-1.27***	-1.98***	0.8	-0.97***	-1.42**	1.16	0.36***
Concept 10	-0.52	-0.78	-0.09	-1.95***	0.6	0.54	0.3***	1.59**	-0.54	-1.75***
OMEGA	0.02***	0.02***	0.02***	0***	0.02***	0***	0***	0.02***	0.02***	0***
ALPHA 1	0.1***	0.1***	0.1***	0.07***	0.1***	0.07***	0.06***	0.1***	0.1***	0.06***
BETA 1	0.88***	0.88***	0.88***	0.93***	0.88***	0.93***	0.93***	0.88***	0.88***	0.93***

Table H: Prediction Errors and Coefficients Alternative Model 2 across all submodels										
Supervision	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MAE_I_LM	0.70	0.70	0.70	0.71	0.71	0.71	0.71	0.71	0.71	0.71
MAE_II_ARIMA	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
MAE_III_ARIMAX	0.74	0.71	0.71	0.71	0.70	0.71	0.72	0.71	0.71	0.75
MAE_IV_GARCH	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70	0.70
MAE_V_GARCHX	0.71	0.70	0.70	0.70	0.70	0.70	0.71	0.70	0.70	0.74
Estimates ARIMAX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
AR1	0.05	0.06	0.06	0.06	0.06	0.06	0.05	0.05	0.04	0.04
AR2	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.93***	0.94***	0.94***	0.94***
AR3	-0.18***	-0.19***	-0.19***	-0.18***	-0.19***	-0.18***	-0.17***	-0.18***	-0.17***	-0.17***
AR4	-0.09***	-0.09***	-0.09***	-0.09***	-0.08***	-0.09***	-0.09***	-0.09***	-0.09***	-0.09***
AR5	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***	0.09***
MA1	0.28***	0.28***	0.27***	0.28***	0.28***	0.28***	0.29***	0.29***	0.29***	0.29***
MA2	-0.78***	-0.78***	-0.78***	-0.78***	-0.79***	-0.78***	-0.79***	-0.79***	-0.79***	-0.79***
MA3	-0.15**	-0.14**	-0.14**	-0.14**	-0.13**	-0.14**	-0.15**	-0.16**	-0.15**	-0.15**
Concept 1	6.26**	4	-3.32	3.38	-1.19	-4.19*	4.72**	-4.25*	-1.56	-9.64***
Concept 2	-5.28*	-2.47	-1.02	0.9	-6.62**	1.2	4.63*	-0.17	-3.58	24.84*
Concept 3	0.01	-5.48***	3.29*	3.28*	1.08	-3.25**	-3.69**	1.51	8.13***	-4.38**
Concept 4	3.23**	4.76**	2.02	-4.97***	0.68	6.53***	8.41***	-10.2***	6.6***	-5.26***
Concept 5	3.28**	0.53	4.52***	3.9	4.07**	-0.47	-5.43**	3.02	4.2*	0.47
Concept 6	-9.05**	-1.35	1.59	1.69	-5.35**	-1.24	-6.72***	2.18	-6.56***	1.3
Concept 7	-11.95**	0.81	-1.52	-1.43	-1.41	5.54**	4	-1.56	-2.93**	4.53***
Concept 8	-15.83**	-2.63**	2.37*	-2.49**	-1.24	6.45**	-0.76	4.16*	-0.94	-4.1***
Concept 9	-1.22	-4.35**	-1.57	2	3.76***	1.49	-1.27	1.4	1.28	4.77***
Concept 10	4.49***	1.57	2.34*	1.31	-1.46	-3.62***	3.43***	8.01***	-0.2	2.14*
Estimates GARCHX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MU	-0.02***	0.03	0.01	0.05*	0.01	0.04*	0.01***	0.03**	0.02**	0.04***
AR1	0.03***	0.01	0.02	0.01	0.05***	0.02	-0.16***	0.09***	0.03	0.04
AR2	0.55***	0.54***	0.53***	0.54***	0.54***	0.54***	0.92***	0.55***	0.54***	0.56***
AR3	-0.36***	-0.37***	-0.36***	-0.36***	-0.47***	-0.37***	-0.05***	-0.49***	-0.38***	-0.36***
AR4	0.04***	0.05	0.05	0.05	0.08***	0.05	-0.09***	0.09***	0.05	0.05*
AR5	0.07***	0.06***	0.06***	0.06***	0.06***	0.06***	0.05***	0.06***	0.06***	0.07***
MA1	0.31***	0.32***	0.32**	0.33***	0.29***	0.31***	0.5***	0.26***	0.31***	0.3***
MA2	-0.42***	-0.4***	-0.4***	-0.41***	-0.42***	-0.41***	-0.73***	-0.44***	-0.41***	-0.43***
MA3	0.17***	0.19	0.18	0.18	0.29***	0.19	-0.23***	0.3***	0.2	0.17**
Concept 1	-5.96***	0.89	1.34	4.82	-5.66	-2.92	-0.56***	-2.44	0.64	-8.25***
Concept 2	-4.01***	-3.71	-1.24	1.88	-2.37	4.47**	6***	-3.6**	-5.27	23.02**
Concept 3	-0.41***	-3.39***	1.74	1.43	0.9	-2.19**	-1.74***	0.53	4.16**	-1.29
Concept 4	1.85***	1.3	-1.47	0	-3.53*	2.38**	3.77***	-5.45***	4.27***	-2.96***
Concept 5	2.5***	1.26	1.34	2.59*	3.56***	-0.38	-2.89***	1.68	0.34	-1.24
Concept 6	-5.14***	-0.3	0.23	0.12	-1.32	-0.63	-4.36***	2.3	-1.16	-0.81
Concept 7	-5.43***	-0.11	-0.76	-1.84*	-0.8	3.21**	1.15***	-1.09	0.06	0.76
Concept 8	-7.83***	-0.7	1.87**	-1.06	1.33	2.46	0.03***	2.05	0.89	-1.82**
Concept 9	-2.26***	-3.66***	0.09	1.04	2.08***	-0.26	0.25***	0.24	-0.35	0.75
Concept 10	0.53***	-0.58	0.99	1.88**	-0.87	-2.5***	0.34***	1.2	-0.02	1.22
OMEGA	0***	0.02***	0***	0.02***	0.02***	0.02***	0***	0***	0.02***	0***
ALPHA 1	0.07***	0.1***	0.09***	0.1***	0.1***	0.1***	0.07***	0.07***	0.1***	0.07***
BETA 1	0.93***	0.88***	0.9***	0.88***	0.88***	0.88***	0.93***	0.93***	0.88***	0.93***

Table K: Prediction Errors and Coefficients Alternative Model 3 across all submodels										
Supervision	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MAE_I_LM	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73	0.73
MAE_II_ARIMA	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
MAE_III_ARIMAX	1.95	1.29	2.01	1.97	2.28	2.81	2.00	2.14	2.60	2.30
MAE_IV_GARCH	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72	0.72
MAE_V_GARCHX	29.47	1.10	35.54	10.95	27.75	5.84	2.85	2.35	2.43	1.10
Estimates ARIMAX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
AR1	0.16***	0.15***	0.16***	0.16***	0.15***	0.16***	0.16***	0.16***	0.17***	0.16***
AR2	0.93***	0.92***	0.93***	0.93***	0.92***	0.93***	0.92***	0.93***	0.93***	0.92***
AR3	-0.3***	-0.29***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***	-0.3***
AR4	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***	-0.04***
AR5	0.08***	0.07***	0.08***	0.07***	0.07***	0.07***	0.07***	0.07***	0.07***	0.07***
MA1	0.13***	0.14***	0.13***	0.13***	0.14***	0.13***	0.13***	0.13***	0.12***	0.13***
MA2	-0.84***	-0.83***	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.84***	-0.85***	-0.84***
Concept 1	-0.14	-4.63***	-0.06	-0.49	0.2	-0.2	-1.01	1.79	-2.25	-0.15
Concept 2	0.29	1.59	0.1	-0.14	-0.44	-2.38*	-2.81*	-0.45	2.7*	0.39
Concept 3	-1.19	-2.56**	-2.27*	1.34	1.73	2.63**	-2.96**	-4.41***	2.61**	7.55***
Concept 4	0.9	-4.13***	4.01***	1.78	-4.46***	0.18	5.27***	7***	-9.61***	5.46***
Concept 5	0.66	1.55	1.32	4.65***	3.46***	4.86***	-1.21	-1.79	1.59	3.66***
Concept 6	-1.09	-1.43	-2.16*	2.63**	0.34	-3.73***	-0.14	-4.14***	0.48	-5.1***
Concept 7	-0.91	3.04***	0.91	-0.51	-0.74	-1.58	2.78**	3.83***	-0.59	-1.15
Concept 8	-0.96	-4.81***	-1.85	1.32	-2.03*	-1.16	4.7***	-0.3	3.02***	-0.18
Concept 9	2.88**	4.16***	-1.2	-0.56	2.29**	3.38***	1.51	-1.51	1.81	0.72
Concept 10	3.58***	1.6	1.07	1.71	-0.88	-0.06	-3.79***	1.89*	6.45***	0.11
Estimates GARCHX	**SV 1**	**SV 2**	**SV 3**	**SV 4**	**SV 5**	**SV 6**	**SV 7**	**SV 8**	**SV 9**	**SV 10**
MU	-0.01	0.05***	-0.04	0.06*	-0.05***	0.02	0.05***	0.01***	0.05***	0.03***
AR1	-0.08***	-0.07***	-0.07***	-0.08***	0.22***	-0.08***	-0.07***	0.22***	-0.06***	-0.07***
AR2	0.76***	0.76***	0.73***	0.76***	0.72***	0.76***	0.76***	0.96***	0.77***	0.76***
AR3	-0.21***	-0.22***	-0.21***	-0.21***	-0.24***	-0.22***	-0.22***	-0.32***	-0.22***	-0.22***
AR4	-0.02***	-0.02***	-0.01***	-0.02***	-0.01***	-0.02***	-0.02***	-0.02***	-0.02***	-0.01***
AR5	0.05***	0.05***	0.05***	0.04***	0.05***	0.04***	0.05***	0.05***	0.05***	0.05***
MA1	0.37***	0.37***	0.37***	0.37***	0.08***	0.37***	0.36***	0.08***	0.36***	0.37***
MA2	-0.62***	-0.63***	-0.59***	-0.62***	-0.66***	-0.62***	-0.63***	-0.91***	-0.63***	-0.62***
Concept 1	-9.53	-3.74***	-13.07**	-4.05	-13.58***	-3.59	-2.06	-2.34***	-2.9	-0.72
Concept 2	-1.11	1.24	-0.9	-0.28	2.64***	0.16	0.7	1.37***	-1.03	-0.59
Concept 3	-0.92	0.76	-0.78	0.26	1***	0.79	-1.94**	-2.08***	0.42	2.8***
Concept 4	0.73	-1.07	1.64*	-1.72**	-0.81***	-2.89***	1.45*	2.24***	-4.32***	2.19***
Concept 5	0.74	-0.88	1.98**	0.84	2.42***	3.25***	-0.2	0.03***	0.61	1.67**
Concept 6	-0.49	-2.08**	-1.52**	2.35***	0.09***	-1.63**	0.3	-0.55***	0.78	-0.92
Concept 7	0.61	0.19	-0.08	-0.19	-2.06***	-0.78	-0.32	2.1***	-0.71	0.69
Concept 8	0.4	-1.65**	-0.09	2.06***	-0.89***	1.21	2.92***	0.69***	0.17	1.16
Concept 9	0.84	0.9	-1.22	0.79	1.47***	2.52***	-0.65	0.6***	-0.8	-0.7
Concept 10	0.75	1.33*	-0.5	0.83	-0.6***	0.24	-2.76***	-0.14***	2.05***	-0.26
OMEGA	0.02***	0.02***	0***	0.02***	0***	0.02***	0.02***	0***	0.02***	0***
ALPHA 1	0.1***	0.1***	0.07***	0.1***	0.07***	0.1***	0.1***	0.06***	0.1***	0.09***
BETA 1	0.88***	0.88***	0.93***	0.88***	0.93***	0.88***	0.88***	0.93***	0.88***	0.91***