R_CODERS

R Analysis For Import Export

FOCUS IMPORT AND EXPORT ISSUES WITH R DEDICATED AFFECTS OF INLATION, CURRENCY AND FOREIGN EXCHANGE RATES

Reason for Project Topic

We started to hear import export amounts, foreign trade deficit, currency rates, inflation and many other financial terms more frequently in our daily life and financial analysis are getting more important day by day. There are a lot of trustable data sources for trading statistics, inflation and currency rates. We selected Turkey export/import analysis as Project topic because of that we want to understand the reasons behind economic status of Turkey and want to make an introduction to visualize and analyze financial data.

Executive Summary

- We investigated import/export amounts, currency rates, inflation rates and customer price index changes from 2010 to April 2018.
- We mainly looked changes in the export import amounts, main sectors for export and import and possible correlations between export/import amounts between currency rates and inflation.
- When export sub-sectors are investigated for last 3 years, we saw that top 3 export subsectors are manufacture of motor vehicles and trailers, manufacture of basic metals and manufacture of textiles.
- When import subsectors are investigated for last 3 years, we saw that top 3 import subsectors are manufacture of basic metals, manufacture of chemicals and chemical products, mining and quarrying.
- A significant decrease in import is seen between 2014-2016, when we investigated currency rates for the same period, we saw that dollar currency rate was increased dramatically which means that increases in currency rates have a decreasing impact over import amount.

- As an overall conclusion we can say that import amounts are always more than export amount which means there is a chronical foreign trade deficit problem, however foreign trade deficit is in a decreasing trend after 2013 except for 2017.

Group Project - Initilization and Project Diagrams

25.11.2018

Group Name : R_Coders

Project Name : Import - Export Analysis

##Project Members

- Büşra Aydemir
- Leyla Yiğit
- Mehmet Ak
- Mercan Karacabey
- Merve Özen Şahin

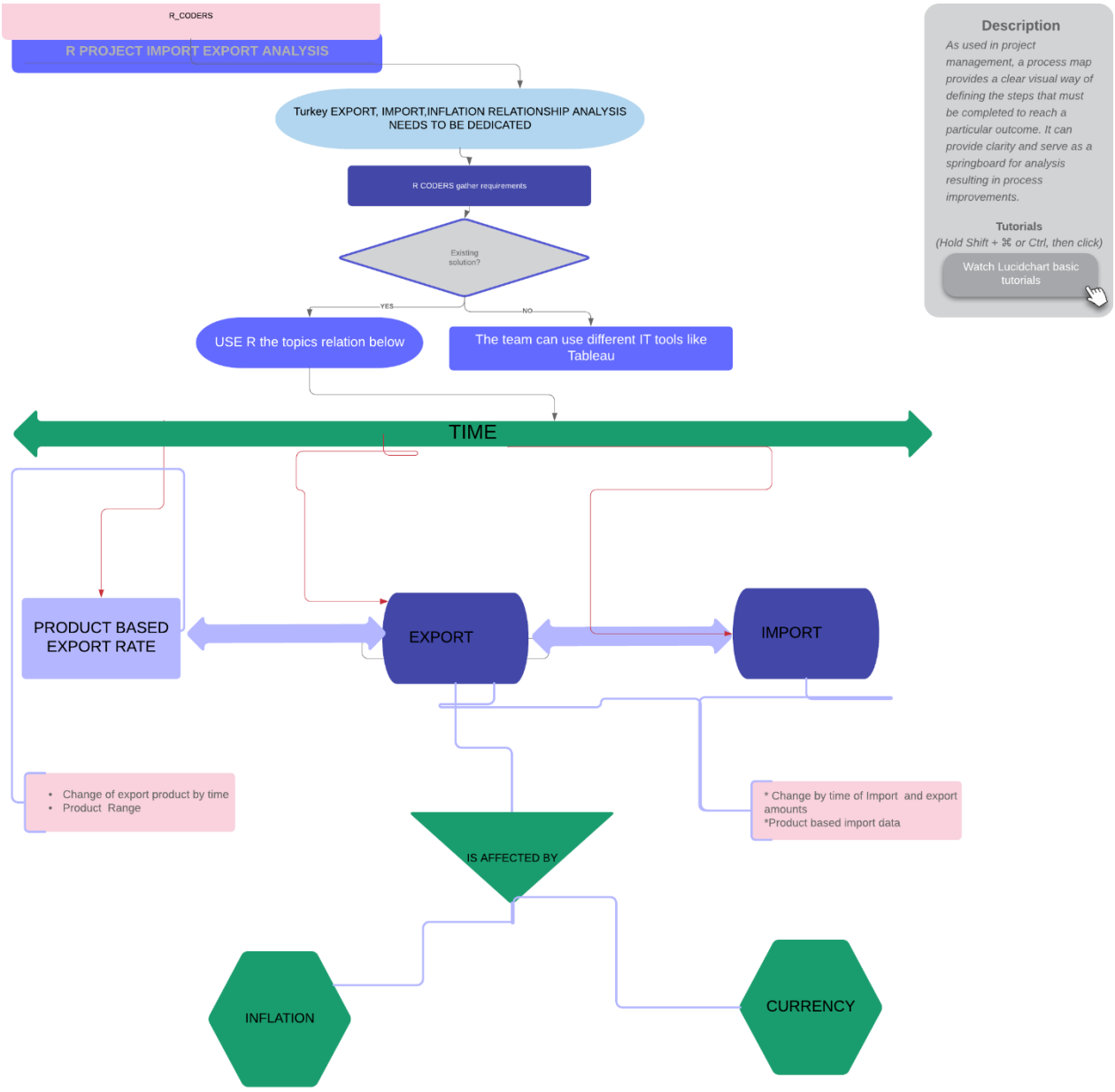
About Project

We will analyze the recent changes in inflation vs. data on exports and imports.We will analyze the factors which affecting export and import in 3 categories:

- Inflation
- Exchange Rates
- Interest Rates

Project Schema:

Project Management System Procurement Process



Details

We will examine export and import sector based. Some of the sectors which we will examine:

- Agriculture
- Production

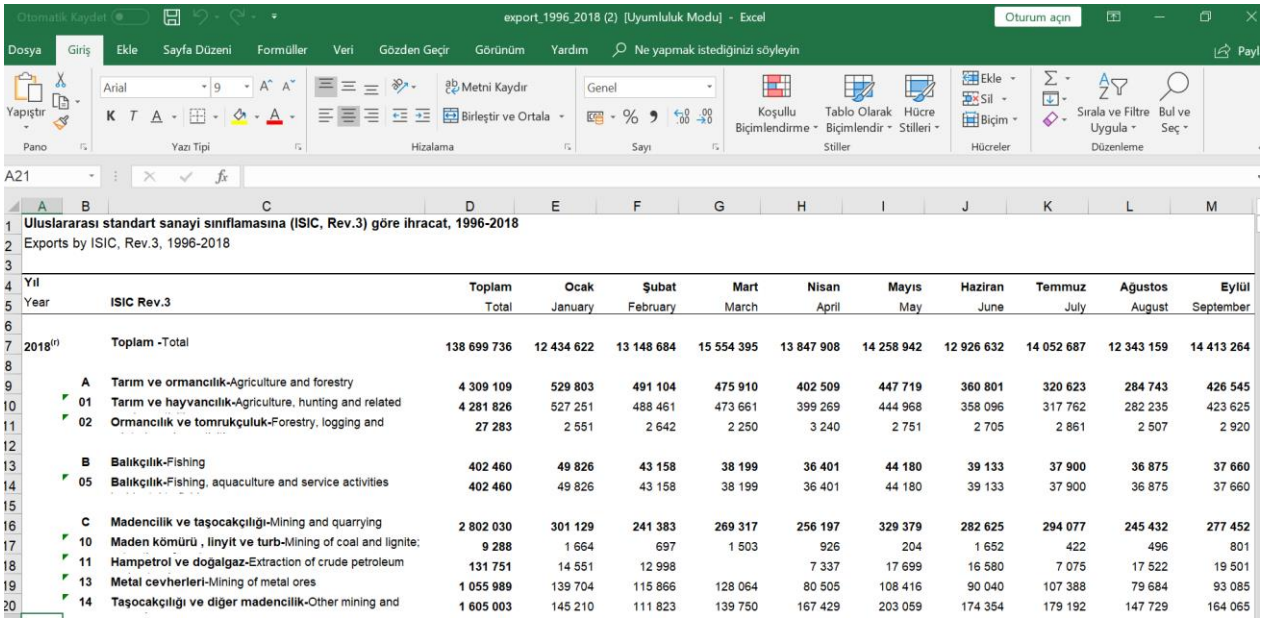
- Textile
- Metal Industry
- Plastic

Where do we get data?

- For Import: Import
- For Export: Export
- For Inflation: Inflation
- For Exchange Rate: ExchangeRates
- For Interest Rate: InterestRate

Import data uploaded Github. Here

Export Raw data structure

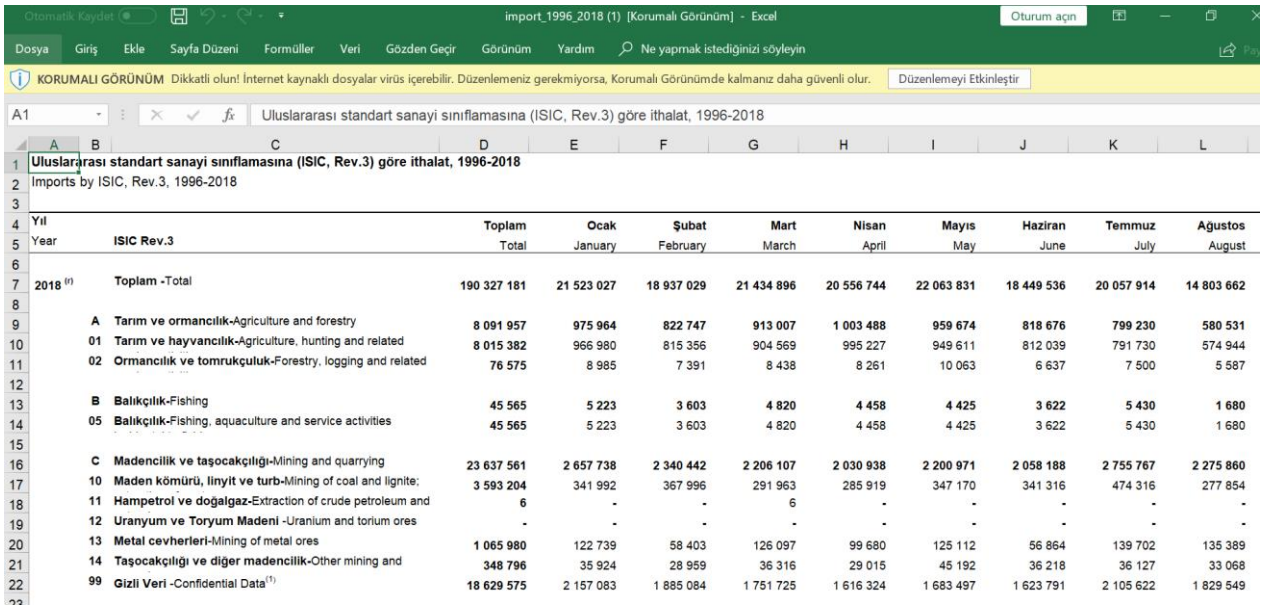


export_1996_2018 (2) [Uyumluluk Modu] - Excel

Uluslararası standart sanayi sınıflamasına (ISIC, Rev.3) göre ihracat, 1996-2018
Exports by ISIC, Rev.3, 1996-2018

Yıl / Year	ISIC Rev.3	Toplam / Total	Ocak / January	Şubat / February	Mart / March	Nisan / April	Mayıs / May	Haziran / June	Temmuz / July	Ağustos / August	Eylül / September
2018(r)	Toplam -Total	138 699 736	12 434 622	13 148 684	15 554 395	13 847 908	14 258 942	12 926 632	14 052 687	12 343 159	14 413 264
A	Tarım ve ormancılık-Agriculture and forestry	4 309 109	529 803	491 104	475 910	402 509	447 719	360 801	320 623	284 743	426 545
01	Tarım ve hayvancılık-Agriculture, hunting and related	4 281 826	527 251	488 461	473 661	399 269	444 968	358 096	317 762	282 235	423 625
02	Ormancılık ve tomrukçuluk-Forestry, logging and	27 283	2 551	2 642	2 250	3 240	2 751	2 705	2 861	2 507	2 920
B	Balıkçılık-Fishing	402 460	49 826	43 158	38 199	36 401	44 180	39 133	37 900	36 875	37 660
05	Balıkçılık-Fishing, aquaculture and service activities	402 460	49 826	43 158	38 199	36 401	44 180	39 133	37 900	36 875	37 660
C	Madencilik ve taşocakçılığı-Mining and quarrying	2 802 030	301 129	241 383	269 317	256 197	329 379	282 625	294 077	245 432	277 452
10	Maden kömürü , linyit ve turb-Mining of coal and lignite;	9 288	1 664	697	1 503	926	204	1 652	422	496	801
11	Hampetrol ve doğalgaz-Extraction of crude petroleum	131 751	14 551	12 998		7 337	17 699	16 580	7 075	17 522	19 501
13	Metal cevherleri-Mining of metal ores	1 055 989	139 704	115 866	128 064	80 505	108 416	90 040	107 388	79 684	93 085
14	Taşocakçılığı ve diğer madencilik-Other mining and	1 605 003	145 210	111 823	139 750	167 429	203 059	174 354	179 192	147 729	164 065

Import Raw Data Structure

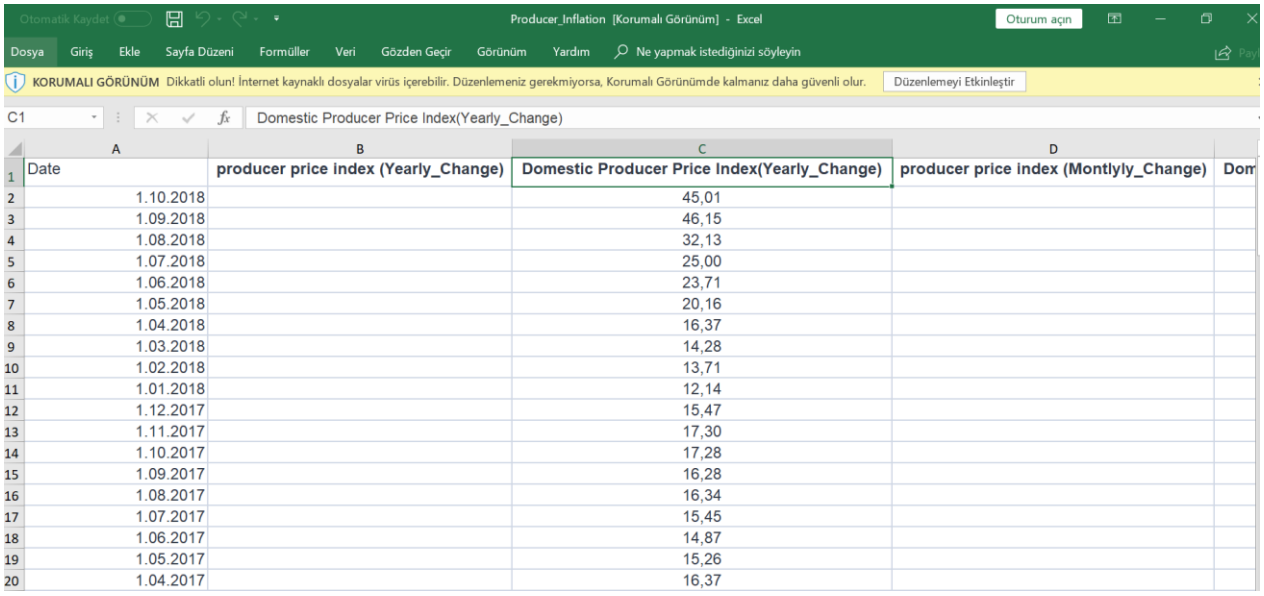


Uluslararası standart sanayi sınıflamasına (ISIC, Rev.3) göre ithalat, 1996-2018

Imports by ISIC, Rev.3, 1996-2018

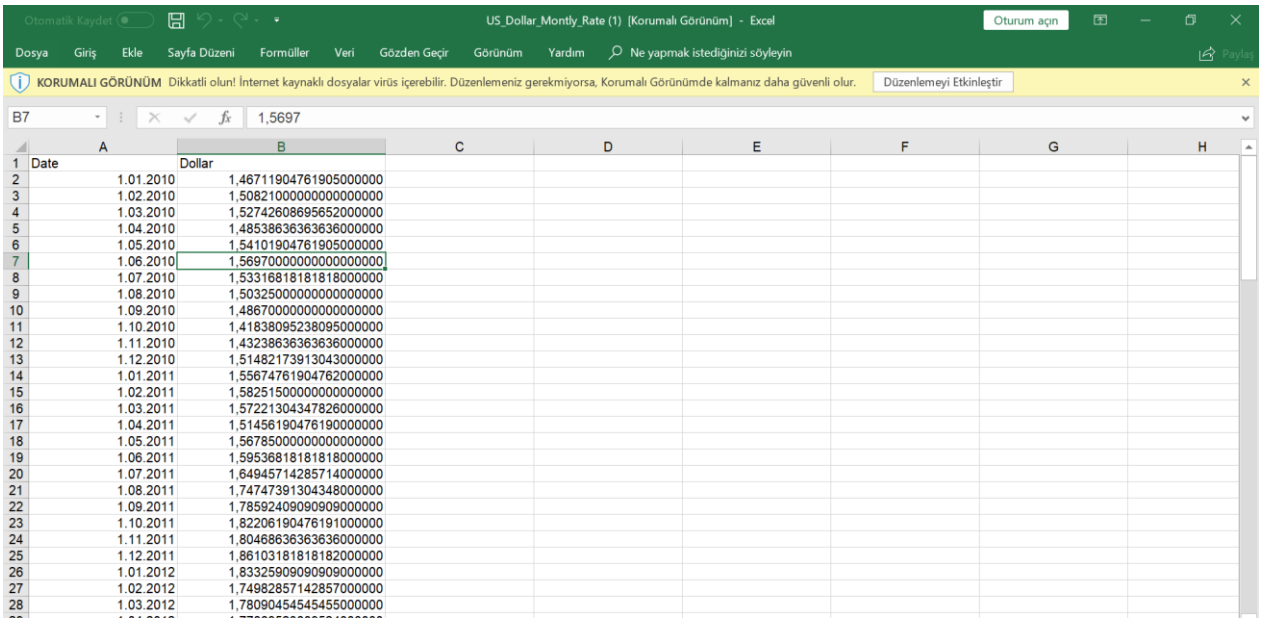
Yıl / Year	ISIC Rev.3	Toplam / Total	Ocak / January	Şubat / February	Mart / March	Nisan / April	Mayıs / May	Haziran / June	Temmuz / July	Ağustos / August
2018 (r)	**Toplam** -Total	190 327 181	21 523 027	18 937 029	21 434 896	20 556 744	22 063 831	18 449 536	20 057 914	14 803 662
A	**Tarım ve ormancılık**-Agriculture and forestry	8 091 957	975 964	822 747	913 007	1 003 488	959 674	818 676	799 230	580 531
01	**Tarım ve hayvancılık**-Agriculture, hunting and related	8 015 382	966 980	815 356	904 569	995 227	949 611	812 039	791 730	574 944
02	**Ormancılık ve tomrukçuluk**-Forestry, logging and related	76 575	8 985	7 391	8 438	8 261	10 063	6 637	7 500	5 587
B	**Balıkçılık**-Fishing	45 565	5 223	3 603	4 820	4 458	4 425	3 622	5 430	1 680
05	**Balıkçılık**-Fishing, aquaculture and service activities	45 565	5 223	3 603	4 820	4 458	4 425	3 622	5 430	1 680
C	**Madencilik ve taşocakçılığı**-Mining and quarrying	23 637 561	2 657 738	2 340 442	2 206 107	2 030 938	2 200 971	2 058 188	2 755 767	2 275 860
10	**Maden kömürü, linyit ve turb**-Mining of coal and lignite;	3 593 204	341 992	367 996	291 963	285 919	347 170	341 316	474 316	277 854
11	**Hampetrol ve doğalgaz**-Extraction of crude petroleum and	6	-	-	6	-	-	-	-	-
12	**Uranyum ve Toryum Madeni** -Uranium and torium ores	-	-	-	-	-	-	-	-	-
13	**Metal cevherleri**-Mining of metal ores	1 065 980	122 739	58 403	126 097	99 680	125 112	56 864	139 702	135 389
14	**Taşocakçılığı ve diğer madencilik**-Other mining and	348 796	35 924	28 959	36 316	29 015	45 192	36 218	36 127	33 068
99	**Gizli Veri** -Confidential Data(1)	18 629 575	2 157 083	1 885 084	1 751 725	1 616 324	1 683 497	1 623 791	2 105 622	1 829 549

Producer Inflation Raw Data



Date	producer price index (Yearly_Change)	Domestic Producer Price Index(Yearly_Change)	producer price index (Montlyly_Change)	Dom
1.10.2018		45,01		
1.09.2018		46,15		
1.08.2018		32,13		
1.07.2018		25,00		
1.06.2018		23,71		
1.05.2018		20,16		
1.04.2018		16,37		
1.03.2018		14,28		
1.02.2018		13,71		
1.01.2018		12,14		
1.12.2017		15,47		
1.11.2017		17,30		
1.10.2017		17,28		
1.09.2017		16,28		
1.08.2017		16,34		
1.07.2017		15,45		
1.06.2017		14,87		
1.05.2017		15,26		
1.04.2017		16,37		

USD Rate Raw Data



US_Dollar_Montly_Rate (1) [Korumalı Görünüm] - Excel

KORUMALI GÖRÜNÜM Dikkatli olun! İnternet kaynaklı dosyalar virüs içerebilir. Düzenlemeniz gerekmiyorsa, Korumalı Görünümde kalmanız daha güvenli olur. Düzenlemeyi Etkinleştir

B7: 1,5697

	A	B
1	Date	Dollar
2	1.01.2010	1,467119047619050000
3	1.02.2010	1,508210000000000000
4	1.03.2010	1,527426086956520000
5	1.04.2010	1,485386363636360000
6	1.05.2010	1,541019047619050000
7	1.06.2010	1,569700000000000000
8	1.07.2010	1,533168181818180000
9	1.08.2010	1,503250000000000000
10	1.09.2010	1,486700000000000000
11	1.10.2010	1,418380952380950000
12	1.11.2010	1,432386363636360000
13	1.12.2010	1,514821739130430000
14	1.01.2011	1,556747619047620000
15	1.02.2011	1,582515000000000000
16	1.03.2011	1,572213043478260000
17	1.04.2011	1,514561904761900000
18	1.05.2011	1,567850000000000000
19	1.06.2011	1,595368181818180000
20	1.07.2011	1,649457142857140000
21	1.08.2011	1,747473913043480000
22	1.09.2011	1,785924090909090000
23	1.10.2011	1,822061904761910000
24	1.11.2011	1,804686363636360000
25	1.12.2011	1,861031818181820000
26	1.01.2012	1,833259090909090000
27	1.02.2012	1,749828571428570000
28	1.03.2012	1,780904545454550000

Data preparation

a. Data Identification

The Data Identification stage is dedicated to identifying the datasets required for the analysis project and their sources. Our analysis examines the change between 2010 and 2018. Our main data used between 2010 - 2018 import and export results. In addition, inflation and currency figures are our main variables.

b. Data Acquisition & Filtering

```
IMPORT Data Source:

(" http://www.oaib.gov.tr/tr/default.html")

EXPORT DATA SOURCES

("http://www.tim.org.tr/tr/ihracat-rakamlari.html")

INFLATION

("https://www.tcmb.gov.tr/wps/wcm/connect/TR/TCMB+TR/Main+Menu/Istatistikler/Enflasyon+Verileri/Tuketici+Fiyatlari")

EXCHANGE RATE

("https://github.com/algopoly/EVDS")

INTEREST RATE
```

```
("https://evds2.tcmb.gov.tr/")
```

c. Data Extraction

Download export and import raw excel files and put them in temp files and read them. And remove the temp files after reading. We use “readxl” library for read data.

First create a temporary file, download file from repository to the temp file and read excel files.

d. Data Validation & Cleansing

Invalid data can skew and falsify analysis results. The Data Validation and Cleansing stage

is dedicated to establishing often complex validation rules and removing any known invalid data.

We removed the NA rows and change the column names of data. Also we don’t use total amount, because of this removed the total amount columns. And convert char data types to numeric data types for all columns except sector type code and sector name columns.

We use “plyr”, “tidyverse”, “stringr” packages for data validation and cleansing stage.

e. Data Aggregation & Representation

Data may be spread across multiple datasets, requiring that datasets be joined together via common fields, for example date or ID. In our cases, the same data fields appear in multiple datasets, such as sector type code and sector name. Either way, a method of data reconciliation is required or the dataset representing the correct value needs to be determined.

Data analysis

Explotary Analysis

R_Coders

December 20, 2018

Libraries

library("tidyverse")

The **tidyverse** is an opinionated collection of **R packages** designed for data science.
All **packages**share an underlying design philosophy, grammar, and data structures.

library("readxl")

The **readxl package** makes it easy to get data out of Excel and into **R**. Compared to many of the existing **packages** (e.g. gdata, xlsx, xlsReadWrite) **readxl** has no external dependencies, so it's easy to install and use on all operating systems.

library("ggplot2")

ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics.

library("plotly")

Plotly's R graphing **library** makes interactive, publication-quality graphs online

#library("xlsx")

An **R package** to read, write, format Excel 2007 and Excel 97/2000/XP/2003 files

library("sqldf")

sqldf is an R package for runing SQL statements on R data frames, optimized for convenience.

library("dplyr")

dplyr is a powerful **R-package** to transform and summarize tabular data with rows and columns.

```
install.packages("tidyverse", repos = "https://cran.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## also installing the dependency 'dplyr'
## package 'dplyr' successfully unpacked and MD5 sums checked
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
install.packages("dplyr", repos = "https://cran.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'dplyr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
```

```
install.packages("readxl", repos = "http://cran.us.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'readxl' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
install.packages("ggplot2", repos = "http://cran.us.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
install.packages("plotly", repos = "http://cran.us.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'plotly' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
install.packages("gapminder", repos = "http://cran.us.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'gapminder' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
#install.packages("xlsx", repos = "http://cran.us.r-project.org")
install.packages("sqldf", repos = "http://cran.us.r-project.org")
## Installing package into 'C:/Users/ozenm/Documents/R/win-library/3.5'
## (as 'lib' is unspecified)
## package 'sqldf' successfully unpacked and MD5 sums checked
##
```

```
## The downloaded binary packages are in
##  C:\Users\ozenm\AppData\Local\Temp\RtmpkrTOIg\downloaded_packages
```

```
library("tidyverse")
```

```
## -- Attaching packages ------------------------------------------------- t
idyverse 1.2.1 --
## <U+221A> ggplot2 3.1.0     <U+221A> purrr   0.2.5
## <U+221A> tibble  1.4.2     <U+221A> dplyr   0.7.8
## <U+221A> tidyr   0.8.2     <U+221A> stringr 1.3.1
## <U+221A> readr   1.3.0     <U+221A> forcats 0.3.0
## -- Conflicts ---------------------------------------------------- tidyver
se_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("readxl")
library("ggplot2")
library("plotly")
```

```
##
## Attaching package: 'plotly'
## The following object is masked from 'package:ggplot2':
##
##     last_plot
## The following object is masked from 'package:stats':
##
##     filter
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library("gapminder")
#library("xlsx")
library("sqldf")
```

```
## Loading required package: gsubfn
## Loading required package: proto
## Loading required package: RSQLite
```

```
library("dplyr")
tmp<-tempfile(fileext=".rds")
```

```
download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/imp_data_final.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

imp_data_final<-read_rds(tmp)

file.remove(tmp)
```

```
## [1] TRUE
```

```
tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/exp_data_final.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

exp_data_final<-read_rds(tmp)

file.remove(tmp)
```

```
## [1] TRUE
```

```
tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/imp_data.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

imp_data<-read_rds(tmp)

file.remove(tmp)
```

```
## [1] TRUE
```

```
tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/exp_data.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

exp_data<-read_rds(tmp)

file.remove(tmp)
```

```
## [1] TRUE
```

```
tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/Producer_Inflation.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

producer_inf<-read_rds(tmp)

file.remove(tmp)
```

```
## [1] TRUE
```

```
# Create a temporary file

tmp=tempfile(fileext=".xls")

# Download file from repository to the temp file

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Excel/export_import_sectors.xls?raw=true",destfile=tmp,mode='wb')

# Read that excel file.

sectors <- read_excel(tmp)
```

```
## readxl works best with a newer version of the tibble package.
## You currently have tibble v1.4.2.
## Falling back to column name repair from tibble <= v1.4.2.
## Message displays once per session.
```

```
# Remove the temp file
file.remove(tmp)
```

```
## [1] TRUE
```

```
tmp<-tempfile(fileext=".rds")
download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/US_Dollar_Montly_Rate.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')
usd_rate<-read_rds(tmp)
file.remove(tmp)
```

```
## [1] TRUE
```

Format Data

```
names(exp_data_final)[names(exp_data_final) == 'Date'] <- 'Export_Date'
names(exp_data)[names(exp_data) == 'Date'] <- 'Export_Date'
names(imp_data_final)[names(imp_data_final) == 'Date'] <- 'Import_Date'
names(imp_data_final)[names(imp_data_final) == 'Export_Total_Amount'] <- 'Import_Total_Amount' #fix
names(imp_data)[names(imp_data) == 'Date'] <- 'Import_Date'
```

```
library("dplyr")
exp_data <- inner_join(exp_data,sectors, by=c("Sector_Type_Code"="Sub_Sector_Type_Code"))
```

```
imp_data <- inner_join(imp_data,sectors, by=c("Sector_Type_Code"="Sub_Sector_Type_Code"))
```

```
exp_data$Export_Year<-as.numeric(format(exp_data$Export_Date,"%Y"))
exp_data$Export_Year_Month<-format(exp_data$Export_Date,"%Y-%m")
exp_data_final$Export_Year<-as.numeric(format(exp_data_final$Export_Date,"%Y"))
exp_data_final$Export_Year_Month<-format(exp_data_final$Export_Date,"%Y-%m")
```

```
imp_data$Import_Year<-as.numeric(format(imp_data$Import_Date,"%Y"))

imp_data$Import_Year_Month<-format(imp_data$Import_Date,"%Y-%m")

imp_data_final$Import_Year<-as.numeric(format(imp_data_final$Import_Date,"%Y")
)

imp_data_final$Import_Year_Month<-format(imp_data_final$Import_Date,"%Y-%m")


imp_data<- imp_data %>%

  select (Import_Date,Sector_Type_Code,Sector_Type_Code.y,Main_Sector_Flag,Sec
tor_Name_Eng,Amount,Import_Year,Import_Year_Month)




exp_data<- exp_data %>%

  select (Export_Date,Sector_Type_Code,Sector_Type_Code.y,Main_Sector_Flag,Sec
tor_Name_Eng, Amount,Export_Year,Export_Year_Month)




colnames(imp_data)[colnames(imp_data) == 'Amount'] <- 'Import_Amount'

colnames(exp_data)[colnames(exp_data) == 'Amount'] <- 'Export_Amount'

colnames(imp_data)[colnames(imp_data) == 'Sector_Type_Code'] <- 'Sub_Sector_Ty
pe_Code'

colnames(exp_data)[colnames(exp_data) == 'Sector_Type_Code'] <- 'Sub_Sector_Ty
pe_Code'

colnames(imp_data)[colnames(imp_data) == 'Sector_Type_Code.y'] <- 'Sector_Type
_Code'

colnames(exp_data)[colnames(exp_data) == 'Sector_Type_Code.y'] <- 'Sector_Type
_Code'

imp_data$Import_Amount[is.na(imp_data$Import_Amount)] <- 0

imp_data_final$Import_Total_Amount[is.na(imp_data_final$Import_Total_Amount)]
<- 0

exp_data$Export_Amount[is.na(exp_data$Export_Amount)] <- 0

exp_data_final$Export_Total_Amount[is.na(exp_data_final$Export_Total_Amount)]
<- 0




exp_data_final <- exp_data_final %>%

  filter(Export_Date<'2018-11-01')
```

```
exp_data <- exp_data %>%
  filter(Export_Date<'2018-11-01')


imp_data_final <- imp_data_final %>%
  filter(Import_Date<'2018-11-01')


imp_data <- imp_data %>%
  filter(Import_Date<'2018-11-01')


saveRDS(imp_data,file="imp_data_v2.rds")
saveRDS(imp_data_final,file="imp_data_final_v2.rds")
saveRDS(exp_data,file="exp_data_v2.rds")
saveRDS(exp_data_final,file="exp_data_final_v2.rds")
```

Review Import Data Structure

```
str(imp_data)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    4236 obs. of  8 variables:
##  $ Import_Date         : Date, format: "2018-01-01" "2018-01-01" ...
##  $ Sub_Sector_Type_Code: chr  "A" "01" "02" "B" ...
##  $ Sector_Type_Code    : chr  "A" "A" "A" "B" ...
##  $ Main_Sector_Flag    : num  1 0 0 1 0 1 0 0 0 0 ...
##  $ Sector_Name_Eng     : chr  "Agriculture and forestry" "Agriculture, hunt
ing and related service activities" "Forestry, logging and related service act
ivities" "Fishing" ...
##  $ Import_Amount       : num  975964 966980 8985 5223 5223 ...
##  $ Import_Year         : num  2018 2018 2018 2018 2018 ...
##  $ Import_Year_Month   : chr  "2018-01" "2018-01" "2018-01" "2018-01" ...
```

```
str(imp_data_final)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  7 variables:
##  $ Import_Date                          : Date, format: "2010-07-01" "2010-08
-01" ...
##  $ Import_Total_Amount                  : num  26002098 24474224 24503919 268
42444 26584903 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4
.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3
0.41 0.73 0.42 0.87 ...
```

```
##  $ USD_Rate                          : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Import_Year                       : num  2010 2010 2010 2010 2010 ...
##  $ Import_Year_Month                 : chr  "2010-07" "2010-08" "2010-09"
"2010-10" ...
```

Review Export Data Structure

```
str(exp_data)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    4452 obs. of  8 variables:
##  $ Export_Date         : Date, format: "2018-01-01" "2018-01-01" ...
##  $ Sub_Sector_Type_Code: chr  "A" "01" "02" "B" ...
##  $ Sector_Type_Code    : chr  "A" "A" "A" "B" ...
##  $ Main_Sector_Flag    : num  1 0 0 1 0 1 0 0 0 0 ...
##  $ Sector_Name_Eng     : chr  "Agriculture and forestry" "Agriculture, hunt
ing and related service activities" "Forestry, logging and related service act
ivities" "Fishing" ...
##  $ Export_Amount       : num  529803 527251 2551 49826 49826 ...
##  $ Export_Year         : num  2018 2018 2018 2018 2018 ...
##  $ Export_Year_Month   : chr  "2018-01" "2018-01" "2018-01" "2018-01" ...
```

```
str(exp_data_final)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  7 variables:
##  $ Export_Date                         : Date, format: "2010-07-01" "2010-08
-01" ...
##  $ Export_Total_Amount                 : num  19129365 17046904 17818461 219
27173 18764739 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4
.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3
0.41 0.73 0.42 0.87 ...
##  $ USD_Rate                            : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Export_Year                         : num  2010 2010 2010 2010 2010 ...
##  $ Export_Year_Month                   : chr  "2010-07" "2010-08" "2010-09"
"2010-10" ...
```

Prepare Data for Import&Export Line Graph

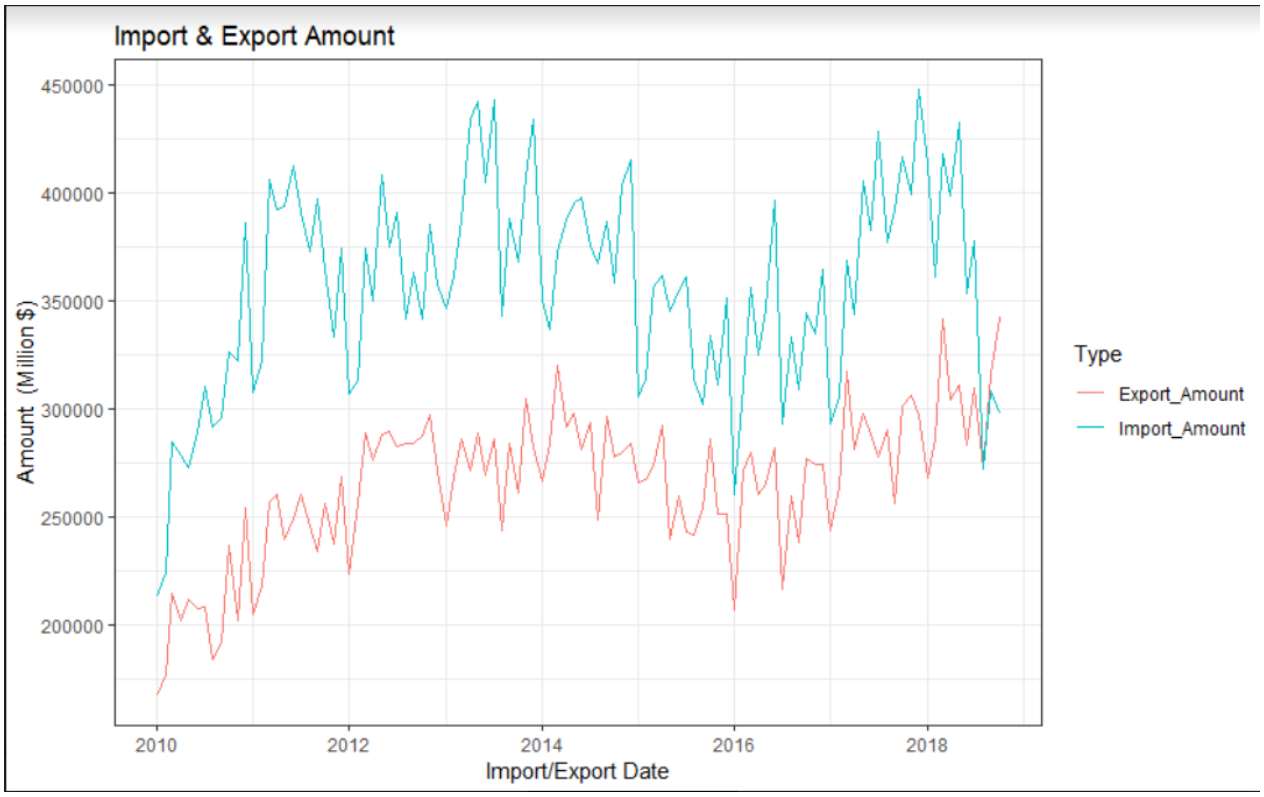
```
imp_and_exp_data <- inner_join(exp_data, imp_data, by=c("Export_Date" = "Impor
t_Date","Sub_Sector_Type_Code"="Sub_Sector_Type_Code"))
```

```
imp_and_exp_data_bymonth <- aggregate(cbind(Import_Amount, Export_Amount) ~ Ex
port_Date, data = imp_and_exp_data, sum)

imp_and_exp_data_bymonth <- gather(imp_and_exp_data_bymonth,

                                   value = "value",

                                   key = "type",

                                   Export_Amount, Import_Amount)


#Rename column names

colnames(imp_and_exp_data_bymonth) <- c("Date","Type","Amount")


#Remove Empty Dates

imp_and_exp_data_bymonth <- imp_and_exp_data_bymonth %>%

  filter(Date<'2018-11-01')
```

01_export_and_import_amount

```
 p<-ggplot(imp_and_exp_data_bymonth,

        aes(x=Date,

            y=Amount/1000,

            color=Type)) +

   geom_line()+

   scale_size_area("Nitrogen") +

   xlab("Import/Export Date") +

   ylab("Amount  (Million $)") +

   ggtitle("Import & Export Amount")

 style(p, text = row.names(imp_and_exp_data_bymonth))
```

2010201220142016201820000300004000
Export_AmountImport_AmountImport & Export AmountImport/Export DateAmount (Million $)Type



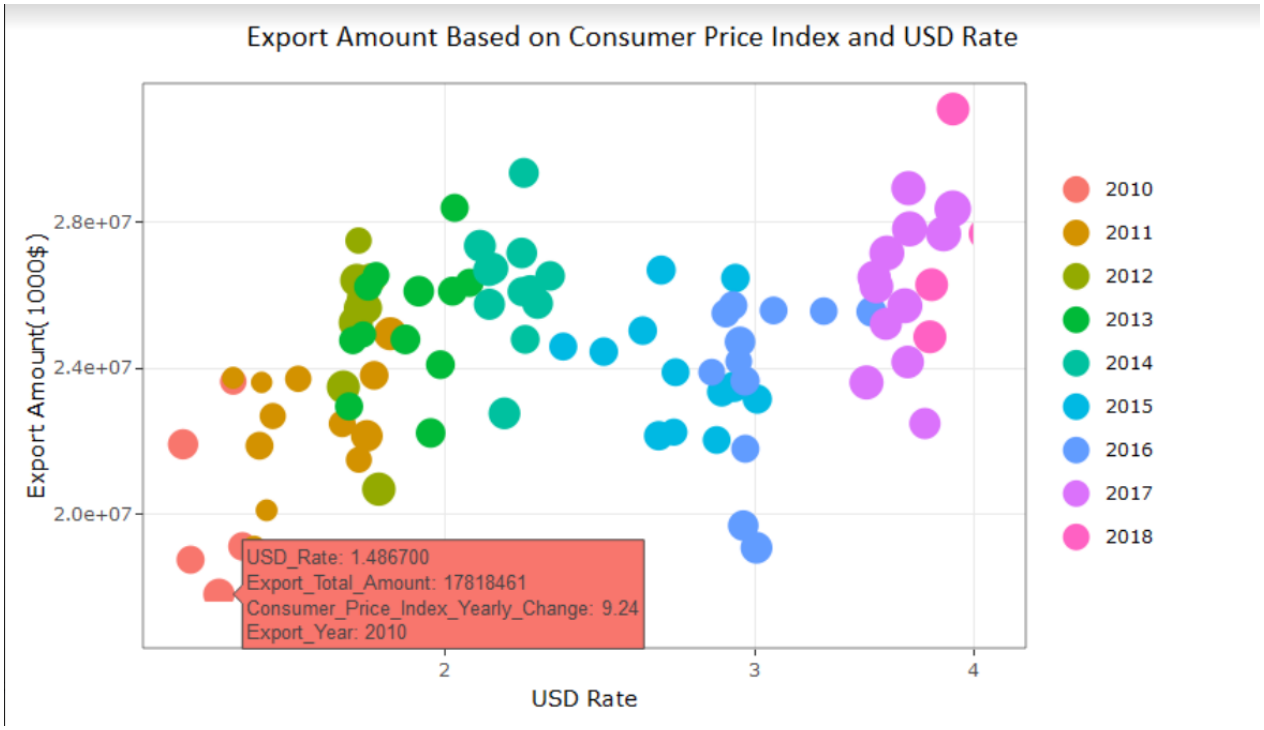
The graph shows the year-based comparison of export and import amount data. A very long period of increases and decreases are moving synchronously. In April 2018, the import amount Line has a major decline, while export amount Line has a major increase.

02_Export_Amount_Based_on_Consumer_Price_Index_And_USD_Rate

```
library(ggplot2)
library(plotly)
library(gapminder)


p <- exp_data_final %>%
  ggplot(aes(USD_Rate, Export_Total_Amount, size = Consumer_Price_Index_Yearly
_Change, color=Export_Year)) +
  geom_point() +
  scale_x_log10() +
  theme_bw()+
  scale_size_area("Nitrogen") +
```

```
  xlab("USD Rate") +
  ylab("Export Amount(1000$)") +
  ggtitle("Export Amounts and Consumer Price Index")


ggplotly(p)
```

2342.0e+072.4e+072.8e+07
Export Amounts and Consumer Price Index201020122014201620182018Export_YearUSD RateExport Amount(1000$)



The graph shows export amount based on consumer price index and USD rate on monthly basis. Each year is indicated in different colors and every point shows monthly variables values.

03_Import_Amount_Based_on_Consumer_Price_Index_And_USD_Rate

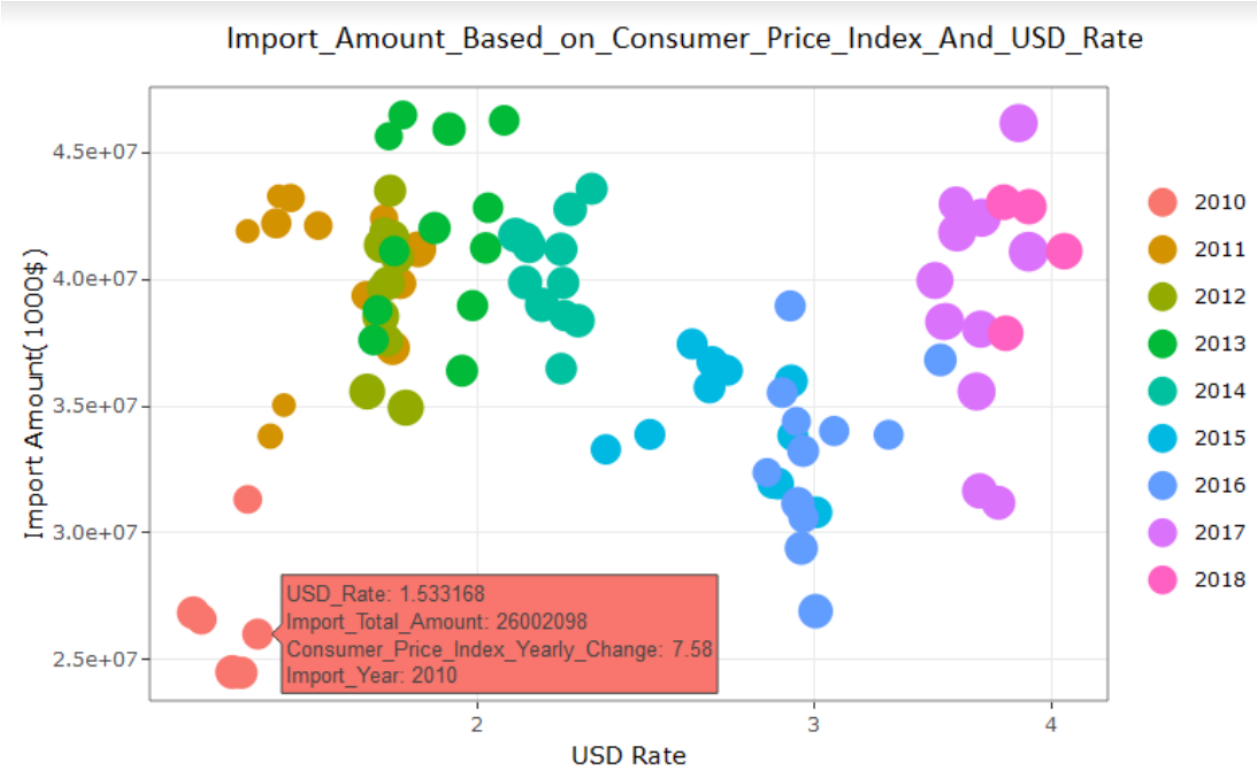
```
p <- imp_data_final %>%
  ggplot(aes(USD_Rate, Import_Total_Amount, size = Consumer_Price_Index_Yearly
_Change, color=Import_Year)) +
  geom_point() +
  scale_x_log10() +
```

```
  theme_bw()+
  scale_size_area("Nitrogen") +
  xlab("USD Rate") +
  ylab("Import Amount(1000$)") +
  ggtitle("Import Amounts and Consumer Price Index")


ggplotly(p)
```

2342.5e+073.0e+073.5e+074.0e+074.5e+07
Import Amounts and Consumer Price Index2010201220142016 2018Import_YearUSD RateImport Amount(1000$)

```
top_import_by_sector <-
  imp_data %>%
  filter(Main_Sector_Flag==1)%>% #& Import_Year==2018) %>%
  group_by(Import_Year,Sector_Type_Code,Sector_Name_Eng) %>%
  summarise(Import_Total_Amount=sum(Import_Amount))%>%
  arrange(desc(Import_Total_Amount))




top_export_by_sector <-
  exp_data %>%
  filter(Main_Sector_Flag==1)%>% #& Export_Year==2018) %>%
  group_by(Export_Year,Sector_Type_Code,Sector_Name_Eng) %>%
  summarise(Export_Total_Amount=sum(Export_Amount))%>%
  arrange(desc(Export_Total_Amount))


trade_deficit_by_sectors<-sqldf('select Export_Year as Year, a.Sector_Type_Cod
e,
a.Sector_Name_Eng, Export_Total_Amount, Import_Total_Amount,
(Import_Total_Amount - Export_Total_Amount) as Trade_Deficit_Amount
    from top_export_by_sector a left join top_import_by_sector b
      on Export_Year = Import_Year
      and a.Sector_Type_Code = b.Sector_Type_Code')
```

```
trade_deficit_by_sectors$Import_Total_Amount[is.na(trade_deficit_by_sectors$Import_Total_Amount)] <- 0

trade_deficit_by_sectors$Trade_Deficit_Amount[is.na(trade_deficit_by_sectors$Trade_Deficit_Amount)] <- 0
```

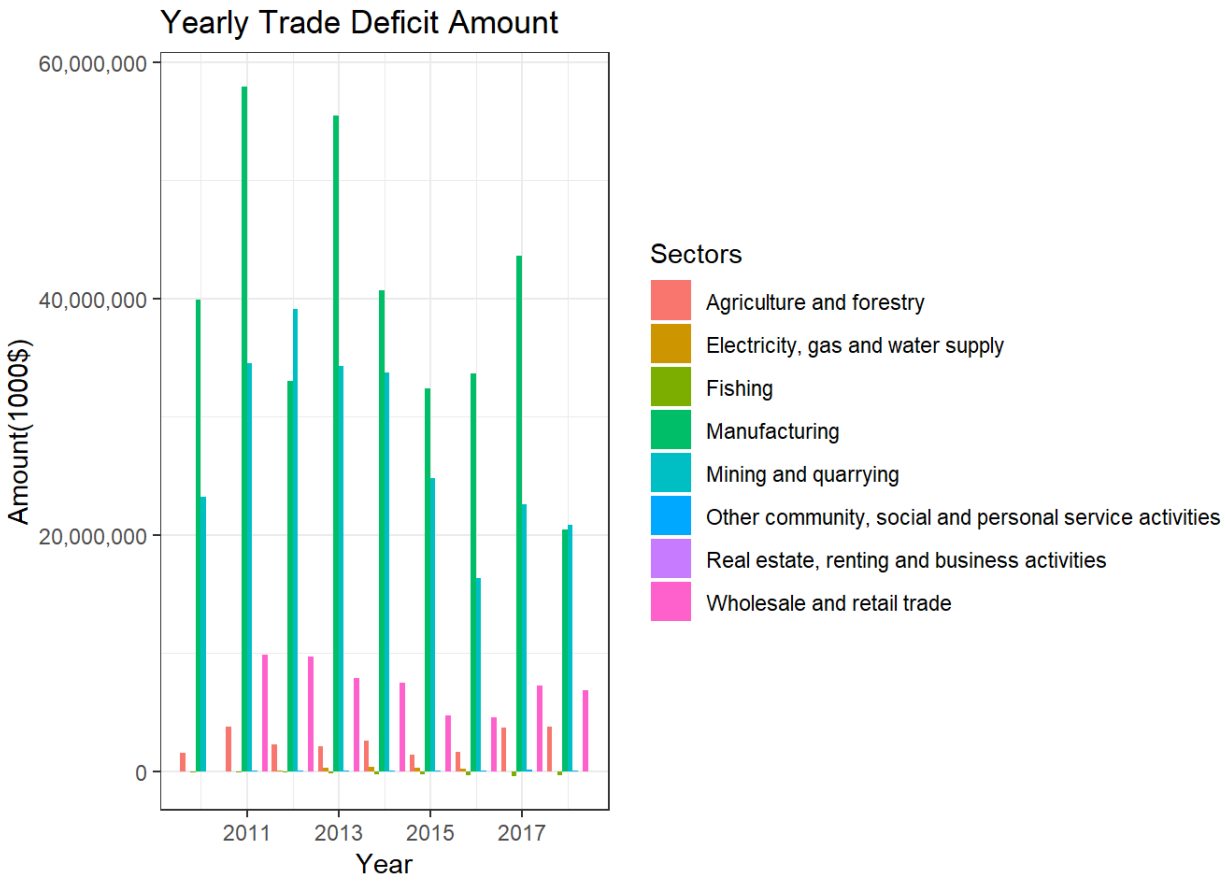


The graph shows import amount based on consumer price index and USD rate on monthly basis. Each year is indicated in different colors and every point shows monthly variables values.

04_yearly_trade_deficit_amount

```
trade_deficit_by_sectors %>%
  #filter(Main_Sector_Flag==1)%>%
  group_by(Year,Sector_Name_Eng) %>%
  summarise(Yearly_Total_Trade_Deficit_Amount=sum(Trade_Deficit_Amount))%>%
  ggplot(data = ., aes(x = Year, y = Yearly_Total_Trade_Deficit_Amount,
                       fill = Sector_Name_Eng)) + geom_bar(stat = "identity",
position=position_dodge()) + aes(x = Year, y =   Yearly_Total_Trade_Deficit_Amount) + labs(x = "", y = "", title = "Yearly Trade Deficit Amount") + theme_bw() +  scale_y_continuous(labels = scales::comma) +   guides(fill=guide_legend(title="Sectors"))+
  xlab("Year") +
```

```
  ylab("Amount(1000$)")
```



Yearly trade deficit amount graph was prepared based on the sector. In general, there is a constant decline. Although a clear result can not be seen due to the incomplete year for 2018, we can see that the year will be closed down again for two months due to the close of the year.

Prepare Data for Yearly Average Export Amount and Other Factors

```
library("dplyr")

exp_data_total_amount_by_year <-

  exp_data %>%

  group_by(Export_Date,Export_Year) %>%

  summarise(Yearly_Export_Total_Amount=sum(Export_Amount))


exp_data_amount_and_others <- inner_join(exp_data_final,exp_data_total_amount_
by_year, by=c("Export_Date" = "Export_Date"))

str(exp_data_amount_and_others)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  9 variables:
##  $ Export_Date                          : Date, format: "2010-07-01" "2010-08
-01" ...
##  $ Export_Total_Amount                  : num  19129365 17046904 17818461 219
27173 18764739 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4
.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3
0.41 0.73 0.42 0.87 ...
##  $ USD_Rate                             : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Export_Year.x                        : num  2010 2010 2010 2010 2010 ...
##  $ Export_Year_Month                    : chr  "2010-07" "2010-08" "2010-09"
"2010-10" ...
##  $ Export_Year.y                        : num  2010 2010 2010 2010 2010 ...
##  $ Yearly_Export_Total_Amount           : num  19129365 17046904 17818461 219
27173 18764739 ...
```

```
library("dplyr")

exp_data_amount_and_others$Export_Year.x <- as.numeric(as.character(exp_data_a
mount_and_others$Export_Year.x))

str(exp_data_amount_and_others)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  9 variables:
##  $ Export_Date                          : Date, format: "2010-07-01" "2010-08
-01" ...
##  $ Export_Total_Amount                  : num  19129365 17046904 17818461 219
27173 18764739 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4
.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3
0.41 0.73 0.42 0.87 ...
##  $ USD_Rate                             : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Export_Year.x                        : num  2010 2010 2010 2010 2010 ...
##  $ Export_Year_Month                    : chr  "2010-07" "2010-08" "2010-09"
"2010-10" ...
##  $ Export_Year.y                        : num  2010 2010 2010 2010 2010 ...
##  $ Yearly_Export_Total_Amount           : num  19129365 17046904 17818461 219
27173 18764739 ...
```

```
colnames(exp_data_amount_and_others)[colnames(exp_data_amount_and_others) == '
Export_Year.x'] <- 'Export_Year'


  exp_data_amount_and_others_yearly <-
```

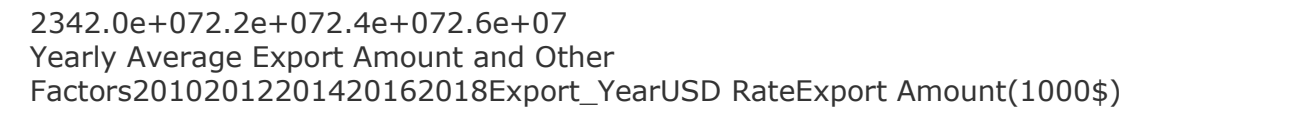
```
  exp_data_amount_and_others %>%
  group_by(Export_Year) %>%
  summarise(Yearly_Avg_Export_Amount=mean(Yearly_Export_Total_Amount),
            Yearly_Avg_Consumer_Price_Index_Yearly_Change = mean(Consumer_Price_Index_Yearly_Change),
            Yearly_Avg_Consumer_Price_Index_Monthly_Change = mean(Consumer_Price_Index_Monthly_Change),
            Yearly_Avg_USD_Rate = mean(USD_Rate))


str(exp_data_amount_and_others)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  9 variables:
##  $ Export_Date                          : Date, format: "2010-07-01" "2010-08-01" ...
##  $ Export_Total_Amount                  : num  19129365 17046904 17818461 21927173 18764739 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3 0.41 0.73 0.42 0.87 ...
##  $ USD_Rate                             : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Export_Year                          : num  2010 2010 2010 2010 2010 ...
##  $ Export_Year_Month                    : chr  "2010-07" "2010-08" "2010-09" "2010-10" ...
##  $ Export_Year.y                        : num  2010 2010 2010 2010 2010 ...
##  $ Yearly_Export_Total_Amount           : num  19129365 17046904 17818461 21927173 18764739 ...
```

05-Yearly Average Export Amount and Other Factors

```
#Export_Yearly_Avg_Values
library("ggplot2")
library("plotly")
  p <- exp_data_amount_and_others_yearly %>%
    ggplot(aes(Yearly_Avg_USD_Rate, Yearly_Avg_Export_Amount, size = Yearly_Avg_Consumer_Price_Index_Yearly_Change, color=Export_Year)) +
    geom_point() +
    scale_x_log10() +
    theme_bw()+
    xlab("USD Rate") +
```

```
    ylab("Export Amount(1000$)") +
    ggtitle("Yearly Average Export Amount and Other Factors")


  ggplotly(p)
```

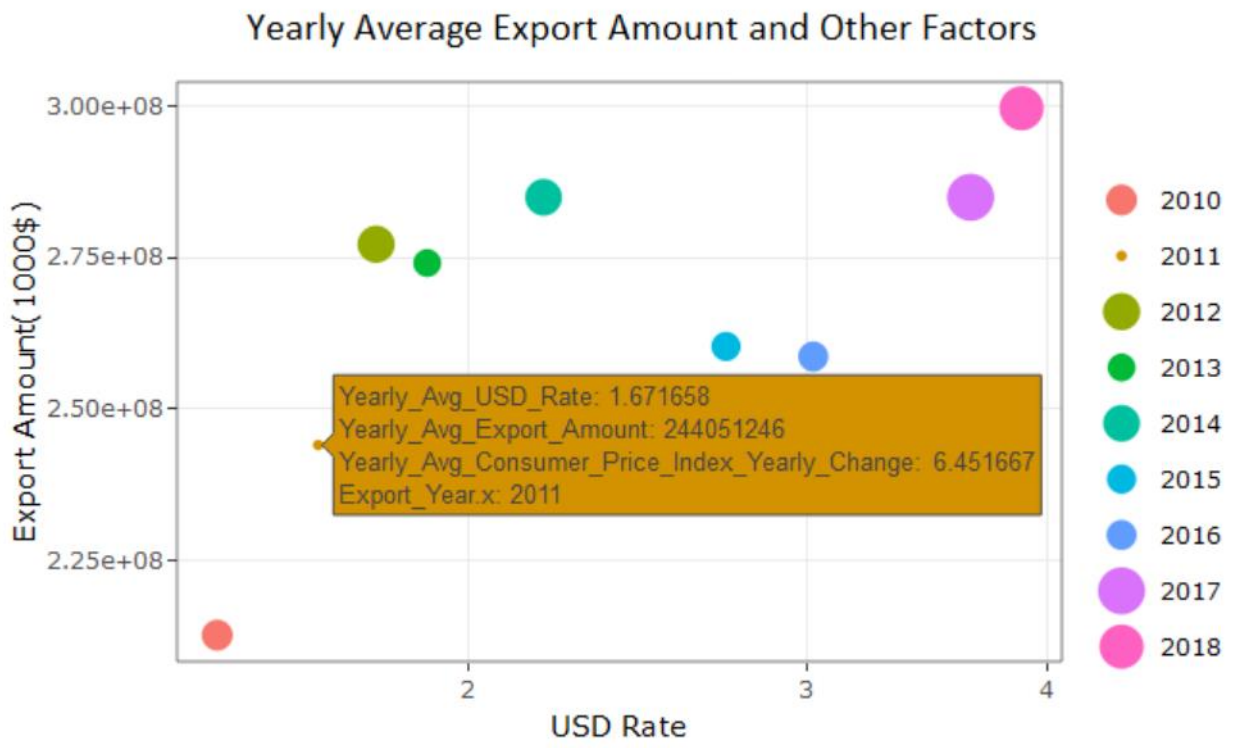



```
exp_min_max_mean_by_sectors<- exp_data %>%
  filter(Main_Sector_Flag==1)%>%
  group_by(Export_Year,Sector_Name_Eng) %>%
  summarise_each(funs(min(.,na.rm=TRUE), round(mean(.,na.rm=TRUE),digits = 3),
max(.,na.rm=TRUE),sum(.,na.rm=TRUE)),Export_Amount)
```

```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over a selection of variables, use `summarise_at()`
```

```
head(exp_min_max_mean_by_sectors)
```

```
## # A tibble: 6 x 6
## # Groups:   Export_Year [1]
##   Export_Year Sector_Name_Eng                     min   round     max     sum
##         <dbl> <chr>                             <dbl>   <dbl>   <dbl>   <dbl>
## 1        2010 Agriculture and forestry        2.87e5 4.11e5  5.70e5  4.93e6
## 2        2010 Electricity, gas and water s~   7.61e3 1.51e4  2.63e4  1.81e5
## 3        2010 Fishing                         8.28e3 1.30e4  2.57e4  1.56e5
## 4        2010 Manufacturing                   7.18e6 8.79e6  1.09e7  1.05e8
## 5        2010 Mining and quarrying            1.48e5 2.24e5  2.60e5  2.69e6
## 6        2010 Other community, social and ~   4.18e1 3.01e2  5.89e2  3.62e3
```



Yearly average export amount and other factors show the inflation rate of the points in the chart. At every point can be seen yearly average USD rate, yearly average export amount, yearly average consumer price index yearly change.

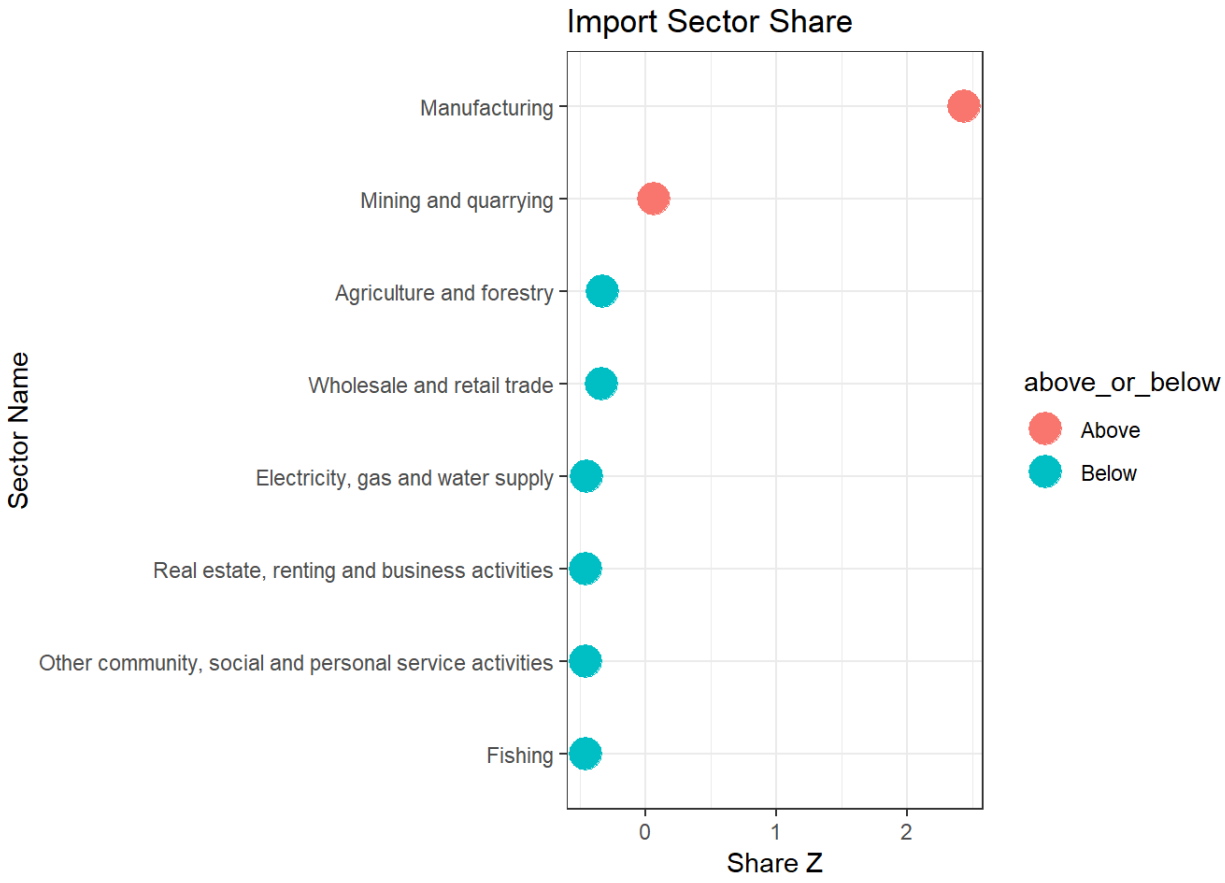
06_Export_Sector_Share

```
exp_share_sectors <-
  exp_data  %>%
  filter(Main_Sector_Flag==1 & Export_Date<'2018-11-01')%>%
  group_by(Sector_Name_Eng) %>%
  summarize(Export_Amount_Share=sum(Export_Amount)) %>%
  mutate (Export_Amount_Share=round((Export_Amount_Share/sum(Export_Amount_Sha
re)),4))



exp_share_sectors$share_z <- round((exp_share_sectors$Export_Amount_Share - me
an(exp_share_sectors$Export_Amount_Share))/sd(exp_share_sectors$Export_Amount_
Share), 2)
```

```
exp_share_sectors$above_or_below <- ifelse(exp_share_sectors$share_z < 0, "Below", "Above")

exp_share_sectors <- exp_share_sectors[order(exp_share_sectors$share_z), ]

exp_share_sectors$Sector_Name_Eng <- factor(exp_share_sectors$Sector_Name_Eng, levels = exp_share_sectors$Sector_Name_Eng)



theme_set(theme_bw())


ggplot(exp_share_sectors, aes(x= share_z   , y=  Sector_Name_Eng, label=share_z)) +
  xlab("Share Z") +
  ylab("Sector Name") +
  ggtitle("Export Sector Share")+
  geom_point(stat='identity', aes(col=above_or_below), size=6) #+
```

06_Import_Sector_Share

```
imp_share_sectors <-

  imp_data  %>%

  filter(Main_Sector_Flag==1 & Import_Date<'2018-11-01')%>%

  group_by(Sector_Name_Eng) %>%

  summarize(Import_Amount_Share=sum(Import_Amount)) %>%

  mutate (Import_Amount_Share=round((Import_Amount_Share/sum(Import_Amount_Sha
re)),4))



imp_share_sectors$share_z <- round((imp_share_sectors$Import_Amount_Share - me
an(imp_share_sectors$Import_Amount_Share))/sd(imp_share_sectors$Import_Amount_
Share), 2)


imp_share_sectors$above_or_below <- ifelse(imp_share_sectors$share_z < 0, "Bel
ow", "Above")

imp_share_sectors <- imp_share_sectors[order(imp_share_sectors$share_z), ]

imp_share_sectors$Sector_Name_Eng <- factor(imp_share_sectors$Sector_Name_Eng,
levels = imp_share_sectors$Sector_Name_Eng)



 theme_set(theme_bw())


 ggplot(imp_share_sectors, aes(x= share_z    , y=  Sector_Name_Eng, label=share
_z)) +

   xlab("Share Z") +

   ylab("Sector Name") +

   ggtitle("Import Sector Share")+

   geom_point(stat='identity', aes(col=above_or_below), size=6)
```



The graph shows the sector shares in the total import amount. Only two sector is above the sector average import amount because of that manufacturing is extremely higher than other sectors. Mining is also important but manufacturing import amount is higher than the total of the other sector import amounts.

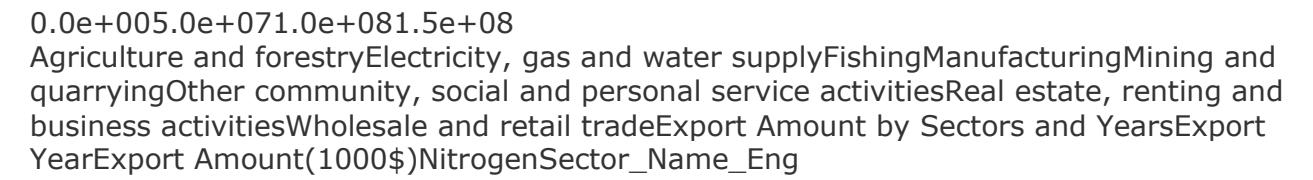
07_Export_Amount_by_Sectors_And_Year

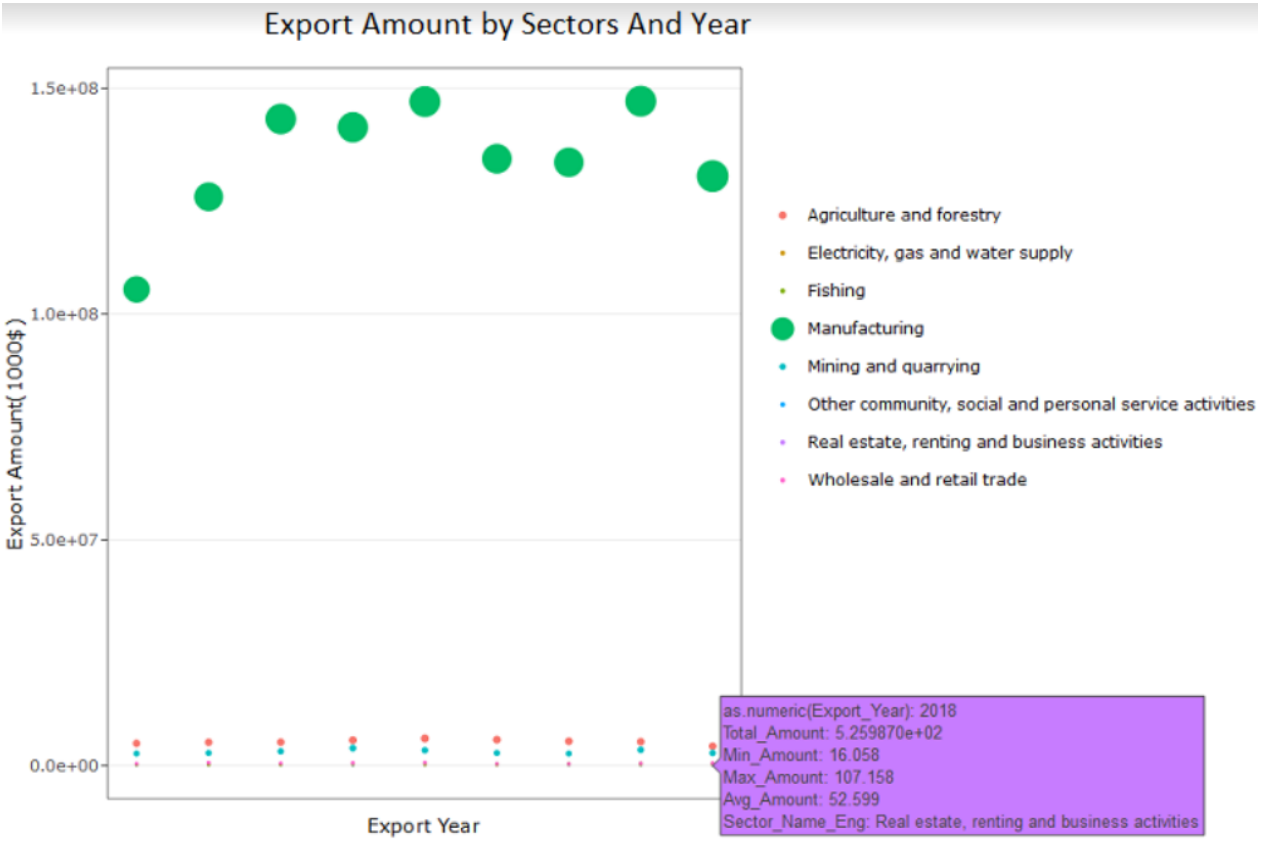
```
exp_agg_by_sectors<- exp_data %>%
  filter(Main_Sector_Flag==1)%>%
  group_by(Export_Year,Sector_Name_Eng) %>%
  summarise_each(funs(min(.,na.rm=TRUE), round(mean(.,na.rm=TRUE),digits = 3), max(.,na.rm=TRUE),sum(.,na.rm=TRUE)),Export_Amount)
```

```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over a selection of variables, use `summarise_at()`
```

```
library(ggplot2)
library(plotly)
```

```
library(gapminder)


colnames(exp_agg_by_sectors)<- c("Export_Year", "Sector_Name_Eng", "Min_Amount
", "Avg_Amount", "Max_Amount", "Total_Amount" )


p <- exp_agg_by_sectors %>%

    ggplot(aes(x = Export_Year, y= Total_Amount, group=Min_Amount, group2 = Ma
x_Amount, size = Avg_Amount, color=Sector_Name_Eng)) +

    geom_point() +

    scale_x_log10() +

    theme_bw()+

    scale_size_area("Nitrogen") +

    xlab("Export Year") +

    ylab("Export Amount(1000$)") +

    ggtitle("Export Amount by Sectors and Years")



ggplotly(p)
```





The graph shows the export amounts by sectors and years. We can see that manufacturing is the main part of the export amount. Manufacturing export amount is sharply increased between 2010-2013 and it is in a cycle of increase-decrease-stable for every 3 year between 2013-2018.

08_Import_Amount_by_Sectors_And_Year

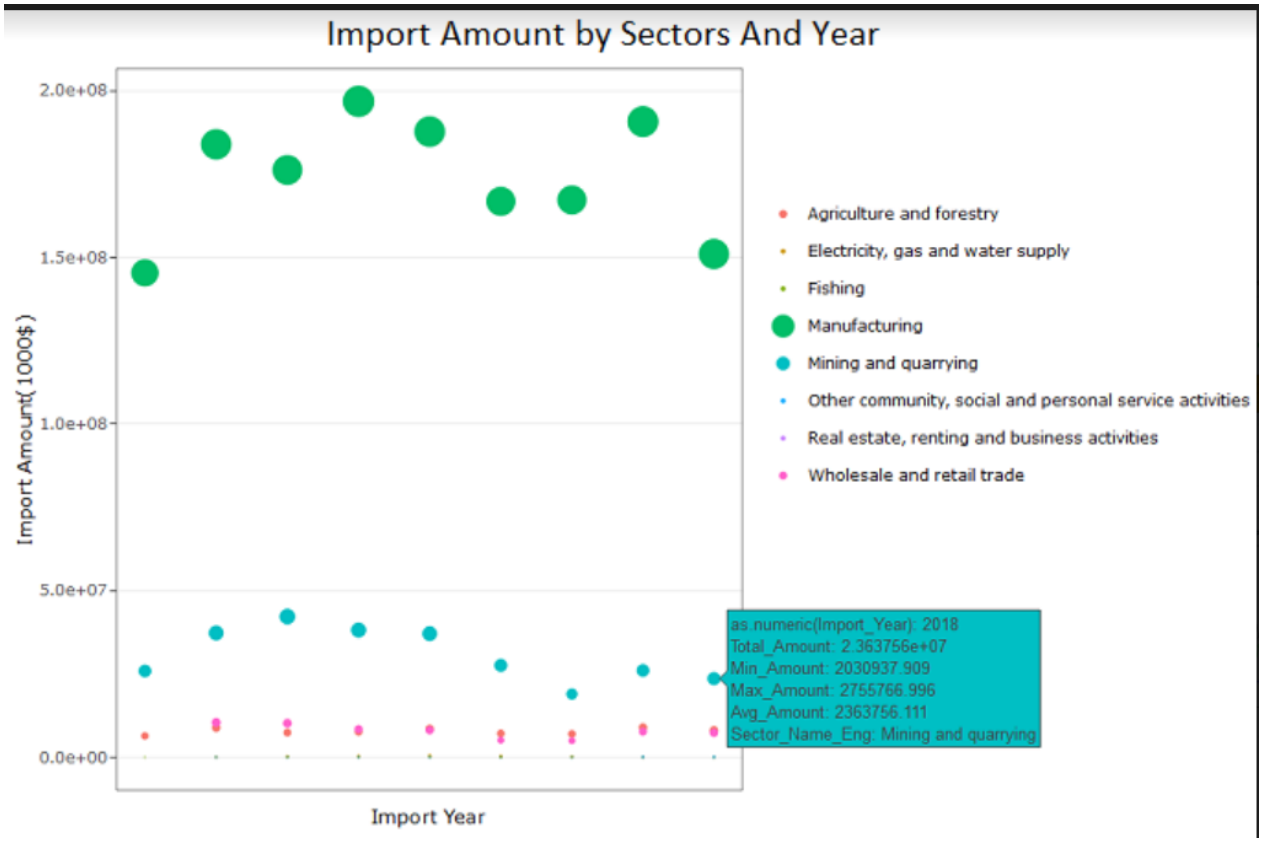
```
imp_agg_by_sectors<- imp_data %>%
  filter(Main_Sector_Flag==1)%>%
  group_by(Import_Year,Sector_Name_Eng) %>%
  summarise_each(funs(min(.,na.rm=TRUE), round(mean(.,na.rm=TRUE),digits = 3),
max(.,na.rm=TRUE),sum(.,na.rm=TRUE)),Import_Amount)
```

```
## `summarise_each()` is deprecated.
## Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
## To map `funs` over a selection of variables, use `summarise_at()`
```

```
colnames(imp_agg_by_sectors)<- c("Import_Year", "Sector_Name_Eng", "Min_Amount
", "Avg_Amount", "Max_Amount", "Total_Amount" )


p <- imp_agg_by_sectors %>%
```

```
    ggplot(aes(x= as.numeric(Import_Year), y= Total_Amount, group=Min_Amount,
group2 = Max_Amount, size = Avg_Amount, color=Sector_Name_Eng)) +
    geom_point() +
    scale_x_log10() +
    theme_bw()+
    scale_size_area("Nitrogen") +
    xlab("Import Year") +
    ylab("Import Amount(1000$)") +
    ggtitle("Import Amount by Sectors and Years")


  ggplotly(p)
```

0.0e+005.0e+071.0e+081.5e+082.0e+08
Agriculture and forestryElectricity, gas and water supplyFishingManufacturingMining and quarryingOther community, social and personal service activitiesReal estate, renting and business activitiesWholesale and retail tradeImport Amount by Sectors and YearsImport YearImport Amount(1000$)NitrogenSector_Name_Eng



The graph shows import amounts by sectors and years. Manufacturing is the main port of our import. Mining amounts gives us a chance to compare manufacturing amounts and mining amount trends. We

can see that there is almost a synchronization between manufacturing and mining trends which may mean overall factors affects these two sectors in the same way and with a nearly equal rate.

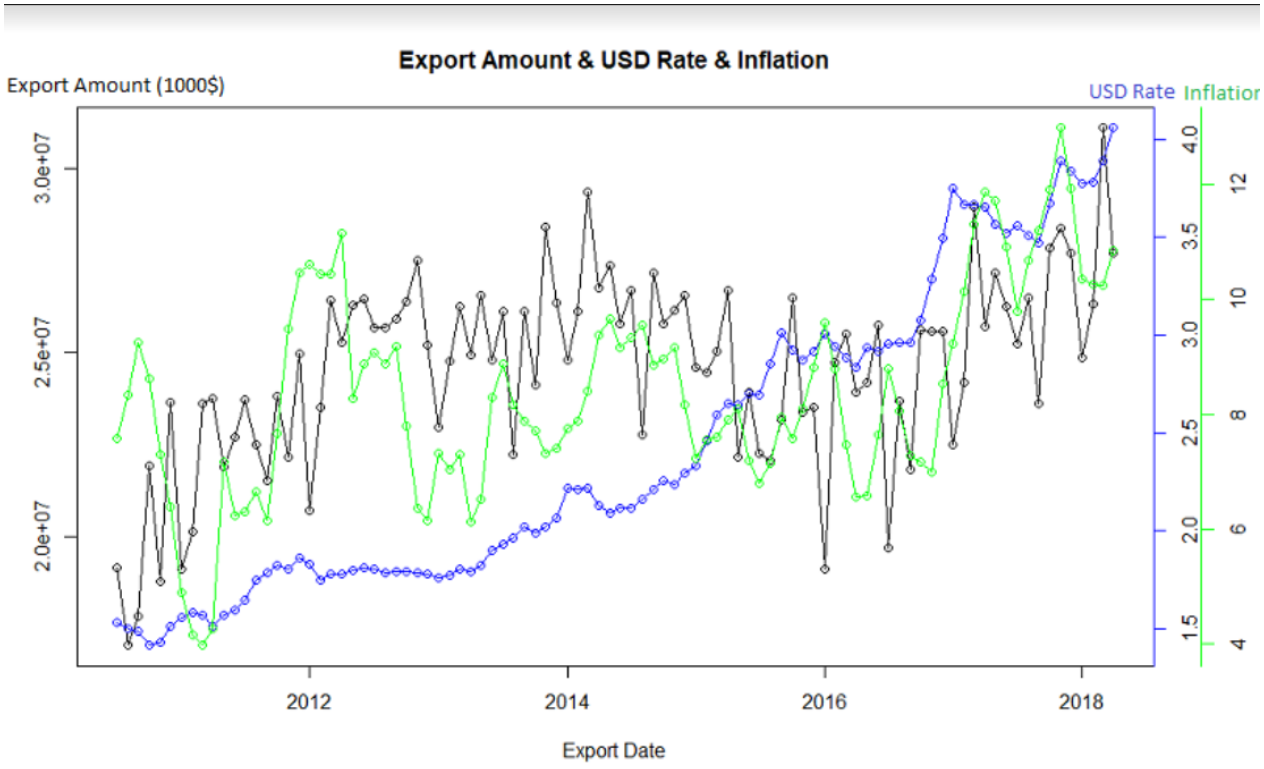
09_Export_Amount_USD_Rate_Inflation

```
colors = c("red", "blue", "green")


# Set the margins of the plot wider
par(oma = c(0, 2, 2, 3))


plot(exp_data_final$Export_Date, exp_data_final$Export_Total_Amount, yaxt = "n
", xlab = "Export Date", main = "Export Amount & USD Rate & Inflation",
     ylab = "")
lines(exp_data_final$Export_Date, exp_data_final$Export_Total_Amount)


# We use the "pretty" function go generate nice axes
axis(at = pretty(exp_data_final$Export_Total_Amount), side = 2)
library("tidyverse")
exp_data_final <- exp_data_final %>%
  select(Export_Date,Export_Total_Amount,USD_Rate,Consumer_Price_Index_Yearly_
Change
         ,Consumer_Price_Index_Monthly_Change,Export_Year,Export_Year_Month)


str(exp_data_final)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    94 obs. of  7 variables:
##  $ Export_Date                       : Date, format: "2010-07-01" "2010-08
-01" ...
##  $ Export_Total_Amount               : num  19129365 17046904 17818461 219
27173 18764739 ...
##  $ USD_Rate                          : num  1.53 1.5 1.49 1.42 1.43 ...
##  $ Consumer_Price_Index_Yearly_Change : num  7.58 8.33 9.24 8.62 7.29 6.4 4
.9 4.16 3.99 4.26 ...
##  $ Consumer_Price_Index_Monthly_Change: num  -0.48 0.4 1.23 1.83 0.03 -0.3
0.41 0.73 0.42 0.87 ...
##  $ Export_Year                       : num  2010 2010 2010 2010 2010 ...
##  $ Export_Year_Month                 : chr  "2010-07" "2010-08" "2010-09"
"2010-10" ...
```

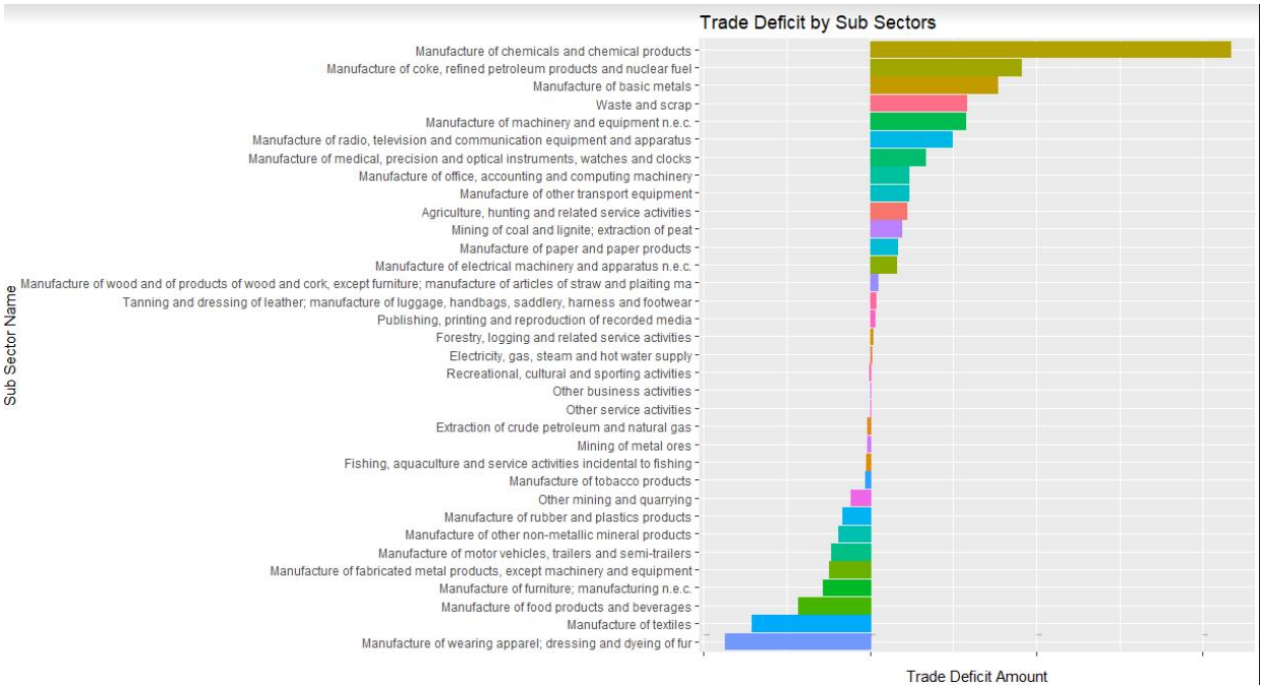
```
# The side for the axes.  The next one will go on
```

```
# the left, the following two on the right side
sides <- list(2, 4, 4)


# The number of "lines" into the margin the axes will be
lines <- list(2, NA, 2)



for(i in 3:5) {
  par(new = TRUE)
  plot(exp_data_final$Export_Date, exp_data_final[[i]], axes = FALSE, col = co
lors[i - 1], xlab = "", ylab = "")
  axis(at = pretty(exp_data_final[[i]]), side = sides[[i-2]], line = lines[[i-
2]],
       col = colors[i - 1])
  #mtext(2,text=colnames(exp_data_final)[i],line=2)
  lines(exp_data_final$Export_Date, exp_data_final[[i]], col = colors[i - 1])
}
```



The graph shows export amounts, USD rate and inflation. First thing that take attention is USD rate is increasing slowly or fast but it is always increasing however export amount and inflation is fluctuated by years. Export amount and inflation is nearly synchronized. If we look at 2014-2016 USD rate is increasing and export amount is decreased significantly however in 2016-2018 USD rate is increasing again and export amount is increased significantly which means that only USD rate is not enough to explain direction and fluctuations in export amounts.

````
10_Trade_Deficit_Amount_by_Sub_Sectors
```{r, warning=FALSE, echo=FALSE}
   emp_by_sectors <- exp_data %>%
   filter(Main_Sector_Flag == 0) %>%
   group_by(Sector_Name_Eng,Sub_Sector_Type_Code) %>%
   summarize(Total_Export_Amount = sum(Export_Amount))


   imp_by_sectors <- imp_data %>%
   filter(Main_Sector_Flag == 0) %>%
   group_by(Sector_Name_Eng,Sub_Sector_Type_Code) %>%
   summarize(Total_Import_Amount = sum(Import_Amount))


 trade_deficit_sub_sector <- inner_join(imp_by_sectors, emp_by_sectors, by =c(
"Sub_Sector_Type_Code", "Sub_Sector_Type_Code")) %>%
   mutate(Trade_Deficit_Amount = Total_Import_Amount - Total_Export_Amount)%>%
   arrange(desc(Trade_Deficit_Amount))


trade_deficit_sub_sector %>%
  ggplot(aes(x=reorder(Sector_Name_Eng.x, Trade_Deficit_Amount),
             y=Trade_Deficit_Amount, color =Sector_Name_Eng.x, fill=Sector_Nam
e_Eng.x)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(x = "Sub Sector Name", y = "Trade Deficit Amount") +
  ggtitle("Trade Deficit by Sub Sectors") +
  theme(legend.position = "none", axis.text.x = element_text(angle = 0, vjust
= 10.0, hjust = 0.0, size = 1))
```
````

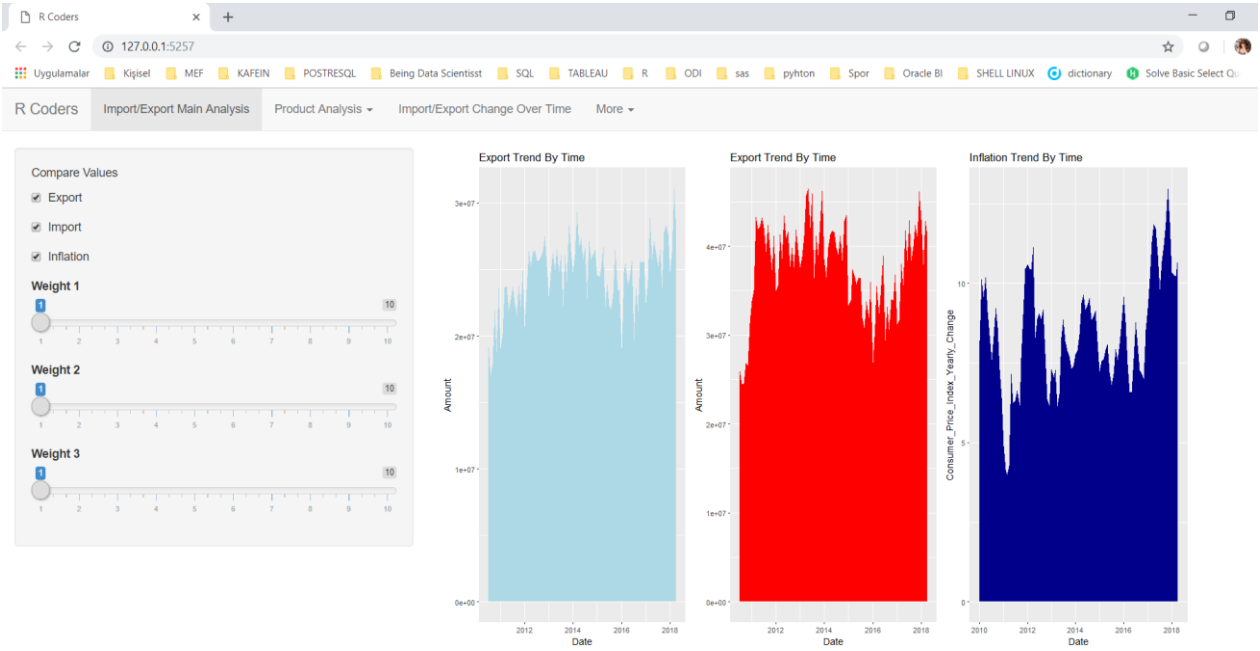


- When export sub-sectors are investigated for last 3 years, we saw that top 3 export subsectors are manufacture of motor vehicles and trailers, manufacture of basic metals and manufacture of textiles.
- When import subsectors are investigated for last 3 years, we saw that top 3 import subsectors are manufacture of basic metals, manufacture of chemicals and chemical products, mining and quarrying.

SHINY APP

The detail version of shiny app can be seen from this link: https://mef-bda503.github.io/gpj18-r_coders/

And also screenshots and analysis of app is like below.



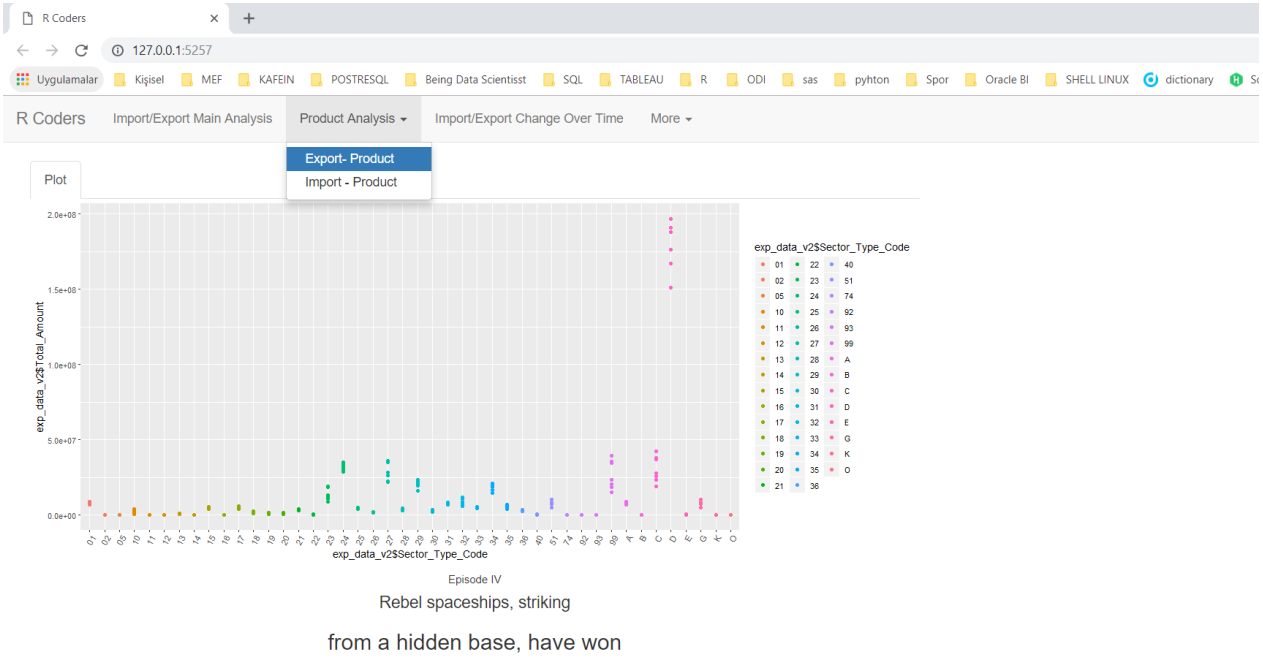
A country's current account falls into a deficit when imports of goods and services are larger than exports, so given the tendencies outlined in the above chart, it's of little surprise that the current account shifts to a surplus soon after major currency devaluation. Imports have suddenly got a lot more expensive. The plunge in Turkey's lira and subsequent reversal of last year's credit boom has had a similar effect of grinding domestic economic activity. Because of all these effects inflation rate is simultaneously increasing.

A country's current account falls into a deficit when imports of goods and services are larger than exports, so given the tendencies outlined in the above chart, it's of little surprise that the current account shifts to a surplus soon after major currency devaluation. Imports have suddenly got a lot more expensive. The plunge in Turkey's lira and subsequent reversal of last year's credit boom has had a similar effect of grinding domestic economic activity. Because of all these effects inflation rate is simultaneously increasing.

This dynamic app is giving the ability to choose which analysis to be shown to the user.

And also the weight of graphs can be determined by the user.

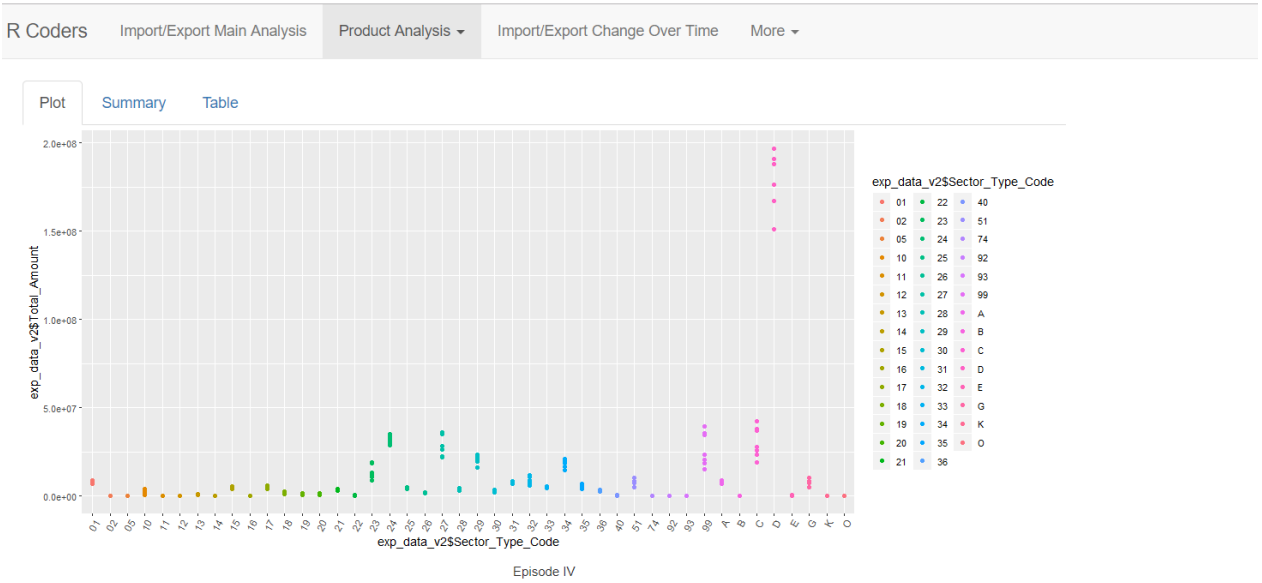
Shiny also gives the opportunity of arrange many different custom manupulations on diagrams like color, size, being smooth. This abilities creates a difference against other visualization tools.

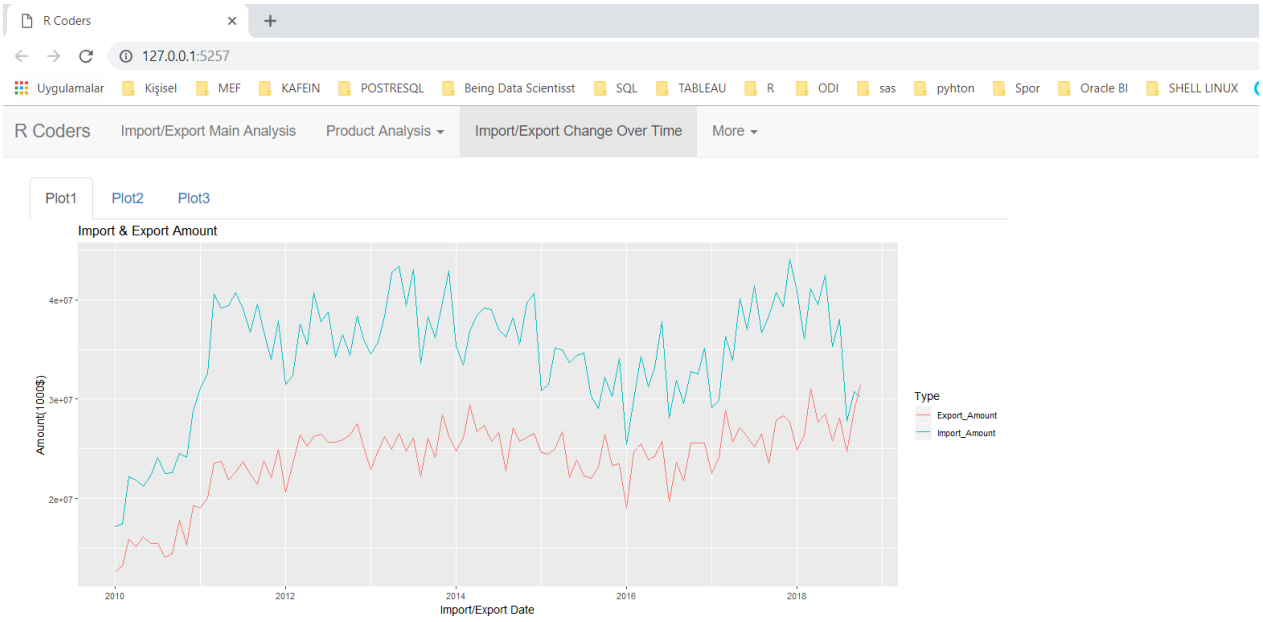


On the Shiny app, dashboard view(As R CODERS we have stole this idea from TABLEAU dashboard structure)

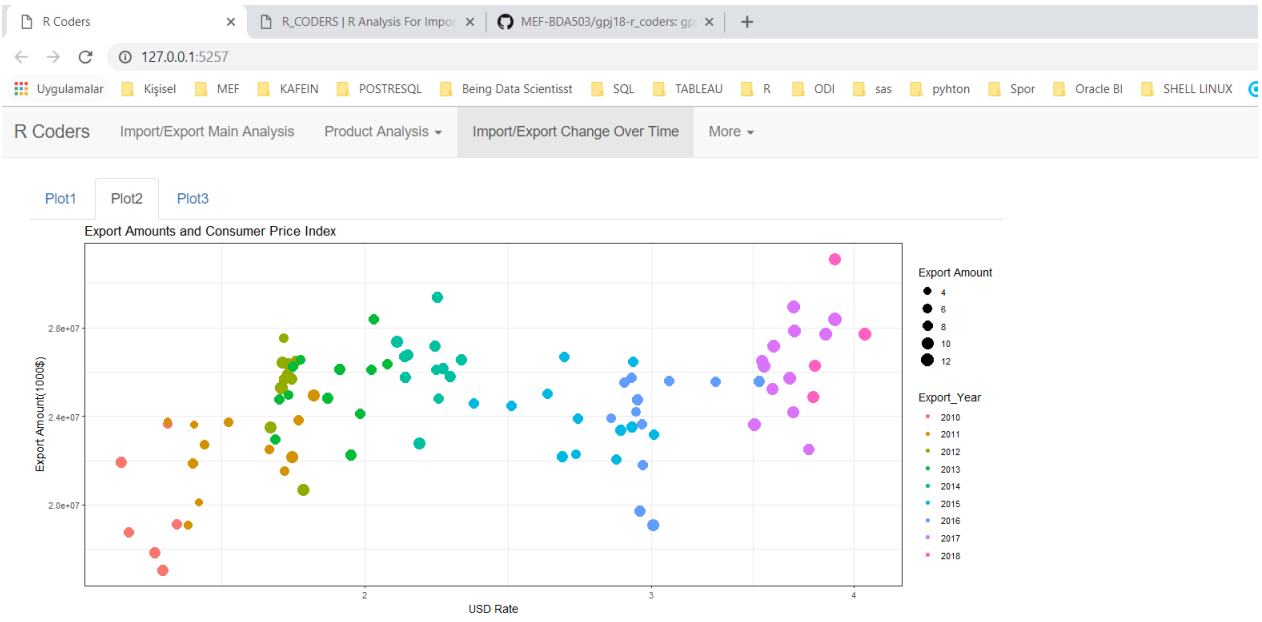
From the perfective of Tableau, R_CODER's second worksheet is the Product analysis tab. The user or analyzer can choose either Export products or import products.

On this diagram product range try to be analyzed.

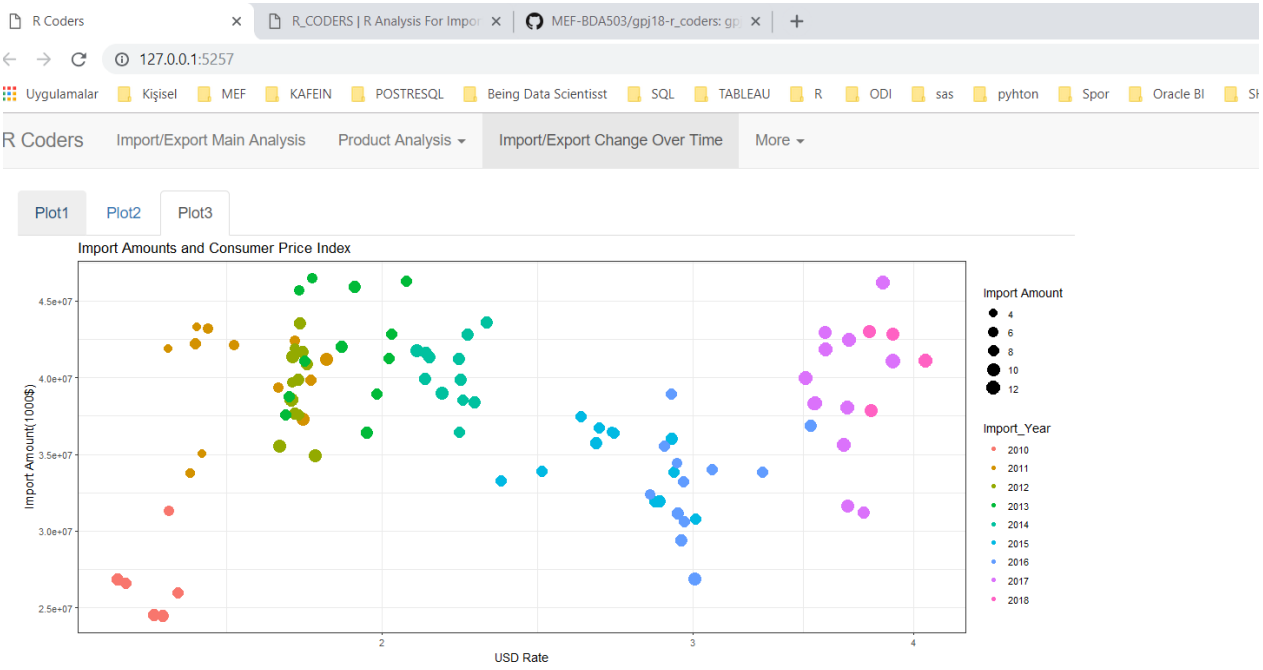




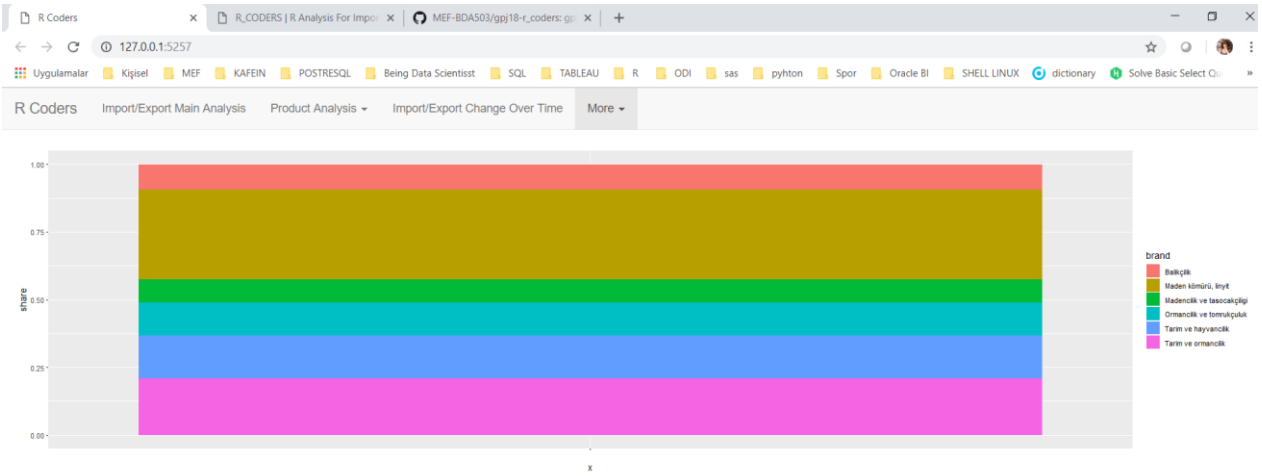
In this diagram Turkey's foreign trade deficit try to be dedicated. So the conclusion as we saw is that while deficit is decreasing, economic indicators of Turkey is decreasing also such as inflation goes up.

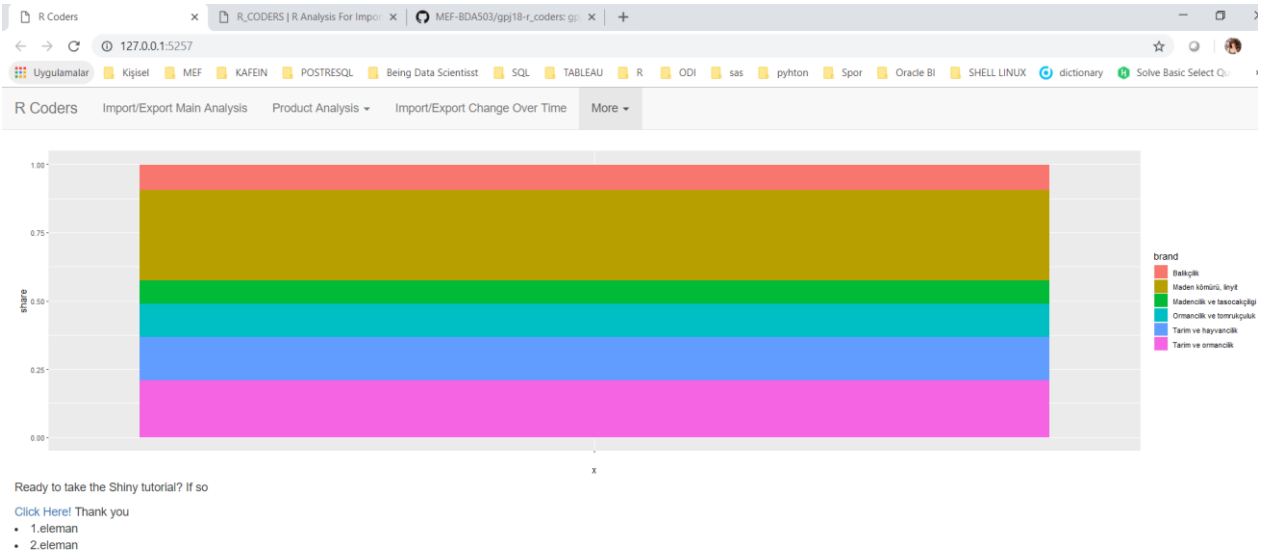


The initial impact crops up in the import data as consumers' buying power contracts, bringing economic output down with it. Following the depreciation at time t, imports fall sharply and export volumes increase as local goods become more competitively priced than those denominated in stronger currencies



These analysis above Show that the scale of products amount in USD. By the time passes the import amount of Turkish economy is increasing.





Above graps are product groups percentage in import and export total amount. These analysis indicate that , As an import product energy has a great percentage

Here is R_CODER's shiny app code:

```
library(shiny)

library(readxl)

library(tidyverse)
```

```
library(ggplot2)

library(dplyr)

library(stringr)

library(rsconnect)

library(plotly)

library(gapminder)

library(gridExtra)


## Import Analysis


tmp<-tempfile(fileext=".xls")


download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Excel/import_1996_2018.xls?raw=true",mode = 'wb',destfile=tmp)


import_data<-readxl::read_excel(tmp,skip=7,col_names=FALSE)


file.remove(tmp)


#Define Colnames

colnames(import_data) <- c("Year","Sector_Type_Code","Sector_Name",          "Total_Amount",
          "January",        "February",       "March",          "April", "May",  "June", "July","August",
          "September",    "October"         ,"November","December")


cols = c(4:15);

import_data[,cols] = suppressWarnings(apply(import_data[,cols], 2, function(x)
as.numeric(as.character(x))));

str(import_data)


print(import_data %>% select(Sector_Name,January,February,March)) %>% mutate(VATotal =
import_data$January + import_data$February + import_data$March) %>% filter(VATotal > 3000000)
```

```
## Print No Import Sectors

import_data %>% select(Sector_Name) %>% mutate(VADiff = import_data$January +
import_data$February + import_data$March ) %>% filter(is.na(VADiff)) %>% distinct()


print(import_data %>% select(Sector_Name) %>% mutate(VADiff = import_data$January +
import_data$February + import_data$March ) %>% filter(is.na(VADiff)) %>%
filter(!(is.na(Sector_Name))) %>% distinct())




tmp<-tempfile(fileext=".xls")


download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Excel/import_1996_2018.xls?raw=true",mode = 'wb',destfile=tmp)


raw_data<-readxl::read_excel(tmp,skip=7,col_names=FALSE)


file.remove(tmp)


colnames(raw_data) <- c("Year","Sector_Type_Code","Sector_Name",    "Total_Amount",
          "January",        "February",       "March",          "April", "May",  "June",  "July","August",
          "September",    "October"         ,"November","December")


cols = c(4:15);

raw_data[,cols] = suppressWarnings(apply(raw_data[,cols], 2, function(x) as.numeric(as.character(x))));


raw_data %>% select(Sector_Name) %>% mutate(VADiff = raw_data$January + raw_data$February)


for (row in 1:nrow(raw_data)) {
```

```
  year <- raw_data[row, "Year"]
  if(!is.na(year) & year == 2017){
    break
  }
  raw_data[row, "Year"] <- 2018
}

v_year <- 2017
for (row in 1:nrow(raw_data)) {
  year <- raw_data[row, "Year"]
  if(!is.na(year) & year == v_year){
    v_year <- v_year - 1
  }
  raw_data[row, "Year"] <- v_year + 1
  if (v_year==2008){
    break
  }
}

exp_data_v2 <- raw_data %>%
  slice(6:391)%>% filter(Sector_Name != "Toplam -Total")

Months <-
c("January","February","March","April","May","June","July","August","September","October","Novembe
r","December")

Values <- c(1000,1200,1100,1600,1800,1000,1200,1300,2000,1300,1200,1100)

Randoms <- c(1020,1300,1130,1500,1080,2000,2200,1350,2500,1350,1220,1101)
```

Import/ Export Union Part

```
tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/imp_data_final.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

imp_data_final <- read_rds(tmp)

file.remove(tmp)


tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/exp_data_final.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

exp_data_final<-read_rds(tmp)

file.remove(tmp)


tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/imp_data.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

imp_data<-read_rds(tmp)

file.remove(tmp)


tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/exp_data.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

exp_data<-read_rds(tmp)

file.remove(tmp)


tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/Producer_Inflation.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')
```

```
producer_inf<-read_rds(tmp)

file.remove(tmp)


# Create a temporary file

tmp=tempfile(fileext=".xls")

# Download file from repository to the temp file

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Excel/export_import_sectors.xls?raw=true",destfile=tmp,mode='wb')

# Read that excel file.

sectors <- read_excel(tmp)

# Remove the temp file

file.remove(tmp)


tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/US_Dollar_Montly_Rate.rds?raw=true?raw=true",destfile=tmp,mode = 'wb')

usd_rate<-read_rds(tmp)

file.remove(tmp)


names(exp_data_final)[names(exp_data_final) == 'Date'] <- 'Export_Date'

names(exp_data)[names(exp_data) == 'Date'] <- 'Export_Date'

names(imp_data_final)[names(imp_data_final) == 'Date'] <- 'Import_Date'

names(imp_data_final)[names(imp_data_final) == 'Export_Total_Amount'] <- 'Import_Total_Amount' #fix

names(imp_data)[names(imp_data) == 'Date'] <- 'Import_Date'


exp_data <- inner_join(exp_data,sectors, by=c("Sector_Type_Code"="Sub_Sector_Type_Code"))


imp_data <- inner_join(imp_data,sectors, by=c("Sector_Type_Code"="Sub_Sector_Type_Code"))
```

```
exp_data$Export_Year<-format(exp_data$Export_Date,"%Y")

exp_data$Export_Year_Month<-format(exp_data$Export_Date,"%Y-%m")

exp_data_final$Export_Year<-format(exp_data_final$Export_Date,"%Y")

exp_data_final$Export_Year_Month<-format(exp_data_final$Export_Date,"%Y-%m")


imp_data$Import_Year<-format(imp_data$Import_Date,"%Y")

imp_data$Import_Year_Month<-format(imp_data$Import_Date,"%Y-%m")

imp_data_final$Import_Year<-format(imp_data_final$Import_Date,"%Y")

imp_data_final$Import_Year_Month<-format(imp_data_final$Import_Date,"%Y-%m")


imp_data<- imp_data %>%

  select
(Import_Date,Sector_Type_Code,Sector_Type_Code.y,Main_Sector_Flag,Sector_Name_Eng,Amount,Imp
ort_Year,Import_Year_Month)


exp_data<- exp_data %>%

  select
(Export_Date,Sector_Type_Code,Sector_Type_Code.y,Main_Sector_Flag,Sector_Name_Eng,Amount,Exp
ort_Year,Export_Year_Month)



colnames(imp_data)[colnames(imp_data) == 'Amount'] <- 'Import_Amount'

colnames(exp_data)[colnames(exp_data) == 'Amount'] <- 'Export_Amount'

colnames(imp_data)[colnames(imp_data) == 'Sector_Type_Code'] <- 'Sub_Sector_Type_Code'

colnames(exp_data)[colnames(exp_data) == 'Sector_Type_Code'] <- 'Sub_Sector_Type_Code'

colnames(imp_data)[colnames(imp_data) == 'Sector_Type_Code.y'] <- 'Sector_Type_Code'

colnames(exp_data)[colnames(exp_data) == 'Sector_Type_Code.y'] <- 'Sector_Type_Code'

imp_data$Import_Amount[is.na(imp_data$Import_Amount)] <- 0

imp_data_final$Import_Total_Amount[is.na(imp_data_final$Import_Total_Amount)] <- 0
```

```
exp_data$Export_Amount[is.na(exp_data$Export_Amount)] <- 0

exp_data_final$Export_Total_Amount[is.na(exp_data_final$Export_Total_Amount)] <- 0


exp_data_final <- exp_data_final %>%

 filter(Export_Date<'2018-11-01')


exp_data <- exp_data %>%

 filter(Export_Date<'2018-11-01')


imp_data_final <- imp_data_final %>%

 filter(Import_Date<'2018-11-01')


imp_data <- imp_data %>%

 filter(Import_Date<'2018-11-01')


saveRDS(imp_data,file="imp_data_v2.rds")

saveRDS(imp_data_final,file="imp_data_final_v2.rds")

saveRDS(exp_data,file="exp_data_v2.rds")

saveRDS(exp_data_final,file="exp_data_final_v2.rds")


imp_and_exp_data <- inner_join(exp_data, imp_data, by=c("Export_Date" =
"Import_Date","Sub_Sector_Type_Code"="Sub_Sector_Type_Code"))


imp_and_exp_data_bymonth <- aggregate(cbind(Import_Amount, Export_Amount) ~ Export_Date, data
= imp_and_exp_data, sum)

imp_and_exp_data_bymonth <- gather(imp_and_exp_data_bymonth,

                    value = "value",

                    key = "type",

                    Export_Amount, Import_Amount)
```

```
exp_data_final_2 <- exp_data_final

imd_data_final_2 <- imp_data_final

#Rename column names

colnames(imp_and_exp_data_bymonth) <- c("Date","Type","Amount")


#Remove Empty Dates

imp_and_exp_data_bymonth <- imp_and_exp_data_bymonth %>%

  filter(Date<'2018-11-01')


df = data.frame("brand" = c("Tarım ve ormancılık","Tarım ve hayvancılık","Ormancılık ve
tomrukçuluk","Balıkçılık","Madencilik ve taşocakçılığı","Maden kömürü, linyit"),

            "share" = c(.2090,.1580,.1210,.0930,.0860,.3320))


##--------------##

#Download rds files

#get import data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Rds/imp_data_final.rds?raw=true",destfile=tmp,mode = 'wb')

imp_data_final<-read_rds(tmp)

file.remove(tmp)


imp_data_final


#get export data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Rds/exp_data_final.rds?raw=true",destfile=tmp,mode = 'wb')

exp_data_final<-read_rds(tmp)

file.remove(tmp)
```

```
exp_data_final


#Get export data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Rds/exp_data.rds?raw=true",destfile=tmp,mode = 'wb')

exp_data<-read_rds(tmp)

file.remove(tmp)


#Get inflation data

#Download rds files

#get import data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Rds/imp_data_final.rds?raw=true",destfile=tmp,mode = 'wb')

imp_data_final<-read_rds(tmp)

file.remove(tmp)


imp_data_final


#get export data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-
r_coders/blob/master/Data_Sources_Rds/exp_data_final.rds?raw=true",destfile=tmp,mode = 'wb')

exp_data_final<-read_rds(tmp)

file.remove(tmp)


exp_data_final
```

```
#get export data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/exp_data.rds?raw=true",destfile=tmp,mode = 'wb')

exp_data<-read_rds(tmp)

file.remove(tmp)


#get inflation data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/Consumer_Inflation.rds?raw=true",destfile=tmp,mode = 'wb')

Inflation_data<-read_rds(tmp)

file.remove(tmp)


#Get $ data

tmp<-tempfile(fileext=".rds")

download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Rds/US_Dollar_Montly_Rate.rds?raw=true",destfile=tmp,mode = 'wb')

US_Dollar_data<-read_rds(tmp)

file.remove(tmp)




#US_Dollar_Montly_Rate

#Download Raw Data

# Create a temporary file

tmp<-tempfile(fileext=".xlsx")

# Download file from repository to the temp file
```

```
download.file("https://github.com/MEF-BDA503/gpj18-r_coders/blob/master/Data_Sources_Excel/US_Dollar_Montly_Rate.xlsx?raw=true",mode="wb",destfile=tmp)

# Read that excel file using readxl package's read_excel function. You might need to adjust the parameters (skip, col_names) according to your raw file's format.

raw_data<-readxl::read_excel(tmp,skip=7,col_names=FALSE)

# Remove the temp file

file.remove(tmp)


colnames(raw_data) <- c("Date","Dollar")

US_Dollar_Montly_Rate<- raw_data


saveRDS(US_Dollar_Montly_Rate, file = "US_Dollar_Montly_Rate.rds")



US_Dollar_Montly_Rate



colnames(imp_data)[which(colnames(imp_data) %in% c("Date") )] <- c("Import_Date")

colnames(exp_data)[which(colnames(exp_data) %in% c("Date") )] <- c("Export_Date")


(imp_data)

head(exp_data)

Inflation_data

US_Dollar_data

imp_data_final

exp_data_final

Export_Import_union_sektor_data


#a nes column type
```

```
imp_data_final<- mutate(imp_data_final,Type="Import")

exp_data_final<- mutate(exp_data_final,Type="Export")

Export_Import_union_data <- rbind.fill(imp_data_final,exp_data_final)

print.data.frame(Export_Import_union_sektor_data)

#change column name as amount

Export_Total_Amount

names(Export_Import_union_data)[names(Export_Import_union_data) == "Export_Total_Amount"] <-
"Total_Amount"


names(imp_data_final)[names(imp_data_final) == "Export_Total_Amount"] <- "Total_Amount"


names(exp_data_final)[names(exp_data_final) == "Export_Total_Amount"] <- "Total_Amount"



names(imp_data_final)[names(imp_data_final) == "Date"] <- "datadate"


names(exp_data_final)[names(exp_data_final) == "Date"] <- "datadate"


names(Inflation_data)[names(Inflation_data) == "Consumer_Price_Index_Montly_Change_%"] <-
"Consumer_Price_Index"


names(Inflation_data)[names(Inflation_data) == "Consumer_Price_Index_Yearly_Change_%"] <-
"Consumer_Price_Index_Yearly_Change"


Export_Import_union_data


imp_data_final

exp_data_final

str(Inflation_data)

Inflation_data
```

```
## UI Part ##


ui <- navbarPage("R Coders",
            tabPanel("Import/Export Main Analysis",
                 sidebarLayout(position = "left",
                          sidebarPanel("Compare Values",
                                   checkboxInput("donum1", "Export", value = T),
                                   checkboxInput("donum2", "Import", value = F),
                                   checkboxInput("donum3", "Inflatıon", value = F),
                                   sliderInput("wt1","Weight 1",min=1,max=10,value=1),
                                   sliderInput("wt2","Weight 2",min=1,max=10,value=1),
                                   sliderInput("wt3","Weight 3",min=1,max=10,value=1)
                          ),
                          mainPanel((plotOutput(outputId="plotgraph", width="900",height="600px")),
                                h4("A country's current account falls into a deficit when imports of goods and services are larger than exports, so given the tendencies outlined in the above chart, it's of little surprise that the current account shifts to a surplus soon after major currency devaluation. Imports have suddenly got a lot more expensive.
                                   The plunge in Turkey's lira and subsequent reversal of last year's credit boom has had a similar effect of grinding domestic economic activity.
                                   Because of all these effects inflation rate is simultaneously increasing."
                                )))
                                ),
            navbarMenu("Product Analysis",tabPanel("Export- Product",mainPanel(
             tabsetPanel(
               tabPanel("Plot", plotOutput("distPlot"),h6("Episode IV", align = "center"),
                    h4("Rebel spaceships, striking", align = "center"),
                    h3("from a hidden base, have won", align = "center"))
             )
```

```
)),
tabPanel("Import - Product",mainPanel(
 tabsetPanel(
  tabPanel("Plot", plotOutput("distPlot_1"),h6("Episode IV", align = "center"),
      h4("Rebel spaceships, striking", align = "center"),
      h3("from a hidden base, have won", align = "center")),
  tabPanel("Summary",
verbatimTextOutput("selected_var"),verbatimTextOutput("summary")),
  tabPanel("Table", tableOutput("table"))
 )
))
),
tabPanel("Import/Export Change Over
Time",mainPanel(tabsetPanel(tabPanel("Plot1",plotOutput("importExportPlot")),
                            tabPanel("Plot2",plotOutput("ExpoloratoryPlot"),
                                h4("The initial impact crops up in the import data as
consumers' buying power contracts, bringing economic output down with it. Following the depreciation
at time t, imports fall sharply and export volumes increase as local goods become more competitively
priced than those denominated in stronger currencies",align = "center")),
                            tabPanel("Plot3",plotOutput("UsdRatePlot"))))),
navbarMenu("More",
      tabPanel("Export- Details",plotOutput("pieChart"),tags$div(class="header", checked=NA,
                                list(
                                 tags$p("Ready to take the Shiny tutorial? If so"),
                                 tags$a(href="shiny.rstudio.com/tutorial", "Click
Here!"),
                                 "Thank you",
                                 tags$li("1.eleman"),
                                 tags$li("2.eleman")
                                ))),
```

```
                    tabPanel("Import- Details",plotOutput("pieChart1"),tags$div(class="header",
checked=NA,
                                                                    list(
                                                                      tags$p("Ready to take the Shiny tutorial? If so"),
                                                                      tags$a(href="shiny.rstudio.com/tutorial", "Click
Here!"),
                                                                      "Thank you",
                                                                      tags$li("1.eleman"),
                                                                      tags$li("2.eleman")
                                                                    ))))
                                        )
## Server Part ##
server <- function(input, output) {


  output$distPlot <- renderPlot({
    ggplot(exp_data_v2,aes(x=exp_data_v2$Sector_Type_Code,y=exp_data_v2$Total_Amount,color =
exp_data_v2$Sector_Type_Code))+geom_point()+theme(axis.text.x = element_text(angle = 60, hjust =
1))
  })


  output$distPlot_1 <- renderPlot({
    ggplot(exp_data_v2,aes(x=exp_data_v2$Sector_Type_Code,y=exp_data_v2$Total_Amount,color =
exp_data_v2$Sector_Type_Code))+geom_point()+theme(axis.text.x = element_text(angle = 60, hjust =
1))
  })


  output$importExportPlot <- renderPlot({
    ggplot(imp_and_exp_data_bymonth,
           aes(x=Date,
               y=Amount,
               color=Type)) +
```

```
    geom_line()+
    scale_size_area("Export Amount") +
    xlab("Import/Export Date") +
    ylab("Amount(1000$)") +
    ggtitle("Import & Export Amount")
  })

  output$ExpoloratoryPlot <- renderPlot({
    ggplot(exp_data_final_2,aes(x=USD_Rate, y = Export_Total_Amount, size =
Consumer_Price_Index_Yearly_Change, color=Export_Year)) +
    geom_point() +
    scale_x_log10() +
    theme_bw()+
    scale_size_area("Export Amount") +
    xlab("USD Rate") +
    ylab("Export Amount(1000$)") +
    ggtitle("Export Amounts and Consumer Price Index")
  })

  output$pieChart <- renderPlot({
    ggplot(df, aes(x="", y=share, fill=brand)) + geom_bar(stat="identity", width=1)
  })

  output$pieChart1 <- renderPlot({
    ggplot(df, aes(x="", y=share, fill=brand)) + geom_bar(stat="identity", width=1)
  })
  ##pie = ggplot(df, aes(x="", y=share, fill=brand)) + geom_bar(stat="identity", width=1)

  output$UsdRatePlot <- renderPlot({
```

```
   ggplot(imd_data_final_2,aes(x = USD_Rate, y = Import_Total_Amount, size =
Consumer_Price_Index_Yearly_Change, color=Import_Year)) +
     geom_point() +
     scale_x_log10() +
     theme_bw()+
     scale_size_area("Import Amount") +
     xlab("USD Rate") +
     ylab("Import Amount(1000$)") +
     ggtitle("Import Amounts and Consumer Price Index")
  })

  output$selected_var <- renderText({
    paste("You have selected",input$Number)
  })

  output$table <- renderTable({
    head(import_data %>% select(Sector_Name,Sector_Type_Code), 100) %>% filter(!is.na(Sector_Name)
& Sector_Name != 'Toplam -Total')  %>% distinct()
  })

  output$summary <- renderPrint({
    dataset <- import_data %>% select(Sector_Name,Sector_Type_Code)
    summary(dataset)
  })

  output$table_import <- renderTable({
    head(import_data %>% select(Sector_Name) %>% mutate(VADiff = import_data$January +
import_data$February + import_data$March ) %>% filter(is.na(VADiff)) %>% distinct() %>%
filter(!(is.na(Sector_Name))), 10)
  })
```

```
  output$table_export <- renderTable({

    head(import_data %>% select(Sector_Name) %>% mutate(VADiff = import_data$January +
import_data$February) %>% filter(VADiff>1000000 & Sector_Name != 'Toplam -Total') %>%
distinct(Sector_Name), 10)

  })


  set.seed(600)


  pt1 <- reactive({

    if (!input$donum1) return(NULL)

    qplot(datadate, Total_Amount, data=exp_data_final,
geom="area",fill=I("lightblue"),binwidth=0.2,main="Export Trend By Time",xlab="Date", ylab='Amount')

  })


  pt2 <- reactive({

    if (!input$donum2) return(NULL)

    qplot(datadate, Total_Amount, data=imp_data_final,
geom="area",fill=I("red"),binwidth=0.2,main="Export Trend By Time",xlab="Date", ylab='Amount')

  })

  pt3 <- reactive({

    if (!input$donum3) return(NULL)

    qplot(Date, Consumer_Price_Index_Yearly_Change, data=Inflation_data,
geom="area",fill=I("darkblue"),binwidth=0.2,main="Inflation Trend By Time",xlab="Date",
ylab='Consumer_Price_Index_Yearly_Change')

  })


  output$plotgraph = renderPlot({

    ptlist <- list(pt1(),pt2(),pt3())

    wtlist <- c(input$wt1,input$wt2,input$wt3)

    # remove the null plots from ptlist and wtlist
```

```
    to_delete <- !sapply(ptlist,is.null)
    ptlist <- ptlist[to_delete]
    wtlist <- wtlist[to_delete]
    if (length(ptlist)==0) return(NULL)

    grid.arrange(grobs=ptlist,widths=wtlist,ncol=length(ptlist))
  })
}
# Create Shiny app ----
shinyApp(ui, server)
```