Business Analytics

– SEMINAR SUMMER SEMESTER 2018 –

Visualizing stock performance and Form 8-K sentiment using R's shiny package

– SEMINAR PAPER –

Submitted by:
Hadi Chami

Advisor:
Prof. Dr. Dirk Neumann

Contents

1 Introduction 1

2 The Shiny Package for R 2

3 Theoretical Background 3
- 3.1 Noise Trader Theory 3
- 3.2 Sentiment Analysis 3

4 Empirical Design 5
- 4.1 Methods 5
- 4.2 Dataset 5

5 Practical application of Shiny in R 6
- 5.1 Shiny Dashboard 6
- 5.2 Program Structure 8
- 5.3 Output 9

6 Critical Discussion 16

7 Conclusion 16

A References i

B List of Figures ii

C List of Tables iii

Abstract

Data analysts has lately promote significant research on the impact of news sentiment in finance. This paper thus explores how different investor types perceive relevant Form 8-K news, based on sentiment scores, and how the stock market reacts to this. Therefore, the noise trader theory and the sentiment analysis approach were introduced to explain the difference among informed investors and noise traders. The powerful shiny package in R was adopt to build an visual interactive environment and nicely give the user the opportunity to see how the different investors decision has an impact on the stock market prices. We can hold on - shiny is under rapid development and is gaining popularity, resulting in more and more capabilities being added.

1 Introduction

The behavior of investors through the impact of financial news has been shown as a growing discussion among finance scientists. In this manner the noise trader theory, which initially was introduced in Blacks seminal work (1986) has been largely deal with the disadvantages, that arise from noise trading.
The central assumption of noise traders is that they decide on their emotions and feelings and that leads to misinterpret of information. Thus, they are noticed as irrational or uniformed investors whose interest for capital and risk cannot be explained by fundamental news [17]. It has for quite some time been a bone of conflict in the field of financial theory whether noise traders have a conduct impact on stock prices. The modern finance pretend that noise traders have a little effect on stock prices as the investment decision made random. Even if noise traders planning to act rational and make a mistake on purpose, informed investors instantly could eliminate it [6]. In such transactions, noise traders would suffer and lose money to informed investors and, in some cases, cut of the stock market [7]. Thus, the price of an asset is identical to its fundamental value and a market efficiency is reached when the market gets back to its equilibrium. According to this theory and approach, the modern financial theory has been established [16].
For us information scientists it is of great interest to study how the news reception differs between these two groups of investors based on their rational abilities. The US-regulated Form 8-K filings provide an interesting case to study how individuals perceive relevant news and how the stock market reacts to this news.

The software program R for statistical computing offers a powerful package for building interactive web applications. This seminar paper will introduce and apply this powerful tool to visualize specific data with the help of several functions within the package. Therefore, a brief overview of the shiny package will be presented in section 2. Section 3 examines the noise trader theory and sentiment analysis approach for understanding the theoretical background behind the visualization part in this seminar paper. The methodology and composition of the dataset will be presented in Section 4. Section 5, the shiny package will be used to visualize and analyze the dataset and the generated output will be presented. Following, a conclusion will be given.

2 The Shiny Package for R

Shiny is an open source R package that allows it easy to visualize interactive web-based applications straight from R. It was created and maintained by RStudio and is available from Comprehensive R Archive Network (CRAN). The shiny package provides a powerful web framework and comes with a variety of widgets for quickly building user-interface without requiring any HTML, CSS or JavaScript knowledge.

Basically, shiny applications are included in a single script called app.R, which has three components:

- a user interface object
- a server function
- a call to the shinyApp() function

The user interface (ui.R) object defines the layout of the application with all sorts of interactions, including widgets. The server function (server.R) contains instructions on how to build and rebuild the R objects displayed in the user interface section. Finally the shinyApp() function is used to combine the ui and server code into a functioning shiny app. This method is more adequate for smaller applications. For larger projects it is recommended to separate the ui.R and server.R files, which makes the code easier to manage [18]. Figure 1 visualizes the package and exhibits the interactive progress of the two components.

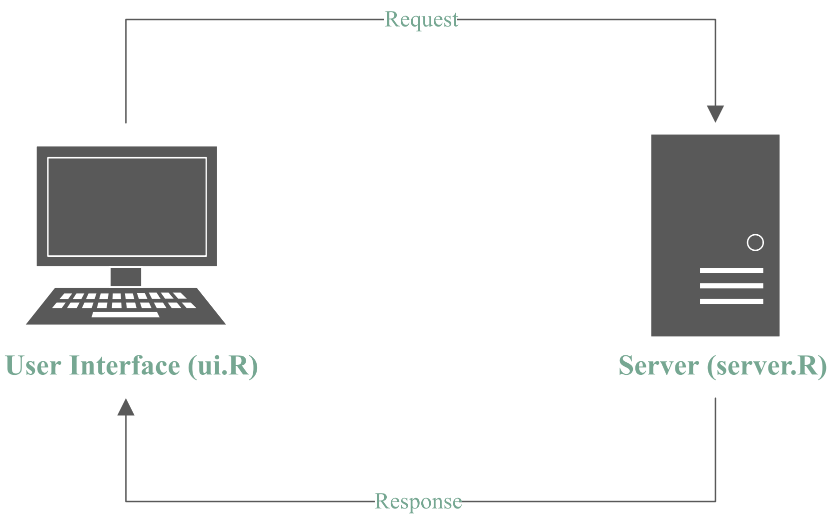


Figure 1: System architecture for the shiny application.

In order to be able to create a shiny application in R, RStudio or other GUIs, it is important to first get and install the package. There are two ways to do this. Either by using the package management functions included in the editor or by typing install.packages("shiny") at the console. Both options lead to the installation of the shiny package on the computer [1].

3 Theoretical Background

In this section, fundamental theoretical background about the noise trader theory and sentiment approach will be explained. Beginning with the theory of noise trader, the difference between the two investor groups - informed and noise investor are clarified. Further, an introduction into the sentiment analysis theory is given, that also explains the difference between the four common dictionaries of the approach and gives a deep understanding of the topic for the further steps of the seminar paper.

3.1 Noise Trader Theory

Over the last 50 years, a lot of research has been committed to the effect of news on the stock market. Financial experts have different views on the behavior of stock markets reaction to new information and on how news can be consolidated in trading and investment strategies. In the 1950s, which was considered as the beginning of the modern finance, investors were almost behave rational in their decisions. Beginning from this rationality assumption, the concept of the Efficient Market Hypothesis (EMH) was established [5]. However, on October 19, 1987 the stock in the EMH crashed together with the rest of the market around the world [17]. According to this crash, there is impressive confirmation that numerous investors try not to take economists suggestion whether to buy or hold a market portfolio. They regularly pick stocks through their own particular research or on the guidance of news or sentiments. Following this behavior, the approach of the noise trader theory has been introduced, for which Black (1986) and Shleifer and Summers (1990) are considered to be the pioneers of this concept [4].

The approach assumes two different groups of market participants:

- Informed investors (rational trader)
- Uninformed investors (noise trader)

Informed investors base their rational decisions on a full set of information and fundamental news. Since the number of informed investors is always smaller than the total number of investors, they have no effect on the competitive equilibrium prices. For instance, even informed investors have the great advantage to separate between information and noise, they cannot anticipate when an over- or an undervalued asset price will get back to their fundamental price. As a consequence, arbitrage trading of informed investors is sensitive risky and limited [10].

Noise trader is characterized in several literature of financial research as a stock trader that operate irrationally and do not use fundamental information for their investment decisions. They often have bad timing, track trends, overreact to incoming news and trade on noise as if it were information. The transactions of noise traders may have an impact on the market and performance of financial instruments, thereby increasing price volatility [3].

3.2 Sentiment Analysis

One aspect that has an important impact on how a stock is noticed by investors is the news media. By trying to stimulate our emotions and interest, the media tend to make active investment decisions and market timing. All of these findings have expanded the interest in sentiment analysis to financial news media [15].

Sentiment analysis is a Natural Language Processing (NLP) task that focus on the automatic identification and classification of investors feelings, opinions, sentiments, emotions and other attributes regarding entities such as events or topics in a given text. This kind of classification can be attributed in 3 categories such as positive, negative and in some cases neutral [2].

Furthermore, there exist several fundamental sentiment techniques to deal with the issue of extracting sentiment automatically. Among a variety of techniques two approaches have been found to be the most appropriate: i) lexicon-based approach and ii) machine learning approach. The lexicon-based approach is based on a sentiment lexicon, accumulation of known and assembled sentiment terms. It is subdivided into dictionary-based approach and corpus-based approach that are predicated on analytical and grammatical methods to identify sentiment polarity. The machine learning approach uses the most common machine learning algorithms and is further based on the use of linguistic features [14].

Besides the just mentioned sentiment techniques, in the financial sector the focus matter on so-called dictionaries. In numerous applications, when data scientists effort to measure the tone of an annual report, initial public offering (IPO) prospectus or earnings press, the first step is choosing a dictionary that is also known as a world list. These dictionaries accumulate applicable words that are related with a positive, negative or emotionally clarification, regardless of the textual content. Documents that rather contain a high number of positive words are noticed as optimistic, and similarly those with a high number of negative words as pessimistic [13].

In a quantity of various literature, researchers mostly focus on four different word lists:

- Harvard University's General Inquirer
- Diction
- Henry [2008]
- Loughran and McDonald [2011]

In addition to the general positive and negative word lists, these four dictionaries also contain subcategories of word topics such as weak modal, uncertainty, constraints, pain, extreme emotion, pleasure and even virtue [12].

News sentiment, delivers us the most recent information about the stock market for investors. Several studies have exhibited that financial news appears in a strong correlation with future stock performance. For that reason, collecting sentiments from financial news is an effective and beneficial way to assist investors in the stock price predictions. One of the possible solutions is to use software systems that manage financial analysis and market sentiment of news articles [11].

Information science relevance refers to the detecting of emotional sentiment maintain in text through specific semantic analysis for a range of intentions. There is a group of research that is fixate on sentiment analysis or so called opinion mining. It is fundamentally used to analyze positive and negative words and processing text with the ambition of categorize its emotional position. Maks and Vossen (2012) work is an example of sentiment analysis which demonstrates a lexicon model of wide sentiment analysis and option mining. Here, we learn to understand how individuals with different rational abilities perceive information in text messages and what semantics have for different influences [9].

4 Empirical Design

In this section, the application and the structure of the obtained dataset will be presented. First the use of a two-stage approach and a LASSO regression will be discussed, that gives information how the dataset was developed. Afterwards the framework and composition of the finished dataset will be elucidated and the most important information explained.

4.1 Methods

A two-stage approach was used to interpret news reception between informed and noise traders. First, the approach is presented to fragment stock prices and a noise residual by the use of the Kalman filter. Second, the fragmentation is delegated to the informed investor and the noise residual to the noise trader. After that the LASSO regression was used to statistically filter relevant words for the two investor groups. Finally, the format and structure of the dataset will be explained in the next section [8].

4.2 Dataset

The provided dataset was obtained from all U.S. Form 8-K fillings and covers a timeline from 2004 to 2013 [8]. After several filtering and text cleaning procedures, it contains 73986 observations and 46 variables. The dataset is structured between informed investor and noise trader and contains sentiment scores. Each of the investor type is further divided into six categories, which now will be explained.

Full:	Sentiment score calculated based on all words in the document independent of categories.
Fact:	Sentiment score calculated based on all words from the Positive and Negative category of the Harvard IV General Inquirer.
Fact.Pos:	Only Positive category from Harvard IV General Inquirer.
Fact.Neg:	Only Negative category from Harvard IV General Inquirer.
Emotions:	Sentiment score calculated based on eight emotions categories from NRC Word-Emotion Association Lexicon.
EmotionsFact:	Sentiment score calculated based on eight emotions categories from NRC Word-Emotion Association Lexicon.

In addition to the just mentioned data, the dataset also contains market data regarding the firm that had published the Form 8-K stock prices.

Abn. Return:	Shows how much more the stock of the company reacts, on which there was a message, compared to the market average.
Date:	Dates of the data of all obtained Form 8-K fillings. The exactly date range extends between 2004-01-02 and 2013-12-31.

5 Practical application of Shiny in R

In this section, the practical application of shiny in R will be presented. First, the construction and advantages of the shiny dashboard will be introduced and explained. Second, a logical structure of the program is given, that briefly describe the main idea behind the construction of the app. Finally, the hole output of each page of the shiny dashboard, represented in the form of code snippets and plot output, is explained and illustrated.

5.1 Shiny Dashboard

The shiny package provides a number of user interface schemes and styles for the generated apps. Early on, however, it was decided to use the standardized dashboard from shiny, which can be generated with the help of the additional package shinydashboard. It offers a very clear assignment of input elements mostly on the left side and a large output area on the right side. Also, on the help pages of shiny the shinydashboard is praised as a preferable progression of the shiny apps.

A shiny dashboard has three parts: a header, a sidebar, and a body. Here is the most minimal possible structure for a dashboard page:

```
library(shiny)
library(shinydashboard)

ui <- dashboardPage(

   # creates the header
   dashboardHeader(),
   # creates the sidebar
   dashboardSidebar(),
   # creates the body
   dashboardBody()
)

server <- function(input, output) { }

shinyApp(ui, server)
})
```

dashboardHeader

The *dashboardHeader* is structured into a title and if desirable a dropdown menu that is separated into three types - messages, notifications, and tasks.

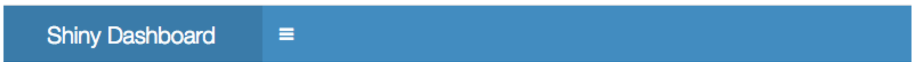


Figure 2: The header part of the shiny dashboard.

Setting the title can be simple done with the use of the *title* argument.

```
dashboardHeader(
   title = "Shiny Dashboard"
)
```

dashboardSidebar

After the header comes the sidebar. A *dashboardSidebar* is usually used for quick navigation. It can contain menu Items that are linked to a different body of the dashboard. For this seminar work, a structuring into 4 menu items has proved to be logical. Each individual menu item represents different visualization options of the dataset and contains different graphical representations.

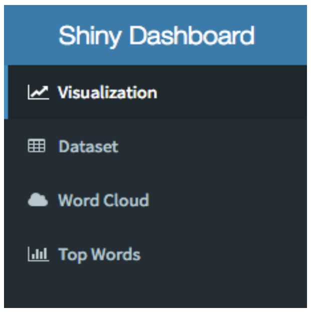


Figure 3: The sidebar part of the shiny dashboard.

In order to create a sidebar, the function *sidebarMenu()* must first be called, within this, then the *menuItem()* function can be used to create several menu items.

```
dashboardSidebar(sidebarMenu(

# Creates a menu item in the sidebar
menuItem(
   # Name of the menu Item
   "Visualization",
   # name of the tab that this menu item will activate
   tabName = "visu",
   # display a icon from fontawesome.com icon list
   icon = icon("line-chart")
)
```

dashboardBody

The *dashboardBody()*, which is the third part of a shiny dashboard, is used to display any shiny content. Working with a dashboard brings a lot of advantages in the way of structuring the body. Therefore the *box()* function was used as the basic building block of the app.

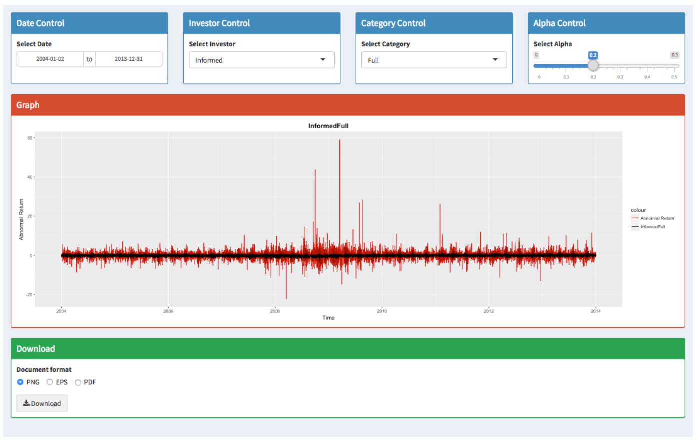

Date Control
Select Date: 2004-01-02 to 2013-12-31
Investor Control
Select Investor: Informed
Category Control
Select Category: Full
Alpha Control
Select Alpha: 0.2
Download
Document format: [✓ PNG] EPS PDF

Figure 4: The body part of the shiny dashboard.

To use the *box()* function inside the *dashboardBody()* we typically need to place the boxes inside a *fluidRow()*.

```
dashboardBody(

#  Define tab Item container
tabItems(
   # link tab Item to menu item "visu"
   tabItem(tabName = "visu",
   fluidRow(
      # create a box
      box(
         # title of the box
         title = "Date Control",
         # items background color
         status = "primary",
         # box header background color
         solidHeader = TRUE,
         # width of the box
         width = 3,
         # height of the box
         height = 142,
         # control widget
         dateRangeInput("daterange", "Select Date",
         start = min(OLS.Data$Date),
         end = max(OLS.Data$Date))
)
```

5.2 Program Structure

The shiny dashboard program for the seminar paper is structured into four menu Items. Every page of a menu item has the same layout and starts with an input area, followed by an output area and at the end a download area. In order to make it as uniform as possible, colors were used to create a clear demarcation between the three components:

- Input box marked in blue
- Output box marked in red
- Download box marked in green

The idea behind a uniform layout is dedicate to the user to enable a simple and clear navigation on every page. Each menu item represents a different visualization of the dataset and shows the use of different powerful and creative functions of the shiny package. The application is divided into the following four pages.

1. Visualization
2. Dataset
3. Word Cloud
4. Top Words

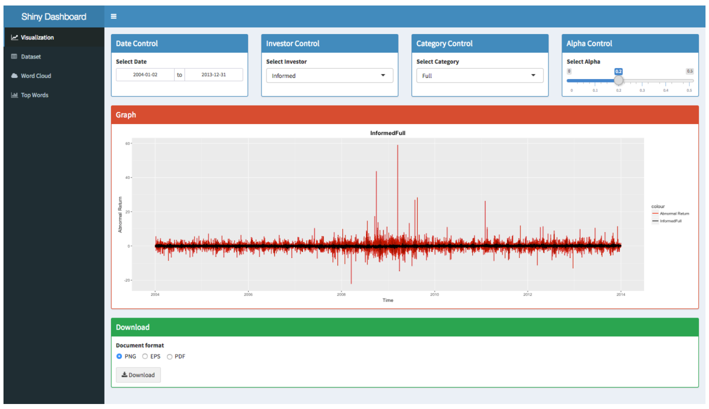
The following section explains the framework and technical implementation of the shiny dashboard application. Each individual menu item is described and explained by detail with code snippets and graphical outputs.

5.3 Output

To navigate through the shiny application the following link can be used:
https://hadichami.shinyapps.io/Shiny_Dashboard/

1. Visualization

The visualization page is the main part of shiny application. It is used to visualize how the two groups of investors, as mentioned in section Noise Trader Theory, perceive relevant Form 8-K news (based on sentiment scores) and how the stock market reacts to this. As explained in the previous section, the visualization page is divided into a series of widgets, the main output and at the end a download box. Figure 5 shows the content and structure of the Visualization page.



Select Date 2004-01-02 to 2013-12-31
Select Investor Informed
Select Category Full
Select Alpha 0.2
Document format [✓ PNG] EPS PDF

Figure 5: Output of the Visualization page.

The main focus of the data visualization was to create different kind of filters and see how the output changes. The decision to implement four filter boxes was seen as the most efficient solution within the given dataset. The user has the choice between setting one filter, setting several filters or setting all filters simultaneously. Each time a filter is selected, a new output based on the new filter is plotted. To understand what kind of filters were created and how they operate, will now be explained in more detail.

Date Control | Inside the Date Control box the *dateRangeInput()* function is used to select, by clicking on the left and right date, between a date period (from - to). To cover the entire period, the start and end options were initialized with the minimum Date and maximum Date of the Date column in the dataset.

```
dateRangeInput("daterange", "Select Date", start = min(OLS.Data$Date),
   end = max(OLS.Data$Date))
```

Investor Control | Inside the Investor Control box the *selectInput()* function is used to choose between the two investor groups - informed and noise trader.

```
selectInput("investor", label="Select Investor", choices = list("Informed" =
   "Informed", "Noise" = "Noise"), selected = "Informed")
```

Category Control | Inside the Category Control box the *selectInput()* function is used to choose between six different categories, given by the dataset, of the two investor groups.

```
selectInput("category", label="Select Category", choices = list("Full" = "Full",
   "Fact" = "Fact", "Fact Positive" = "Fact.Pos", "Fact Negative" = "Fact.Neg",
   "Emotions" = "Emotions", "Emotions Fact" = "EmotionsFact"), selected = "Full")
```

Alpha Control | Inside the Alpha Control box the *sliderInput()* function is used to create a slider with a minimum and maximum value, which apply a transparency effect on the plot.

```
sliderInput("alpha", label = "Select Alpha", min = 0, max = 1, value = 0.8,
   step = 0.1)
```

After the control widget boxes have been explained and illustrated, the graphic implementation and visualization of the dataset will be discussed. A very powerful tool of the shiny package is the *reactive()* function. A reactive expression uses the widget input and returns a value. It will automatically update the returned value whenever the original widget changes. In our case, the user can immediately see the different outputs by changing different filters.

To get the graphical output in shiny, a code section in the user interface file (ui.R) and server file (server.R) is needed. The ui.R file is used to create the box frame, whereas the server.R file creates the output which is assigned to this box.

ui.R

```
box(
   title = "Graph",
   status = "danger",
   solidHeader = TRUE,
   width = 12,
   plotOutput("plotVisu"))
```

To plot the data from the dataset the powerful *ggplot()* function was used. It can be applied to generate quick and complex plots and includes several opportunities and advantages compared to the basic *plot()* function in R. To see how different investor groups, based on their sentiment score perceive news and how the stock markets reacts, an overlaying of two graphs has been produced plus an alpha slider, which is responsible for the transparency of the plot line.

server.R

```
output$plotVisu <- renderPlot({

      ggplot(aes(x = Date, y = NYSE)) + geom_line() +
      ggtitle(paste(input$investor,input$category,sep = "")) +
      theme(plot.title = element_text(hjust = 0.5,face="bold")) +
      labs(x = "Time", y = "S&P500") +
      geom_line(aes(x = Date, y = OLS.Data.filtered[parts$partC]), color="red",
      alpha = rep(as.numeric(input$alpha), nrow(OLS.Data.filtered[parts$partC])))
})
```

In order to see how different filter affects the output, six graphs are used to show the various transformation of the plot.

The results between the two investor groups and a total date range with a filter on the category widget can be considered in Figure 6.

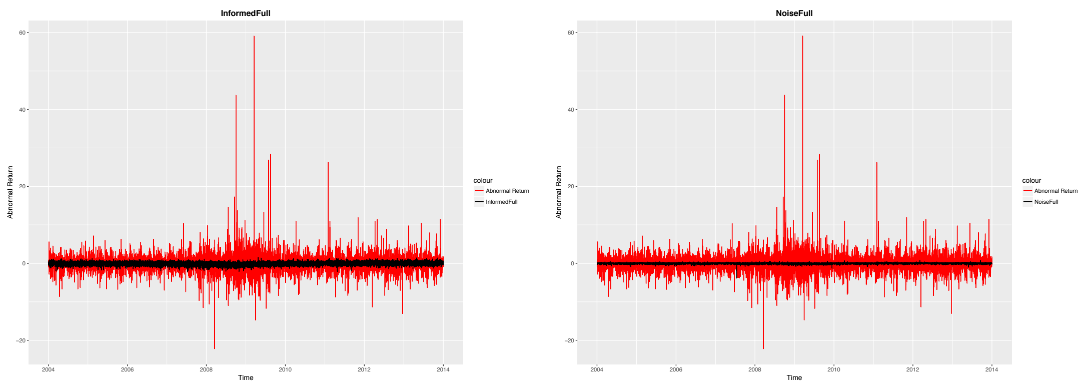


Figure 6: Output of a full informed and full noise investor.

The results between the two investor groups and a fixed date range with a filter on the category widget can be considered in Figure 7.

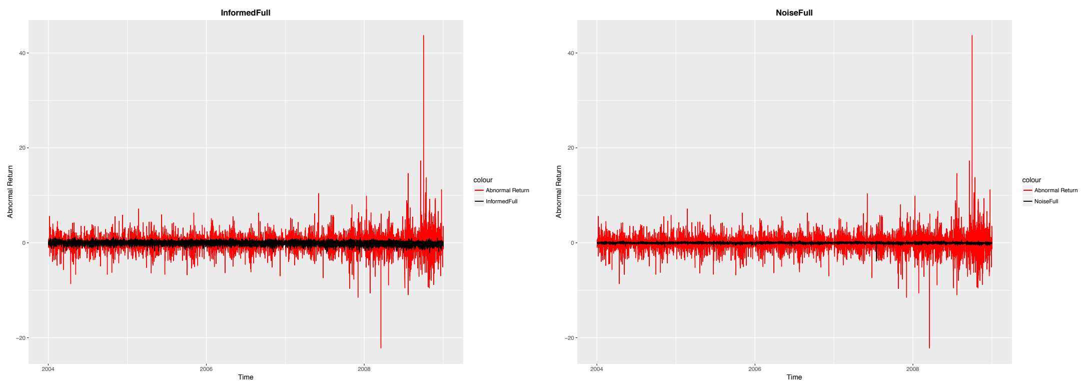


Figure 7: Output of a full informed and full noise investor with a fixed date range.

The results between the two investor groups and a total date range with a filter on the category widget and a lower selected alpha value can be considered in Figure 8.

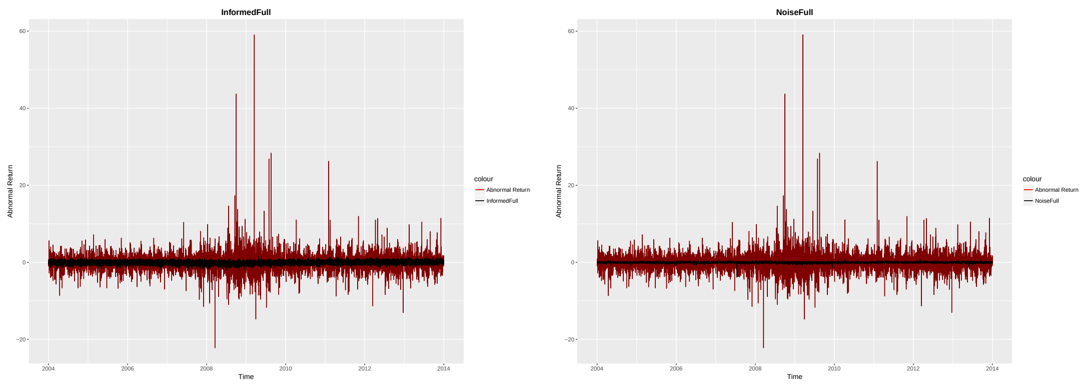
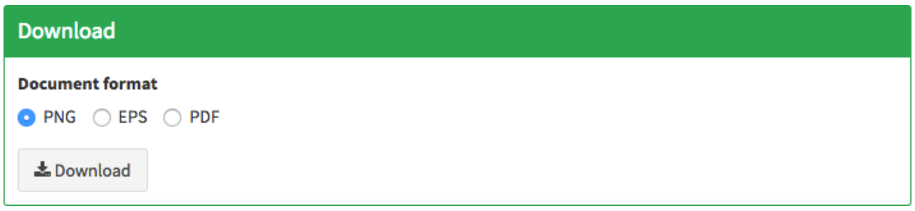


Figure 8: Output of a full informed and full noise investor with a lower alpha value.

After the visualization of the dataset, each page contains a download area, where the generated output can be downloaded in various formats.



Download

Document format

[✓ PNG] EPS PDF

Download

Figure 9: Download box of the Visualization page.

As Figure 9 shows, the following formats can be selected to save the output:

- PNG (Portable Network Graphics)
- EPS (Encapsulated Postscript)
- PDF (Portable Document Format)

The download area was generated as before with the *box()* function that includes three radio buttons to select between the mentioned formats and furthermore a download button to save the plot. For saving the output the *ggsave()* function of the shiny package was used. It saves the last generated plot that is displayed within the Graph Box, and print it into a format of DIN A4 with a dpi of 600 for a higher resolution.

ui.R

```
radioButtons("formatVisu", "Document format",
   c("PNG"="png", "EPS"="eps", "PDF"="pdf"),

   inline = TRUE, selected = "png"),

   downloadButton("downloadVisu")
)
```

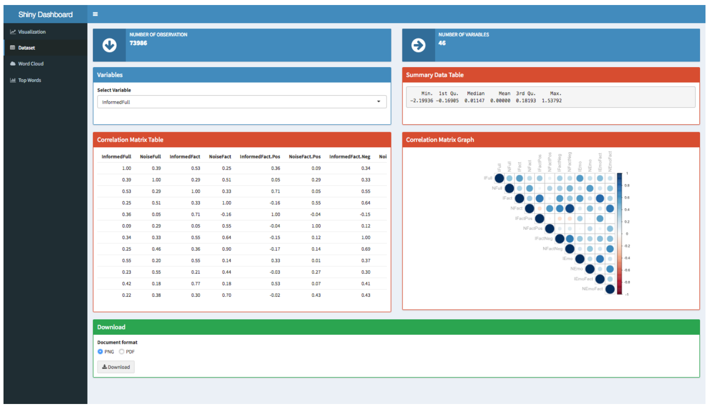
server.R

```
fn <- reactive({paste(parts$partC,input$formatVisu,sep = ".")})
d <- reactive({input$formatVisu})

output$downloadVisu <- downloadHandler(
   filename = fn,
   content = function(file) {

   ggsave(file, device=d(), dpi = 600, width = 297, height = 210, units = "mm")
})
```

In order to generate high-resolution graphics, the selection of the EPS format is recommended as one of the key features and benefits of vector graphics is infinite and lossless scalability.

2. Dataset (https://hadichami.shinyapps.io/Shiny_Dashboard/)

The Dataset page is an extension of the seminar paper and is used to illustrate the structure and composition of the dataset. It shows further possibilities which are realizable with the shiny package and in addition to the graphical output, it also displays a data table and statistics of the selected observation. Figure 10 shows the content of the Dataset page.



```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-2.19936 -0.16905  0.01147  0.00000  0.18193  1.53792
```

InformedFull	NoiseFull	InformedFact	NoiseFact	InformedFact.Pos	NoiseFact.Pos	InformedFact.Neg	Noi
1.00	0.39	0.53	0.25	0.36	0.09	0.34	
0.39	1.00	0.29	0.51	0.05	0.29	0.33	
0.53	0.29	1.00	0.33	0.71	0.05	0.55	
0.25	0.51	0.33	1.00	-0.16	0.55	0.64	
0.36	0.05	0.71	-0.16	1.00	-0.04	-0.15	
0.09	0.29	0.05	0.55	-0.04	1.00	0.12	
0.34	0.33	0.55	0.64	-0.15	0.12	1.00	
0.25	0.46	0.36	0.90	-0.17	0.14	0.69	
0.55	0.20	0.55	0.14	0.33	0.01	0.37	
0.23	0.55	0.21	0.44	-0.03	0.27	0.30	
0.42	0.18	0.77	0.18	0.53	0.07	0.41	
0.22	0.38	0.30	0.70	-0.02	0.43	0.43	

[✓ PNG] PDF

Figure 10: Output of the Dataset page.

At the beginning, two *infobox()* functions give a brief overview about the number of observations and the number of variables of the dataset. The function can be used to display numbers or also a short text with minimal information. Thereafter, a statistical calculation within the *summary()* function was added. For any selected variable the following six numbers are calculated: Minimum, First quartile, Median, Third quartile, Maximum. To analyze the relationship between different variables, a correlation matrix has been added in a table form and also as a graphical output. The calculation of the correlation matrix can be done by the use of the *cor()* function.

```
m <- cor(OLS.Data[,c(1:12)])
```

And be plotted by the *corrplot()* function.

```
corrplot(m,method = "circle",type = "upper",tl.col = "darkgrey")
```

The output of the correlation matrix can be downloaded in different formats (png, pdf).

3. Word Cloud (https://hadichami.shinyapps.io/Shiny_Dashboard/)

The Word Cloud page is a further extension of the seminar paper. A word cloud is a visual representation of text data, based on single words and their importance and frequency. As more important a word is, as bigger - based on the font size - it is shown in the word cloud. Furthermore, not only the font size, but also different colors are used to demarcation between the words. Figure 11 shows the content of the Word Cloud page.

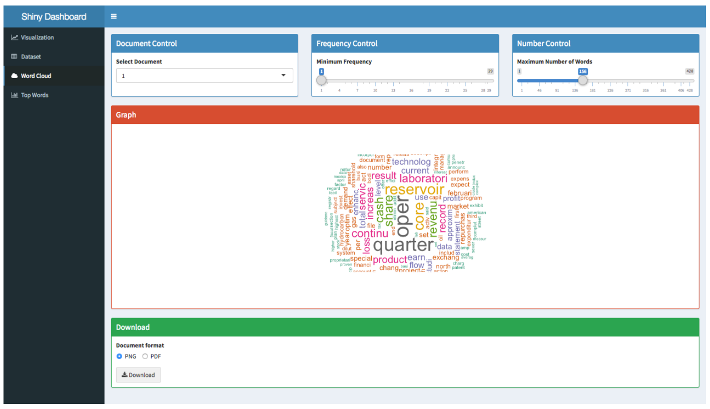


Figure 11: Output of the Word Cloud page.

The Word Cloud page is built with three widget controls - Document Control, Frequency Control and Number Control. With the Document Control a selecting between different documents in the dataset can be done. In order to not affect the speed of the shiny application, the dataset has been reduced from 33647383 observation to 8758, which exactly corresponds to 20 documents. Beside that the Frequency Control can be used to slider between the desired frequency that is attributed to a word. With every selected document, the maximal number of the frequency slider is automatically updated with the maximal frequency of the word within this document. The same procedure was also implemented on the Number Control widget. It is used to slide between desired number of words. The maximal words value is also automatically updated within the selected document.

To create and plot a word cloud in shiny the *wordcloud()* function can be used. Within the function different arguments can be used to determine the representation, color, scale, frequency and much more.

```
filtered <- reactive({
   Wcloud.Data.filtered <- Wcloud.Data %>%
   filter(document == input$doc)
})

output$plotWcloud <- renderPlot({

   wordcloud(words = filtered()$term,
   freq = filtered()$count,
   min.freq = input$minFreq,
   max.words = input$maxNum,
   scale=c(3.5,0.25),
   random.order=FALSE,
   rot.per=0.35, colors=brewer.pal(8, "Dark2"))
})
```

As we see in the code, different arguments can be used to make the output more useful and clearer. In order to print a large number of words, the scale argument was used, which is responsible for the scaling of the words. The result is illustrated in Figure 12.

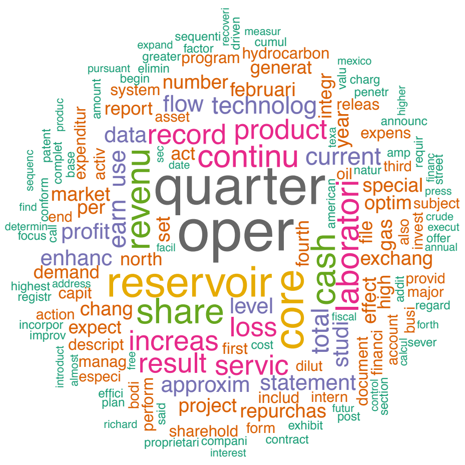


Figure 12: Word Cloud plot of document 1.

Figure 12 shows, that the combination of font size and the color describes a lot about the data. The bigger the word and darker the color, the higher the frequency of the word within the document.

If desired, the wordcloud plot can be downloaded in different formats (png, pdf).

4. Top Words (https://hadichami.shinyapps.io/Shiny_Dashboard/)

The Top Words page is also a further extension of the seminar paper. It can be used to quickly see the top 15 positive and negative words between informed investor and noise trader. Figure 13 shows the content of the Top Words page.

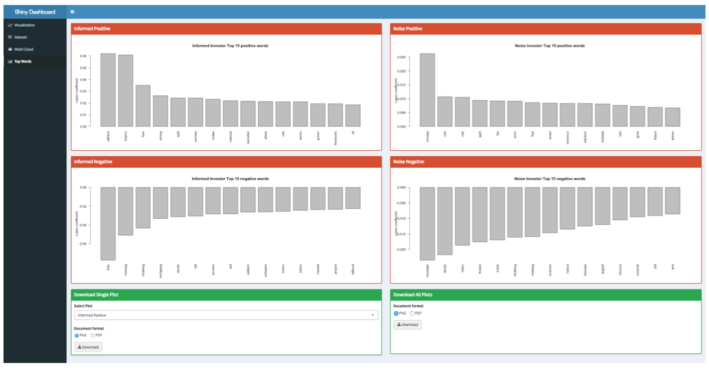


Figure 13: Output of the Top Words page.

The data for the visualization, which is created by using the *barplot()* function, was obtained from this paper [8]. The x-axis demonstrates the top 15 words, while the y-axis shows the lasso coefficient for each word and based on that the ranking is built.

```
output$myPlotTwords1 <- renderPlot({
   barplot(TopWords$lassoInfPos, las = 2, names.arg = TopWords$informedPos,
      main = "Informed Investor Top 15 positive words", ylab = "Lasso coefficient")})
```

A very useful feature is that the client can either choose to download one of the four plots or by clicking on the right download box, all of them. Also, as in the other pages of the shiny dashboard the use between the formats png and pdf can be selected.

```
par(mfrow=c(2,2))
   barplot(TopWords$lassoInfPos, las = 2, names.arg = TopWords$informedPos,
      main = "Informed Investor Top 15 positive words", ylab = "Lasso coefficient")
dev.off()
```

6 Critical Discussion

This section illustrates the advantages and disadvantages of the shiny package when creating web applications. There are limitations during the programming process as well as when starting the app.

Advantages	Disadvantages
Powerful Tool: Shiny is a powerful tool without any need of HTML, CSS or JavaScript skills.	***Restricted Design:*** Shiny offers limited possibilities in changing the design of the app.
Interactive web apps: Shiny makes it possible to create interactive web applications that automatically adjust to changes in the data.	***Limited Widgets:*** There are only limited available control widgets to manipulate the data.
Full access to R packages: During the work with shiny a full access to all other R packages is possible.	***Performance problem:*** When setting a filter within the shiny app, a so-called performance problem occurs. Hereby, the complete dataset is loaded first and then filtered. For larger datasets, this leads to long loading times. To avoid this, an if-else query can be set before the filter request.

Table 1: Advantages and Disadvantages of the shiny package.

7 Conclusion

The shiny package is a powerful tool for exploring interactive data visualization in R. With the help of shiny a quick development of HTML and JavaScript-based web application is possible. The potential application scenarios are manifold: reporting, deployment of statistical analyzes or interactive visualization of data stocks. When creating a Shiny application basically no programming knowledge is necessary because the complete programming of the app takes place directly in R. Only when customizing the application, that is, when customizing colors, logos, fonts and layouts, basic knowledge of HTML, CSS and JavaScript is required. Thus, shiny offers a really great opportunity for data analysts and non-R users to quickly generate output in an elegant and professional way. Shiny is under rapid development and is gaining popularity, resulting in more and more capabilities being added.

A References

[1] C. Beeley. *Web Application Development with R Using Shiny - Second Edition*. Community Experience Distilled. Packt Publishing, Limited, 2016.

[2] L. Bing. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers, 2012.

[3] F. Black. *Noise*. In: *Journal of Finance*, Vol. 41, No. 3 (1986), pp. 529–543.

[4] J. B. De Long et al. *Noise Trader Risk in Financial Markets*. In: *Journal of Political Economy*, Vol. 98, No. 4 (1990), pp. 703–738.

[5] E. F. Fama. *Efficient Capital Markets: A Review of Theory and Empirical Work*. In: *The Journal of Finance*, Vol. 2, No. 25 (1970), pp. 283–417.

[6] E. F. Fama. *The Behavior of Stock-Market Prices*. In: Vol. 38, No. 1 (1965), pp. 34–105.

[7] M. Fredman. *The case for flexible exchange rates*. In: Vol. 38, No. 1 (1953), pp. 34–105.

[8] F. Hannemann et al. *Noise Trader Behavior - A disaggregated approach to understanding news reception in financial markets*. In: *Twenty-Sixth European Conference on Information Systems (ECIS2018)* (2018), pp. 1–16.

[9] A. Khadjeh Nassirtoussi et al. *Text mining for market prediction: A systematic review*. In: *Expert Systems with Applications*, Vol. 41, No. 16 (2014), pp. 7653–7670.

[10] C. M. Lee, A. Shleifer, and R. H. Thaler. *Investor Sentiment and the Closed-End Fund Puzzle*. In: *Journal of Finance*, Vol. 46, No. 1 (1991), pp. 75–109.

[11] T. Li Im et al. *Analysing market sentiment in financial news using lexical approach*. In: *2013 IEEE Conference on Open Systems (ICOS)* (2013), pp. 145–149.

[12] T. Loughran and B. McDonald. *Textual Analysis in Accounting and Finance: A Survey*. In: *Journal of Accounting Research*, Vol. 54, No. 4 (2016), pp. 1187–1230.

[13] T. Loughran and B. McDonald. *The Use of Word Lists in Textual Analysis*. In: *Journal of Behavioral Finance*, Vol. 16, No. 1 (2015), pp. 1–11.

[14] W. Medhat, A. Hassan, and H. Korashy. *Sentiment analysis algorithms and applications: A survey*. In: *Ain Shams Engineering Journal*, Vol. 5, No. 4 (2014), pp. 1093–1113.

[15] J. R. Nofsinger. *The Psychology of Investing*. Pearson series in finance. Routledge, 2017.

[16] W. G. Schwert. *Anomalies and Market Efficienc*. In: Vol. 1 (2002), pp. 939–974.

[17] A. Shleifer and L. H. Summers. *The Noise Trader Approach to Finance*. In: *Journal of Economic Perspectives*, Vol. 4, No. 2 (1990), pp. 19–33.

[18] J. Wojciechowski, A. Hopkins, and R. Upton. *Interactive Pharmacometric Applications Using R and the Shiny Package*. In: *CPT: Pharmacometrics and Systems Pharmacology*, Vol. 4, No. 3 (2015), pp. 146–159.

B List of Figures

1 System architecture for the shiny application. 2
2 The header part of the shiny dashboard. 7
3 The sidebar part of the shiny dashboard. 7
4 The body part of the shiny dashboard. 8
5 Output of the Visualization page. 9
6 Output of a full informed and full noise investor. 11
7 Output of a full informed and full noise investor with a fixed date range. 11
8 Output of a full informed and full noise investor with a lower alpha value. 12
9 Download box of the Visualization page. 12
10 Output of the Dataset page. 13
11 Output of the Word Cloud page. 14
12 Word Cloud plot of document 1. 15
13 Output of the Top Words page. 15

C List of Tables

1 Advantages and Disadvantages of the shiny package. 16