Universität Ulm
Fakultät für Mathematik und
Wirtschaftswissenschaften

Do Expert Reviews Affect the Demand for Wine
An interactive RTutor Problem Set

Bachelorarbeit
in Wirtschaftswissenschaften

vorgelegt von
Hagedorn, Sascha
(Matrikelnummer 803557)
am 02.08.2016

Gutachter

Prof. Dr. Sebastian Kranz

Table of Contents

Problem Set Do Expert Reviews Affect the Demand for Wine

Author: Sascha Hagedorn

Exercise 1 Introduction

Thank you for taking the time and giving my problem set a chance. I would like to start with a short introduction about the main question.

In this paper **Do Expert Reviews Affect the Demand for Wine?** Richard Friberg and Erik Grönqvist examine if reviews effect the demand of wine by using the Swedish wine market. They use five years of weekly data on sales, advertising and expert reviews. The public data as well as the article are provided on the website of the ***American Economic Association***. You can simply click ***here*** to download it.

So this question should give you a good imagination about the topic. To get a better understanding, let us take a look at the conditions we have and about the market that we use to examine the effects.

The paper uses the Swedish market because it has certain conditions that make the estimation possible. First of all, we have an institutional setting and secondly, we have really detailed data about the market, which is obviously necessary to do such an analysis. We take the reviews of the six leading print media into account (Year 2002 - 2007). The reviews are getting combined with the weekly sales volumes and advertising expenditures for some weeks.

The regular assortment of all the Swedish market consist of 500 different wines. The definition of a wine depends on the producer, grape varietal, the container size and any other indication on the label. It happens that some just differ in terms of the container (bottle, bag-in-box).

This problem set leads you through different exercises and chapter that uses different statistical methods and calculations in order to explore the data itself and the impact of reviews, good reviews, bad reviews and advertising expenditures. We are also going to see that it depends on the media how impactful a review is. Furthermore, we are going to examine reviews that are referring to certain categories, for example wines that are in a medium or high price category.

It is important to mention, that obviously some reviews might be corrupted. That is a reasonable thought. For example, informer that have no economic benefit from their reviews have more influence on the customer. Consumers might also buy certain wines because of herding behavior. They choose a product that many people buy. This even happens to customers that have signals for better fits. Therefore, a lot of customers get locked into the certain products. The cause is mostly lack of information and this leads to the following of other's opinions.

However, Customer-to-customer information have an effect on demand for experience goods (Chevalier and Mayzlin 2006, sales rank data of books on Amazon[1]). This indicates that expert reviews might be impactful, therefore the estimations are worth the effort.

[1] Chevalier, Judith A., and Dina Mayzlin. 2006. “The Effect of Word of Mouth on Sales: Online Book Reviews.” Journal of Marketing Research, 43 (3) : 345–54.

Problemset

The problem set contains descriptive statistics and quite a lot of regressions. A lot of these regressions are very similar to each other. This is kind of obvious because the main purpose is the same, but if you take a closer look at each regression, then it is very interesting.

The problem set uses different data sets. Most of the data preparation is not going to be part of the exercises. In case you are interested, you can take a look at exact code of the preparation in the appendix.

The problem set main tool are little code chunks that you have to solve. With every new chunk, you will discover new results and gather more and more interesting interpretations and in the end of the set, you can call yourself a Swedish whine market expert. Isn't that exciting?

Your task is to edit every chunk with the edit button. Afterwards, you can check you answer with the check button if your code was correct. Some chunks are going to be optimal, because they explain an alternative method or are just for code showing purposes. It will always say if the chunk is optional. Most of the exercises will start with loading the corresponding data set. It is a bot annoying but necessary, just check the chunk and proceed.

Chapters & Exercises

- **1. Introduction**
- **2. Data Overview**
- **3. Shares of the distribution levels**
- **4. Descriptive statistics across wines**
- **5. Price changes**
- **6. Wine Reviews in Swedish Print Media**
- **7. Insightful Plots**
- **8. The Empirical Model**
- **9. Effects of Reviews and Advertising on Demand for Wine**
- **10. AoM Wines**
- **11. Categorical impact of Reviews**
- **12. Conclusion**
- **13. References**

So let's get started. Click on *Go to next exercise...* to start and later to continue.

Exercise 2 Data overview

a) Load & sort the data

We have to load the data into our working space in order to be able to work with the data.

The data is stored as a RDS file. RDS files are single R object connections (usually a file) where R objects are stored. We can just load the data with the readRDS() function. Name the data set data.

The function just requires the RDS filename, meaning readRDS(file=filename). For more information regarding RDS files and function click on the info button.

Info: readRDS()

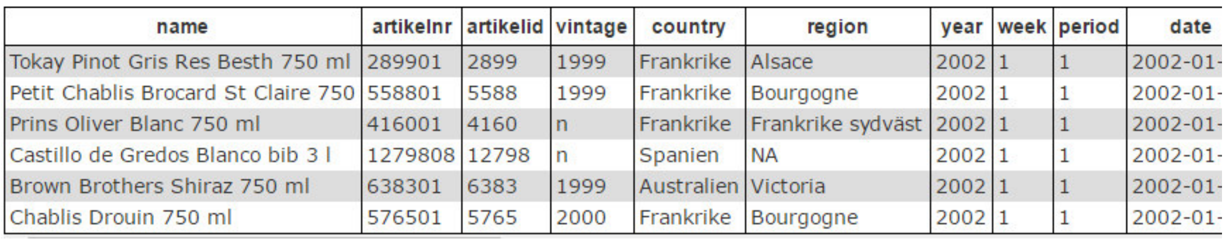
These functions provide the means to save a single R object to a connection (typically a file) and to restore the object, quite possibly under a different name. This differs from save and load, which save and restore one or more named objects into an environment. More information in the source link.

Source: Website of Inside-r.org

```
# Use readRDS() to load the datanonexp into the workspace.
data <- readRDS(file="datadta.RDS")
```

Now let's take a look at the first rows of the data set. Use the head() function to display the first 5 rows.

```
# Display the first 5 rows with the head() function.
head(data)
```



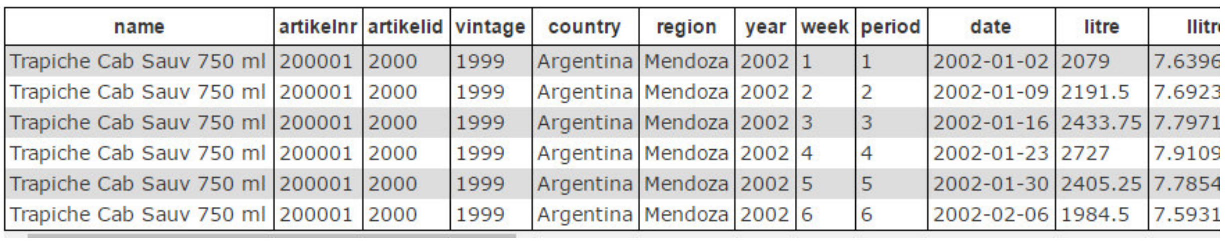
name	artikelnr	artikelid	vintage	country	region	year	week	period	date
Tokay Pinot Gris Res Besth 750 ml	289901	2899	1999	Frankrike	Alsace	2002	1	1	2002-01-
Petit Chablis Brocard St Claire 750	558801	5588	1999	Frankrike	Bourgogne	2002	1	1	2002-01-
Prins Oliver Blanc 750 ml	416001	4160	n	Frankrike	Frankrike sydväst	2002	1	1	2002-01-
Castillo de Gredos Blanco bib 3 l	1279808	12798	n	Spanien	NA	2002	1	1	2002-01-
Brown Brothers Shiraz 750 ml	638301	6383	1999	Australien	Victoria	2002	1	1	2002-01-
Chablis Drouin 750 ml	576501	5765	2000	Frankrike	Bourgogne	2002	1	1	2002-01-

We finished loading the first dataset into the workspace. As you can see the data is not really sorted. We want to sort the data by artikelnr and period. We need to sort by period because we have data over time.

The new data frame is supposed to have the name datasort. We can use the arrange() function in order to sort a data frame. We want to order by the artikelnr & period. We can just separate the sorting criteria by comma. The arrange() function belongs to the dplyr package. This package has a lot of powerful functions that we will use later on. The package also introduced the operator %>% that creates a chain. You can find more information to the package under the following link:

- Introduction to dplyr
- Cran R Project PDF all functions explained

```
# Use arrange() to sort the data.
# Assign the new data frame to datasort.
datasort <- data %>%
  arrange(artikelnr, period)
head(datasort)
```



name	artikelnr	artikelid	vintage	country	region	year	week	period	date	litre	llitr
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	1	1	2002-01-02	2079	7.6396
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	2	2	2002-01-09	2191.5	7.6923
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	3	3	2002-01-16	2433.75	7.7971
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	4	4	2002-01-23	2727	7.9109
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	5	5	2002-01-30	2405.25	7.7854
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	6	6	2002-02-06	1984.5	7.5931

The data is now sorted by the artikelnr and period.

This exercise is pretty basic and just want you to understand that it is important to take a first look at the data and then sort it if necessary. After we sorted the data, it also got expanded in the preparations. We are not going to do this here, because the problem set should not have too much data preparation tasks, however it is important to understand that data preparation is a big part of data analysis. It is often not the norm to have structured data. For example, the mentioned expanding is important because the base data that we have, has period gaps, so it is important to fill the gaps even though it is just placeholder data.

b) The rest of the datasets

This problem set uses different data sets that got prepared differently due to their purposes. You can look up the data preparations in a different file.

Click the info button for short explanations regarding every data set.

- datadta Original data set
- datasort Original sorted data set
- datafilter Expanded (`periods`) and filtered original data set. Observations that have the value NA for either any explanatory-, dependent-, lead variables or dist will be filtered out. Also observations that do not fulfil the requirement dist > 4. These filter get explained later on.
- datacc Datasort with country names changed to the english spelling.
- datafilterreg Expanded (`periods`) and filtered original data set with lagged variables. Filter checks for NA in the lead variables and dist. Also dist > 4 got checked. It is the base data set for the regressions.
- dataaom Same as datafilterreg just with different lagged variables.

The filters and components is going to be explained throughout the problem set. It is just an overview, in case it is too confusing, just skip it.

The filters are going to be explained throughout the problem set.

c) Data investigation

We already took a first look and the data and decided to sort it, because we have data over time. The next recommended step would be to take a deeper look at the data to have an even better understanding. Well, it is clearly important to know all the data to actually be able to analyze it. So first let us take a look at the column names, most of them will give us directly an understanding what the column is about. However, some Colum names aren’t that obvious because they use certain acronyms or syntax.

Use the colnames() command to see the name of the columns.

```
# Use colnames() to take a closer look.
colnames(datasort)

##  [1] "name"       "artikelnr"  "artikelid"
##  [4] "vintage"    "country"    "region"
##  [7] "year"       "week"       "period"
## [10] "date"       "litre"      "llitre"
## [13] "price"          "lp"             "rprice_litre"
## [16] "dist"           "taste_segment"   "segm"
## [19] "price_segm"      "time_segm_price" "artikpr"
## [22] "old"            "ma_split"        "v10_a"
## [25] "v10_dn"         "v10_di"          "v10_exp"
## [28] "v10_svd"        "v10_aom"         "v10_am"
## [31] "v10_dnm"        "v10_dim"         "v10_expm"
## [34] "v10_svdm"       "v10_aomm"        "v10_all"
## [37] "rev_all"        "rev_all_hi"      "rev_all_lo"
## [40] "rev_eve"        "rev_eve_hi"      "rev_eve_lo"
## [43] "rev_ex"         "rev_ex_hi"       "rev_ex_lo"
## [46] "rev_nyaom"      "rev_nyaom_hi"    "rev_nyaom_lo"
## [49] "rev_all_p50"    "rev_all_p80"     "rev_all_p20"
## [52] "m_rev"          "m_rev_hi"        "m_rev_lo"
## [55] "nrarom"         "pri_m"           "ms_segm"
## [58] "ind"
```

Now, we can see all the column names. Most of the columns are pretty much self-explained. The important variables are the v10_ variations and espeically the rev_ variables. The v10_s are the normalized review grades of the different print medias. The rev_s are the indicators of a wine being reviewed. There are different indicators. Some indicate a good review, a review from a tabloid or even combinations.

Click on the "Info" button for variable descriptions.

Info: Variables

- name = Name of wine
- artikelnr = ID number of wine (given by Systembolaget)
- artikelid = ID number of brand (given by Systembolaget)
- vintage = Vintage (if any)
- country = Country
- region = Region
- year = Year
- week = Week no. (1-53)
- period = Weekly time indicator (1-263)
- date = Date
- litre = Weekly sale in litres
- llitre = Weekly sale in log litre
- price = Price in SEK
- lp = Log price in SEK
- rprice_litre = Real litre price in SEK (base Jan 2004)
- dist = Level of distribution of wine
- taste_segment = Taste segment of wine (16 groups)
- segm = Color segment of wine
- price_segm = Price segment of wine (high, medium, low, bib)
- time_segm_price = Period-color-price segment-package indicator
- artikpr = Product number-price-vintage combination
- old = Indicator for the wine being distributed longer than the previous 2 years
- ma_split = Advertising expenditures a specific week for a wine

- v10_a = Normalized review (0-10): Aftonbladet
- v10_dn = Normalized review (0-10): Dagens Nyheter v10_di Normalized review (0-10): Dagens Industri
- v10_exp = Normalized review (0-10): Expressen
- v10_svd = Normalized review (0-10): Svenska Dagbladet
- v10_aom = Normalized review (0-10): Allt om Mat (AoM)
- v10_am = Mean normalized review during the weeks the wine is distributed: Aftonbladet
- v10_dnm = Mean normalized review during the weeks the wine is distributed: Dagens Nyheter
- v10_dim = Mean normalized review during the weeks the wine is distributed: Dagens Industri
- v10_expm = Mean normalized review during the weeks the wine is distributed: Expressen
- v10_svdm = Mean normalized review during the weeks the wine is distributed: Svenska Dagbladet
- v10_aomm = Mean normalized review during the weeks the wine is distributed: Allt om Mat
- v10_all = Weekly average normalized review (all media)
- rev_all = Indicator of the wine being reviewed (all media)
- rev_all_hi = Indicator of the wine receiving a good review (all media)
- rev_all_lo = Indicator of the wine receiving a bad review (all media)
- rev_eve = Indicator of the wine being reviewed in tabloids
- rev_eve_hi = Indicator of the wine receiving a good review in tabloids
- rev_eve_lo = Indicator of the wine receiving a bad review in tabloids
- rev_ex = Indicator of the wine being reviewed in AoM
- rev_ex_hi = Indicator of the wine receiving a good review in AoM
- rev_ex_lo = Indicator of the wine receiving a bad review in AoM
- rev_nyaom = Indicator of review is not in AoM yearly special
- rev_nyaom_hi = Indicator of good reviews not in AoM yearly special
- rev_nyaom_lo = Indicator of bad reviews not in AoM yealy special
- rev_all_p50 = Indicator of the wine receiving a higher than median review
- rev_all_p80 = Indicator of the wine recepving a higher than p80 review
- rev_all_p20 = Indicator of the wine receiving a lower than p20 review
- m_rev = Indicator of the wine receiving multiple reviews in a week
- nrarom = Number of wines reviewed in Allt om Mat each week
- pri_m = Mean real litre price in SEK (base jan 2004) during the weeks the wine is distributed
- ms_segm = Mean market share within color during weeks the wine is distributed
- ind = Indicator for the first time the wine is observed in data

There are certain variables in the analysis that show if a certain condition is true. These are called dummy variables and have the value one or zero.[2]

---

[2] Richard Friberg and Erik Grönqvist (2012). Readme file for Do expert reviews affect the demand for wine? page 1, 2

Info: Self-created dummy variables

- rev_all_lag_x = The indicator rev_all lagged by x
- rev_all_lead_x = The indicator rev_all led by x
- rev_all_hi_lag_x = The indicator rev_all_hi lagged by x
- rev_all_hi_lead_x = The indicator rev_all_hi led by x
- rev_all_lo_lag_x = The indicator rev_all_lo lagged by x
- rev_all_lo_lead_x = The indicator rev_all_lo led by x
- ma_split_lag_x = The indicator ma_split lagged by x
- ma_split_lead_x = The indicator ma_split led by x
- maind = The indicator of advertising expenditures
- indcross = The indicator for a new artikelnr
- pdum = The indicator of a price change
- medhigh = The indicator of the wine category medium or high price
- newworld = The indicator of the wine category new world
- ma_brands = The indicator of the wine category brand with marketing
- legal_m = The indicator of the wine category marketing became legal
- new = The indictaor of a wine being new
- vint_brands = The indicator of a wine being a vintage
- highvar_q = The indicator of a wine from one of the high variability regions
- Review = The indicator of the wine receiving a review
- GoodReview = The indicator of the wine receiving a good review
- BadReview = The indicator of the wine receiving a bad review Every category has a Review, GoodReview and a BadReview variable
- categories: medhigh, newworld, ma_brands, legal_m, new, vint_brands, highvar_q and tabloid.

A dummy variable is an artificial variable constructed such that it takes the value unity whenever the qualitative phenomenon it represents occurs, and zero otherwise. Once created, these proxies, or "dummies" as they are called, are used in classical statistical methods.[3]

Info: Calculated Variables

- Advertising_Expenditure = The mean of advertising expendituress (ma_split) during weeks the wine is distributed
- AdvertExpend = Same as Advertising_Expenditure just with NA instead of zero in order to ignore the zeros
- Price_per_liter = The mean of real litreprice (rprice_litre) during weeks the wine is distributed
- Liters_per_week = The mean of litres (litre) sold during weeks the wine is distributed
- lagprice = price lagged by 1
- lagartikelnr = artikelnr lagged by 1
- pchange & pchangefinal = Price change between two observations (price - lagprice)

[3] Peter Kennedy, A Guide to Econometrics, 6th edition Wiley-Blackwell, 2008

- achange = Indicates change of a artikelnr (artikelnr - lagartikelnr)
- nrpchange = The sum of pdum
- v10_all_hi = The value of v10_all where v10_all is larger than 7.999 and not NA
- v10_all_lo = The value of v10_all where v10_allis smaller or equal 4

There are variables that are used to store certain calculations.

One example would be the pchange variable that shows if a price change occurred in the given week.

You can also use the str() command to get a really detailed look at the data. The str() command stands for structure and basically gives us the structure of the data frame. It gives us the amount of objects and variables. It also lists the variable names, types and the first couple of values for each column. Some other information will be provided but these are not relevant for our problem set.

```r
# Use the str() to take a closer look at datasort.
str(datasort)
```

```
## 'data.frame':    193574 obs. of  58 variables:
##  $ name           : chr  "Trapiche Cab Sauv 750 ml" "Trapiche Cab Sauv 750 ml" "Trapiche Cab Sauv 750 ml" "Trapiche Cab Sauv 750 ml" ...
##  $ artikelnr      : int  200001 200001 200001 200001 200001 200001 200001 200001 200001 200001 ...
##  $ artikelid      : int  2000 2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
##  $ vintage        : chr  "1999" "1999" "1999" "1999" ...
##  $ country        : chr  "Argentina" "Argentina" "Argentina" "Argentina" ...
##  $ region         : chr  "Mendoza" "Mendoza" "Mendoza" "Mendoza" ...
##  $ year           : num  2002 2002 2002 2002 2002 ...
##  $ week           : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ period         : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ date           : Date, format: "2002-01-02" "2002-01-09" ...
##  $ litre          : num  2079 2192 2434 2727 2405 ...
##  $ llitre         : num  7.64 7.69 7.8 7.91 7.79 ...
##  $ price          : num  64 64 64 64 64 64 64 64 64 64 ...
##  $ lp             : num  4.41 4.41 4.41 4.41 4.41 ...
##  $ rprice_litre   : num  82.5 82.5 82.5 82.5 82.5 ...
##  $ dist           : num  4 4 4 4 4 4 4 4 4 4 ...
##  $ taste_segment  : chr  "Fruktiga & Smakrika" "Fruktiga & Smakrika" "Fruktiga & Smakrika" "Fruktiga & Smakrika" ...
##  $ segm           : chr  "red" "red" "red" "red" ...
##  $ price_segm     : chr  "l" "l" "l" "l" ...
##  $ time_segm_price: num  3 16 29 42 55 68 81 94 107 120 ...
##  $ artikpr        : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ old            : chr  "1" "1" "1" "1" ...
##  $ ma_split       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ v10_a          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_dn         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_di         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_exp        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_svd        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_aom        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_am        : num  7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 ...
##  $ v10_dnm       : num  7.08 7.08 7.08 7.08 7.08 ...
##  $ v10_dim       : num  7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 7.5 ...
##  $ v10_expm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_svdm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ v10_aomm      : num  4 4 4 4 4 4 4 4 4 4 ...
##  $ v10_all       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ rev_all       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_all_hi    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_all_lo    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_eve       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_eve_hi    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_eve_lo    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_ex        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_ex_hi     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_ex_lo     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_nyaom     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_nyaom_hi  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_nyaom_lo  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_all_p50   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_all_p80   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ rev_all_p20   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ m_rev         : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ m_rev_hi      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ m_rev_lo      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ nrarom        : num  0 0 0 2 0 0 0 0 22 0 ...
##  $ pri_m         : num  90.9 90.9 90.9 90.9 90.9 ...
##  $ ms_segm       : num  0.000709 0.000709 0.000709 0.000709 0.000709 ...
##  $ ind           : num  NA NA NA NA NA NA NA NA NA NA ...
##  - attr(*, "datalabel")= chr ""
##  - attr(*, "time.stamp")= chr "12 May 2011 00:36"
##  - attr(*, "formats")= chr  "%35s" "%10.0g" "%10.0g" "%9s" ...
##  - attr(*, "types")= int  35 253 253 4 22 23 254 254 254 254 ...
##  - attr(*, "val.labels")= chr  "" "" "" "" ...
##  - attr(*, "var.labels")= chr  "" "artikelnr" "" "" ...
##  - attr(*, "expansion.fields")=List of 18
##   ..$ : chr  "_dta" "iis" "artikelnr"
##   ..$ : chr  "_dta" "tis" "period"
##   ..$ : chr  "_dta" "_TSitrvl" "1"
##   ..$ : chr  "_dta" "_TSdelta" "+1.0000000000000X+000"
##   ..$ : chr  "_dta" "_TSpanel" "artikelnr"
##   ..$ : chr  "_dta" "_TStvar" "period"
##   ..$ : chr  "artikelnr" "destring" "Characters removed were:"
##   ..$ : chr  "artikelid" "destring" "Characters removed were:"
##   ..$ : chr  "artikelnr" "tostring" "converted to string"
##   ..$ : chr  "_dta" "ReS_i" "artikelnr vecka year volym"
##   ..$ : chr  "_dta" "ReS_ver" "v.2"
##   ..$ : chr  "_dta" "ReS_j" "karaktärtyp"
##   ..$ : chr  "_dta" "ReS_str" "1"
##   ..$ : chr  "_dta" "ReS_Xij" "karaktärbeskrivning"
##   ..$ : chr  "artikelid" "tostring" "converted to string"
##   ..$ : chr  "week" "destring" "Characters removed were:"
##   ..$ : chr  "year" "destring" "Characters removed were:"
##   ..$ : chr  "old" "tostring" "converted to string"
##  - attr(*, "version")= int 12
```

As you can see, the name has the type chr, artikelnr is an integer and period is from type num. This is not really surprising.

Before we proceed to actually do some statistics, let us take another look, in fact let us take a look at the important variables in this data set. We know that the article is about the impact of the reviews and advertising expenditures. It is pretty obvious what a varibale with advertising expenditures looks like. But the situation about the reviews is a little bit different. We will just select certain columns since our data set is quite big.

```
# Displaying statsort with head().
head(datasort)
```

name	artikelnr	artikelid	vintage	country	region	year	week	period	date	litre
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	1	1	2002-01-02	2079
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	2	2	2002-01-09	2191.
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	3	3	2002-01-16	2433.
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	4	4	2002-01-23	2727
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	5	5	2002-01-30	2405.
Trapiche Cab Sauv 750 ml	200001	2000	1999	Argentina	Mendoza	2002	6	6	2002-02-06	1984.

```
# Selecting certain columns for better readability.
datasample <- datasort %>%
  select(name,period,year,week,v10_di,v10_aom,rev_all,rev_all_hi,rev_all_lo) %>%
  # Filtering for rev_all_hi == 1 to show the column meaning.
  filter(rev_all_hi == 1)
# Displaying the datasample.
head(datasample)
```

name	period	year	week	v10_di	v10_aom	rev_all	rev_all_hi	rev_all_lo
Trapiche Cab Sauv 750 ml	197	2005	40	7.5	10	1	1	0
Douglas Green Pinotage 750 ml	234	2006	25	NA	NA	1	1	0
Bellingham Shiraz 750 ml	41	2002	41	NA	10	1	1	0
Bellingham Shiraz 750 ml	47	2002	47	NA	NA	1	1	0
Bellingham Shiraz 750 ml	125	2004	21	NA	NA	1	1	0
Bellingham Shiraz 750 ml	147	2004	43	NA	NA	1	1	0

The result gives us more knowledge about the structure of the data set.

You can see in the first sample that we have weekly data. he week column represents the week in the given year, thus starting from 1 every new year. The column period however goes up to 261 which represents all weeks stacked up from the data period year 2002 - the first two weeks of 2007. this one period is also one week.

The filtered second sample focuses on the review columns. We can see that the rev_ columns are dummies that represent a certain review, thus if it has the value 1 a review appeared in the given week. The v10_ columns give us the grade of the review.

So, you can see that the wine Trapiche Cab Sauv 750 ml got reviewed by the print media Dagens Industri and Allt om Mat in week 40 of year 2005. The grades are 7.5 and 10. Therefore, rev_all has the value 1 and indicates a review in the given week. If at least one review appears in any print media, then the indicator rev_all has the value 1. rev_all_hi has also the value 1, which indicates that a good review happened. rev_all_lo has the value 0 because no bad review was published. A grade is considered good if larger than 8 and bad if smaller or equal 4.

Now, let us investigate how many unique wines we have. Use the n_distinct() command from the dplyr package to get the amount of unique wines. We can use the data frame name and the name of the column to indicate the data that we want to take a look at. The artikelnr is the right choice here.

```
# Use n_distinct command to count the numbers of unique artikelnr
n_distinct(datasort$artikelnr)

## [1] 1162
```

The results show that we have 1162 different wines in the data set. However, not all of the wines are going to be used for estimations respectively statistics. This is going to be explained later.

This exercise showed and explained you the different variables, gave you an idea about the structure about the data set and did the first small analysis on how many different wines we have. Let's continue to the next exercise that will explain why we use a certain filter to limit the dataset.

Exercise 3 Shares of distribution level

a) Introduction

As already mentioned, the paper uses retail sales data from Jan 2002 - first two weeks of 2007. It takes red, white and sparkling wine (750 ml bottles and 3 l bag-in-box) into account. These package sizes account for more than 96 percent of the volume. There is just a state owned monopoly retailer called Systembolaget.

Systembolaget is the retailer for wine, spirits and beer with an alcohol % by volume larger than 3.5. Its responsibility is to minimize alcohol related health problems.

There are six distribution levels in the regular assortment for wine:

- Tier 1: all 420 stores
- Tier 2: 325 stores
- Others 195, 95, 45 stores

Which means that a wine that offered in all 420 stores, belongs to Tier 1.

The different shares clearly affect sales. The paper only takes attention to the first two tiers in order to have a better result. The results would not be consistent if all the wines would not be in most of the stores. The store coverage can only change twice every year with the spring & fall catalogue, what is also good for the estimation. The volumes are overall rather stable, with some cyclical patterns for some types. For example, sparkling wine has more sales during new years' eve and white wine is more popular during the summer, which is going to be addressed, but we will come later to that. So this as a first overview.

Among other preparations, we filtered out all observations that have the value of dist is larger than 4 and not NA. The reason for this is the above mentioned volume of the Tier 1 and 2 wines. So as the next exercises let us take a look at the distribution level and the volumes to understand why we filter out most the wines in the following exercises.

b) Shares in terms of amount

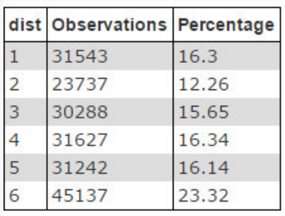
Let us take a look at the shares of distribution level in terms of the amount and the volume, the first step is to load the data, of course. Just check and proceed.

```
# Loading up datasort.
datasort <- readRDS(file="datasort.RDS")
```

Now, we need to group datasort with the group_by() function from the already used dplyr package. This package is very powerful and got already mentioned. It is going to be grouped by dist.

Next, we will summarize the distribution level by count and the percentage of the corresponding count. The functions summarise() and mutate() from the dplyr package also will help us there. The chain operator %>% will connect the different steps again. So we just need to pass the correct functions to the summarise() to get the desired result. It is important to mention that the summarise() function get red of all the variable, so after the summary you only will have the summary and the group variables at this point in the chain. The mutate() function adds new variables according to the given expressions and preserves the existing ones.

```
# Summarise datasort as follows:
# Name it sum1.
# Group the data by dist.
# Count the observations
# Calculate the percentage and round the result.
sum1 <- datasort %>%
  group_by(dist) %>%
  summarise('Observations' = n()) %>%
  mutate('Percentage' = round(Observations / sum(Observations)*100, digits = 2))
sum1
```



dist	Observations	Percentage
1	31543	16.3
2	23737	12.26
3	30288	15.65
4	31627	16.34
5	31242	16.14
6	45137	23.32

```
sum(sum1$Percentage[5:6])

## [1] 39.46
```

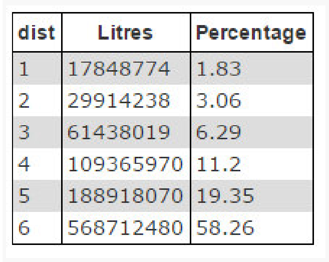
The results show that the observations of Tier 1 and Tier 2 account for approximately 40% in terms of amount. This is definitely a quite big amount, since there are 6 distribution levels. Tier 1 even being the biggest amount with ~ 23.32%.

c) Shares in terms of volume

Now let us take a look at the volumes (litres per week). Summarize by dist again. Here it is important that you use the command na.omit() because we need to exclude all observations with the value NA for litre. Also, in case there is even one NA in the sum function for a certain dist. The whole sum would be NA. So it is always recommendable to use na.omit() when summing up.

So group by dist, then summarize 'Litres' as the sum of litre. Use na.omit() here. Then calculate the percentage, display the summary and calculate the addition of Tier 1 & 2 (dist > 4). Use the same syntax as above.

```
# Use the same syntax as above.
# Name the summary sum2.
# Group by dist.
# Summarise the sum of litre, name it Litres.
# Calculate the 'Percentage' with the mutate function.
# Make sure to round the percentages by 2 digits.
# Display sum2.
# Display the sum of tier 1 and 2.
sum2 <- datasort %>%
  group_by(dist) %>%
  summarise('Litres' = sum(na.omit(litre))) %>%
  mutate('Percentage' = round(Litres / sum(Litres)*100, digits = 2))
sum2
```



dist	Litres	Percentage
1	17848774	1.83
2	29914238	3.06
3	61438019	6.29
4	109365970	11.2
5	188918070	19.35
6	568712480	58.26

```
sum(sum2$Percentage[5:6])

## [1] 77.61
```

As you can see, the volume of litres of Tier 1 & 2 is even higher than just the share in amount. The volume of Tier 1 & 2 account of approximately 77%, which is not a big surprise since the wines of Tier 1 & 2 are offered in most of the stores. This is the reason why we only use the observations of Tier 1 & 2, this makes sure that we will have a better estimation.

Let’s take a look at some questions to make sure you understood the purpose of the exercise.

Quiz: On what variable are the results focusing on?

- Liters [ ]
- Percentage [ ]
- dist [x]
- Observations [ ]

Quiz: What is one condition that refers to the variable and is essential for the selection of the used wines in the paper?

- smaller than 2 [ ]
- smaller than 4 [ ]
- larger than 4 [x]

d) Liters over time

So it was mentioned, that the volumes are overall rather stable with some cyclical patterns. So let us create a line chart that shows the overall volume per year. We are going to create a special line chart with the googleVis package. Click the info button for some more information.

Info: GoogleVision

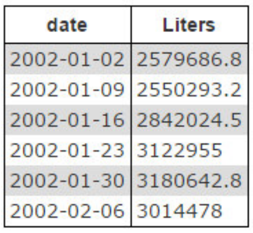
The package googleVis gives the ability to visualize the data in an interactive fashion in the form of a Google Chart. The corresponding function creates an html & javascript code referring to the Google Visualisation API that can be used in the browser or embedded in anything that can handle html & javascript. It usually surpasses normal R plots, and thus it is a great way to visualize and analyze the data.

Source: Cran R Project Link

We are going to use the gvisLineChart() function to create this special line chart. We just need to pass at least a data frame and the x and y variables. Further options are available, that you can also find in the link in the info section. We will just use the option to change some colors later on.

First we need to summarize the data to get a suitable data frame to pass by. We want to have a line chart that represents the liters over time. So we want to have a time variable as x and the liters as y. Therefore, we need to group the data by date and sum litre as Liters.

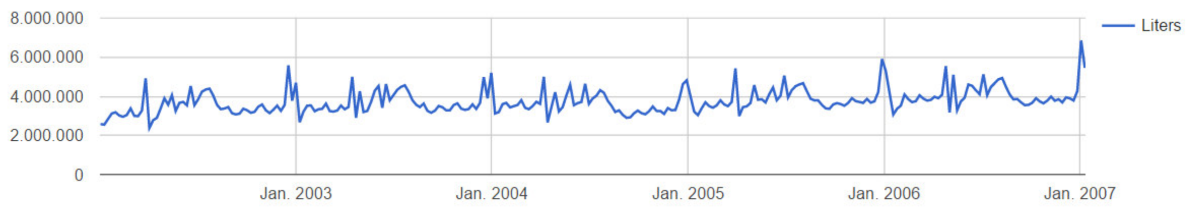
```
# Name the new dataframe datastable.
# Group by date.
# Summarise the sum of litre as Liters.
# We need na.omit() while summing up.
datastable <- datasort %>%
  group_by(date) %>%
  summarise(Liters = sum(na.omit(litre)))
head(datastable)
```



date	Liters
2002-01-02	2579686.8
2002-01-09	2550293.2
2002-01-16	2842024.5
2002-01-23	3122955
2002-01-30	3180642.8
2002-02-06	3014478

Perfect, now we can apply the gvisLineChart() function to create the corresponding object that we can pass to the plot() function in order to plot the special line chart.

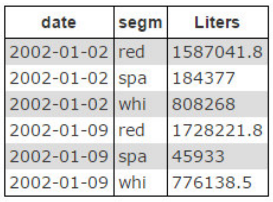
```
# Pass datastable to gvisLineChart.
# Name it lineplot1.
# Declare date as xar and Liters as yvar.
lineplot1 <- gvisLineChart(datastable, xvar ="date", yvar = "Liters")
# Pass lineplot1 to plot.
# Tag = chart.
plot(lineplot1, tag = "chart")
```



As you can see, the volume is quite stable at around a bit under 4000000 liters with some cyclical peaks. It is also noticeable that there is always a peak around January which suggests that these are the sales around Christmas and new year's eve. However, this chart just represents the total volume not the volume for each segment (red, white & sparkling). So let's create another one that shows the volume for each segment.

The syntax for the summary is the same as above. The difference is that we also group by segm though.

```
# Name the new dataframe datastablesegm.
# Group by date and segm
# Summarise the sum of litre as Liters.
# Display datastablesegm.
datastablesegm <- datasort %>%
  group_by(date, segm) %>%
  summarise(Liters = sum(na.omit(litre)))
head(datastablesegm)
```



date	segm	Liters
2002-01-02	red	1587041.8
2002-01-02	spa	184377
2002-01-02	whi	808268
2002-01-09	red	1728221.8
2002-01-09	spa	45933
2002-01-09	whi	776138.5

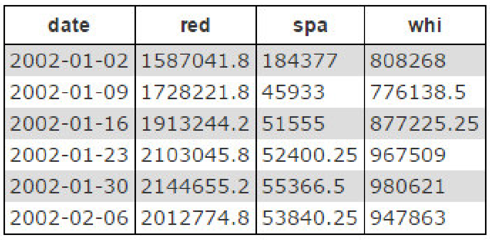
Perfect, the summary is done. But it seems that the structure of the data frame is not correct. We need to have the segments as columns with the corresponding Liters. We will use the function dcast() in order to get the right strucuture for plotting. dcast() is a function from the reshape2 package.

You can read more about the package under the following links:

Seananderson.ca Link An introduction to reshape2. Cran.r-project.org Link Package 'reshape2' PDF.

dcast() makes it possible to split segm into multiple columns. The dcast() function needs to have a data frame, a formula for the transposition and value variable for the variable that gets transposed.

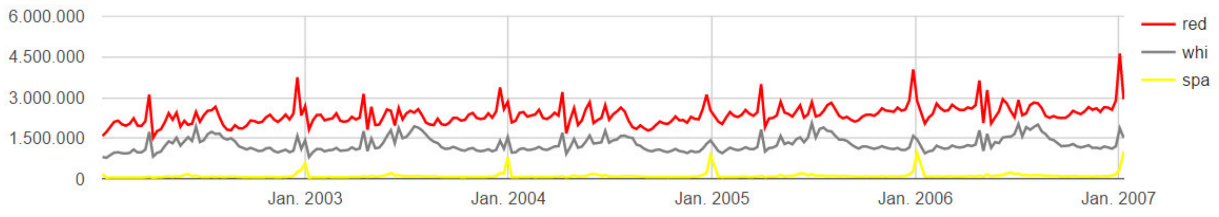
```
# Pass datastablesegm.
# Forumla: date ~ segm.
# value.var = Liters.
dcastsample <- dcast(datastablesegm, date ~ segm, value.var = 'Liters')
head(dcastsample)
```



date	red	spa	whi
2002-01-02	1587041.8	184377	808268
2002-01-09	1728221.8	45933	776138.5
2002-01-16	1913244.2	51555	877225.25
2002-01-23	2103045.8	52400.25	967509
2002-01-30	2144655.2	55366.5	980621
2002-02-06	2012774.8	53840.25	947863

Alright, we have the correct structure. So let's pass this data frame to the gvisLineChart as before. We are going to have 3 lines for each segment (red, white & sparkling) this time, meaning we need to pass 3 columns to yvar this time. We also going to define a color option for the segments. The syntax for the options is options=list(). So we just need to add the color option to the list as colors="['red', 'grey', 'yellow']" The colors can also be in the hexadecimal color format.

```
# Pass datastable to gvisLineChart.
# Name it lineplot2.
# xvar as date, yvar as c("red","whi","spa").
# options=list(colors="['red', 'grey', 'yellow']"))
lineplot2 <- gvisLineChart(dcastsample, xvar ="date", yvar = c("red","whi","spa"),
options=list(colors="['red', 'grey', 'yellow']"))
# Pass lineplot to plot.
# Declare tag = "chart".
plot(lineplot2, tag = "chart")
```



The line chart for the different segments is quite similar to the overall result. The volumes are rather stable and have cyclical peaks. We also have peaks around January every time. The volume for red wines is the highest, followed by the white wine. The sparkling wines have the least amount of volume. Very noticeable is that sparkling wines only have peaks around January and are very stable for the rest of the time. The reason for that is obviously new year's eve and also Christmas to some extent.

So this exercise showed is why it is reasonable to filter out most of the wines in the data and just estimate for tier 1 & 2 in order to have a better result. We also saw that the volume is quite stable even though there are some cyclical patterns. However, these patters get addressed in the regressions, we will come to that later.

Continue to the next exercise where we will create a descriptive statistics table regarding the different segments.

Exercise 4 Descriptive statistics across wines

a) Introduction

Let's take a look at some descriptive statistics across the wines. We saw already the volumes over time for the different segments. This time we want to create a table that gives us the mean, sd, min, medium, max and the amount of observations of Liters per week, Price per liter and

Advertising expenditures. It will give us an even better understanding about the differences among the segments which is quite important when you take a look at a wine market and its dynamic.

Click the info buttons for more information about these statistical values. Min, Max and Observations are rather self-explained and not listed as an info section.

Info: Mean

The mean is one of the most common used statistical value and is used in a lot of statistical models. The mean is the arithmetic average of a sample that is roughly a summary of the distribution. However, it does not really give information about how spread the samples are. In addition, extreme values (low or high) in a sample can really mess with the mean, thus it is always important to look at more statistical values.

Formula for calculating the Mean of a sample

$$\overline{X} = \frac{\sum X}{n}$$

where $\overline{X}$ is the sample of mean, X is an individual score in the distribution and n is the number of scores in the sample.[4]

Info: Median

The median is the value that represents the 50% percentile, which means that 50% of the sample fall above the median and 50% below. Thus it divides the sample into two equal groups (also called a median split). The median is also a useful statistic to examine when the values in a distribution are weird or when there are a few extreme values (high or low). So it is also a support for the mean. It captures the real middle of the values, which is often not possible with the mean.

Formula:

$$median\ x = \begin{cases} x_{(n+1)/2} & n \text{ odd} \\ \frac{1}{2}\left(x_{(n/2)} + x_{(n+1)/2}\right) & n \text{ even} \end{cases}$$ [5]

Info: Standard deviation

The best way to understand a standard deviation is to look at the meaning of the words standard and deviation.

---

[4] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010, page 13 & 14

[5] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010, page 26 & Hogg, R. V. and Craig, A. T. Introduction to Mathematical Statistics, 5th ed. New York: Macmillan, 1995

Deviation refers to the difference between an individual score in a distribution and the average score for the distribution. So let's say the average score for a distribution is 10, and an individual child has a score of 12, the deviation is 2.

The other word in the term standard deviation is standard. Standard means typical, or average. So a standard deviation is the typical, or average, deviation between individual scores in a distribution and the mean for the distribution.

Estimate based on a sample:

$$s = \sqrt{\frac{\sum(X - \overline{X})^2}{n-1}}$$

where X is a score in the distribution, $\overline{X}$ is the sample mean, n is the number of cases in the sample. [6]

---

b) Summarising

This time we are going to use a different dataset. This dataset is called datafilter. It is the original dataset that got expended, and filtered in order to get get rid of all observations where any of the lags or leads for rev_all, rev_all_hi, rev_all_lo, ma_split or litre have the value NA, thus are empty. We need to do this in order to only take the observations into account that are used in the regressions. The regression function filters these observations automatically. Furthermore, we use the standard filter dist > 4 and not NA. We learned the reason for that in the parts before. Just let this explanation sink, it is okay, if this is a bit confusing, because it is not really important for this exercise and will be more clear later on.

The terms **lags** and **leads** weren't introduced that. It is going to be explained much more detail later because it is an important part of the regressions. Shortly, said a **lag** or **lead** is a slid column by a certain value. For a lag the column is going to be slid up and for a lead it is going to be down. Let's continue.

Load up the data set datafilter and name it tab1.

```
# Load datafilter into tab1.
tab1 <- readRDS(file="datafilter.RDS")
```

The next step is to create the dummy maind. The default value should be 0. This dummy indicates if a wine had advertising expenditures or not. We only want to include wines with advertising expenditures for the expenditure calculations.

We are going to change the value to 1 if ma_split is larger than 0 and not NA. We will use the mutate() function in combination with ifelse() to create maind and adapt the values.

```
# Mutate tab1.
# Maind is 1 if > 0 and not NA, 0 otherwise.
# Display the first 10 rows of ma_split and maind.
tab1 <- tab1 %>%
```

[6] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010, p . 20-22

```
  mutate(maind = ifelse(ma_split > 0 & !(is.na(tab1$ma_split)),1,0))
head(tab1[,c("ma_split", "maind")], n = 10)
```

ma_split	maind
160.25829	1
79.852524	1
78.841728	1
0	0
158.69426	1
79.397842	1
0	0
0	0
0	0
0	0

As you can see, the indicator for ma_split works and marks the wines that have advertising expenditures. The next steps are not super intuitive, because first we are going to calculate the means of ma_split, rprice_litre and litre for each artikelnr. And then calculate the above mentioned statistical values for each mean and segment. The first step makes sure that we take the "weight" of every artikelnr into account.

Within the mean calculation of ma_split, maind is going to be used, because we only want to take the wines into account that have advertising expenditures. So we have the means for the advertising expenditures, the price per liter and liters per week.

So, first group by artikelnr and segm. We include segm because we do not want to lose the variable while summarizing. And since every artikelnr only has one segment, it does not make a difference. Afterwards, we will summarise the mean of ma_split*maind as Advertising_Expenditures, the mean of rpice_litre as Price_per_liter and the mean of litre as Liters_per_week. We obviously will use the summarise() function.

```
# Grouped by artikelnr and segm.
# Calculate and add the mean of Advertising Expenditure, Price per liter and Liters per week
with the summary function.
tab1sum <- tab1 %>%
  group_by(artikelnr,segm) %>%
  summarise(
        'Advertising_Expenditure' = mean(ma_split*maind),
        'Price_per_liter' = mean(rprice_litre),
        'Liters_per_week' = mean(litre))
# Displaying the first 10 observations.
head(tab1sum, n = 10)
```

artikelnr	segm	Advertising_Expenditure	Price_per_liter	Liters_per_week
200401	red	5.516753	108.00337	7841.1316
201001	red	2.216277	81.389396	7922.0417
201101	red	0	129.36342	2035.6111
201301	whi	3.6959234	82.785689	3703.9247
201501	whi	0	113.53376	4282.375
202001	whi	0.9726059	92.149947	2731.0343
202008	whi	0.75633494	67.39612	9142.8387
202401	red	3.93268	73.84666	10303.177
202408	red	3.8136386	66.335496	38315
202701	whi	1.296349	80.021484	1852.1667

Alright, now we have the means for every artikelnr.

Next, we need to make sure the zeros in Advertising_Expenditures don't get taken into account, because these artikelnr respectively wine do not have any advertising expenditures during the whole period. So we simple change the values from 0 ti NA. We were not able to declare NA at the beginning as the default value because would not have had the correct divisor and the means would have been very high and wrong.

```
# Change the value of Advertising_Expenditure to NA if it is 0.
# Use the mutate function in combination with ifelse().
# The new variables is called AdvertExpend.
tab1sum <- tab1sum %>%
  mutate(AdvertExpend = ifelse(Advertising_Expenditure == 0, NA, Advertising_Expenditure))
head(tab1sum)
```

artikelnr	segm	Advertising_Expenditure	Price_per_liter	Liters_per_week	AdvertExpend
200401	red	5.516753	108.00337	7841.1316	5.516753
201001	red	2.216277	81.389396	7922.0417	2.216277
201101	red	0	129.36342	2035.6111	NA
201301	whi	3.6959234	82.785689	3703.9247	3.6959234
201501	whi	0	113.53376	4282.375	NA
202001	whi	0.9726059	92.149947	2731.0343	0.9726059

Afterwards, we will calculate the above mentioned statistical values mean, sd, min, medium, max and the amount of observations for each mean of tab1sum. So it is a bit nested, but makes sense if you think about it.

The function summarise() is already familiar to us. dplyr has even more functions that bring some extra features. So we will use summarise_each() were you list a couple of variable that will get passed to the declared functions.

The syntax for the whole summary is not really obvious so just take a look at the code in the chuck. We used na.omit() to make sure NA are not a problem. The "." just makes sure the variables get passed to the function. It is needed because we have nested functions. Check the code and proceed.

```
# Name the summary tab1summary.
# Group by segm.
# Pass the functions.
# Pass the variables.
tab1summary <- tab1sum %>%
  group_by(segm) %>%
  summarise_each(funs(
    "Mean" = mean(na.omit(.)),
    "SD" = sd(na.omit(.)),
    "Min" = min(na.omit(.)),
    "Median" = median(na.omit(.)),
    "Max" = max(na.omit(.)),
    'Observations' = length(na.omit(.))),
    Liters_per_week,Price_per_liter,AdvertExpend)
# Display the colnames.
colnames(tab1summary)
```

```
## [1] "segm"                       "Liters_per_week_Mean"
## [3] "Price_per_liter_Mean"        "AdvertExpend_Mean"
## [5] "Liters_per_week_SD"          "Price_per_liter_SD"
## [7] "AdvertExpend_SD"             "Liters_per_week_Min"
## [9] "Price_per_liter_Min"         "AdvertExpend_Min"
## [11] "Liters_per_week_Median"      "Price_per_liter_Median"
## [13] "AdvertExpend_Median"         "Liters_per_week_Max"
## [15] "Price_per_liter_Max"         "AdvertExpend_Max"
## [17] "Liters_per_week_Observations" "Price_per_liter_Observations"
## [19] "AdvertExpend_Observations"

# Display the first 6 columns.
tab1summary[,1:6]
```

segm	Liters_per_week_Mean	Price_per_liter_Mean	AdvertExpend_Mean	Liters_per_week_SD	Price_per_liter_SD
red	9609.2913	96.456103	7.9220701	12874.321	37.412276
spa	2968.7986	197.81948	2.9676222	3802.0197	136.13963
whi	7087.3701	86.960079	3.515211	8177.9346	30.679754

So we have a 3x19 data frame as the result. 3 because of the grouped segments and 19 because of 6 functions, 3 variables and the segments.

c) Summarising with better format

The dataframe contains all the desired values but not in a convinient format. A modified summarise_each() function can help us to get the right format right away. It is from the package dplyrExtras and is called xummarise_each(). The syntax is basically the same, just with the addition of .long = "vars", .wide ="funs"..long = "vars"and thearrange()` function.

So .wide ="funs" obviously makes sure that the functions are the wide argument. .long=vars makes sure that the variales are the long argument. In the end we will apply the arrange() function and pass the segm variable in order to sort by segm. All this will give us a convinient format. You can read more about the long and wide format in the already mentioned link about the reshape2 package.

```
# Naming the summary tab1summaryx.
# Group by segm.
# Passing functions and variables.
tab1summaryx <- tab1sum %>%
  group_by(segm) %>%
  xsummarise_each(funs(
    "Mean" = mean(na.omit(.)),
    "SD" = sd(na.omit(.)),
    "Min" = min(na.omit(.)),
    "Median" = median(na.omit(.)),
    "Max" = max(na.omit(.)),
    'Observations' = length(na.omit(.))),
    Liters_per_week,Price_per_liter,AdvertExpend,
    # Define the long and wide arguments.
    .long = "vars", .wide ="funs") %>%
  # Order by segm with the arrange function.
  arrange(segm)
# Displaying the summary.
tab1summaryx
```

.var	segm	Mean	SD	Min	Median	Max	Observations
Liters_per_week	red	9609.2913	12874.321	9.75	5155.6002	96367.655	293
Price_per_liter	red	96.456103	37.412276	49.369346	90.137444	324.96537	293
AdvertExpend	red	7.9220701	13.553271	0.04811343	3.2476077	96.342066	160
Liters_per_week	spa	2968.7986	3802.0197	285.03226	1707.2727	19686.054	35
Price_per_liter	spa	197.81948	136.13963	66.411384	126.57227	450.24167	35
AdvertExpend	spa	2.9676222	3.1723709	0.0180792	1.3103389	9.3446102	19
Liters_per_week	whi	7087.3701	8177.9346	183.11111	3651.921	46679.418	198
Price_per_liter	whi	86.960079	30.679754	48.955347	80.021996	267.63455	198
AdvertExpend	whi	3.515211	5.5311252	0.03063515	1.8188053	42.102694	91

As you can see, this format is way better more convenient. It is also nice to round the results, we can just use the round() function. Important while rounding is to exclude the non-numeric values. Use the following syntax: tablename[,-1:-2], digits = 2. In this example, we are excluding the first two rows and round to two digits.

```
# Rounding up the numeric values except of the first two columns (strings).
#
tab1summaryx[,-1:-2] <- round(tab1summaryx[,-1:-2], digits = 2)
tab1summaryx
```

.var	segm	Mean	SD	Min	Median	Max	Observations
Liters_per_week	red	9609.29	12874.32	9.75	5155.6	96367.66	293
Price_per_liter	red	96.46	37.41	49.37	90.14	324.97	293
AdvertExpend	red	7.92	13.55	0.05	3.25	96.34	160
Liters_per_week	spa	2968.8	3802.02	285.03	1707.27	19686.05	35
Price_per_liter	spa	197.82	136.14	66.41	126.57	450.24	35
AdvertExpend	spa	2.97	3.17	0.02	1.31	9.34	19
Liters_per_week	whi	7087.37	8177.93	183.11	3651.92	46679.42	198
Price_per_liter	whi	86.96	30.68	48.96	80.02	267.63	198
AdvertExpend	whi	3.52	5.53	0.03	1.82	42.1	91

Perfect, we have the final version. It was quite some steps, which sort of include data preparation with no direct results. But it is really interesting and also important to see the steps to understand better what the end results are and how to interpret them. We of course could skip the summary with the normal summarise_each() function, but it is part of the exercise to show the difference and the advantage of xsummarise_each(), so you can use the appropriate function for your needs.

Before we move on to the result interpretation. Try to answer these quick questions.

Quiz: Why do we have less number of observations for the advertising expenditures?

- The observations did not meet filter requirements [ ]
- Only observations with advertising expenditures got taken into account [x]
- We do not have enough data [ ]

Quiz: Which variable directly influenced these numbers?

- maind [x]
- segm [ ]
- ma_split [ ]

d) Results

Our filtered data set consists of 293 red wines with a volume of 9609 liters per week. The amount of white wines is with 198 lower. The sales with an average volume of 7987 liters per week are lower as well. Sparkling wines are not so common with an amount of 35 and a average sales volume of 2969 liters per week.

The average price of red and white wines is pretty similar with around SEK 90 (~ USD 12.2). In comparison, the sparkling wines are more expensive (SEK 198, ~ USD 27) because of the large amount of Champagnes. However, the median price is smaller than the mean price in all three segments, which means that there is a larger amount of higher priced wines in all categories.

We can also see that the amount of observations is smaller for Advertising_Expenditures. Remember: we only took observations into account where ma_split was larger than 0 and not empty. This means, that we only took advertised wines into account. The numbers show that not all the wines, even in Tier 1 & 2, got advertised. Further it is important to mention that wines are advertised by the importer (retailer does not engage in wine advertising) in Sweden.

So now, we have a quite some knowledge about the different segments. Also this exercise showed that the red wines are the most popular one before the white and sparkling wines. So we focused on volumes and advertising expenditures to investigate the data so far. Next, we will take a look at the price changes that can also have an impact on the demand. Continue to the next exercise.

This exercise refers to page 197-199 in the paper

Exercise 5 Price changes

a) Introduction

The limited possibilities to change the price and the institutional setting makes it easier to estimate the effect of the reviews, because we have a more controlled environment and do not have some sort of random impacts that might influence our estimation too much.

There are also mark up rules to all prices. The prices in the store are basically the ones from the wholesalers, which are the profit-maximizing importers. The stores itself do not have the possibiliy to change the prices.

These importers would like to change prices as a response to demand shocks. But there is little scope to change the prices. The prices are also the same in the whole country.

The prices can change only with new catalogues (April & September / October, some years also December / Summer). The prices usually change even less frequently than possible. As already stated, this is good for our estimation, because too many price changes could affect the impact of the reviews and also the estimations.

b) Data preparation

Let’s take a look at a bar chart to examine the price changes. As always load the data into the workspace. We are going to need datafilter. We also used this dataset the exercise before.

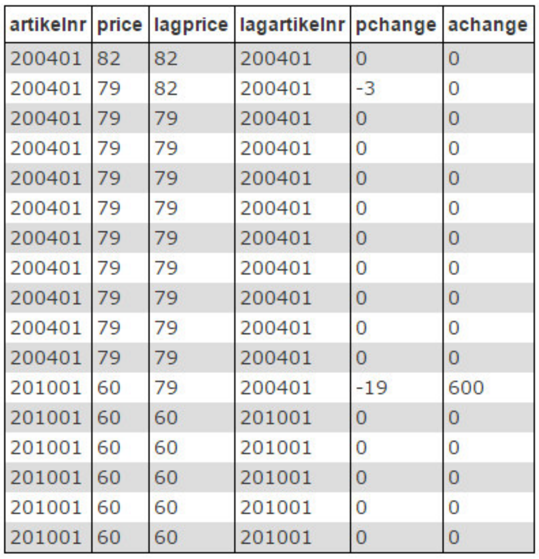
```r
# Loading the data into fig1.
fig1 <- readRDS(file="datafilter.RDS")
```

Next, we need to do some data preparations again. Again, it is more valuable than you might think. It gives us a better understanding.

So we want to have the number of price changes. We cannot just summarize for unique prices for each artikelnr, because we lose the price changes if the same prices appear again. So we need a bit of a more complicated method.

So we will create a lag variable (by 1) for the price and the artikelnr. Then, we calculate the change while calculating the difference. The mutate() function will come handy again to add these mentioned variables.

```
# Create lagprice, lagartikelnr, pchange, achange.
fig1 <- fig1 %>%
  mutate(lagprice = lag(price, n = 1),
         lagartikelnr = lag(artikelnr, n = 1),
         pchange = price - lagprice,
         achange = artikelnr - lagartikelnr)
# Display a sample with selected columns.
fig1[104:120,c("artikelnr","price","lagprice","lagartikelnr","pchange","achange")]
```

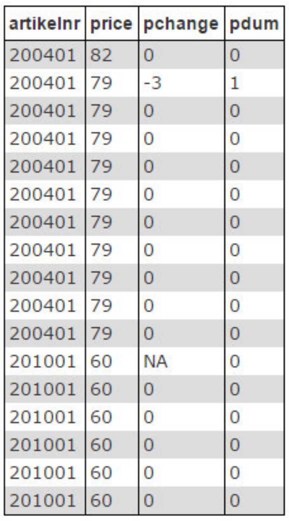


artikelnr	price	lagprice	lagartikelnr	pchange	achange
200401	82	82	200401	0	0
200401	79	82	200401	-3	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
200401	79	79	200401	0	0
201001	60	79	200401	-19	600
201001	60	60	201001	0	0
201001	60	60	201001	0	0
201001	60	60	201001	0	0
201001	60	60	201001	0	0
201001	60	60	201001	0	0

As you can see, the calculations are correct. However, we do not want to have a price change when we change the artikelnr, because it is obviously not a price change. So, we need to change pchangeaccording to achange.

Then we will create the variable pdum that will indicate a price change. There are different ways to do this, but this is one is quite fast and easy. So pdum will be 1 if pchange is not 0 and NA.

```
# Change pchange to NA if achange is not 0.
# Create pdum and give it the value 1 if pchange is not 0 and NA.
fig1 <- fig1 %>%
  mutate(pchange = ifelse(achange == 0, pchange, NA),
         pdum = ifelse(pchange != 0 & !(is.na(pchange)),1,0))
# Display a sample with selected columns.
fig1[104:120,c("artikelnr","price","pchange","pdum")]
```



artikelnr	price	pchange	pdum
200401	82	0	0
200401	79	-3	1
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
200401	79	0	0
201001	60	NA	0
201001	60	0	0
201001	60	0	0
201001	60	0	0
201001	60	0	0
201001	60	0	0

As you can see, the whole procedure worked and we are almost done.

c) Summarise the data

Now, we need to summarize the data and plot the price changes. So since we want to plot the price changes for each artikelnr, we need to group by it. We will summarize the data by summing up pdum since it indicates price changes.

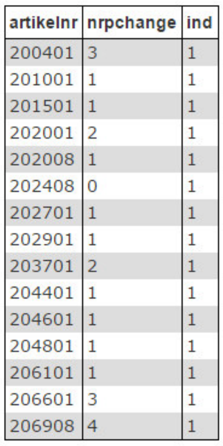
We also need to sum the indicator ind. ind is the indicator for the first time the wine is observed in data. It is possible to use the sum here because it appears maximally once for every artikelnr, keep the name ind.

It is necessary to use na.omit() for summing up ind.

Further, we only want to use the observations where ind is 1. Use the filter command. In the end we are going to display a sample as always in order to take a look at the results.

So let us get started, you should already be familiar with the summarise syntax.

```
# Name the summary figplot.
# Group by artikelnr.
# Summarise by summing pdum and ind.
# Use na.omit() for the sum of ind.
# Filter for ind == 1.
# Then, display the first 15 observations with the head function.
fig1plot <- fig1 %>%
  group_by(artikelnr) %>%
  summarise(
    'nrpchange' = sum(pdum),
    'ind' = sum(na.omit(ind))) %>%
  filter(ind == 1)
head(fig1plot, n = 15)
```



artikelnr	nrpchange	ind
200401	3	1
201001	1	1
201501	1	1
202001	2	1
202008	1	1
202408	0	1
202701	1	1
202901	1	1
203701	2	1
204401	1	1
204601	1	1
204801	1	1
206101	1	1
206601	3	1
206908	4	1

figplot gives us the number of price changes for each artikelnr.

d) Plot the bar chart

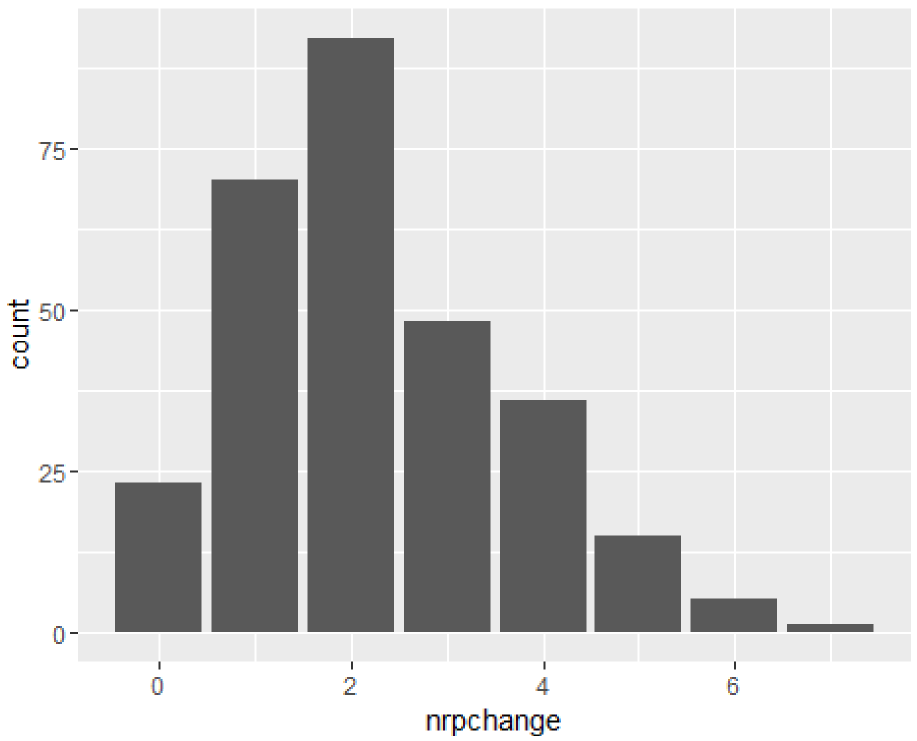
The last step is to plot the summary. We can use the ggplot command for that. It belongs to the ggplot2 package and is very powerful. You can combine ggplot with a lot of commands to customize your plot. We also include the scales package in order to be able to work with percentages.

For further information and examples about the whole package, click this docs.ggplot2.org Link You can inform yourself about the package and its functions such as ggplot, geom_bar, geom_text, scale_continuous and labs.

First, just try to plot a normal bar chart with ggplot. ggplot needs our data frame fig1plot, the aes() function and the geom_bar() function. The aes()docs.ggplot2.org Link function defines the aesthetic mappings. They describe how the variables are mapped to visual properties of geoms. Due to the fact that we want to plot a bar chart, we just need to pass our x variable nrpchange. Afterwards, we just need to add the geom_bar() docs.ggplot2.org Link with the + operator. We don't need to pass something to the function, if we just want to visualize the count. The geom_bar() function uses stat="count" by default for the height. stat is the statistical transformation to use on the data and get's passed as a string.

So, we need to pass figplot and nrpchange within the aes() function to ggplot(). Afterwards add the geom_bar() function with the + operator.

```r
# Loading the ggplot package.
# Pass fig1plit and nrpchange to the ggplot function.
# Add the geom_bar() function to the ggplot() function.
ggplot(fig1plot, aes(nrpchange)) +
  geom_bar()
```



Now, we learned the basics of the ggplotfunction. Next, we want to use percentages instead of counts. We need the scales package in order to do so.

For information regarding the scales package, download this PDf cran r-project Link

Further we want to change the labels of the x & y axis and add a title, we also want to have percentage labels on top of the bars and change the y axis value to percentages as well. Changing the labels of the x & y axis and adding a title is simply done with the labs(title,x,y) function.

The next step is to pass aesthetic mappings to the geom_bar() function because we do not want to have the default counts, of course. We need to pass a y as percentage within the aes() function. The ..count.. command is going to be very helpful, it counts the amount for each x. So we just following calculation for the percentages (..count..)/sum(..count..).

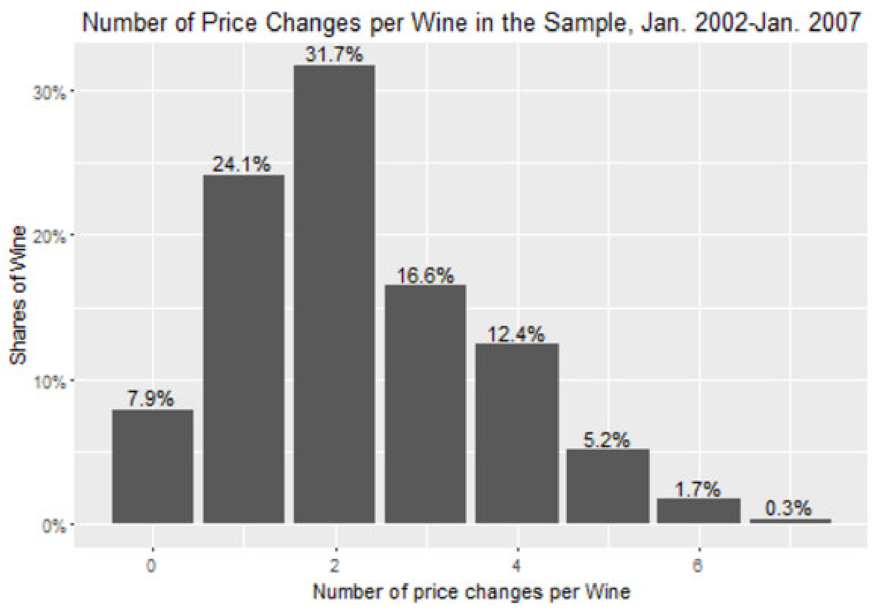
note maybe another step inbetween

Afterwards, we need to work with the geom_text() docs.ggplot2.org Link function to add labels on top of the bars. We will pass the same y as before to geom_text(), then we are going to add a label command. Both of these steps are included in aes. Further, we need to define stat and vjust. We are going to use "count" here. vjust is just used for text alignments. vjust basically stands for vertical justification. A positive number (between 0 & 1) moves the text down and a negative up.

The last function that we need is scale_y_continuous(). We just pass label setting percent to it in order to get a percentage scale at the y axis.

So let's get started.

```
# Basic ggplot with data and aes.
 ggplot(fig1plot, aes(x = nrpchange)) +
  # Bar function with y as percentages as aesthetics.
  geom_bar(aes(y = (..count..)/sum(..count..))) +
  # Function in order to add labels to the bar chart.
  # Again, y & labels as percentages as aesthetics.
  # Satistical information count and vjust as text alignment.
  geom_text(aes(y = ((..count..)/sum(..count..)), label =
scales::percent((..count..)/sum(..count..))), stat = "count", vjust = -0.25) +
  # Change the y axix scaling to percentages
  scale_y_continuous(labels = percent) +
  # Adds title and changes x & y axis labels
  labs(title = "Number of Price Changes per Wine in the Sample, Jan. 2002-Jan. 2007", x =
"Number of price changes per Wine", y = "Shares of Wine")
```



```
# Calculate the median of nrpchange
median(fig1plot$nrpchange)

## [1] 2
```

The result gives us good information about the price change situation in the Swedish wine market. The median wine changed the price twice during the period of Jan 2002 - first two weeks of 2007. This is the exact number of markup changes at Systembolaget.

So if we add up the numbers from 0 to 3 changes, we get approximately 80.3 %. We have data of round about 5 years, which means the majority of wines changed their price 0 to 0.6 times a year, which means rougly every 0 to 7 month. This is not really that often, since - remember - the prices can change twice a year with the new catalogues in April & September / October. Some years also have a decemeber or summer catalogue.

All this means that we also have rather stable prices which is great in order to get good results, since price changes as a response to for example demand shocks, can influence the demands and therefore the results. We do not want to have a too big of a impact from price changes on demands since the reviews are the important variable.

In the next exercise, we are going to take a look where the reviews actually come from and what differences exist among the reviews. Thus, we will take a look at the different print media.

Proceed to the next exercise.

This exercise refers to page 199 in the paper

Exercise 6 Wine Reviews in Swedish Print Media

a) Introduction

The paper used the grades of the six major print media.

The target group of theses reviews is the general consumer market, the consumer that looks for a good wine for the weekend dinner rather than the latest prestige vintage wine. However, there are indications that also suggest a prestige focused customer.

The six major print media consists of:

- **Two leading tabloids (nationwide distribution)** Aftonbladet circulation of 452.300 Expressen circulation of 363.00
- **Two leading morning papers (nationwide distribution, even though mainly greater Stockholm area)** Dagens Nyheter circulation of 368.200 Svenska Dagbladet circulation of 180.800
- **Main business daily (nationwide distribution)** Dagens Industri circulation of 116.700
- **Food and beverage magazine** Allt om Mat (AoM) circulation of 129.300, 20 issues a year

The grades from the print media indicate if a wine is a "good" buy. The numerical grades set the quality in relation to the price.

The grades get converted into a 0 - 10 scale in order to have a consistent picture.

If a grade exists, then at least one review appeared in the given week. If a wine has several reviews from several sources, then a circulation weighted average is calculated.

Info: Weighted arithmetic mean

The weighted arithmetic mean is similar to an ordinary arithmetic mean (the most common type of average), except that instead of each of the data points contributing equally to the final average, some data points contribute more than others.

So we have a set of data $x_1, x_2, \ldots, x_n$. Each x has an individual weight w, meaning: $\frac{w_1}{\sum_{n=1}^{n} w_i}$.

If we combine both we are going to have:

$$\overline{x} = \frac{\sum_{n=1}^{n} w_i x_i}{\sum_{n=1}^{n} w_i}$$

which results in:

$$\overline{x} = \frac{w_1 x_1 + w_2 x_2 + \ldots + w_n x_n}{w_1 + w_2 + \ldots + w_n}$$

Source: Wikipedia Link

Let’s get an overview about the grades among the print medias. We are going to create a table in order to summarise the grades of all print medias. The summary will consist of the mean, standard deviation, minimum, median, maximum and amount of observations for each of the print media grades.

b) Calculated variables: Good and bad grades

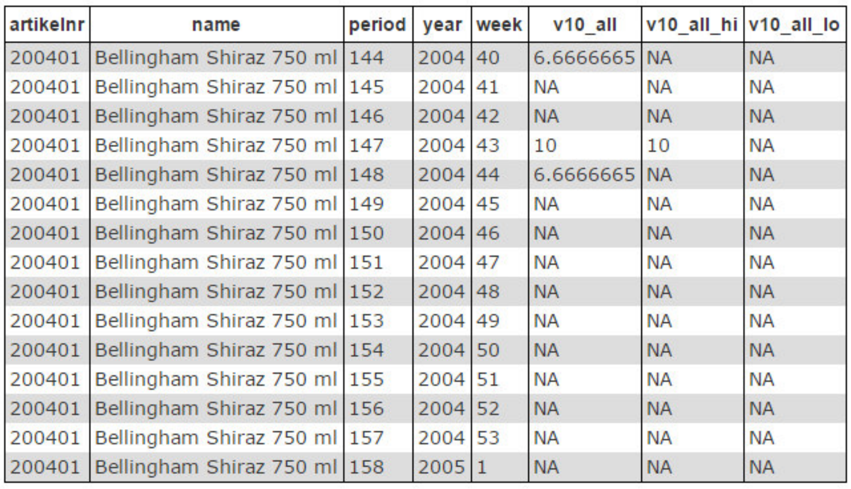
Same procedere, load the data set that we need. We are going to use datafilter again.

```r
# Loading data for tab2.
tab2 <- readRDS(file="datafilter.RDS")
```

The next step is to create two new variables. One will represent a high grade and the other a low grade. Remember: a good grade is considered a grade larger than 7.999 and a bad grade is considered equal or smaller than 4. The grade in a given week is stored in v10_all which is therefore our variable to check. So cases can basically happen. The grade is larger than 7.999, which means the new variable v10_all_hi takes the given value. Analogically, the same happens in the case of equal or smaller 4, just that the grade gets stored in v10_all_lo. The third case happens if neither of these conditions are true, here the new variables will get assigned to value NA.

So let us create these variables. We are going to use the mutate() and ifelse() function in combination as already known. In the end, it is always good to display your results in order to make sure all is correct.

```r
# Assign the value of v10_all to the dummy variable v10_all_hi if it is larger than 7.999, NA otherwise.
# Assign the value of v10_all to the dummy variable v10_all_lo if it is smaller or equal 4, NA otherwise.
tab2 <- tab2 %>%
  mutate(v10_all_hi = ifelse(v10_all > 7.999 & !(is.na(v10_all)),v10_all,NA),
         v10_all_lo = ifelse(v10_all <= 4,v10_all,NA))
# Displaying first 15 observations with selected columns
head(tab2[,c("artikelnr","name","period","year","week","v10_all","v10_all_hi","v10_all_lo")], n = 15)
```



artikelnr	name	period	year	week	v10_all	v10_all_hi	v10_all_lo
200401	Bellingham Shiraz 750 ml	144	2004	40	6.6666665	NA	NA
200401	Bellingham Shiraz 750 ml	145	2004	41	NA	NA	NA
200401	Bellingham Shiraz 750 ml	146	2004	42	NA	NA	NA
200401	Bellingham Shiraz 750 ml	147	2004	43	10	10	NA
200401	Bellingham Shiraz 750 ml	148	2004	44	6.6666665	NA	NA
200401	Bellingham Shiraz 750 ml	149	2004	45	NA	NA	NA
200401	Bellingham Shiraz 750 ml	150	2004	46	NA	NA	NA
200401	Bellingham Shiraz 750 ml	151	2004	47	NA	NA	NA
200401	Bellingham Shiraz 750 ml	152	2004	48	NA	NA	NA
200401	Bellingham Shiraz 750 ml	153	2004	49	NA	NA	NA
200401	Bellingham Shiraz 750 ml	154	2004	50	NA	NA	NA
200401	Bellingham Shiraz 750 ml	155	2004	51	NA	NA	NA
200401	Bellingham Shiraz 750 ml	156	2004	52	NA	NA	NA
200401	Bellingham Shiraz 750 ml	157	2004	53	NA	NA	NA
200401	Bellingham Shiraz 750 ml	158	2005	1	NA	NA	NA

So you can clearly see that v10_all_hi took the value 10 of v10_all in row 4. Also important and visible is the row 1 and 5 did not pass the value 6.666667 to v10_all_hi or v10_all_lo, which means the grade is neither good nor bad, because it is between 7.9999 and 4.

c) Summarise the data

The next step is to summarize the data. Remember: The summary will consist of the mean, standard deviation, minimum, median, maximum and amount of observations for each of the print media grades. The corresponding print media variables are:

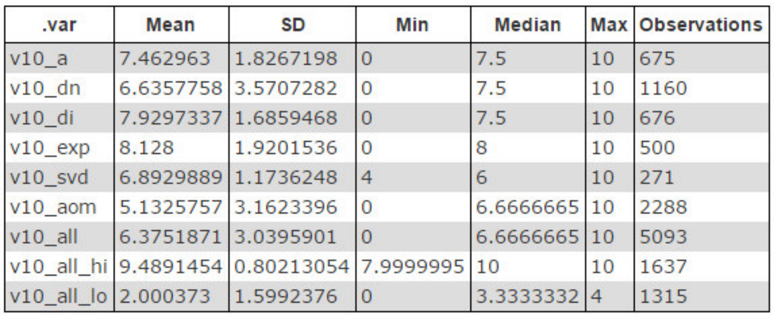
- v10_a Normalized review (0-10): Aftonbladet
- v10_dn Normalized review (0-10): Dagens Nyheter
- v10_di Normalized review (0-10): Dagens Industri
- v10_exp Normalized review (0-10): Expressen
- v10_svd Normalized review (0-10): Svenska Dagbladet
- v10_aom Normalized review (0-10): Allt om Mat (AoM)

The first thought would probably again to use the function summarise_each(). However, we would have the same problem as in chapter 4. We would get a 1x54 data frame as the resul. 1 because of no grouped data and 54 because of 6 functions and 9 variables. This is not really the format that we want, always be careful and choose your functions wisely even though this takes sometimes a bit of a trial and error time.

So the solution is to use xsummarise_each() in order to get the right format.

First, let us summarise the data with the desired values. You should be familiar with the xsummarise_each() syntax. So pass the functions, then the variables and the long & wide arguments. We will use na.omit() also again at this point. The variables are going to be the above mentioned print media grades and the new two variables.

```
# Summarising tab2.
tab2summaryx <- tab2 %>%
 # Passing functions.
 xsummarise_each(funs("Mean" = mean(na.omit(.)),
                 "SD" = sd(na.omit(.)),
                 "Min" = min(na.omit(.)),
                 "Median" = median(na.omit(.)),
                 "Max" = max(na.omit(.)),
                 "Observations" = length(na.omit(.))),
            # Passing variables.
            v10_a, v10_dn,v10_di,v10_exp,v10_svd,v10_aom,v10_all,v10_all_hi,v10_all_lo,
            # Long and wide arguments.
            # Variables as long / id argument, functions as the wide argument.
            .long = "vars", .wide ="funs")
# Displaying the summary.
tab2summaryx
```

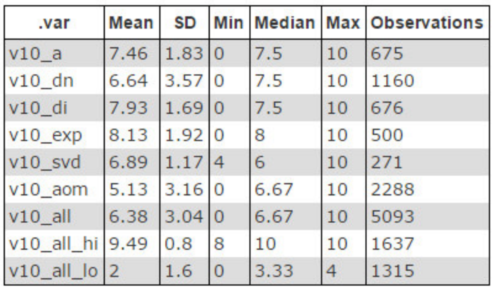


.var	Mean	SD	Min	Median	Max	Observations
v10_a	7.462963	1.8267198	0	7.5	10	675
v10_dn	6.6357758	3.5707282	0	7.5	10	1160
v10_di	7.9297337	1.6859468	0	7.5	10	676
v10_exp	8.128	1.9201536	0	8	10	500
v10_svd	6.8929889	1.1736248	4	6	10	271
v10_aom	5.1325757	3.1623396	0	6.6666665	10	2288
v10_all	6.3751871	3.0395901	0	6.6666665	10	5093
v10_all_hi	9.4891454	0.80213054	7.9999995	10	10	1637
v10_all_lo	2.000373	1.5992376	0	3.3333332	4	1315

The transformation gave us the expected readability. But again we need to round the results.

You know the round() function already, remember to exclude the first column because it is non-numeric. This time it us just one variable. Of course, it is not necessary to round the results, but it is always better to have a good looking result, even though some formatting that are basically extras should have the least priority.

```
# Round the values to 2 digits.
# Then display tab2result
tab2summaryx[,-1] <- round(tab2summaryx[,-1], digits = 2)
tab2summaryx
```



.var	Mean	SD	Min	Median	Max	Observations
v10_a	7.46	1.83	0	7.5	10	675
v10_dn	6.64	3.57	0	7.5	10	1160
v10_di	7.93	1.69	0	7.5	10	676
v10_exp	8.13	1.92	0	8	10	500
v10_svd	6.89	1.17	4	6	10	271
v10_aom	5.13	3.16	0	6.67	10	2288
v10_all	6.38	3.04	0	6.67	10	5093
v10_all_hi	9.49	0.8	8	10	10	1637
v10_all_lo	2	1.6	0	3.33	4	1315

Perfect, we have the final version.

The result shows that there is considerable dispersion of the grades across all media. It seems that papers like Allt om Mat or Dagens Nyheter give averagely lower grades, which could mean that they are simply stricter. They also have the biggest amount of observations.

Summing the numbers of observations of the six print media gives us 5570. In comparison, the number of grades is 5093, meaning the majority of wines got reviewed just in one source in a given week.

This exercise gave us a better understanding what the differences of the print media are. The grades are obviously important for our estimations because they are the essential part of the review. Later you will see that the main estimators are the variables for a review, good review and bad review, so the grades influence to what category the review belongs to.

Before you proceed to the next exercise, let's take a look at some questions to test your knowledge.

d) Questions

Quiz: To what print media is v10_a referring?

- Aftonbladet [x]
- Dagens Industri [ ]
- Expressen [ ]
- Svenska Dagbladet [ ]

Quiz: To what print media is v10_svd referring?

- Aftonbladet [ ]

- Dagens Industri [ ]
- Expressen [ ]
- Svenska Dagbladet [x]

Quiz: Does it make a difference for the good grade indicator if a grade is 8 or 9?

- Yes [ ]
- No [x]

This exercise refers to page 199-201 in the paper

Exercise 7 Insightful Plots

a) The Number of Wines reviewed per Week

This part of the exercise follows up to the previous exercise. We looked at the different print medias that provide the reviews and grades. Now, we will take a look at the number of reviews per week. This will give us an idea if the numbers are rather stable or have some insightful peaks. This is interesting because assuming the reviews have a significant impact on the demand, the demand should increase more during these peaks.

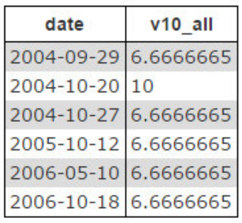
As preparation, load the "original" dataset datasort first.

```
datasort <- readRDS(file="datasort.RDS")
```

We also have to quickly apply the standardfilter, select the needed columns and get rid of pairs that have NA in it. The so called standardfilter, is the filter for tier 1 & 2. Then we need to select only date and v10_all, because we want to plot the amount of reviews over time on a weekly basis. Remember: v10_all is the weekly average normalized review (all media).

We can use the filter() and select() functions to do most of it. Afterwards, the complete.cases() function to get rid of "NA pairs". This function belongs to the stats package. If you want to read more about the package, click on the following Stat ethz Link.

```
fig2 <- datasort %>%
  filter(dist > 4 & na.omit(dist)) %>%
  select(date, v10_all)
fig2 <- fig2[complete.cases(fig2),]
head(fig2)
```

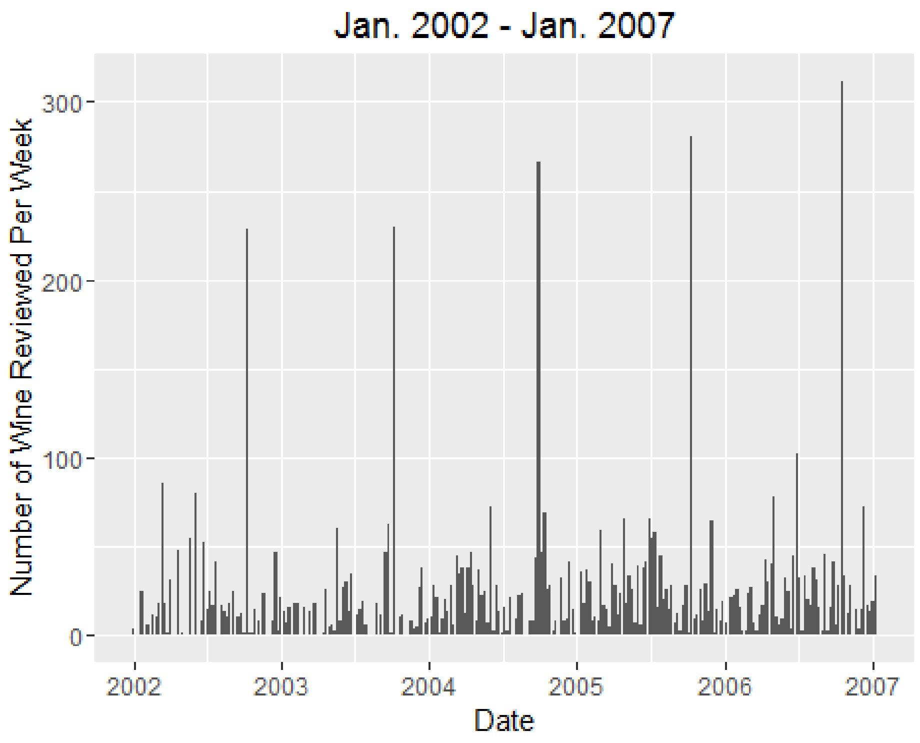


date	v10_all
2004-09-29	6.6666665
2004-10-20	10
2004-10-27	6.6666665
2005-10-12	6.6666665
2006-05-10	6.6666665
2006-10-18	6.6666665

Now, let's come to the important part: the bar plot. We already created a bar plot in one of the previous exercises, so you should remember some parts of the syntax. This time we also want to plot the counts, in fact the counts of the reviews per week. Thus, the x argument is going to be the date and the y argument respectively the count argument is going to be v10_all.

ggplot is our function of choice. So we will pass fig2 as the data frame and date as the x within the aes function. Afterwards, geom_bar will be added in order to create the bar plot. For a better readability, we can a title and labels for the axis with labs().

```r
# ggplot.
# Pass fig2 and date as x within aes().
# Add geom_bar().
# Add labs() with the title = "Jan. 2002 - Jan. 2007".
# Date as x label and Number of Wine Reviewed Per Week as the y label.
ggplot(fig2, aes(x = date)) +
  geom_bar() +
  labs(title = "Jan. 2002 - Jan. 2007", x = "Date", y = "Number of Wine Reviewed Per Week")
```



The bar plot shows that the reviews are spread across the time. It is clearly visible that the amount of reviews peaks at around October every year due to the AoM wine festival, which is the yearly wine tasting of the print media Allt om Mat. As mentioned, basically all red and white wines are being reviewed during that festival. To put it into numbers, in 2004 95 percent of red and 94 percent of white wines were reviewed during the AoM festival.

One might think again that the wholesalers might want to adapt prices right after the festival in order to influence the demand and as a response to the reviews. This is not possible though, because the festival happens shortly after the publication of the catalogue, where, remember, the prices get determined. So it is not possible for the wholesalers to change their prices in response to the reviews. This is really important for our estimations, since we do not want to have too much or better no influence from other factors as the main estimators.

This exercise refers to page 201 $ 202 in the paper

b) Geo Chart

The next part of the exercise is to create a geo chart / map with the amount of volumes that gets important from the countries all over the world. It is not really a follow up to the previous exercises, but this exercises focuses rather on insightful plots that give us some extra information about the Swedish wine market and the circumstances.

Again, first check the chunk to load our data. We are going to use datacc which basically is datasort with changed country names that can be recognized by the functions that we are going to use. Most of the country names do not have the American spelling, because the paper is from Scandinavian authors.

```
# Loading datacc, cc stands for changed countrynames due to scandinavian spellings.
# Name it datacc.
datacc <- readRDS(file="datacc.RDS")
```

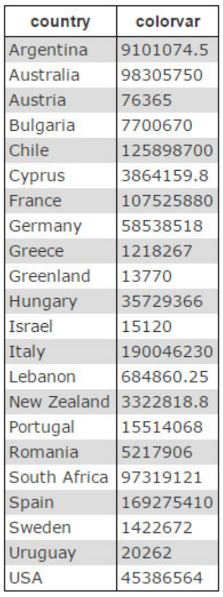
Next, let us create a map where we color the countries according to the amount of liters imported. We are going to use the google vision package. We already used a function from the google vision package to create line charts.

This time we will use the function gvisGeoChart() to create the desired GeoChart. But first we need to summarise the data in order to have a useable data frame. So since we want to have the amount of liters for each country, it is obvious that we will group by country. Then summarise with the sum of litre as colorvar. na.omit() needs to be used again, remember the reason (NAs).

```
# Group by country.
# Summarise the sum of litre as colorvar.
# Filter out all the NA in colorvar.
geosummary <- datacc %>%
  group_by(country) %>%
  summarise(colorvar = sum(na.omit(litre)))
```

Perfect, let us take a look at the data. Display geosummary.

```
#Display geosummary.
geosummary
```

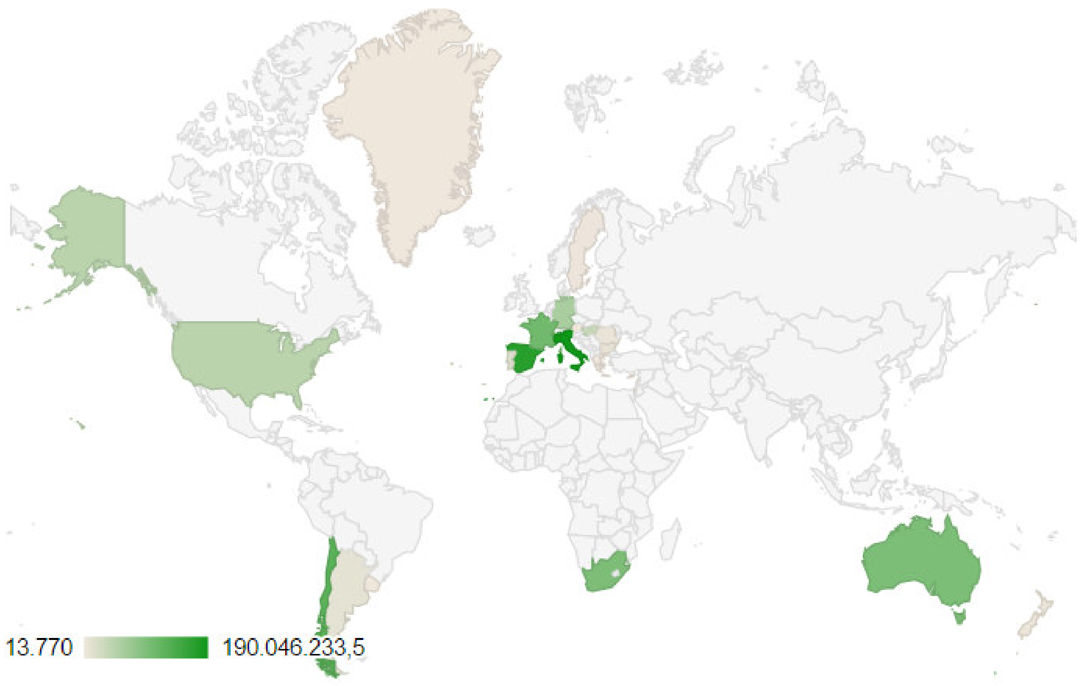


country	colorvar
Argentina	9101074.5
Australia	98305750
Austria	76365
Bulgaria	7700670
Chile	125898700
Cyprus	3864159.8
France	107525880
Germany	58538518
Greece	1218267
Greenland	13770
Hungary	35729366
Israel	15120
Italy	190046230
Lebanon	684860.25
New Zealand	3322818.8
Portugal	15514068
Romania	5217906
South Africa	97319121
Spain	169275410
Sweden	1422672
Uruguay	20262
USA	45386564

As you can see the biggest wine provider are *Italy*, *Spain*, *Chile* and *France*. However, Australia and South Africa are quite close to France. After that the numbers get quite smaller. At this point, it is important to know that Greenland is actually a placeholder for *others*.

Now let's create the Geo Chart for these numbers. We need to pass at least the summarized data frame, a locationvar and a colorvar. The locationvar is the variable that has all the country names. The countries can for example exist in the format of latitude: longitude, as string with the names or even as uppercase ISO-3166 code. colorvar is for the variable that indicates the scale as color. The function will create a color gradient according to min and max value with the default colors. After we created the gvisGeoChart object, we are going to pass this object to the plot() function with chart as the tag. This makes sure that we get the html code as the output for the problem set.

```r
# Passing ccsummary, country and colorvar.
# plot g as gvisGeoChart object.
plotg <- gvisGeoChart(geosummary,locationvar = "country", colorvar = "colorvar")
# Plotting plotg with the plot function, tag has to be chart.
plot(plotg, tag = "chart")
```



The geo chart indicates the amount of liters with corresponding color. You can see the scale in the bottom left corner. *Italy* has the highest amount and therefore has the darkest green. You can also hover the mouse of the country to see the actual amount.

c) Motion chart

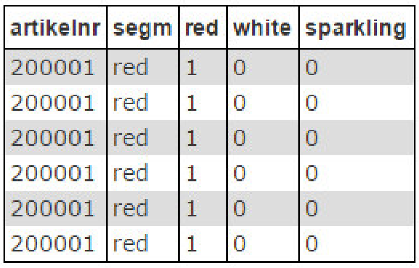
In this part of the exercise, we want to create a motion chart with the gvisMotionChart function. It is from the googleVis package as well. We need to pass at least the data frame, an id variable, a time variable and variables for the x & y axis. A color and size variable is optional. Those would determine the color and size of the aesthetics such as points.

But first we need to prepare with some short steps a data frame with the correct values and format.

So we want to pass a data frame that is grouped by year and country. We also want to have the sums of each segments (segm) in order to have values for the axis.

Before we can do this, we need to split the segm column into three columns in order to sum up the amount of segments. Let's use the mutate() function.

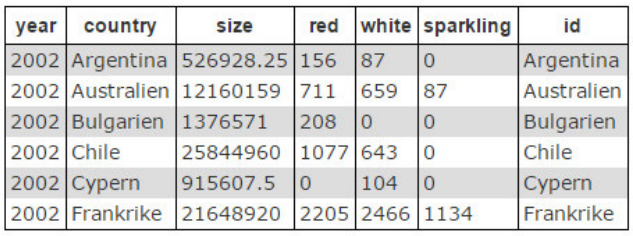
```
# Use datasort as the base dataset.
# The new dataframe should be called datamoch.
# Split segm into 3 different columns.
# Check for the corresponding value in segm.
# Use ifelse() to assign 1 if yes and NA if no.
datamoch <- datasort %>%
  mutate(red = ifelse(segm == 'red',1,0),
         white = ifelse(segm == 'whi',1,0),
         sparkling = ifelse(segm == 'spa',1,0))
# Display a sample with head().
head(datamoch[,c("artikelnr","segm","red","white","sparkling")])
```



artikelnr	segm	red	white	sparkling
200001	red	1	0	0
200001	red	1	0	0
200001	red	1	0	0
200001	red	1	0	0
200001	red	1	0	0
200001	red	1	0	0

As you can see, it works perfectly. Now, we need to group by year and country. Then, we will summarize the size as the sum of litre. Further, we will sum up the amount of each segment.

```
# Use datamoch.
# Group by year and country.
# Summarise size as the sum of litre.
# red as the sum of red, the same procedure applies to white and sparkling.
# Use na.omit().
# Mutate id as country.
# Ungroup datamoch.
# Display datamoch with the head() function.
datamoch <- datamoch %>%
  group_by(year,country) %>%
  summarise(size = sum(na.omit(litre)),
            red = sum(na.omit(red)),
            white = sum(na.omit(white)),
            sparkling = sum(na.omit(sparkling))) %>%
  mutate(id = country)
datamoch <- ungroup(datamoch)
head(datamoch)
```

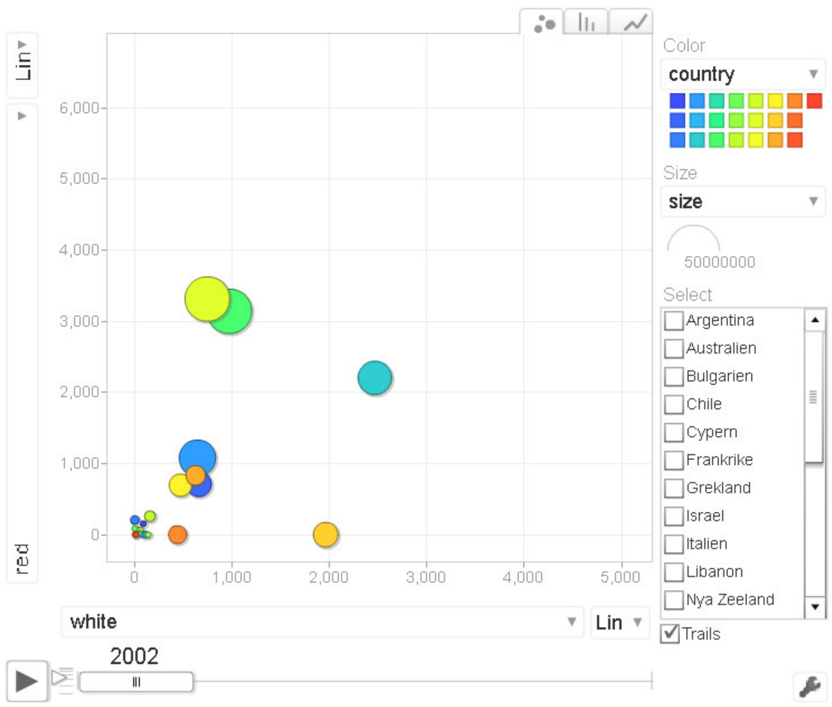


year	country	size	red	white	sparkling	id
2002	Argentina	526928.25	156	87	0	Argentina
2002	Australien	12160159	711	659	87	Australien
2002	Bulgarien	1376571	208	0	0	Bulgarien
2002	Chile	25844960	1077	643	0	Chile
2002	Cypern	915607.5	0	104	0	Cypern
2002	Frankrike	21648920	2205	2466	1134	Frankrike

Perfect, now we have the data frame that we want. The we duplicated the country column as id because we want to use the country as id and also as the color variable. The ungrouping is necessary because gvisMotionChart() would not accept the data frame due to the "wrong" format.

Let's pass the data frame and variables to gvisMotionChart().

```
# Pass datamoch to gvisMotionChart, name the object plotmoch.
# Pass id as idvar, year as timevar, red as xvar, white as yvar, country as colorvar and size as sizevar.
plotmoch <- gvisMotionChart(datamoch, idvar = "id", timevar = "year", xvar = "red", yvar = "white" , colorvar = "country" , sizevar = "size")
# Plot plotmoch with plot() and tag being "chart".
plot(plotmoch, tag = "chart")
```



The motion chart gives you a lot of options. You have a bubble chart, a column chart and line chart tab. All of these tabs interact with the timeline. So if you press the play button all years are going to be displayed and the chart will change according to the year. You can also just slide the timeline as you wish or change the arguments for the size etc. So there are a lot of options, discover them as you like.

When we have red and white as x & y, use size as the size variable and press the play button, we can see that most if the bubbles move to the top right corner. This indicates that the amount of red and white wines became larger over time. However, the size of the bubbles does not change that significantly, which suggest that the volumes stay relatively stable. Remember: we had the same results in previous exercises.

Well, so far, we looked at some statistics in order to investigate and interpret the data and to understand the settings we have for our regressions regarding the impacts of reviews and advertising expenditures on demands.

Proceed to the next exercise.

Exercise 8 The Empirical Model

a) The Equation

As already mentioned, we did quite some data investigation to have a better understanding about the settings we have. Before we proceed to do some regressions, we need to take a look at the theoretical model behind the regressions we are going to do.

Let's take a look at the main equation (1):

$$lnQ_{ijkt}$$

$$= \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}^{good} R_{it-l}^{good} + \sum_{l=-4}^{25} \alpha_{t-l} R_{it-l} + \sum_{l=-4}^{25} \alpha_{t-l}^{b} adR_{it-l}^{b}ad + \sum_{l=-4}^{25} \gamma_{t-l} ADVERT_{it-l}$$

$$+ \eta_{ijkt}$$

lnQ is the natural log of liters sold of wine i

α is a fixed effect, defined by wine x vintage x price combination.

δ is a sparate fixed effect, defined by week t x segment k (compensation for time trends & sales)

For example: One fixed effect captures bottled red wines sold at a price above SEK 110 per bottle in week 128

R is the review variable, the value is 1 if wine got reviewed in given week t, otherwise it’s just 0

R^{good} indicates a good review. Remember: it is a good review if the grade is > 7.999

R^{bad} indicates a bad review. Remember: it is a bad review if the grade is < 4

$ADVERT$ stands for the advertisement expenditures (in SEK) for wine i and week t

η is the error term with wine i, vintage x price combination j and week t

So this is our linear regression equation. For more information to linear regressions models, click the info buttons.

Info: The Linear Regression Model with one Regressor

This model explaines a linear relationship between X and Y. The slope of the line relating X and Y is the effect of a one-unit change in X on Y. Just as the mean of Y is an unknown characteristic of the population distribution of Y. Thus, the slope of the line relating X and Y is also an unknown characteristic in the given population. So in our case we want to examine the relationship between reviews / advertisment and the liters of sales. So, we need to examine with the sample data the slope that represent the effect on Y of a unit change in X. (1)

$$\beta_X = \frac{\text{change in} Y}{\text{change in} X} = \frac{\Delta Y}{\Delta X}$$

where the greek letter δ(delta) stands for "change in." That is,

$$\Delta Y = \beta_X * \Delta Y$$

The equation (1) is the definition of the slope of a straight line relating Y and X. This straight line can be written:

$$Y = \beta_0 + \beta_X * X$$

where β_0 is the intercept of this straight line and, as before, β_X is the slope.

In most scenarios, there are more factors that might influence Y, thus we have to account for that. We add these factors to the equation. For example, in our problem set this could be demand shocks. We also add indices for the different samples i. Every sample has its own Y_i, X_i and u_i. We get:

$$Y_i = \beta_0 + \beta_1 * X_i + u_i$$

For each i where β_0 is the intercept of the line and β_1 is the slope.

This equation is called the **linear regression model with a single regressor**, in which Y is the dependent variable and X the **independent variable** or the **regressor**. The first part of equation, $\beta_0 + \beta_1 * X_i$, is the population regression line or the population regression function.

This is the relationship that holds between Y and X on average over the population. Thus, if you knew the value of X, you can estimate the value of the dependent variable Y which gets defined by $Y = \beta_0 + \beta_1 * X$. The **intercept** β_0 and the **slope** β_1 are the **coefficients**.

The slope β_1 is the change in Y associated with a unit change in X. The intercept is the value of the when $X = 0$

The term u_i for the other factors in the equation is called the **error term**. The error term accounts for the factors that are different amoung the samples i and the value predicted by the population regression line. This error term contains all the other factors besides X that determine the value of the dependent variable for a specific observation i.[7]

However, we have multiple regressors which means, we need a multiple regression model.

Info: The Multiple Regression Model

The **multiple regression model** extends the single variable regression model and give the opportunity to include more variables as regressors.

[7] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p.107-111

The Population Multiple Regression Model

The population regression line $E(Y_i|X_{1i} = x_1, X_{2i} = x_2) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$ is the relationship between Y and X_1 and X_2. Also in this case, just as in the caser of a single regressor regressioin, there are other factors that influence Y_i besides the regressors. Analogously, all these factors are included in the "error" term u_i. This error term is the deviation of a particular observation i. Accordingly, we have:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + u_i, i = 1, \ldots, n$$

where the subscript i indicates the i^{th} of the n observations in the sample. The equation is the **population multiple regression model** when there are only two regressors, X_{1i} and X_{2i}.

We showed the case with two regressors. But the model can of course include more regressors, in fact a model that includes k regressors. So, the multiple regression model can have k regressors, X_{1i}, X_{2i}, X_{ki}.

The definitions of homoscedasticity and heteroscedasticity in the multiple regression model are extensions of their definitions in the single-regressor model. The error term u_i in the multiple regression model is **homoscedastic** if the variance of the conditional distribution of u_i given X_{1i}, X_{2i}, X_{ki}, $var(u_i|X_{1i}, X_{2i}, X_{ki})$, is constant for $i = 1, \ldots, n$ and thus does not depend on the values of X_{1i}, X_{2i}, X_{ki}. Otherwise, the error term is **heteroskedastic**.

The Multiple Regression Model

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + u_i, i = 1, \ldots, n$$

where

Y_i is the i^{th} observation on the dependent variable; X_{1i}, X_{2i}, X_{ki} are the i^{th} observation on each of the k regressor; and u_i is the error term. The population regression line is the relationship that hold between Y and the $X's$ on average in the population:

$$E(Y|X_{1i} = x_1, X_{2i} = x_2, \ldots, X_{ki} = x_k) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$$

*The intercept β_0 is the expected value of Y when all the $X's$ equal 0. The intercept can be thought of as the coefficient on a regressor, X_{0i}, that equals 1 for all i.[8]

We are going to use a finite distributed lag model. This allows the possibility for non-simultaneous effects of reviews. Therefore, we allow effects to last up to 25 weeks.

Click on the info buttons for information regarding *lags* and the *distributed lag model*.

Info: Lags

The observation on the time series variable Y made at date t is denoted Y_t, and the total number of observations is denoted T. The interval between two observations $t, t+1$ is defined by the time variable such as weeks, month or years. The time variable in our dataset is period and has

[8] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p. 186-189

the range from to 261 for each artikelnr. This range represents the whole examining time period. One period is one-week long.

The future and past values have special notations. A **lagged variable** is denoted Y_{t-j} with t as the current time period and j as the j^{th}lagged value or simply its j^{th} **lag**. Analogically, the values are denoted as Y_{t+j} as the j^{th}lead value.[9]

Info: Distributed Lag Model

A *distributed-lag model* is a dynamic model in which we want to examine the effect of X on Y over time. We are going to use **lags** and **leads** in order to do so.

In the simple case of one explanatory variable and a linear relationship, we can write the model as

$$y_t = \alpha + \beta(L)x_t + u_t = \alpha + \sum_{s=0}^{\infty} \beta_s x_{t-s} + u_t,$$

where u_t is the error term. This formula does not include "negative lags" as leads, but it as absolutley possible to include these.

The individual coefficients β_s are called *lag weights* and the collectively describe the *lag distribution*. They define the pattern of how x affects y over time. We do not really have weights in our model, we use individual fixed effects in the same spot instead.

Further, we cannot estimate an infinite number of coefficients. One practical method is to limit the lag to finite length, which is appropriate if the lag distribution is effectively zero beyond q periods.

Source: Reed.edu PDF Link (Visited: 28.07.2016)

The regression is estimated by ordinary least squares (OLS). This accounts for possible correlation in the error term over time period and across vintages, prices and container types. We clustered the standard errors at the brand level artikelid.

artikelid is the cluster variable for the regression. artikelid indicates the ID number of brand (given by Systembolaget).

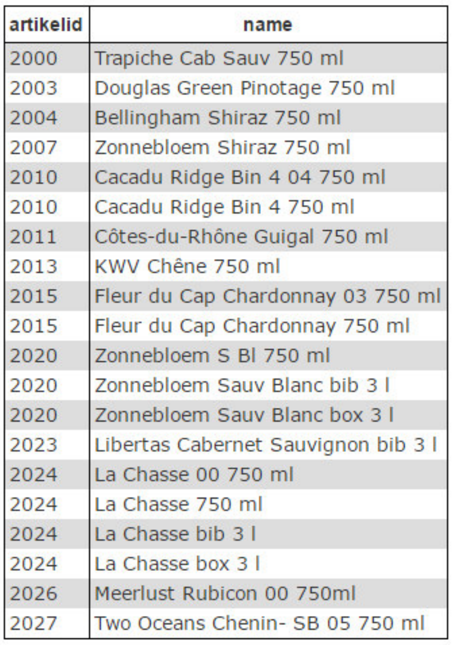
Let us take a closer look at the artikelidto understand better why we cluster at the brand level. First load up datasort.

```
# Load datasort into datasort.
datasort <- readRDS(file="datasort.RDS")
```

[9] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p. 520

Now, we need to group the data by artikelid and name to see what differences are within each artikelid respectively brand. Then summarize it with the summarise() function. We do not need to pass a function to summarise().

```r
# Name the summary cluster.
# Group by artikelid and name.
# Summarise.
# Display the first 20 rows of cluster with the head() function.
cluster <- datasort %>%
  group_by(artikelid, name) %>%
  summarise()
head(cluster, n = 20)
```



artikelid	name
2000	Trapiche Cab Sauv 750 ml
2003	Douglas Green Pinotage 750 ml
2004	Bellingham Shiraz 750 ml
2007	Zonnebloem Shiraz 750 ml
2010	Cacadu Ridge Bin 4 04 750 ml
2010	Cacadu Ridge Bin 4 750 ml
2011	Côtes-du-Rhône Guigal 750 ml
2013	KWV Chêne 750 ml
2015	Fleur du Cap Chardonnay 03 750 ml
2015	Fleur du Cap Chardonnay 750 ml
2020	Zonnebloem S Bl 750 ml
2020	Zonnebloem Sauv Blanc bib 3 l
2020	Zonnebloem Sauv Blanc box 3 l
2023	Libertas Cabernet Sauvignon bib 3 l
2024	La Chasse 00 750 ml
2024	La Chasse 750 ml
2024	La Chasse bib 3 l
2024	La Chasse box 3 l
2026	Meerlust Rubicon 00 750ml
2027	Two Oceans Chenin- SB 05 750 ml

As you can see the wines within an artikelid group have just a different name or container. For example, the artikelid 2024 has two different names for the 750 ml bottle and for the 3l box. But this means that the wines within each group can be correlated, since it is basically the same wine. Therefore, we use clustered standard errors to account for these correlations. However, it is important that values are not correlated across the groups.

Info: Standard error

There are two definitions for the standard error.

First: A standard error is the standard deviation of the sampling distribution of some statistic (e.g., the mean, the difference between two means, the correlation coefficient, etc.)

It is now important to understand what a sampling distribution is. An example will bring more light into that. If we were to take 1,000 different random samples of women, each of 100, and compute the average shoe size of each sample, these 1,000 sample means would form their own distribution. This distribution would be called the sampling distribution of the mean. So this sampling distribution has a mean and standard deviation as well, so as said this standard deviation is the standard error. The standard deviation got already introduced in chapter 3.

Second: The standard error is the denominator in the formulas used to calculate many inferential statistics. The reason for that is that the standard error is the measure of how much random variation we would expect from samples of equal size drawn from the same population.

Formula for calculating the standard error of the mean.

$$\sigma_{\overline{x}} = \frac{\sigma}{\sqrt{n}}$$

where σ is the standard deviation for the population and n the size of the sample.[10]

Info: The Ordinary Least Squares Estimator

The OLS estimator predicts the regression coefficients so that the estimated regression line is as close as possible to the observed data, where closeness is measured by the sum of the squared mistakes made in predicting Y given X.

The sample mean $\overline{Y}$,is the least squares estimator of the population mean. $\overline{Y}$ minimizes the total squared estimation mistakes $\sum_{i=1}^{n}(Y_i - m)^2$ among all possible estimators m.

The OLS estimator adapts this concept to the linear regression model. Let b_0 oand b_i be some estimators of β_0 and β_1. According to our linear regression model, the regression line is then defined as $Y_i = b_0 + b_1X$.

Thus the mistake made in predicting the i^{th} observation is $Y_i - (b_0 + b_1X_i) = Y_i - b_0 - b_1X_i$.The sum of these squared prediction mistakes over all n observations is

$$\sum(Y_i - b_0 - b_1X_i)^2$$

The estimators of the intercept and slope that minimize the sum of squared mistakes in expression (1) are called the **ordinary least squares (OLS) estimators** of β_0 and β_1.

OLS has its own special notation and terminology. The OLS estimator of β_0 is denoted as $\hat{\beta_0}$, and the OLS estimator of β_1 is denoted as $\hat{\beta_1}$. The **OLS regression line**, also called the **sample regression line** or **sample regression function**, is the straight line constructed using the OLS estimators: $\hat{\beta_0} + \hat{\beta_1}X$.

The **predicted value** of Y_i given X_i, based on the OLS regression line, is $Y_i = \beta_0 + \beta_1X_i$.

The OLS estimators for the slope β_1 and the intercept β_0 are

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2} = \frac{s_{XY}}{s_{X^2}}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1\bar{X}$$

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1X_i, i = 1, \ldots, n$$

[10] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010

The estimated intercept ($\hat{\beta}_0$), slope ($\hat{\beta}_1$) are computed from a sample of n n observations of X_i and $Y_i, i = 1, \ldots, n$. These are the estimates of the unknown true population intercept ($\hat{\beta}_0$) and slope ($\hat{\beta}_1$).[11]

Info: The Ordinary Least Squares Estimator with multiple regressors

We saw how to estimate the intercept and slope coefficients in the single-regressor model by applying OLS to a sample of observations of Y and X. The key concept is to minimize the sum of squared prediction mistakes by choosing the estimators b_0 and b_1 so as to minimize $\sum_{i=1}^{n}(Y_i - b_0 - b_1 X_i)^2$ in order to predict the coefficients. These estimators are called $\hat{\beta}_0$ and $\hat{\beta}_1$.

The method can also be used to estimate multiple coefficients $\beta_0, \beta_1, \ldots, \beta_k$ as in the multiple regression model. Meaning, $b_0, b_1, \ldots, b_k$ are the estimators of $\beta_0, \beta_1, \ldots, \beta_k$. Analog to the OLS in the single regressor model, the predicted value of Y_i is $b_0 + b_1 X_{1i} + \ldots + b_k X_{ki}$. Thus, the mistake in predicting Y_i is $Y_i - (b_0 + b_1 X_{1i} + \ldots + b_k X_{ki}) = Y_i - b_0 - b_1 X_{1i} - \ldots - b_k X_{ki}$. The sum of these squared prediction mistakes over all n observations thus is

(1)

$$\sum(Y_i - b_0 - b_1 X_{1i} - \ldots - b_k X_{ki})^2$$

The estimators of the coefficients $\beta_0, \beta_1, \ldots, \beta_k$ that minimize the sum of squared mistakes in Expression (1) are called the **ordinary least squares (OLS) estimators** of $\beta_0, \beta_1, \ldots, \beta_k$.

The OLS estimators are denoted $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k$. The **OLS regression line** is the straight line constructed using the OLS estimators: $\hat{\beta}_0 + \hat{\beta}_1 X_1 + \ldots + \hat{\beta}_k X_k$. The **predicted value** of Y_i given $X_{1i}, \ldots, X_{ki}$, based the OLS regression line is $\overline{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{1i} + \ldots + \hat{\beta}_k X_{ki}$.[12]

Wine is the unit of observation, but is actually only present as the index i. It is allowed for new vintage or new nominal prices to affect the demand by the inclusion of fixed effects α (that are different for every wine, vintage and price combination). We control demand shocks with the fixed effects δ for each week x segment.

Info: Fixed effect

$$Y_{it} = \beta_1 X_{it} + \alpha_i + u_{it}, i = 1, \ldots, n, t = 1, \ldots, T$$

The error term has conditional mean zero, given all T values of X for that entity. This assumption implies that there is no omitted variable bias. The requirement that the conditional mean of u_{it} not depend on any of the values of X for that entity-past, present, or future-adds. This assumption is violated if current u_{it}, is correlated with past, present, or future values of X.

[11] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p. 114f

[12] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p. 190

$$u_{it}\text{has conditional mean zero:}\, E(u_{it}|X_{i1}, X_{i2}, ..., X_{iT}, \alpha_i) = 0$$

For multiple regressors, X_{it} should be replaced by the full list $X_{1,it}, X_{2,it}, ..., X_{k,it}$[13]

---

b) The Error Term & Demand shocks

Let us take a closer look at the error term.

$$E(\eta|\alpha, \delta, R, R^{good}, R^{bad}, ADVERT) = 0$$

This condition would not hold if reviews were simultaneously correlated with demand shocks. For example, Champagnes are rather reviewed in the week before New Years' eve. Demand shocks are included in the error term \eta. These demand shocks are not allowed to correlate with the reviews, since they are the main explanatory variable. Otherwise we would have an **omitted variable bias**. Omitted variable bias appear when a variable is correlated with the dependent and one or more explanatory variables, but is omitted from the regression. So if the demand shocks are actually correlated with the reviews, but are included in the error term, then there is omitted variable bias.

The consequence would be that the conditional mean would not be zero. Thus, we had false results. So, it is always very important to prevent situations where you have omitted variable bias.

So there are possibilities that demand shocks trigger a review and this can influence our results.

We have two ways to address this problem.

1. Included four leads on all time-varying explanatory variables in our regressions.

For example, if a review appears as a response to popularity then the lead is going to be positive. This because a lead is basically a review indicator in the future. For example, if we have a review indicator in period 2. Then the lead by 1 would give us this indicator in period 1. So if we assume that a wine is very popular and therefore got high sales in period 1 and the review would appear as a response to that popularity. Then the lead by 1 would be positive. * 2. Redid analysis

Only the wines from the AoM event are being used. AoM (Allt om Mat), is the anual wine tasting issue. Most of the wines in the regular assortment are being reviewed on that event. However, this basically means the majority of red and white wines are included, since basically all of them are being reviewed on this event. These wines have a predetermined publication date. This makes sure that the timing of the reviews is independent to demand shocks. We will come back to this later on.

In order to make the estimation even more robust and to control wine x vintage x price fixed effects as well as demand shocks, we used weekly data. The high frequency most likely makes sure that there is a low correlation regarding advertising and unusual demand shocks.

[13] Stock, J. H., Watson, M. W. (2010): Introduction to Econometrics. Third Edition, Pearson Addison Wesley, page 362 & 363

This exercise refers to page 202 & 203 in the paper

Exercise 9 Effects of Reviews and Advertising on Demand for Wine

a) Introduction

Now that we know the theoretical model behind the regressions, let us take a look at the results for the full sample of wines and reviews.

We estimated the effects based on equation (1):

$$lnQ_{ijkt} = \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}^{good} R_{it-l}^{good} + \sum_{l=-4}^{25} \alpha_{t-l} R_{it-l} + \sum_{l=-4}^{25} \alpha_{t-l}^{bad} R_{it-l}^{bad} + \sum_{l=-4}^{25} \gamma_{t-l} ADVERT_{it-l} + \eta_{ijkt}$$

In the first estimation we are going to create scatter plots for the effect of reviews itself, good reviews, bad reviews and advertising expenditures. The plot will contain the 4 leads and 25 lags in order to see how long the effect lasts. Every estimation for every week will have a 95 % confidence interval. Click on the info button for detailed information regarding the scatter plot.

Info: Scatter plot

A scatter plot is a plot of n observations on (X_i, X_i) and (Y_i, Y_i) in which each observation is represented by the point $(X_iX_i\ ,\ Y_iY_i)$.[14]

---

b) Regression

The next step is to execute the regression in order to get the estimations for the scatter plot. The package lfe contains the function felm(). This function allows us the implementation of clustered standard error and multiple group fixed effects.

Info: felm

The first part of the felm function required the formula for the regression. Afterwards, we can list the factors, the factors are separated with a | from the formula. Multiple factors are linked with a +.

We can also state independent variables after a next |. We are not going to do that in this regression. We will just write down a 0.

After the next |, we are able to state the cluster variables.

The felm function basically looks like this:

$$y \sim x_1 + \ldots + x_n \mid factor_1 + \ldots + factor_n \mid 0 \mid cluster_1 + \ldots + cluster_n$$

For further information, click this Inside-R.org link

[14] Stock, J. H., Watson, M. W. (2010): Introduction to Econometrics. Third Edition, Pearson Addison Wesley. page 91

Let's create our formula. We want to estimate the effects for reviews, good reviews, bad reviews and advertising expenditures. This means we need to add the corresponding variables.

- rev_all is the indicator of the wine being reviewed in all media. This is the variable for a review per se.
- rev_all_hi is the indicator of the wine receiving a good review in all media. Clearly the variable for good reviews.
- rev_all_lo is the indicator of the wine receiving a bad review in all media. Variable for bad reviews.
- ma_split is the variable for advertising expenditures in a specific week for a wine.

Due to the fact that we also take a look at the leads and lags of each variable, we are going to have a quite long formula that looks a bit complicated but it is fairly easy. We want to estimate how the mentioned variables effect the sales therefore we need to have the llitre at the beginning of the formula. llitre is litre in log, so the weekly sales in log litre.

Info: Logarithms

A possible way to specify a nonlinear regression is by using the natural logarithm of Y and/or X. The logharitm converts the changes in variables into percentage changes. The estimators usually show then the amount of change when the corresponding estimator is increased by 1%.[15]

The full formula needs to look like this:

llitre ~ rev_all + rev_all_lag_1 + rev_all_lag_2 + rev_all_lag_3 + ... + rev_all_lag_25 + rev_all_lead_1 + ... + rev_all_lead_4 + rev_all_hi + rev_all_hi_lag_1 + rev_all_hi_lag_2 + rev_all_hi_lag_3 + ... + rev_all_hi_lag_25 + rev_all_hi_lead_1 + ... + rev_all_hi_lead_4 + rev_all_lo + rev_all_lo_lag_1 + rev_all_lo_lag_2 + rev_all_lo_lag_3 + ... + rev_all_lo_lag_25 + rev_all_lo_lead_1 + ... + rev_all_lo_lead_4 + ma_split + ma_split_lag_1 + ma_split_lag_2 + ma_split_lag_3 + ... + ma_split_lag_25 + ma_split_lead_1 + ... + ma_split_lead_4 which is:

$$llitre_{ijkt}$$

$$= \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}\ \text{rev\_all}_{it-l} + \sum_{l=-4}^{25} \alpha_{t-l}^{good}\ \text{rev\_all\_hi}_{it-l} + \sum_{l=-4}^{25} \alpha_{t-l}^{bad}\ \text{rev\_all\_lo}_{it-l}$$

$$+ \sum_{l=-4}^{25} \gamma_{t-l}\ \text{ma\_split}_{it-l} + \eta_{ijkt}$$

α is a fixed effect, defined by wine x vintage x price combination, which is artikpr with the indices j.

δ is a sparate fixed effect, defined by week t x segment k (compensation for time trends & sales), which is time_segm_price with the indices kt.

[15] James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley, 2010, p.265

η is the error term with wine i, vintage x price combination j and week t, which is clustered at the brand level artikelid.

As already explained, the regression contains two fixed effects.

The first fixed effect is time_segm_price. It is the Period-color-price segment-package indicator. It accounts for separate fixed effects which accommodated different time trends in sales. It captures for example, bottled red wines sold at price above SEK 110 per bottle in week 128.

The second fixed effect is artikpr. It is Product number-price-vintage combination. It makes sure that a new vintage or new price is associated with a new fixed effect.

In order to get the correct results from the ordinary least squares model, we need one particular condition to hold.

$$E(\eta_{ijkt}|\alpha, \delta, R, R^{good}, R^{bad}, ADVERT) = 0$$

or

$$E(\eta_{ijkt}|\text{artikpr}, \text{time\_segm\_price}\delta, \text{rev\_all}, \text{rev\_all\_hi}, \text{rev\_all\_lo}, \text{ma\_split}) = 0$$

This condition only holds if the reviews are not simultaneously correlated with demand shocks, that we do not capture with any time effects. Otherwise we had a omitted variable bias.

As already mentioned, the paper used only a certain sample for certain estimations in order to get better results.

This time we are going to load the dataset datareg. We applied the standard filter regarding distribution levels and filtered out observations with empty leads for rev_all.

Remember: rev_all is the indicator of the wine being reviewed in all media. Now load datareg into a new dataset named fig3.

Just check check the chunk to load the data in case you want to do the two optional chunks.

```
# Load datareg into fig3.
fig3 <- readRDS(file="datafilterreg.RDS")
```

The following chunk shows the code for the required regression that would give us the desired estimators for the regressors and the corresponding clustered standard errors.

Demonstration chuck, do not run

```
# Executing regression
fig3reg <- felm(llitre ~ rev_all + rev_all_lag_1 + rev_all_lag_2 + rev_all_lag_3 + rev_all_lag_4 +
rev_all_lag_5 + rev_all_lag_6 + rev_all_lag_7 + rev_all_lag_8 + rev_all_lag_9 + rev_all_lag_10
+ rev_all_lag_11 + rev_all_lag_12 + rev_all_lag_13 + rev_all_lag_14 + rev_all_lag_15 +
rev_all_lag_16 + rev_all_lag_17 + rev_all_lag_18 + rev_all_lag_19 + rev_all_lag_20 +
rev_all_lag_21 + rev_all_lag_22 + rev_all_lag_23 + rev_all_lag_24 + rev_all_lag_25+
rev_all_lead_1 + rev_all_lead_2 + rev_all_lead_3 + rev_all_lead_4 + rev_all_hi +
rev_all_hi_lag_1 + rev_all_hi_lag_2 + rev_all_hi_lag_3 + rev_all_hi_lag_4 + rev_all_hi_lag_5 +
rev_all_hi_lag_6 + rev_all_hi_lag_7 + rev_all_hi_lag_8 + rev_all_hi_lag_9 + rev_all_hi_lag_10 +
rev_all_hi_lag_11 + rev_all_hi_lag_12 + rev_all_hi_lag_13 + rev_all_hi_lag_14 +
rev_all_hi_lag_15 + rev_all_hi_lag_16 + rev_all_hi_lag_17 + rev_all_hi_lag_18 +
rev_all_hi_lag_19 + rev_all_hi_lag_20 + rev_all_hi_lag_21 + rev_all_hi_lag_22 +
rev_all_hi_lag_23 + rev_all_hi_lag_24 + rev_all_hi_lag_25 + rev_all_hi_lead_1 +
rev_all_hi_lead_2 + rev_all_hi_lead_3 + rev_all_hi_lead_4 + rev_all_lo + rev_all_lo_lag_1 +
rev_all_lo_lag_2 + rev_all_lo_lag_3 + rev_all_lo_lag_4 + rev_all_lo_lag_5 + rev_all_lo_lag_6 +
rev_all_lo_lag_7 + rev_all_lo_lag_8 + rev_all_lo_lag_9 + rev_all_lo_lag_10 + rev_all_lo_lag_11
+ rev_all_lo_lag_12 + rev_all_lo_lag_13 + rev_all_lo_lag_14 + rev_all_lo_lag_15 +
rev_all_lo_lag_16 + rev_all_lo_lag_17 + rev_all_lo_lag_18 + rev_all_lo_lag_19 +
rev_all_lo_lag_20 + rev_all_lo_lag_21 + rev_all_lo_lag_22 + rev_all_lo_lag_23 +
rev_all_lo_lag_24 + rev_all_lo_lag_25 + rev_all_lo_lead_1 + rev_all_lo_lead_2 +
rev_all_lo_lead_3 + rev_all_lo_lead_4 + ma_split + ma_split_lag_1 + ma_split_lag_2 +
ma_split_lag_3 + ma_split_lag_4 + ma_split_lag_5 + ma_split_lag_6 + ma_split_lag_7 +
ma_split_lag_8 + ma_split_lag_9 + ma_split_lag_10 + ma_split_lag_11 + ma_split_lag_12 +
ma_split_lag_13 + ma_split_lag_14 + ma_split_lag_15 + ma_split_lag_16 + ma_split_lag_17 +
ma_split_lag_18 + ma_split_lag_19 + ma_split_lag_20 + ma_split_lag_21 + ma_split_lag_22 +
ma_split_lag_23 + ma_split_lag_24 + ma_split_lag_25 + ma_split_lead_1 + ma_split_lead_2 +
ma_split_lead_3 + ma_split_lead_4 | time_segm_price + artikpr | 0 | artikelid, data = fig3)
```

c) Prepare a dataframe for plotting

The next step is to prepare a dataset for the plotting.

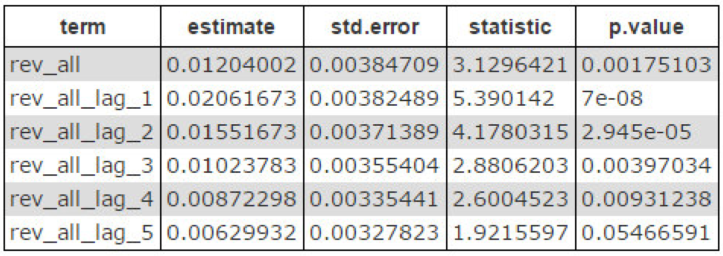
We would extract the estimations, standard errors etc. to a new data frame in order to make further calculations and adjustments. We could easily do this with the function tidy() from the broom package. The tidy() function constructs a data frame that summarizes the model's statistical findings. For more information, click on the following Cran R Project Link

Load up the stored results from tidyfig3.

```
# Load up tidyfig3.RDS into dffig3.
dffig3 <- readRDS(file="tidyfig3.RDS")
```

Let's take a look at the results. Display dffig3 with the head() function.

```
# Display dffig3 with the head function.
head(dffig3)
```



term	estimate	std.error	statistic	p.value
rev_all	0.01204002	0.00384709	3.1296421	0.00175103
rev_all_lag_1	0.02061673	0.00382489	5.390142	7e-08
rev_all_lag_2	0.01551673	0.00371389	4.1780315	2.945e-05
rev_all_lag_3	0.01023783	0.00355404	2.8806203	0.00397034
rev_all_lag_4	0.00872298	0.00335441	2.6004523	0.00931238
rev_all_lag_5	0.00629932	0.00327823	1.9215597	0.05466591

As you can see, the tidy() function from the broom package extracted the terms respectively the repressor, the estimation, the clustered standard error, the t value and the p value. We just need estimation and the clustered standard error.

Since we want to plot a 95% interval, we need to make some calculations. We need to calculate an upper and a lower value for each estimation to get the confidence interval range.

For the upper value of the interval:

$$Estimation + 1.96 * standarderror$$

For the lower value of the interval:

$$Estimation - 1.96 * standarderror$$

So why are we using this formula. The calculation for the confidence interval.

$$CI_{95} = \overline{X} + -(t_{95})(s_{\overline{x}})$$

where CI_{95} is the 95% confidence interval, $\overline{X}$ is the sample mean, $s_{\overline{x}}$ is the standard error and t_{95} is the t value for a two-tailed test, alpha level of .05 with a given degrees of freedom.

We just exchange $\overline{X}$ with our estimator. The 1.96 is the value for t_{95} with $df = \infty$ and $\alpha = .05$[16]

For more information regarding the terms above, click on the corresponding info button.

Info: 95% Confidence interval

Confidence intervals show us the statistical significance. It indicates how meaningful our results from the statistical analysis are, thus more and more reports include confidence intervals to support the meaning of the results. When someone a sample data to make estimations about the population, he usually does not know the actual value. Probabilities and the confidence interval help to make an educated prediction.

Formula for calculating the 95% confidence intervals for the mean:

$$CI_{95} = \overline{X} \pm (t_{95})(s_{\overline{x}})$$

where CI_{95} is the 95% confidence interval, $\overline{X}$ is the sample mean, $s_{\overline{x}}$ is the standard error and t_{95} is the t value for a two-tailed test, alpha level of .05 with a given degrees of freedom.

So when we calculate CI_{95}, we would be able to say with 95% confidence in what interval the mean of the sample is.[17]

Info: Degree of freedom

The degree of freedom is a main concept for estimating statistic of populations. It is usually abbreviated to df. It is considered as mathematical restriction when estimating one statistic from an estimator of another.

Most of the time when we use the degree of freedoms for regressions, analysis of variance or other estimations, we loose of degree of freedom.

In the case of the t-value, the degree of freedom is $n - 1$ where n is our sample size.

Source: Statsdirect (visited 26.07.2016)

---

[16] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010 & James H. Stock, Mark W. Watson, Introduction to Econometrics, Third edition Addison-Wesley (2010), p. 151f.

[17] Timothy C. Urdan, Statistics in Plain English, Third Edition Routledge, 2010

Info: t-value

The t-statistic

The standardized sample average $\frac{\overline{Y}-\mu_{Y,0}}{\mathrm{SE}(\overline{Y})}$ plays a central role in testing statistical hypotheses and has a special name, **t-statistic** or **t-ratio**.

$$t = \frac{\overline{Y}-\mu_{Y,0}}{SE(\overline{Y})}$$

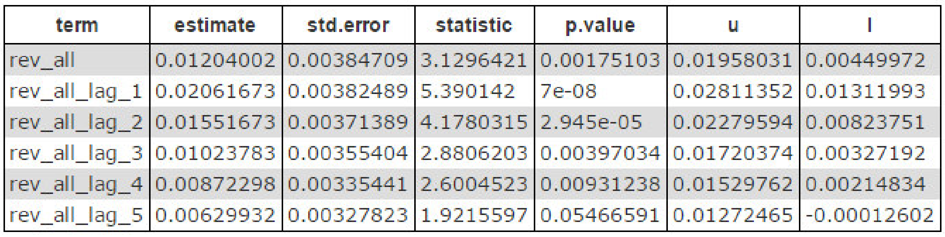
In general, a **test statistic** is a statistic used to perform a hypothesis test. The t-statistic is an important example of a test statistic.

Large-sample distribution of the t-statistic. When n is large, s_Y^2 is close to σ_Y^2 with high probability. Thus the distribution of the t-satistic is approximately the same as the distribution of $\frac{\overline{Y}-\mu_{Y,0}}{\sigma_{\overline{Y}}}$.

t is approximately distributed $N(0,1)$ for lage n.[18]

So let us calculate the upper and lower values for our confidence interval.

```
# Calculating the upper value.
dffig3$u <- dffig3$estimate+1.96*dffig3$std.error
# Calculating the lower value.
dffig3$l <- dffig3$estimate-1.96*dffig3$std.error
# Displaying a part of the data frame.
head(dffig3)
```



term	estimate	std.error	statistic	p.value	u	l
rev_all	0.01204002	0.00384709	3.1296421	0.00175103	0.01958031	0.00449972
rev_all_lag_1	0.02061673	0.00382489	5.390142	7e-08	0.02811352	0.01311993
rev_all_lag_2	0.01551673	0.00371389	4.1780315	2.945e-05	0.02279594	0.00823751
rev_all_lag_3	0.01023783	0.00355404	2.8806203	0.00397034	0.01720374	0.00327192
rev_all_lag_4	0.00872298	0.00335441	2.6004523	0.00931238	0.01529762	0.00214834
rev_all_lag_5	0.00629932	0.00327823	1.9215597	0.05466591	0.01272465	-0.00012602

Perfect, now we have our 95% confidence interval for the estimations.

So let us take a look what that means in terms of the value range. We are going to interpret the values later with the different plots. So we would say with 95% confidence that the population estimation for rev_all, thus the effect of a review, ranges from approximately 0.0196 to 0.0045 with a sample estimation of approximately 0.0120.

Furthermore, we used a finite distributed lag model with 25 lags and 4 leads to allow for the possibility that not all the effects of a review are simultaneous. Thus, we need to add a week variable in order to be able to plot the lags and leads.

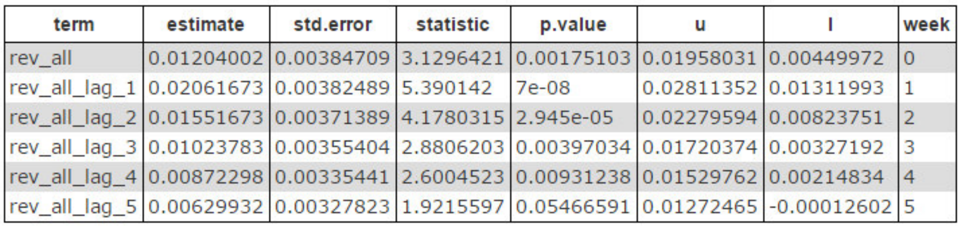
We can just use the $ in addition to the data frame name dffig3 to add a week column. We also now the structure of column that represents the lag and lead names. Take a look at your last

[18] Stock, J. H., Watson, M. W. (2010): Introduction to Econometrics. Third Edition, Pearson Addison Wesley, page 75

result. It starts with rev_all and continues with the first lag afterwards. Then it goes all the way up to the 25th lag and the 4 leads. All the explanatory variables have the same structure. This means that we need a sequence such as 0,1,2,...,25,-1,...,-4. We can simply define this with c(0:25,-1:-4).

So let us create the week column to dffig3.

```
# Add the week column with the explained sequence to dffig3.
dffig3$week <- c(0:25,-1:-4)
head(dffig3)
```



term	estimate	std.error	statistic	p.value	u	l	week
rev_all	0.01204002	0.00384709	3.1296421	0.00175103	0.01958031	0.00449972	0
rev_all_lag_1	0.02061673	0.00382489	5.390142	7e-08	0.02811352	0.01311993	1
rev_all_lag_2	0.01551673	0.00371389	4.1780315	2.945e-05	0.02279594	0.00823751	2
rev_all_lag_3	0.01023783	0.00355404	2.8806203	0.00397034	0.01720374	0.00327192	3
rev_all_lag_4	0.00872298	0.00335441	2.6004523	0.00931238	0.01529762	0.00214834	4
rev_all_lag_5	0.00629932	0.00327823	1.9215597	0.05466591	0.01272465	-0.00012602	5

Perfect. Now, data preparations are finished. Next, we want to create the first plot for the estimations of rev_all, so the estimations for the affect of a review.

d) The plots

We are going to use the plot command. The prepared data frame dffig3 has all the values we need. But remember, dffig3 has the estimations for all the variables. We only need the first 30 (25 lags, 4 leads & the variable itself) values for the review plot.

plot(x, y, ...) is a function to plot R objects.

The function needs values for the x and y coordinates, so it is basically a scatter plot, which already got explained in an info section. Further arguments can be passed to manipulate the appearance.

- main determinates the title. xlab & ylab are the labels for the x & y coordinates.
- xlim & ylim define the limits of the x & y coordinates.
- pch is a graphical parameter and defines the shape of the point.
- bg determinates the color of the points.

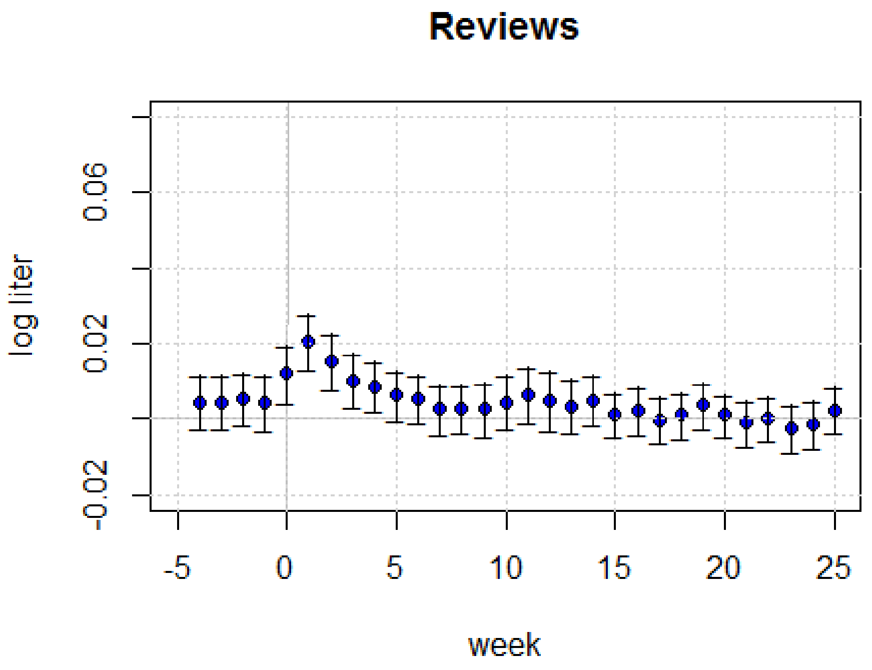
We also added lines around the (0,0) points for better visualization. This is done with the combination of the panel.first and abline commands. *panel.first is an expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids or scatterplot smooths. Note that this works by lazy evaluation: passing this argument from other plot methods may well not work since it may be evaluated too early.

For further information, click this Inside-R.org link or Inside-R.org link

So let's plot the first estimations.

```
# Reviews plot.
# Only the first 30 values describe the rev_all variable.
# Using the limits -0.02 & 0.08 for the y coordinates and -5 & 25 for the x coordinates.
plot(dffig3$week[1:30],dffig3$estimate[1:30], main = 'Reviews', xlab = 'week', ylab = 'log
liter',ylim=c(-0.02,0.08), xlim=c(-5,25), pch = 21, bg = 'blue', panel.first = c(abline(a = 0, b = 1,
lty = 1, col = 'grey'), abline(a = 0, b = 0, lty = 1,  col = 'grey')))
```

```
grid(nx = NULL, ny = NULL)
# Plotting a line between the upper and lower value of the 95% confidence interval.
segments(dffig3$week[1:30], dffig3$l[1:30], dffig3$week[1:30], dffig3$u[1:30])
# Plotting points in the shape of '_' for the upper and lower value of the 95% confidence interval.
points(dffig3$week[1:30],dffig3$l[1:30], pch = '_')
points(dffig3$week[1:30],dffig3$u[1:30], pch = '_')
```



The commands grid, segments and points were also used. These commands are rather intuitive. grid creates a grid on the plot, segments connects two points with a straight line and points plotted points on the plot.

As you can see, a review increases the sales by 1.2 percent in the same week of the review appearance. It goes up to 2 percent within the first week and decreases slowly the weeks after. The effect is quite stable but becomes basically zero after five weeks. You can also see the upper and lower value of the 95% confidence interval ("_" points) and the line in-between those that represents the range of the interval. To repeat it, we can say with 95% confidence that the population estimation for the corresponding value is within this range.

By adding up the betas of the reviews give an increment of 555 liters, this translates into a raise of SEK 55278 (~ USD 7537) .

So let us plot the estimations for good reviews, bad reviews and advertising expenditures next in the same way. Remember: dffig3 has all the values we need which means we just need to change the indices and the title. Except for the advertising expenditures. We also need to change ylim due to its values. The code was stored in a string named codeplot which we simply manipulate to get our desired results. Let us keep codeplot untouched and name the new string codeplot1 and so on for the different plots.

Load up codeplot as the code for the plots as a string.

```r
# Loading the code for the plot as string, basically the same as above.
codeplot <- readRDS(file="codeplot.RDS")
```

The function gsub() from the base package is a good way to manipulate the string. We just need to pass a pattern, the replacement and the string we want to manipulate. For further information, click here Inside-r.org Link

Afterwards, we can just use the manipulated string as code. We need to combine two base functions to do so. First we need the parse() function for parsing. We are just going to pass the string as the text argument. Click the info button for some detailed information.

Info: parse()

parse(file = "", n = NULL, text = NULL, prompt = "?", keep.source = getOption("keep.source"), srcfile, encoding = "unknown")

text: character vector. The text to parse. Elements are treated as if they were lines of a file. Other R objects will be coerced to character if possible.

Source: Stat.ethz.ch Link

Then, we are going to pass the parsed string to the eval() function. This function basically executes the parsed string as code. For more information, click the info button.

Info: eval()

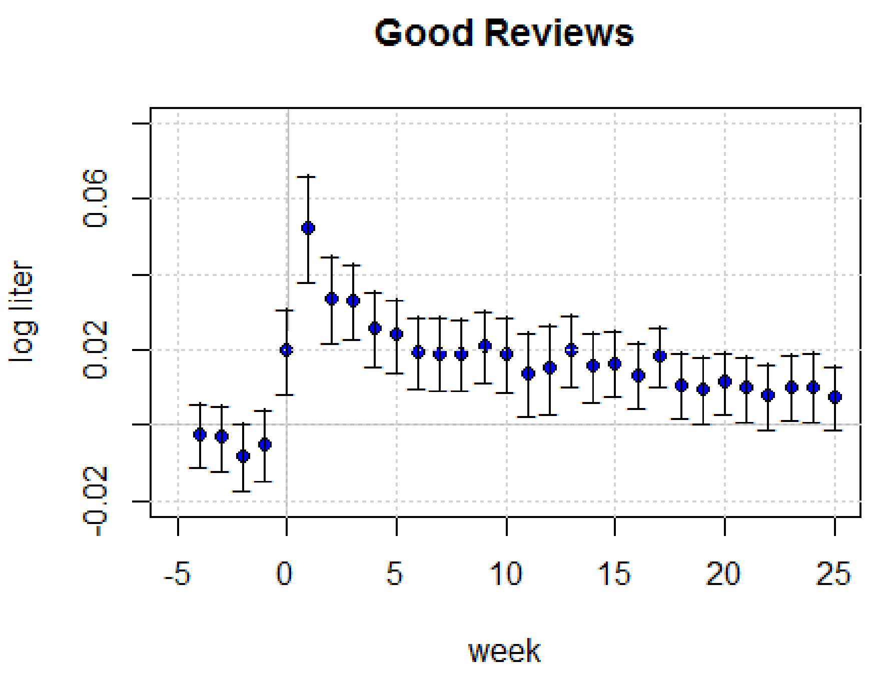
Evaluate an R expression in a specified environment.

eval(expr, envir = parent.frame(), enclos = if(is.list(envir) || is.pairlist(envir)) parent.frame() else baseenv())

More explanations under "Details" at the source.

Source: Stat.ethz.ch Link

```r
# Good reviews.
# Changing the indices.
codeplot1 <- gsub('1:30','31:60',codeplot)
# Changing the title.
codeplot1 <- gsub('Reviews','Good Reviews',codeplot1)
# Running the code.
eval(parse(text = codeplot1))
```

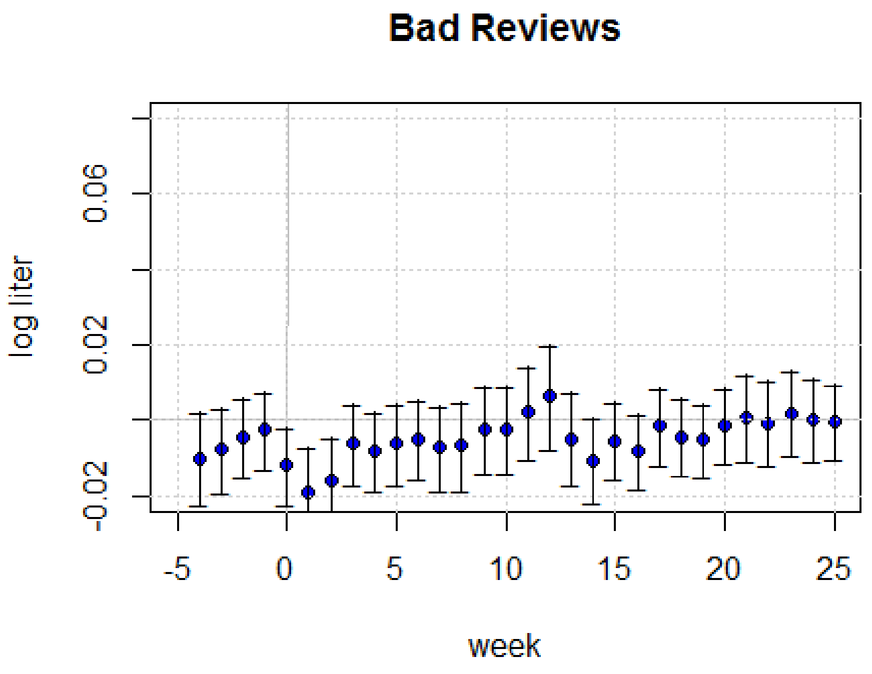


The estimation shows that a good review increases the sales by an additional 2 percent. The effect even peaks one week after at 5.2 percent increment. Although it decreases to approximately 3.3 percent, it stays stable afterwards and the effect stays significant up to 22 weeks.

The same calculation for the betas of the reviews and good reviews and an evaluation at a mean volume of 8218 liters per week gives us an increment of 4418 liters volume. This is approximately half a week of sales. This translated with a mean price of SEK 99.6 means a raise of SEK 440000 (~ USD 60000).

Now the plot for bad reviews.

```
# Bad Reviews.
# Changing the indices.
codeplot2 <- gsub('1:30','61:90',codeplot)
# Changing the title.
codeplot2 <- gsub('Reviews','Bad Reviews',codeplot2)
# Running the code.
eval(parse(text = codeplot2))
```

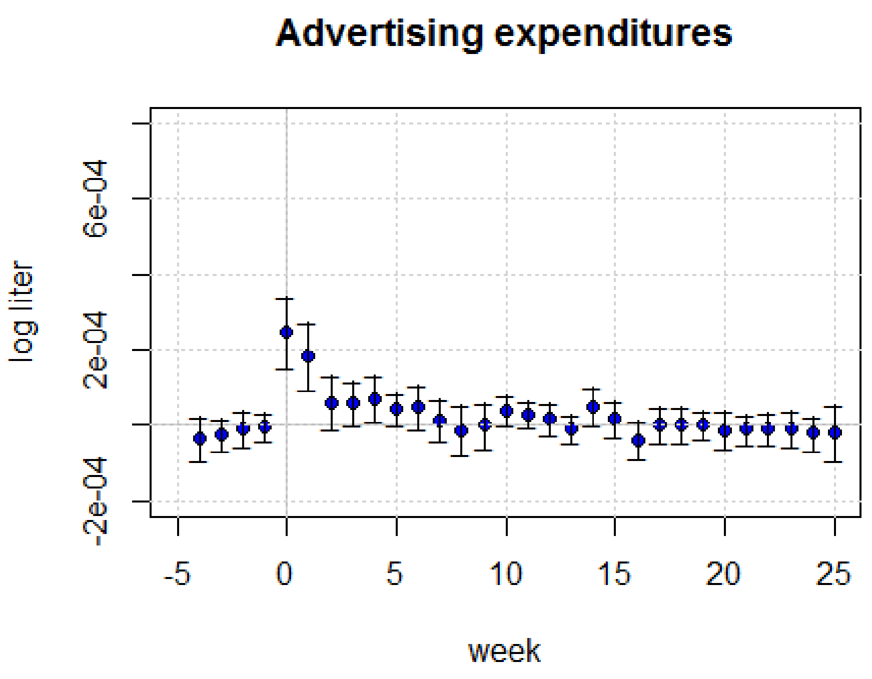


A bad review has indeed a negative coefficient, but the effect is weak and counterbalanced by the effect of being reviewed at all.

Considering reviews and bad reviews together, the negative impact is close to zero, while the positive impact of a review per se slightly outweighs the negative impact which leads to an increment of 170 liters, meaning SEK 16932 (~ USD 2308).

Now the advertising expenditures.

```
# Advertising expenditures.
# Changing the indices.
codeplot3 <- gsub('1:30','91:120',codeplot)
# Changing the title.
codeplot3 <- gsub('Reviews','Advertising expenditures',codeplot3)
# Changing ylim.
codeplot3 <- gsub('-0.02,0.08','-0.0002,0.0008',codeplot3)
# Running the code.
eval(parse(text = codeplot3))
```



The effect of advertising expenditures is rather short lived and low. Every SEK 1000 spent on advertising increases the sales by 0.025 percent in the same week of appearance. The effect is lightly lower the week after and pretty much close to zero after two weeks.

Even though we are also discussing advertising expenditures. The main focus are the reviews. As a side note: The endogeneity of advertising should not worry too much due to the fact that we have data with high frequency in terms of weekly periods. Furthermore, remember we also control the fixed effects of the segments.

Info: Endogeneity

When there is correlation between a repressor and the error term, that repressor is said to be endogenous; when no such correlation exists the repressor is said to be exogenous. It is important to prevent possible endogeneity.[19]

Furthermore, advertising is introduced in a rather simple fashion in this article, meaning it does not take possible threshold effects into account. That being said, the impact of advertising is really small. At the mean advertising level, the volume would just increase by 320 liters for a week of advertising, translating into SEK 31872 (~ USD 4346). However, the average weekly advertising expenditure is only SEK 6900 (~ USD 940). In comparison, the wine with the highest average advertising expenditures spends on average SEK 96000 (~ USD 13000) per week. This leads to a increment of 525 liters, meaning SEK 52290 (~ USD 7130).

[19] Peter Kennedy, A Guide to Econometrics, Sixth Edition Blackwell, 2008, p. 139.

This exercise refers to page 203-206 in the paper

Exercise 10 AoM wines

a) Indicators

Let us also take a look at wines that are being "good" reviewed on the AoM event and that are not. Due to fact that we have new indicators, we need to create a new base data frame. We also need lags and leads for following variables:

- rev_ex_hi is the ndicator of the wine receiving a good review in AoM.
- rev_ex_lo is the indicator of the wine receiving a bad review in AoM.
- rev_nyaom is the indicator of review is not in AoM yearly special.
- rev_nyaom_hi is the indicator of good reviews not in AoM yearly special.
- rev_nyaom_lo is the indicator of bad reviews not in AoM yealy special.
- rev_all is the indicator of the wine being reviewed (all media).
- ma_split Advertising expenditures a specific week for a wine.

b) Wines from AoM yearly special

Load up the reequired datasets first. We will use the dataset dataaom that consists of all required lags and leads. We also need to load dffig3 from the previous exercise. Check the chunk.

```
# Load dataaom.
# Load dffig3.
dataaom <- readRDS(file="dataaom.RDS")
dffig3 <- readRDS(file="dffig3.RDS")
```

So first let us estimate the effect for the wines that were being reviewed on tha yearly AoM event. We are going to use the explanatory variables rev_ex_hi, rev_ex_lo and ma_split. The formula consists again of all 25 lags, 4 leads and the base variable.

$$llitre_{ijkt} = \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}\ \text{rev\_ex\_hi}_{it-l} + \sum_{l=-4}^{25} \alpha_{t-l}^{bad}\ \text{rev\_ex\_lo}_{it-l} + \sum_{l=-4}^{25} \gamma_{t-l}\ \text{ma\_split}_{it-l} + \eta_{ijkt}$$

The fixed effects and the cluster variable are the same with:

α is a fixed effect, defined by wine x vintage x price combination, which is artikpr with the indices j.

δ is a sparate fixed effect, defined by week t x segment k (compensation for time trends & sales), which is time_segm_price with the indices kt.

η is the error term with wine i, vintage x price combination j and week t, which is clustered at the brand level artikelid.

Demonstration chunk, do not run

```
# Regression for wines that have been reviewed on the yearly AoM special.
aomreg <- felm(llitre ~ rev_ex_hi + rev_ex_hi_lag_1 + rev_ex_hi_lag_2 + rev_ex_hi_lag_3 +
rev_ex_hi_lag_4 + rev_ex_hi_lag_5 + rev_ex_hi_lag_6 + rev_ex_hi_lag_7 + rev_ex_hi_lag_8 +
rev_ex_hi_lag_9 + rev_ex_hi_lag_10 + rev_ex_hi_lag_11 + rev_ex_hi_lag_12 +
rev_ex_hi_lag_13 + rev_ex_hi_lag_14 + rev_ex_hi_lag_15 + rev_ex_hi_lag_16 +
rev_ex_hi_lag_17 + rev_ex_hi_lag_18 + rev_ex_hi_lag_19 + rev_ex_hi_lag_20 +
rev_ex_hi_lag_21 + rev_ex_hi_lag_22 + rev_ex_hi_lag_23 + rev_ex_hi_lag_24 +
rev_ex_hi_lag_25 + rev_ex_hi_lead_1 + rev_ex_hi_lead_2 + rev_ex_hi_lead_3 +
rev_ex_hi_lead_4 + rev_ex_lo + rev_ex_lo_lag_1 + rev_ex_lo_lag_2 + rev_ex_lo_lag_3 +
rev_ex_lo_lag_4 + rev_ex_lo_lag_5 + rev_ex_lo_lag_6 + rev_ex_lo_lag_7 + rev_ex_lo_lag_8 +
rev_ex_lo_lag_9 + rev_ex_lo_lag_10 + rev_ex_lo_lag_11 + rev_ex_lo_lag_12 +
rev_ex_lo_lag_13 + rev_ex_lo_lag_14 + rev_ex_lo_lag_15 + rev_ex_lo_lag_16 +
rev_ex_lo_lag_17 + rev_ex_lo_lag_18 + rev_ex_lo_lag_19 + rev_ex_lo_lag_20 +
rev_ex_lo_lag_21 + rev_ex_lo_lag_22 + rev_ex_lo_lag_23 + rev_ex_lo_lag_24 +
rev_ex_lo_lag_25 + rev_ex_lo_lead_1 + rev_ex_lo_lead_2 + rev_ex_lo_lead_3 +
rev_ex_lo_lead_4 + ma_split + ma_split_lag_1 + ma_split_lag_2 + ma_split_lag_3 +
ma_split_lag_4 + ma_split_lag_5 + ma_split_lag_6 + ma_split_lag_7 + ma_split_lag_8 +
ma_split_lag_9 + ma_split_lag_10 + ma_split_lag_11 + ma_split_lag_12 + ma_split_lag_13 +
ma_split_lag_14 + ma_split_lag_15 + ma_split_lag_16 + ma_split_lag_17 + ma_split_lag_18 +
ma_split_lag_19 + ma_split_lag_20 + ma_split_lag_21 + ma_split_lag_22 + ma_split_lag_23 +
ma_split_lag_24 + ma_split_lag_25 + ma_split_lead_1 + ma_split_lead_2 + ma_split_lead_3 +
ma_split_lead_4 | time_segm_price + artikpr | 0 | artikelid, data = dataaom)
```

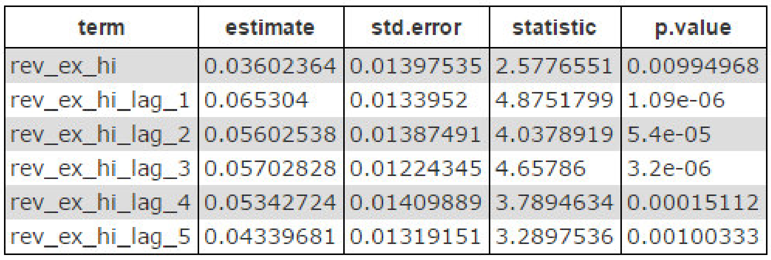
We should already be familiar with the structure of the summary from the last regression. Again, the next step is to prepare a data frame for more plots.

The following chunk will load the result of the summary done by `tidy()`.

```
# Load up tidyfig3.RDS into dffig3.
tidyaom <- readRDS(file="tidyaom.RDS")
```

Let's take a look at the results. Display dfaom with the head() function.

```
# Display dffig3 with the head function.
head(tidyaom)
```



term	estimate	std.error	statistic	p.value
rev_ex_hi	0.03602364	0.01397535	2.5776551	0.00994968
rev_ex_hi_lag_1	0.065304	0.0133952	4.8751799	1.09e-06
rev_ex_hi_lag_2	0.05602538	0.01387491	4.0378919	5.4e-05
rev_ex_hi_lag_3	0.05702828	0.01224345	4.65786	3.2e-06
rev_ex_hi_lag_4	0.05342724	0.01409889	3.7894634	0.00015112
rev_ex_hi_lag_5	0.04339681	0.01319151	3.2897536	0.00100333

Perfect, we are almost done. In this exercise we only need the first 30 values because we are just going to plot the estimates for good reviews. So extract the first 30 values to a new data frame, because as you can see, it starts with the variable rev_ex_hi.

```
# Extract the first 30 values
# Name the data frame dfaomfinal.
dfaom <- tidyaom[1:30,]
```

Perfect. The next step is go calculate the upper and lower value of the 95% confidence interval. Also we will add the week column right away. Remember the appropriate sequence.

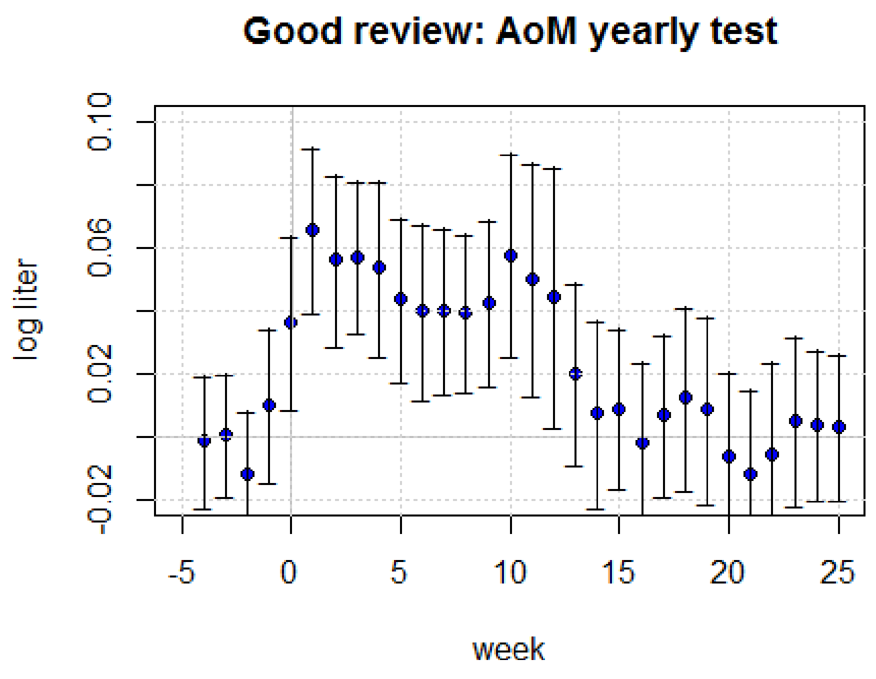
```
# Calculate the upper and lower value of the confidence interval. Name it u and l.
# Add the week column with the appropriate sequence.
# Displaying a part of the data frame with the head function.
dfaom$u <- dfaom$estimate+1.96*dfaom$std.error
dfaom$l <- dfaom$estimate-1.96*dfaom$std.error
dfaom$week <- c(0:25,-1:-4)
head(dfaom)
```

term	estimate	std.error	statistic	p.value	u	l	week
rev_ex_hi	0.03602364	0.01397535	2.5776551	0.00994968	0.06341534	0.00863195	0
rev_ex_hi_lag_1	0.065304	0.0133952	4.8751799	1.09e-06	0.09155858	0.03904941	1
rev_ex_hi_lag_2	0.05602538	0.01387491	4.0378919	5.4e-05	0.08322021	0.02883056	2
rev_ex_hi_lag_3	0.05702828	0.01224345	4.65786	3.2e-06	0.08102545	0.03303112	3
rev_ex_hi_lag_4	0.05342724	0.01409889	3.7894634	0.00015112	0.08106107	0.02579341	4
rev_ex_hi_lag_5	0.04339681	0.01319151	3.2897536	0.00100333	0.06925216	0.01754145	5

Perfect, we finished the first preparations.

The next step is to plot the estimations. You also should be familiar with this syntax. But you need to adapt the data frame, x, y and title. We do not need indices though. You should also change ylim to c(-0.02,0.10) due to the results. The title should be "Good review: Non-AoM yearly test".

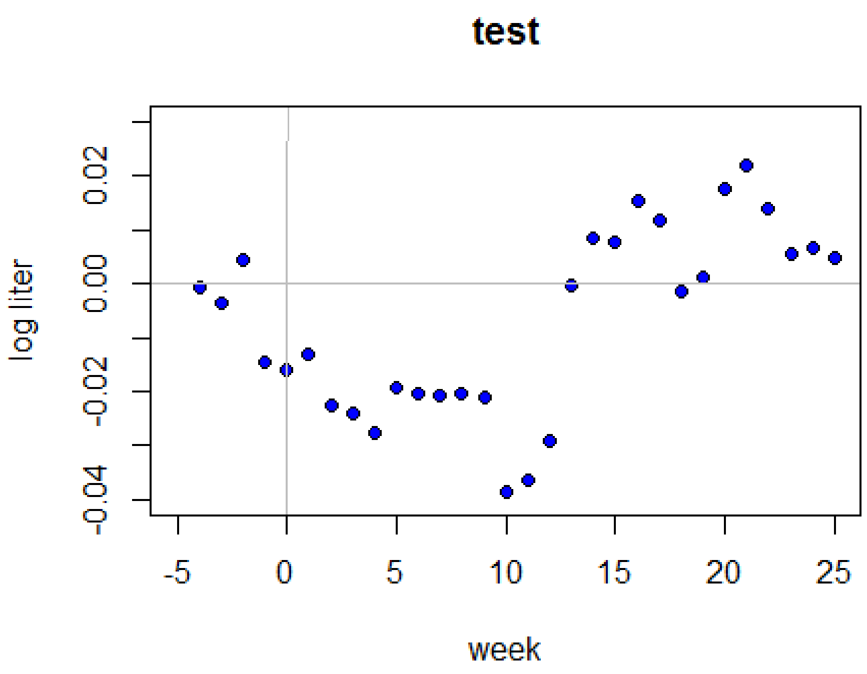
```
# Adapt the data frame, x, y, title and ylim.
# We do not need indices.
plot(dfaom$week,dfaom$estimate, main = "Good review: AoM yearly test", xlab = "week", ylab =
"log liter",ylim=c(-0.02,0.10), xlim=c(-5,25), pch = 21, bg = "blue", panel.last = c(abline(a = 0, b
= 1, lty = 1, col = "grey"),abline(a = 0, b = 0, lty = 1,  col = "grey")))
grid(nx = NULL, ny = NULL)
segments(dfaom$week, dfaom$l, dfaom$week, dfaom$u)
points(dfaom$week,dfaom$l, pch = "_")
points(dfaom$week,dfaom$u, pch = "_")
```



Congratulations, you created the plot. The results show that the good reviews for a wine reviewed on the AoM yearly test increase the demand up to roughly 6 percent within the first week. Afterwards it decreases a bit but stays fairly stable until week 4. We see some decrement from week 4 to 9. But then we see a short rise. After this, it falls down to around zero.

Let us compare the good reviews in general with the good reviews from the AoM yearly test with a scatter plot. We will create a new data frame named comparison. The data frame is going to have the appropriate week column and the difference of both estimations.

```
# Creating data frame with week sequence.
comparison <- data.frame(weeks = c(0:25,-1:-4))
# Calculation difference.
comparison$Difference <- dffig3$estimate[31:60]-dfaom$estimate
# Plot the differences.
plot(comparison$week,comparison$Difference, main = "test", xlab = "week", ylab = "log
liter",ylim=c(-0.04,0.03), xlim=c(-5,25), pch = 21, bg = "blue", panel.last = c(abline(a = 0, b = 1,
lty = 1, col = "grey"),abline(a = 0, b = 0, lty = 1,  col = "grey")))
```



The scatter plot shows that the good reviews from the AoM yearly test have a bigger impact the first 12 weeks after the review appears. You can see it because the values in the plot are negative. Afterwards, the values begin to be positive, thus the effect of the good reviews in general, not just "AoM wines", became bigger and therefore also lasted longer even though it is smaller in the first weeks.

c) Wines not in the AoM yearly special

Next, let's continue with the the good reviews of "non-AoM" wines. The regression will take the variables with the corresponding lags leads of rev_nyaom, rev_nyaom_hi, rev_nyaom_lo and ma_split into account. The regression was already done due to the long calculation time, but you can take a look at the code.

Demonstraion chunk, do not run

```
nonaomreg <- felm(llitre ~ rev_nyaom + rev_nyaom_lag_1 + rev_nyaom_lag_2 +
rev_nyaom_lag_3 + rev_nyaom_lag_4 + rev_nyaom_lag_5 + rev_nyaom_lag_6 +
rev_nyaom_lag_7 + rev_nyaom_lag_8 + rev_nyaom_lag_9 + rev_nyaom_lag_10 +
rev_nyaom_lag_11 + rev_nyaom_lag_12 + rev_nyaom_lag_13 + rev_nyaom_lag_14 +
rev_nyaom_lag_15 + rev_nyaom_lag_16 + rev_nyaom_lag_17 + rev_nyaom_lag_18 +
rev_nyaom_lag_19 + rev_nyaom_lag_20 + rev_nyaom_lag_21 + rev_nyaom_lag_22 +
rev_nyaom_lag_23 + rev_nyaom_lag_24 + rev_nyaom_lag_25  + rev_nyaom_lead_1 +
rev_nyaom_lead_2 + rev_nyaom_lead_3 + rev_nyaom_lead_4 + rev_nyaom_hi +
rev_nyaom_hi_lag_1 + rev_nyaom_hi_lag_2 + rev_nyaom_hi_lag_3 + rev_nyaom_hi_lag_4 +
rev_nyaom_hi_lag_5 + rev_nyaom_hi_lag_6 + rev_nyaom_hi_lag_7 + rev_nyaom_hi_lag_8 +
rev_nyaom_hi_lag_9 + rev_nyaom_hi_lag_10 + rev_nyaom_hi_lag_11 + rev_nyaom_hi_lag_12
+ rev_nyaom_hi_lag_13 + rev_nyaom_hi_lag_14 + rev_nyaom_hi_lag_15 +
rev_nyaom_hi_lag_16 + rev_nyaom_hi_lag_17 + rev_nyaom_hi_lag_18 +
rev_nyaom_hi_lag_19 + rev_nyaom_hi_lag_20 + rev_nyaom_hi_lag_21 +
rev_nyaom_hi_lag_22 + rev_nyaom_hi_lag_23 + rev_nyaom_hi_lag_24 +
rev_nyaom_hi_lag_25 + rev_nyaom_hi_lead_1 + rev_nyaom_hi_lead_2 +
rev_nyaom_hi_lead_3 + rev_nyaom_hi_lead_4 + rev_nyaom_lo + rev_nyaom_lo_lag_1 +
rev_nyaom_lo_lag_2 + rev_nyaom_lo_lag_3 + rev_nyaom_lo_lag_4 + rev_nyaom_lo_lag_5 +
rev_nyaom_lo_lag_6 + rev_nyaom_lo_lag_7 + rev_nyaom_lo_lag_8 + rev_nyaom_lo_lag_9 +
rev_nyaom_lo_lag_10 + rev_nyaom_lo_lag_11 + rev_nyaom_lo_lag_12 +
rev_nyaom_lo_lag_13 + rev_nyaom_lo_lag_14 + rev_nyaom_lo_lag_15 +
rev_nyaom_lo_lag_16 + rev_nyaom_lo_lag_17 + rev_nyaom_lo_lag_18 +
rev_nyaom_lo_lag_19 + rev_nyaom_lo_lag_20 + rev_nyaom_lo_lag_21 +
rev_nyaom_lo_lag_22 + rev_nyaom_lo_lag_23 + rev_nyaom_lo_lag_24 +
rev_nyaom_lo_lag_25 + rev_nyaom_lo_lead_1 + rev_nyaom_lo_lead_2 +
rev_nyaom_lo_lead_3 + rev_nyaom_lo_lead_4 + ma_split + ma_split_lag_1 + ma_split_lag_2 +
ma_split_lag_3 + ma_split_lag_4 + ma_split_lag_5 + ma_split_lag_6 + ma_split_lag_7 +
ma_split_lag_8 + ma_split_lag_9 + ma_split_lag_10 + ma_split_lag_11 + ma_split_lag_12 +
ma_split_lag_13 + ma_split_lag_14 + ma_split_lag_15 + ma_split_lag_16 + ma_split_lag_17 +
ma_split_lag_18 + ma_split_lag_19 + ma_split_lag_20 + ma_split_lag_21 + ma_split_lag_22 +
ma_split_lag_23 + ma_split_lag_24 +ma_split_lag_25 + ma_split_lead_1 + ma_split_lead_2 +
ma_split_lead_3 + ma_split_lead_4 | time_segm_price + artikpr | 0 | artikelid, data = dataaom)
```

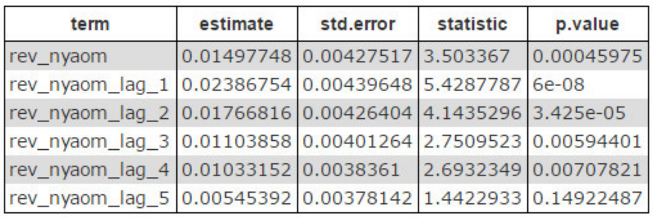
We need to prepare a data frame for confidence interval as before. Afterwards, we are going to plot.

First, load up the required results from the nonaomreg. Load up tidynonaom that was created with the already know, tidy() function.

```
# Load up tidynonaom.
tidynonaom <- readRDS(file = "tidynonaom.RDS")
```

Display tidynonaom to get a quick overview.

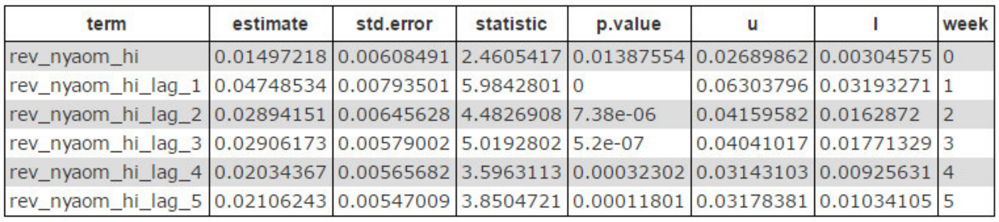
```
# Display tidynonaom with the head() function.
head(tidynonaom)
```



term	estimate	std.error	statistic	p.value
rev_nyaom	0.01497748	0.00427517	3.503367	0.00045975
rev_nyaom_lag_1	0.02386754	0.00439648	5.4287787	6e-08
rev_nyaom_lag_2	0.01766816	0.00426404	4.1435296	3.425e-05
rev_nyaom_lag_3	0.01103858	0.00401264	2.7509523	0.00594401
rev_nyaom_lag_4	0.01033152	0.0038361	2.6932349	0.00707821
rev_nyaom_lag_5	0.00545392	0.00378142	1.4422933	0.14922487

As you can see, the first 30 values refer to the review per se variable of the non-AoM wines, so rev_nyaom. The good review dummy is listed after it, meaning indices 31 to 60.

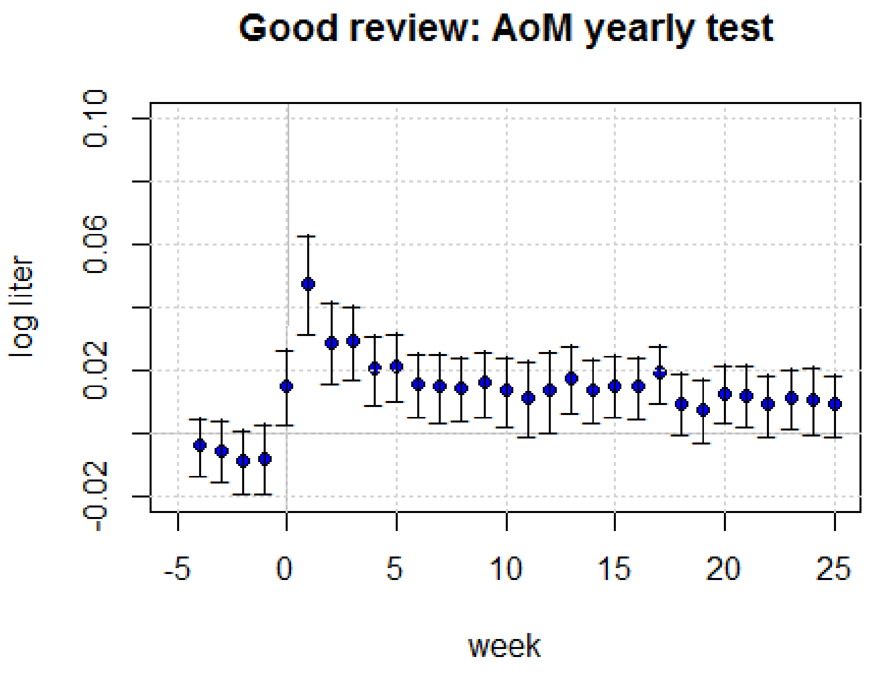
```
# Create the data frame dfnonaom and extract the coefficients of the summary tidynonaom.
# Remember: we only need the values of rev_nyaom_hi. Use indices.
# Calculate the upper and lower value of the confidence interval. Name it u and l.
# Add the week column with the appropriate sequence.
# Displaying a part of the data frame with the head function.
dfnonaom <- tidynonaom[31:60,]
dfnonaom$u <- dfnonaom$estimate+1.96*dfnonaom$std.error
dfnonaom$l <- dfnonaom$estimate-1.96*dfnonaom$std.error
dfnonaom$week <- c(0:25,-1:-4)
head(dfnonaom)
```



term	estimate	std.error	statistic	p.value	u	l	week
rev_nyaom_hi	0.01497218	0.00608491	2.4605417	0.01387554	0.02689862	0.00304575	0
rev_nyaom_hi_lag_1	0.04748534	0.00793501	5.9842801	0	0.06303796	0.03193271	1
rev_nyaom_hi_lag_2	0.02894151	0.00645628	4.4826908	7.38e-06	0.04159582	0.0162872	2
rev_nyaom_hi_lag_3	0.02906173	0.00579002	5.0192802	5.2e-07	0.04041017	0.01771329	3
rev_nyaom_hi_lag_4	0.02034367	0.00565682	3.5963113	0.00032302	0.03143103	0.00925631	4
rev_nyaom_hi_lag_5	0.02106243	0.00547009	3.8504721	0.00011801	0.03178381	0.01034105	5

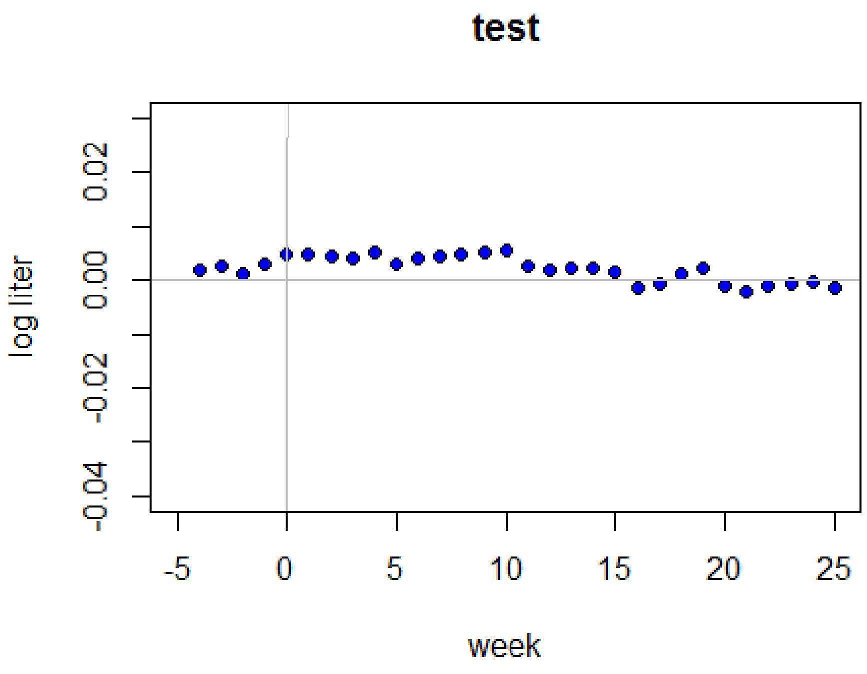
Perfect, next step: plotting. Choose the appropriate code.

```
# Adapt the data frame, x, y, title and ylim.
# We do not need indices.
plot(dfnonaom$week,dfnonaom$estimate, main = "Good review: AoM yearly test", xlab =
"week", ylab = "log liter",ylim=c(-0.02,0.10), xlim=c(-5,25), pch = 21, bg = "blue", panel.last =
c(abline(a = 0, b = 1, lty = 1, col = "grey"),abline(a = 0, b = 0, lty = 1,  col = "grey")))
grid(nx = NULL, ny = NULL)
segments(dfnonaom$week, dfnonaom$l, dfnonaom$week, dfnonaom$u)
points(dfnonaom$week,dfnonaom$l, pch = "_")
points(dfnonaom$week,dfnonaom$u, pch = "_")
```



Awesome, you created the correct plot. Let us compare also the impact of good reviews in general and the good reviews of "non-Aom-wines". We use the same procedure as before by creating a comparison data frame and plotting the differences.

```
# Creating data frame with week sequence.
comparisonnon <- data.frame(weeks = c(0:25,-1:-4))
# Calculation difference.
comparisonnon$Difference <- dffig3$estimate[31:60]-dfnonaom$estimate
# Plot the differences.
plot(comparisonnon$week,comparisonnon$Difference, main = "test", xlab = "week", ylab = "log
liter",ylim=c(-0.04,0.03), xlim=c(-5,25), pch = 21, bg = "blue", panel.last = c(abline(a = 0, b = 1,
lty = 1, col = "grey"),abline(a = 0, b = 0, lty = 1,  col = "grey")))
```



The scatter plot shows that the Estimates are almost exactly the same which means that both of those review types have almost exactly the same impact on the sales.

Let us also talk about the estimates on the leads in all these plots. All these leads are generally close to zero and therefore not significant. Thus, we can assume that segment-specific period fixed effects successfully capture temporary demand shocks and seasonal variation. Meaning, we find no indications for reviews as a response to unusual trends.

d) Interpretations

Even though the advertising expenditures are rather low, it has shown that it tends to matter more how persuasive the advertising is and not how much is spent. (See, for instance, Vakratsas and Ambler (1999) or Tellis, Chandy, and Thaivanich (2000)). Further, the content of advertisement in Sweden is regulated, therefore we see rather informative content. This might explain the low impact.

In comparison, the impact of a good review lasts way longer than the impact of advertising. This might be explained with the characteristics of the medium. Advertising is rather informative and therefore fulfils the same role as a neutral review. A good review, on the other hand, focuses more on details and the quality of a product, thus is more persuasive.

The premise of the article is that wine specific demand shocks are not correlated with fact of a wine being reviewed in the same week. Therefore, we also estimated the effect of wines that are only reviewed on the AoM event and that do not, to examine if we capture the casual effect of wine reviews. The similarities of both plots and therefore estimations give us enough support to the regression captures a casual effect reviews.

This exercise refers to page 206 in the paper

Exercise 11 Categorical Impact of Reviews

So this chapter is a chapter where you rather just check / run the code than trying to solve yourself. The reason for this is that the code is a bit more complicated and that the regressions take quite some time, especially the regression in the end with some iterations. A simpler version would have been way too long. However, the chapter and the corresponding code is really interesting and will show you some tricks, so try to follow and understand the code.

a) Introduction

We have noticed that different types of advertising (informative & persuasive) have a different impact on the sales. The article also tested if there are differences among some specific categories. We took a look at the first 10 weeks. The defined categories of interest are as the following:

- Medium and high price wines medhigh
- Wines from certain countries newworld
- Wines with advertising ma_brands
- Wine with reviews after advertising became legal legal_m
- New wines, defined as not older than 2 years new
- Wines that come in vintages vint_brands
- High quality variations highvar_q
- Wines that got reviewed in tabloids tabloid

The equation (1) that we introduced got modified for the explained purpose. Equation (2)

$$lnQ_{ijkt} = \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}^{m} R_{it-l}^{m} + \sum_{l=0}^{10} D_g\, \alpha_{t-l}^{m} R_{it-l}^{m} + \sum_{l=-4}^{25} \gamma_{t-l}\, ADVERT_{it-l} + \theta_{ijkt*}$$

m super index and stands for good review, bad review and review D_g Indicator that captures the different categories

Let's move on to the explanation of the dummies that will give you some insight how the dummies are defined.

b) Explanation of the category dummies

Medium and high price wines medhigh

This category is based on the price_segm variable. If a wine is in the high h or medium m price segment then the dummy gets the value 1 as well.

Wines from certain countries newworld

For the category it was checked if the wine was imported from a certain country that is considered as *new world*

- The new world countries are: Argentina, Australien, Chile, Nya Zeeland, Portugal, Sydafrika and USA
- Countries that get the value NA are being ignored: Sverige, Grekland, Libanon, Oesterrike, Uruguay, Isreal, Oevriga ursprung, Bulgarien, Rumaenien and Cypern*

The rest will have the value 0.

Wines with advertising ma_brands

When a wine had advertising expenditures, then it belongs to this category. So the variable ma_split got checked and therefore indicates which wines have the value 1 in the corresponding dummy.

Wine with reviews after advertising became legal legal_m

As already mentioned, there was a time where advertising for wine was prohibited in Sweden. This changed on the day after the 15^{th} of May in 2003. This means that all the wines observed after this date belong to the legal_m category.

New wines, defined as not older than 2 years new

The category new is for wines that are not older than 2 years old. The reason for this definition is that the set of wines in the top tiers, this the wines that get distributed in most of the stores, are quite stable and it is hard to achieve a *good* number of observations. So it got decided to use this defintion in order to compare new vs old.

*Wines that come in vintages** vint_brands

The vintage brands category is based on the vintage variable. If the column contains a year for the given wine, then the wine belongs to the vintage wines.

High quality variations highvar_q

The next category is not so obvious. It indicates if a wine comes from a region that exhibits high quality variations. The quality got measured by grades, so we used v10_all which is the weekly average normalized review (all media). The standard deviation of the grades represents the variation in quality. So if a region has in a specific year has high standard deviation in grades then it has a high quality variation. A standard deviation larger than the 75% percentile was considered as high. It got broken down to country, region and year.

Click on the info button for a detailed look at the code.

Info: Code for high_var

do not run the chunks, they are just for demonstrations

The first step is to group by country, region and year. Then, we summarize by calculating the standard deviation of v10_all for the grouped "rows". Afterwards, we create the dummy and check for the 75% percentile. *A number instead of a percentile calculation is used here because there was a small difference between the original calculations of the article in STATA and in R, which would result in a small but still significant difference in observations tagged as high_var*

```r
# Grouping by country, region and year.
# Calculating the standard deviation for non-NA values of v10_all.
# Counting the amount of observation for each standard deviation.
highvar <- tab3 %>%
  group_by(m_country,region,year) %>%
  summarize("standard_deviation" = sd(na.omit(v10_all))) %>%
  # highvar_q is going to be 0 if the grouped standard deviation of v10_all is NA or smaller than 3.040.
  # It is 1 otherwise
  mutate(highvar_q = ifelse(is.na(standard_deviation) | standard_deviation < 3.040,0,1))
```

The next step is to merge the summary values to the base dataframe, so all the wines get the corresponding value.

```
# Merging tab3 and highvar by country, region and year (the group critira).
mergehighvar <- merge(tab3, highvar, by = c("m_country","region","year"))

# Sorting the data frame by artikelnr.
# This is just done because of "safety" purposes.
tab3dummies <- mergehighvar[order(mergehighvar$artikelnr),]
```

Wines that got reviewed in tabloids tabloid

The tabloid category is different than the other ones. The category refers to wines that got reviewed in a tabloid and is based on the different types of rev_eve. This is the pre-existing indicator of the wine being reviewed in a tabloid.

- rev_eve is the indicator of the wine being reviewed in tabloids
- rev_eve_hi is indicator of the wine receiving a good review in tabloids
- rev_eve_lo is indicator of the wine receiving a bad review in tabloids

c) Reviews, Good Reviews and Bad Reviews

Another important part of this categorical regression is that we need to differentiate among the different reviews. As you already know, there are dummies for a Review, a Good Review and a Bad Review.

As you already can guess, the tabloid category already has the classifications with all three types of rev_eve, which are basically category and review indicator together.

The other categories need a combination of category and review indicator.

$$lnQ_{ijkt} = \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}^{m} R_{it-l}^{m} + \sum_{l=0}^{10} D_g\, \alpha_{t-l}^{m} R_{it-l}^{m} + \sum_{l=-4}^{25} \gamma_{t-l}\, ADVERT_{it-l} + \theta_{ijkt*}$$

When you take a look at the part $\sum_{l=0}^{10} D_g\, \alpha_{t-l}^{m} R_{it-l}^{m}$ is this the exact combination of categorical indicator D_g and review indicator R. The sum shows that we also include all the lags and leads into our regression.

The category indicators just got explained. The review indicator is also already known. It is rev_all, the indicator of the wine being reviewed (all media).

Every regression for each category need to have the indicators for a review, a good review and a bad review. They will all have the same name, however every category has its own regression uses different indicator combinations, as previously explained.

- Review
- Good Review
- Bad Review

These indicators got created by checking the category variable and the lags (up to 10) of rev_all.

- rev_all is the indicator of the wine being reviewed.
- rev_all_hi is indicator of the wine receiving a good review.
- rev_all_lo is indicator of the wine receiving a bad review.

If the category variable and the corresponding rev_all have the value 1, then the "combined" indicator will get assigned a 1 as well.

Click on the info button to take a look at the code for the tabloid category.

Info: Tabloid Category Review Indicators

do not run the chunks, they are just for demonstrations

This is the code for the tabloid category. As you can see, the code just checks for the rev_eves.

```r
# Assigning dummy data frame to tabloid.
tabloid <- tab3dummies

# Creating the dummies with 0 as defailt.
tabloid$Review <- 0

tabloid$GoodReview <- 0

tabloid$BadReview <- 0

# Code as string for assigning of the values to the indicators.
# It is a string because of the convenience of string manipulation.
codecate <- "tabloid <- tabloid %>%
  mutate(Review = ifelse(rev_eve_lag_1 == 1,1,Review),
         GoodReview = ifelse(rev_eve_hi_lag_1 == 1,1,GoodReview),
         BadReview = ifelse(rev_eve_lo_lag_1 == 1,1,BadReview))"
# Looping from lag_1 to lag_10
for(i in 2:11){
  eval(parse(text = codecate))
  old <- paste0('lag_',i-1)
  new <- paste0('lag_',i)
  codecate <- gsub(old,new,codecate)
}

# Checking the non-lagged variable, which is basically lag_0.
tabloid <- tabloid %>%
  mutate(Review = ifelse(rev_eve == 1,1,Review),
         GoodReview = ifelse(rev_eve_hi == 1,1,GoodReview),
         BadReview = ifelse(rev_eve_lo == 1,1,BadReview))

# Sorting tabloid by artikelnr.
tabloidsort <- tabloid[order(tabloid$artikelnr),]
```

---

d) Regression example - Tabloids

$$llitre_{ijkt} = \alpha_j + \delta_{kt} + \sum_{l=-4}^{25} \alpha_{t-l}^{m}\, R_{it-l}^{m} + \sum_{l=0}^{10} D_g\, \alpha_{t-l}^{m} R_{it-l}^{m} + \sum_{l=-4}^{25} \gamma_{t-l}\, \text{ma\_split}_{it-l} + \text{artikelid}_{ijkt*}$$

R includes rev_all, rev_all_hi and rev_all_lo m super index and stands for good review, bad review and review D_g Indicator that captures the different categories R in combination with D_g yields the variables Review, GoodReview and BadReview for the corresponding category.

Also this regression contains the same two fixed effects as the regressions before.

Remember: The first fixed effect is time_segm_price. It is the Period-color-price segment-package indicator. It accounts for separate fixed effects which accommodated different time trends in sales. It captures for example, bottled red wines sold at price above SEK 110 per bottle in week 128.

The second fixed effect is artikpr. It is Product number-price-vintage combination. It makes sure that a new vintage or new price is associated with a new fixed effect.

artikelid is again the cluster variable for the regression. artikelid indicates the ID number of brand (given by Systembolaget).

We already saw an example, how to translate our equation into the regression formula for felm().

Demonstration chunk, do not run

```
# Assign the regression result to tabloidreg.
tabloidreg <- felm(llitre ~ Review + GoodReview + BadReview + rev_all + rev_all_lag_1 +
rev_all_lag_2 + rev_all_lag_3 + rev_all_lag_4 + rev_all_lag_5 + rev_all_lag_6 + rev_all_lag_7 +
rev_all_lag_8 + rev_all_lag_9 + rev_all_lag_10 + rev_all_lag_11 + rev_all_lag_12 +
rev_all_lag_13 + rev_all_lag_14 + rev_all_lag_15 + rev_all_lag_16 + rev_all_lag_17 +
rev_all_lag_18 + rev_all_lag_19 + rev_all_lag_20 + rev_all_lag_21 + rev_all_lag_22 +
rev_all_lag_23 + rev_all_lag_24 + rev_all_lag_25  + rev_all_lead_1 + rev_all_lead_2 +
rev_all_lead_3 + rev_all_lead_4 + rev_all_hi + rev_all_hi_lag_1 + rev_all_hi_lag_2 +
rev_all_hi_lag_3 + rev_all_hi_lag_4 + rev_all_hi_lag_5 + rev_all_hi_lag_6 + rev_all_hi_lag_7 +
rev_all_hi_lag_8 + rev_all_hi_lag_9 + rev_all_hi_lag_10 + rev_all_hi_lag_11 + rev_all_hi_lag_12
+ rev_all_hi_lag_13 + rev_all_hi_lag_14 + rev_all_hi_lag_15 + rev_all_hi_lag_16 +
rev_all_hi_lag_17 + rev_all_hi_lag_18 + rev_all_hi_lag_19 + rev_all_hi_lag_20 +
rev_all_hi_lag_21 + rev_all_hi_lag_22 + rev_all_hi_lag_23 + rev_all_hi_lag_24 +
rev_all_hi_lag_25 + rev_all_hi_lead_1 + rev_all_hi_lead_2 + rev_all_hi_lead_3 +
rev_all_hi_lead_4 + rev_all_lo + rev_all_lo_lag_1 + rev_all_lo_lag_2 + rev_all_lo_lag_3 +
rev_all_lo_lag_4 + rev_all_lo_lag_5 + rev_all_lo_lag_6 + rev_all_lo_lag_7 + rev_all_lo_lag_8 +
rev_all_lo_lag_9 + rev_all_lo_lag_10 + rev_all_lo_lag_11 + rev_all_lo_lag_12 +
rev_all_lo_lag_13 + rev_all_lo_lag_14 + rev_all_lo_lag_15 + rev_all_lo_lag_16 +
rev_all_lo_lag_17 + rev_all_lo_lag_18 + rev_all_lo_lag_19 + rev_all_lo_lag_20 +
rev_all_lo_lag_21 + rev_all_lo_lag_22 + rev_all_lo_lag_23 + rev_all_lo_lag_24 +
rev_all_lo_lag_25 + rev_all_lo_lead_1 + rev_all_lo_lead_2 + rev_all_lo_lead_3 +
rev_all_lo_lead_4 + ma_split + ma_split_lag_1 + ma_split_lag_2 + ma_split_lag_3 +
ma_split_lag_4 + ma_split_lag_5 + ma_split_lag_6 + ma_split_lag_7 + ma_split_lag_8 +
ma_split_lag_9 + ma_split_lag_10 + ma_split_lag_11 + ma_split_lag_12 + ma_split_lag_13 +
ma_split_lag_14 + ma_split_lag_15 + ma_split_lag_16 + ma_split_lag_17 + ma_split_lag_18 +
ma_split_lag_19 + ma_split_lag_20 + ma_split_lag_21 + ma_split_lag_22 + ma_split_lag_23 +
ma_split_lag_24 + ma_split_lag_25 + ma_split_lead_1 + ma_split_lead_2 + ma_split_lead_3 +
ma_split_lead_4 | time_segm_price + artikpr | 0 | artikelid, data = tabloidsort)
```

Here you can see the summarise regression results.

	Dependent variable:
	llitre
	Tabloid
Review	-0.0006
	(0.0060)
GoodReview	-0.0191**
	(0.0083)
BadReview	-0.0062
	(0.0186)
Observations	64,863
R^2	0.9910
Adjusted R^2	0.9903
Note:	*p<0.1; **p<0.05; ***p<0.01

This summary was created with the stargazer package. Click the info button for more information about stargazer and the code.

The stargazer command produces LaTeX code, HTML code and ASCII text for well-formatted tables that summarize regression results from several models. Stargazer supports a large number model objects from a variety of packages.

Use the following link to look up more information.

- Cran R Project PDF stargazer: beautiful LATEX, HTML and ASCII tables from R statistical output by Marek Hlavac, Harvard University, July 14, 2015
- Cran R Project PDF Package 'stargazer'

The chunk below shows the stargazer code that created the summary. The regression object gets passed to the function. Then we can choose some options.

- The type describes the output format
- Omit and omit.stat declare variables & statistics being omitted
- Digits defines the amount of decimal places
- Column.labels is used to choose the names for the columns

Demonstration chunk, do not run

```
# Load codetabloid.Rds into codetabloid.
# Run the regression.
stargazer(tabloidreg,
        type = "html",
        omit = exclude,
        omit.stat = 'ser',
        digits = 4,
        column.labels = "Tabloid")
```

We will interpret the results later on when all categories are included.

Let's move on to the rest of the categories since we know the details of the categories & indicators and have a good understanding about the regression model.

e) Rest of the categories

The next step is to run the regressions for the rest of the categories. As mentioned, they are different from the tabloid category. The review, good review and bad review variable of these categories refer to rev_all,rev_all_hi, rev_all_lo and the corresponding dummy variable. For example, tabloid was just referring to the variations of rev_eve. But the other categories also refer to the actual category dummy. So, Review for the category of wines that have a medium or high price refers to medhigh and rev_all (and its lags)

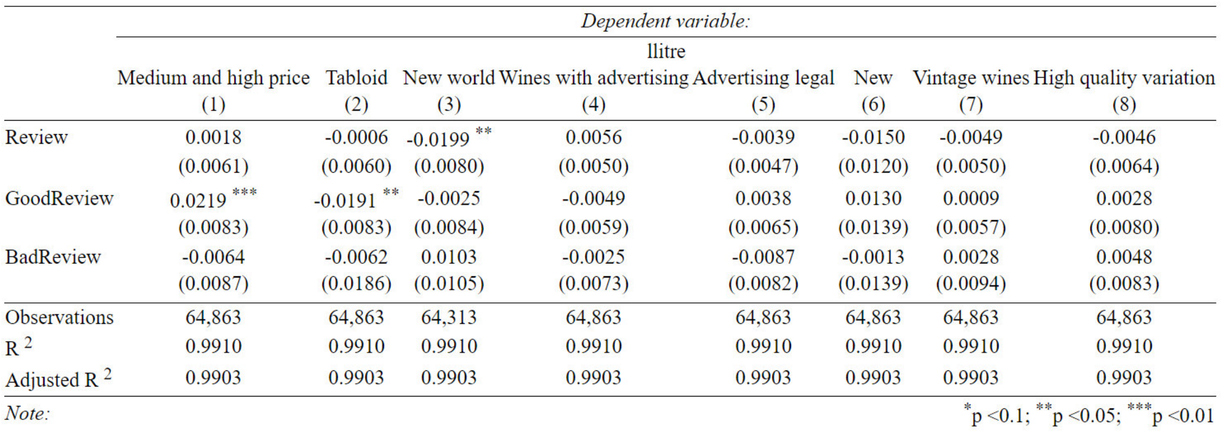
This part is also quite long, therefore you do not solve the chunks on your own. However, try to take the time and understand how it is done.

Demonstration chunk, do not run

```r
# Code for loop due to limit regarding string nesting.
# Code belongs to the for loop within the string: code.
codeloop <- "
  eval(parse(text = codecate))
  olds <- paste0('lag_',i-1)
  news <- paste0('lag_',i)
  codecate <- gsub(olds,news,codecate)"
code <- paste0(
  "
  # Loading tab3dummes, dataset with dummies etc.
  medhigh <- tab3dummies

  # Further indicators.
  medhigh$Review <- 0
  medhigh$GoodReview <- 0
  medhigh$BadReview <- 0","

  # Changing the value of the indicators above according to the category dummy and lag
indicator.
  # So if the category and lag indicator have the value one, then the review indicator will be 1 as
well.
  codecate <- 'medhigh <- medhigh %>%
  mutate(Review = ifelse(rev_all_lag_1 == 1 & medhigh == 1,1,Review),
  GoodReview = ifelse(rev_all_hi_lag_1 == 1 & medhigh == 1,1,GoodReview),
  BadReview = ifelse(rev_all_lo_lag_1 == 1 & medhigh == 1,1,BadReview))'",
  "\n
  # Looping from 2 to basically 10, 1 is included as the starting point.
  # i = 11 will not be executed, but we need it to reach the lag 10.
  # i alters for example rev_all_lag_1 from 2 to 10, we use the paste and gsub function, look at
codeloop above.
  # Remember: eval and parse run a string as a R code.
  ",
  "for(i in 2:11){
    eval(parse(text = codeloop))
  }
  # Checking the non-lagged variable, which is basically lag_0.
  # We also need to check the non-lagged indicators.
  medhigh <- medhigh %>%
  mutate(Review = ifelse(rev_all == 1 & medhigh == 1,1,Review),
  GoodReview = ifelse(rev_all_hi == 1 & medhigh == 1,1,GoodReview),
  BadReview = ifelse(rev_all_lo == 1 & medhigh == 1,1,BadReview))
  medhighsort <- medhigh[order(medhigh$artikelnr),]

  # Regression code, should be familiar.
  medhighreg <- felm(llitre ~ Review + GoodReview + BadReview+rev_all + rev_all_lag_1 +
rev_all_lag_2 +
  rev_all_lag_3 + rev_all_lag_4 + rev_all_lag_5 + rev_all_lag_6 + rev_all_lag_7 + rev_all_lag_8
+ rev_all_lag_9 +
  rev_all_lag_10 + rev_all_lag_11 + rev_all_lag_12 + rev_all_lag_13 + rev_all_lag_14 +
rev_all_lag_15 +
  rev_all_lag_16 + rev_all_lag_17 + rev_all_lag_18 + rev_all_lag_19 + rev_all_lag_20 +
rev_all_lag_21 +
  rev_all_lag_22 + rev_all_lag_23 + rev_all_lag_24 + rev_all_lag_25  + rev_all_lead_1 +
rev_all_lead_2 +
  rev_all_lead_3 + rev_all_lead_4 + rev_all_hi + rev_all_hi_lag_1 + rev_all_hi_lag_2 +
rev_all_hi_lag_3 + rev_all_hi_lag_4 +
  rev_all_hi_lag_5 + rev_all_hi_lag_6 + rev_all_hi_lag_7 + rev_all_hi_lag_8 + rev_all_hi_lag_9 +
rev_all_hi_lag_10 +
  rev_all_hi_lag_11 + rev_all_hi_lag_12 + rev_all_hi_lag_13 + rev_all_hi_lag_14 +
rev_all_hi_lag_15 +
  rev_all_hi_lag_16 + rev_all_hi_lag_17 + rev_all_hi_lag_18 + rev_all_hi_lag_19 +
rev_all_hi_lag_20 +
  rev_all_hi_lag_21 + rev_all_hi_lag_22 + rev_all_hi_lag_23 + rev_all_hi_lag_24 +
rev_all_hi_lag_25 +
  rev_all_hi_lead_1 + rev_all_hi_lead_2 + rev_all_hi_lead_3 + rev_all_hi_lead_4 + rev_all_lo +
rev_all_lo_lag_1 + rev_all_lo_lag_2
  + rev_all_lo_lag_3 + rev_all_lo_lag_4 + rev_all_lo_lag_5 + rev_all_lo_lag_6 + rev_all_lo_lag_7
+ rev_all_lo_lag_8 +
  rev_all_lo_lag_9 + rev_all_lo_lag_10 + rev_all_lo_lag_11 + rev_all_lo_lag_12 +
rev_all_lo_lag_13 + rev_all_lo_lag_14
  + rev_all_lo_lag_15 + rev_all_lo_lag_16 + rev_all_lo_lag_17 + rev_all_lo_lag_18 +
rev_all_lo_lag_19 + rev_all_lo_lag_20
  + rev_all_lo_lag_21 + rev_all_lo_lag_22 + rev_all_lo_lag_23 + rev_all_lo_lag_24 +
rev_all_lo_lag_25 + rev_all_lo_lead_1 +
  rev_all_lo_lead_2 + rev_all_lo_lead_3 + rev_all_lo_lead_4 + ma_split +
  ma_split_lag_1 + ma_split_lag_2 + ma_split_lag_3 + ma_split_lag_4 + ma_split_lag_5 +
ma_split_lag_6 +
  ma_split_lag_7 + ma_split_lag_8 + ma_split_lag_9 + ma_split_lag_10 + ma_split_lag_11 +
ma_split_lag_12 +
  ma_split_lag_13 + ma_split_lag_14 + ma_split_lag_15 + ma_split_lag_16 + ma_split_lag_17 +
ma_split_lag_18 +
  ma_split_lag_19 + ma_split_lag_20 + ma_split_lag_21 + ma_split_lag_22 + ma_split_lag_23 +
ma_split_lag_24 +
  ma_split_lag_25 + ma_split_lead_1 + ma_split_lead_2 + ma_split_lead_3 + ma_split_lead_4 |
time_segm_price +
  artikpr | 0 | artikelid, data = medhighsort)
  ")
# Creating a list with the strings: medhigh, newworld, ma_brands, legal_m and highvar_q.
clist <- c("medhigh","newworld","ma_brands","legal_m","vint_brands","highvar_q")
# Setting old to "medhigh", this is the starting point, because we use it as the base code.
old <- "medhigh"
# We need a loop to run through all the words / categories in the list.
for (c in clist) {
  # Replacing the old string with the next one in clist.
  # So we just replace the old category with the new one.
  # We also change the regression name with gsub, meaing we will have all different category
regressions saved.
  code <- gsub(old,c,code)
  # Converting the sring to a runable code and executing it.
  eval(parse(text = code))
  # Setting the just executed category as the cold category.
  old <- c
}

# Loading the regression code for the category new.
# The code is almost the same as above, the difference is:
# Review + GoodReview + BadReview + new + rev_all
# The category variable was also included into the regression
codenew <- readRDS(file="codenew.RDS")

# Converting and running the string code.
eval(parse(text = codenew))
```

The results of all categories got summarized in one big table. The stargazer() function got used again. All the regressions results got passed to the function. The settings regarding type, omits and digits were the same. Just the column label option got passed a vector with all the category names.

- Medium and high price
- Tabloid
- New World
- Wines with advertising
- Advertising legal
- New
- Vintage wines
- High quality variation



	Dependent variable:							
	llitre							
	Medium and high price	Tabloid	New world	Wines with advertising	Advertising legal	New	Vintage wines	High quality variation
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
Review	0.0018	-0.0006	-0.0199 **	0.0056	-0.0039	-0.0150	-0.0049	-0.0046
	(0.0061)	(0.0060)	(0.0080)	(0.0050)	(0.0047)	(0.0120)	(0.0050)	(0.0064)
GoodReview	0.0219 ***	-0.0191 **	-0.0025	-0.0049	0.0038	0.0130	0.0009	0.0028
	(0.0083)	(0.0083)	(0.0084)	(0.0059)	(0.0065)	(0.0139)	(0.0057)	(0.0080)
BadReview	-0.0064	-0.0062	0.0103	-0.0025	-0.0087	-0.0013	0.0028	0.0048
	(0.0087)	(0.0186)	(0.0105)	(0.0073)	(0.0082)	(0.0139)	(0.0094)	(0.0083)
Observations	64,863	64,863	64,313	64,863	64,863	64,863	64,863	64,863
R^2	0.9910	0.9910	0.9910	0.9910	0.9910	0.9910	0.9910	0.9910
Adjusted R^2	0.9903	0.9903	0.9903	0.9903	0.9903	0.9903	0.9903	0.9903
Note:								*p <0.1; **p <0.05; ***p <0.01

Amazing, now we have a really good and clear summary to interpret the results. The stargazer automatically puts the values of Review, GoodReview and BadReview on the same row, since the variables have purposely the same name in each category. But this is exactly what we want, because it represents the same review type even though it is not really the same variable.

Click on the info button for a closer look at the used code:

As explained, almost the same syntax as in the previous code regarding stargazer(). We just passed more regressions and labels.

Demonstration chunk, do not run the code

```
# Pass the regressions to the function.
# Use html type, exclude the selected variable and ....
# Round up to 4 digits.
# Pass the category names as labels.
stargazer(medhighreg, tabloidreg, newworldreg, ma_brandsreg, legal_mreg, newreg,
vint_brandsreg, highvar_qreg,
        type = "html",
        omit = exclude,
        omit.stat = 'ser',
        digits = 4,
        column.labels = c("Medium and high price", "Tabloid","New world"," Wines with
advertising","Advertising legal", "New", "Vintage wines", "High quality variation"))

## Error in .stargazer.wrap(..., type = type, title = title, style = style, : Objekt 'medhighreg' nicht
gefunden
```

Now, let's move on to the interpretation of the results.

f) Interpretation of the results

We can see that wines medium and high prices have a greater effect on good reviews. This is consistent with the idea that quality information matter in higher price classes. To continue this thought, advertising is often a part of the consumption. It is fair to assume it is a greater pleasure to offer a whine to their guests that has a good review. In comparison, the results show that a good review in a tabloid matter less. This is also consistent with the prestige theory. If we also take a look at the neutral review of tabloid, we can see that the sender in terms of neutral reviews is less important. Remember: the role of a neutral review is just to inform about the existence.

The newworld category examined, except of Portugal, only non-European wines. It seems that a review per se matter less it is about a non-European wine, but there is no significant difference for a good review. This suggests that non-European wines are better known and therefore that the review has less effect. This is consistent with the informative view of the review per se. This indicates that advertised wines should be better known as well, however the estimation from column 4 is essentially zero and not significant.

Another category examines if the impact of a review is different when advertising was allowed and not. If the advertising was persuasive and would create loyal customers, then reviews would matter less after advertising was allowed. However, this theory is not supported.

Further, wines that are already established on the market and therefore tested by a lot of costumers, reviews should matter less in these cases. Due to the fact that the wines that are used in this article are already relatively established, we defined a new wine as a wine that was introduced not more than two years before the appearance of the corresponding review. However, we could not find a significant difference.

An idea is that reviews have a greater impact on wines where the quality matters a lot and uncertainty exists. Thus, we tested wines that come in vintages (vs. non vintages). However, the numbers do not really support this idea. Also the estimation for wines that come from a region that exhibits high quality variation cannot support with numbers either. Both estimations are close to zero and insignificant.

Before we move on to the final conclusion of the paper and problem set. Try to answer some questions.

Quiz: How many categories refer to rev_all?

- 8 [ ]
- 6 [ ]
- 7 [x]
- 5 [ ]

Quiz: How many significant results do we have?

- 5 [ ]
- 3 [x]
- 2 [ ]
- 4 [ ]

Quiz: Do we have a 2.19 % decrement for good reviews in the medium / high price category?

- Yes [ ]
- No [x]

Question 3:
Look at the tabloid category. Add the correct parts to the sentence: "A good review in a tabloid is associated with a (answer1) percent (answer2) in sales / demand"

-- Omitted Quiz---

This exercise refers to page 207-209 in the paper

Exercise 12 Conclusion

So you are almost at the end of the problem set and I hope you liked it and learned a couple of things.

But before you go on and read the conclusion, check for one last time and see how many rewards you will get.

```
awards(as.html=TRUE)
```

You have earned 0 awards

Award: Problem Set Complete

Congratulations, You've earned this award for completing the Problem Set!

As a conclusion we found out that a good review has a significant impact on the demand of wines. The effect lasts roughly up to 22 weeks which is quite long. This means that good reviews in print medias are really valuable for the wine importers that also mainly decide the prices. But not only a good review has an effect, also just being reviewed has a small positive effect which is smaller and is more transient but is definitely there and valuable. A bad review is approximately zero, strictly speaking it is slightly negative but the effect of the review itself counterweights. Advertising expenditures also have an impact, but the effect is similar to a neutral review, which is rather minor and short lived.

But overall, the results suggest that a good review provide information about the wine and its quality which consumers act on, which means that the parties of interest obviously have an eye on every review about their products. As a comparison, the mentioned effect of a neutral review or advertising expenditure suggest that they rather bring the wine to the attention of customers, so this is more an informative channel and not a persuasive one.

Good reviews also have a higher impact on higher priced wines, and even stronger if the review publishes in the morning or targeted press than in tabloid. This is due to the prestige view of the review. The review basically is a part of the consumption, meaning it is a pleasure to serve a wine that has a good review to guests.

Even though the results show a significant impact of the reviews, especially because we also compared the impact of reviews on demand for newer versus older wines. However, the impact is most likely less for wines that have been on the market for a long time and are known to most of the consumers. One explanation might be that there are enough customers that just discovered a particular wine that has been one the market for quite some time. These can be people that are new to wine reviews and just shopped casually before or they are even completely new wine consumers.

What else can explain the impact. There is also a possibility that we have such an impact of reviews because the market lacks persuasive advertising or retailer sales promotion. But this is also because of the limits on the market. Remember that the monopoly retailer Systembolaget itself do not really have a possibility to enhance the sales.

Exercise 13 References

Bibliography

- Friberg, Richard and Erik Grönqvist. 2012. "Do Expert Reviews Affect the Demand for Wine?" American Economic Journal: Applied Economics, 4(1): 193-211. DOI: 10.1257/app.4.1.193
- James H. Stock, Mark W. Watson, Introduction to Econometrics, 3rd Edition Addison-Wesley (2010)
- Timothy C. Urdan, Statistics in Plain English, 3rd Edition, Routledge (2010)
- Peter Kennedy, A Guide to Econometrics, 6th Edition Wiley-Blackwell (2008)
- Hogg, R. V. and Craig, A. T. Introduction to Mathematical Statistics, 5th ed. New York: Macmillan, 1995.
- Chevalier, Judith A., and Dina Mayzlin. 2006. “The Effect of Word of Mouth on Sales: Online Book Reviews.” Journal of Marketing Research, 43 (3) : 345–54.

R and packages in R

- Kranz, Sebastian (2015): RTutor. Creating R problem sets with automatic assement of student's solutions, R package version 2015.12.22 Github Link
- R Core Team. (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. R-project.org/
- Jeremy Stephens (2014). yaml: Methods to convert R data to YAML and back. R package version 2.1.13 Cran R Project Link
- Wickham, Hadley, & Romain Francois. (2016). dplyr: A Grammar of Data Manipulation. R package version 0.5.0 Cran R Project Link
- Kranz, Sebastian (2016): dplyrExtras. Extra functionality for dplyr like mutate_rows for mutation of a subset of rows, R package version 0.1.3 2016.07.07 https://github.com/skranz/dplyrExtras
- Hadley Wickham, Winston Chang & RStudio. (2016). ggplot2: An Implementation of the Grammar of Graphics version 2.1.0 Cran R Project Link
- R Core Team (2015): foreign. Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, Weka, dBase, . R package version 0.8-66 Cran R Project Link
- R Core Team and contributors worldwide (2016). stats: R statistical functions version 3.4.0 Stat ethz.ch Link
- Hadley Wickham & RStudio. (2016). scales: Scale Functions for Visualization version 0.4.0 R Project Link
- Simen Gaure & Ragnar Frisch Centre for Economic Research. (2011-2016). lfe: Linear Group Fixed Effects version 2.5-1998. Cran R Project Link
- Hadley Wickham & RStudio. (2016). tidyr: Easily Tidy Data with 'spread()' and 'gather()' Functions version 0.5.1 Cran R Project Link
- David Robinson, (2016). broom: Convert Statistical Analysis Objects into Tidy Data Frames version 0.4.1 Cran R Project
- Marek Hlavac, (2015). stargazer: Well-Formatted Regression and Summary Statistics Tables version 5.2 Cran R Project PDF Link
- Markus Gesmann, Diego de Castillo & Joe Cheng, (2016). googleVis: R Interface to Google Charts version 0.6.0 Cran R Project Link

Websites

- Hadley Wickham, Winston Chang & RStudio. ggplot2 version 2.1.0 http://docs.ggplot2.org/current/
- Stata: Data Analysis and Statistical Software, Manuals13 Stata.com

Licence

Author: Sascha Hagedorn

Academic Value Added

The problem set gives students the opportunity to expand their knowledge in economics, statistics and R. They will be able to improve their programming skill in the language R by solving or understand small chunks that will step by step analyse a particular question. A variety of packages help the student to be as efficient as possible. It is going to be very beneficial to be familiar with the different packages used in the problem set.

The base of the problem set is the article “Do Expert Reviews Affect the Demand for Wine?” by Richard Friberg and Erik Grönqvist. All of the results are part of the problem set and the user will interactively try to create the results by solving the small chunks.

The problem set starts with an introduction about the main question of the paper. Afterwards, some facts about the paper are stated, so the user get’s a first impression about the topic. In the end of the introduction, it is explained how the problem set works and some instructions are given. The next exercise will introduce some basic such as loading a dataset or sorting. The exercise will also explain the used data sets and their variables. It also will let the user do some basic data investigations.

With the first knowledge, the user will continue to some more advanced chunks that will give the user and deeper knowledge about the data set and circumstances of the analysis. The user has to summarise and calculate shares of the wine distribution levels in terms of amount volumes. As a conclusion, the user will create line charts regarding the volumes with the help of the Google visualization API.

In the next exercise the user will proceed to some descriptive statistic across the different wines and segments (red, white & sparkling). The descriptive statistics contain statistical values such as mean, median and standard deviation about advertising expenditures, price per liter and liters per week. Afterwards, an exercise will be about the possible price changes of wines in the Swedish market. The user will summarise data and create a detailed bar plot.

The following exercise focuses on the grades for the wines provided from the six major print media in Sweden. Also here, a summary will give some insight to the user and the base to understand the regressions better later on. Because in fact, the grades are obviously the main aspect of the reviews and therefore also for the regressions.

Before the user comes to the main part of the problem set, some insightful plots will be created that show the number of reviews per week, a geo chart showing the volumes for each country and an interactive motion chart that examines volumes and numbers of segments. Then a chapter will present the empirical model which provides the main DNA of the regressions. The user will learn about different terms such as fixed effects, distributed lag model or ordinary least square that will provide deep knowledge about econometrics.

Afterwards, the user will dive with the first exercise into the first regression that will examine the impact of reviews & advertising expenditures in the sales & demands. The user will learn about the required R package and will also plot the results with confidence intervals in a scatter plot.

The same concepts and models will also be applied by the user in following exercise, but just for different variables. It will give the user the possibility to compare results and interpret them.

The last and most complicated part is a regression about categorical impact of reviews. This part is rather a part where the user just follows the code and does not solve it himself. This is due to the complexity, but even though, the user will learn a lot about R programming tricks, dummy variables and presenting of regression results. To finish, a conclusion will explain and interpret results of the problem set.

After finishing the problem set, the user will be capable to understand and apply the main models of the paper. He or she will have a deeper knowledge about economics, R and statistics and will be able to answer respectively solve questions with more confidence regarding the presented topics.

Conclusion

The possibilities that the package RTutor, that basically creates the problem set, R and R Markdown provide are really impressive.

RTutor makes it possible to create such an interactive problem set and to share the work by hosting it on shinyapps.io. It is also possible to solve the problem set offline by using R studio and a compatible browser. As already explained, the problem set gives insight about R and a field of study. The problem set does not have to be exclusively for economics. It also can be about finance or social science.

Also, the process of just creating the problem set has so much benefits for the creator. Personally, I learned a lot about R, didactical thinking and economics respectively statistics. It was such a good experience to create the problem set and my knowledge about R, R packages and economics will defiantly help me in the future.

I highly recommend students to take the chance to create a problem set. It will take some effort to create one, but I can confidently say that it will benefit the student strongly.

My problem set “Do Expert Reviews Affect the Demand for Wine” can be found on GitHub.

https://github.com/saschahsp/RTutorExpertReviewsWineDemand

You can also try the problem set yourself on shinyapps.io.

https://saschahsp.shinyapps.io/RTutorExpertReviewsWineDemand

Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Ich bin mir bewusst, dass eine unwahre Erklärung rechtliche Folgen haben wird.

Büchen, den 02.08.2016 ______________________________

(Unterschrift)