SectionA

EMATM0061,TB1 2022

YujieWang

2023-01-08

Contents

Introduction **3**

The solutions of Section A **3**

A.1 3

A.2 4

A.3 6

A.4 6

A.5 7

A.6 7

A.7 9

A.8 10

Introduction

This is the Section A, which includes eight parts to study the data wrangling with some finance data. The data wrangling is the process of transforming data from one form to another in preparation for another downstream task, it consists of 6 data wrangling operations-selecting, filtering, mutating, arranging, summarising, joining.The example is done with two important R packages including the `dplyr` package and the `tidyverse` package.

The solutions of Section A

To finish the experiment,the "tidyverse" package should be loaded. The "tidyverse" package is a collection of R packages that are designed for data science.The "dplyr" package is an R package designed for data wrangling.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
```

A.1

Start by downloading the "finance_data_2022.csv" file,this file contains data about the cumulative commitments funds from the International Finance Corporation (IFC) as well as Loan & Guarantee participations, across different IFC regions and countries.

Then, by using the function `read.csv()`, load the "finance_data_2022.csv" file into a R data frame called "data_original",the top ten columns of the data frame can be displayed below.It illustrated that the data frame has 8 columns which includes 'IFC.Region', 'Country', 'Loan...Guarantee.participations. Cumulative.Commitments..US..Thousands.', 'TOTAL', 'Fiscal.Year', 'As.of.Date'.

```
data_original<-read.csv("finance_data_2022.csv") # load a .csv file into R
data_original%>%head(10) # return the top ten rows of data frame
```

```
##                   IFC.Region                          Country
## 1  East Asia and the Pacific                         Cambodia
## 2  East Asia and the Pacific                            China
## 3  East Asia and the Pacific                             Fiji
## 4  East Asia and the Pacific                        Indonesia
## 5  East Asia and the Pacific                         Kiribati
## 6  East Asia and the Pacific                Korea, Republic of
## 7  East Asia and the Pacific Lao People's Democratic Republic
## 8  East Asia and the Pacific                         Malaysia
```

```
## 9  East Asia and the Pacific                          Mongolia
## 10 East Asia and the Pacific                           Myanmar
##    Number.of.enterprises IFC.Cumulative.Commitments..US..Thousands.
## 1                     13                                  316463.25
## 2                    264                                 8199672.61
## 3                     10                                   52493.22
## 4                    126                                 4068991.86
## 5                      1                                    1798.00
## 6                     51                                  868449.18
## 7                     13                                   66026.45
## 8                     12                                  154868.40
## 9                     18                                  335725.36
## 10                     4                                   87140.66
##    Loan...Guarantee.participations.Cumulative.Commitments..US..Thousands.
## 1                                                                  155000
## 2                                                                 1830109
## 3                                                                    2500
## 4                                                                 2512055
## 5                                                                       0
## 6                                                                  195700
## 7                                                                       0
## 8                                                                    5389
## 9                                                                   31000
## 10                                                                      0
##          TOTAL Fiscal.Year As.of.Date
## 1    471463.25        FY15 06/30/2015
## 2  10029781.90        FY15 06/30/2015
## 3     54993.22        FY15 06/30/2015
## 4   6581047.23        FY15 06/30/2015
## 5      1798.00        FY15 06/30/2015
## 6   1064149.18        FY15 06/30/2015
## 7     66026.45        FY15 06/30/2015
## 8    160257.52        FY15 06/30/2015
## 9    366725.36        FY15 06/30/2015
## 10    87140.66        FY15 06/30/2015
```

By using the function `nrow()` and `ncol()`, the number of rows and columns of this data frame can be computed,it shows that the data frame has 1580 rows and 8 columns.

```
data_original%>%nrow() # the number of rows
```

```
## [1] 1580
```

```
data_original%>%ncol() #the number of columns
```

```
## [1] 8
```

A.2

Before renaming the columns, the names of all columns could be shown by using the function `colnames()`.

```
colnames(data_original) # show the names of all columns
```

```
## [1] "IFC.Region"
## [2] "Country"
## [3] "Number.of.enterprises"
## [4] "IFC.Cumulative.Commitments..US..Thousands."
## [5] "Loan...Guarantee.participations.Cumulative.Commitments..US..Thousands."
## [6] "TOTAL"
## [7] "Fiscal.Year"
## [8] "As.of.Date"
```

Then, to generate a new data frame called "finance_data" with 5 columns by using the data frame "data_original", the function `select()` and the function `rename()` should be used.

The function `select()` could selecting a subset of columns and generating a new dataset. In this case, the function `select()` could select the corresponding 5 columns and generate a new data frame called "finance_data".

```
finance_data<-data_original%>%
  rename(IFC=IFC.Region,   #rename the required columns
         IFC_CC=IFC.Cumulative.Commitments..US..Thousands.,
         Loan_Guarantee_CC=Loan...Guarantee.participations.Cumulative.Commitments..US..Thousands.,
         Date=As.of.Date)%>%
  select(IFC,IFC_CC,Country,Loan_Guarantee_CC,Date)
  # select the required columns to generate a new data frame called finance_data
finance_data%>%head(10) # show the top ten rows of the new data frame
```

```
##                          IFC     IFC_CC                          Country
## 1  East Asia and the Pacific  316463.25                         Cambodia
## 2  East Asia and the Pacific 8199672.61                            China
## 3  East Asia and the Pacific   52493.22                             Fiji
## 4  East Asia and the Pacific 4068991.86                        Indonesia
## 5  East Asia and the Pacific    1798.00                         Kiribati
## 6  East Asia and the Pacific  868449.18                Korea, Republic of
## 7  East Asia and the Pacific   66026.45 Lao People's Democratic Republic
## 8  East Asia and the Pacific  154868.40                         Malaysia
## 9  East Asia and the Pacific  335725.36                         Mongolia
## 10 East Asia and the Pacific   87140.66                          Myanmar
##    Loan_Guarantee_CC       Date
## 1             155000 06/30/2015
## 2            1830109 06/30/2015
## 3               2500 06/30/2015
## 4            2512055 06/30/2015
## 5                  0 06/30/2015
## 6             195700 06/30/2015
## 7                  0 06/30/2015
## 8               5389 06/30/2015
## 9              31000 06/30/2015
## 10                 0 06/30/2015
```

A.3

In this part, firstly, the case creates a new data frame called "data_part1" by choosing a subset of the data frame "finance_data".Using the function `filter()`, the case chooses the rows satisfying that the values of 'IFC_CC' are no less than 300000 and the values of 'Loan_Guarantee_CC' are no more than 500000 and then sorts the rows that the values in the column 'IFC_CC' are in descending order by using the function `arrange()`.

Secondly, the case uses the function `head(4)` and the function `select()` to display a subset of the `data_part1`, which consists of the first 4 rows and the three columns 'IFC', 'IFC_CC', and 'Loan_Guarantee_CC'.

```
data_part1<-finance_data%>%  #data_part1 from a subset of finance_data
  filter(IFC_CC>=300000 & Loan_Guarantee_CC<=500000)%>%
  #choose the rows which IFC_CC>=300000 and Loan_Guarantee_CC <=500000
  arrange(desc(IFC_CC))
  #sort the rows that the values in the column IFC_CC are in descending order
data_part1%>%
  head(4)%>% # head(4) shows the first 4 rows of the data frame
  select(IFC,IFC_CC,Loan_Guarantee_CC) # 'select()' shows the required three columns.
```

```
##                   IFC   IFC_CC Loan_Guarantee_CC
## 1           Worldwide 13280154          330206.2
## 2           Worldwide 11399022          330206.0
## 3 Sub-Saharan Africa 10426234          477155.0
## 4 Sub-Saharan Africa  9863582          456155.0
```

A.4

In this part, the case uses the function `mutate()` to create a new column 'IFC_ratio' for the data frame "finance_data". For each row of the data frame, the element of the 'IFC_ratio' column is computed by $\alpha/(\alpha+\beta)$ where α denotes the element of the 'IFC_CC' column, and β denotes the element of the 'Loan_Guarantee_CC' column.

The code chunk below shows the number of columns for the new data frame by using the function `ncol()`. It illustrates that the new data frame "finance_data" has 6 columns.

```
finance_data<-finance_data%>%
  mutate(IFC_ratio=IFC_CC/(IFC_CC+Loan_Guarantee_CC)) # create a new column IFC_ratio
finance_data%>%ncol() # show the number of columns
```

```
## [1] 6
```

Finally, the case displays a subset of the data frame "finance_data" consisting of the first 5 rows and the 4 columns 'IFC', 'IFC_CC', and 'Loan _Guarantee_CC' and 'IFC_ratio' by using the function `select()` and `head()`.

```
finance_data%>%head(5)%>%
  # head(5) shows the first 4 rows of the data frame
  select(IFC,IFC_CC,Loan_Guarantee_CC,IFC_ratio)
```

```
##                          IFC     IFC_CC Loan_Guarantee_CC IFC_ratio
## 1 East Asia and the Pacific  316463.25             155000 0.6712363
## 2 East Asia and the Pacific 8199672.61            1830109 0.8175325
## 3 East Asia and the Pacific   52493.22               2500 0.9545399
## 4 East Asia and the Pacific 4068991.86            2512055 0.6182895
## 5 East Asia and the Pacific    1798.00                  0 1.0000000
```

```
  # the function 'select()' shows the required four columns
```

A.5

In this part, the month and year are separated by the forward slash character '/'. This case uses the function `separate()` separates the 'Date' column into three columns called 'day','month','year',respectively. To make sure each of the 'day','month','year' columns is of numeric type rather than characters, the function `separate()` set 'convert=True' to convert columns into numeric types.

Then, the case uses the function `head()` and `select()` to display a subset of the data frame "finance_data" consisting of the first 5 rows and the 4 columns 'IFC_CC','day','month', and 'year'.

```
# the 'separate()' function separate the 'Date' column into three columns
# "convert=TRUE" convert columns into numeric types
finance_data<-finance_data%>%
  separate(Date,into=c("day","month","year"),sep="/",convert=TRUE)
finance_data%>%
  head(5)%>% # head(5) shows the first 5 rows of the data frame
  select(IFC_CC,day,month,year) # 'select()' shows the required 4 columns.
```

```
##       IFC_CC day month year
## 1  316463.25   6    30 2015
## 2 8199672.61   6    30 2015
## 3   52493.22   6    30 2015
## 4 4068991.86   6    30 2015
## 5    1798.00   6    30 2015
```

A.6

Next generate a summary data frame called "summary_data" from the "finance_data".

Firstly, the summary data frame should have 7 rows corresponding to the different IFC regions specified in the IFC column of "finance_data" by using the function `group_by()`.

Secondly, the function `summarise()` is used to get the 7 columns:

1. 'IFC' - The IFC regions: "East Asia and the Pacific", "Europe and Central Asia", "Latin America and the Caribbean", "Middle East and North Africa", "South Asia", "Sub- Saharan Africa", "Worldwide".

   To get the the values of 'IFC', the case uses the function `group_by()`.

2. 'ifc_mn' - the mean of "IFC Cumulative Commitments (US$ Thousands)" for the corresponding IFC region.

   The case uses the function `mean()` to get the values of 'ifc_mn'.

3. 'ifc_21q' - the 0.21-quantile of "IFC Cumulative Commitments (US$ Thousands)" for the corresponding IFC region.

   The case uses the function `quantile()` to get the values of 'ifc_21q'.

4. 'ifc_var' - the variance of "IFC Cumulative Commitments (US$ Thousands)" for the corresponding IFC region.

   The case uses the function `var()` to get the values of 'ifc_var'.

5. 'lg_mn' - the mean of "Loan & Guarantee participations Cumulative Commitments (US$Thousands)" for the corresponding IFC region.

   The case uses the function `mean()` to get the values of 'lg_mn'.

6. 'lg_21q' - the 0.21-quantile of "Loan & Guarantee participations Cumulative Commitments (US$ Thousands)" for the corresponding IFC region.

   The case uses the function `quantile()` to get the values of 'lg_21q'.

7. 'lg_var' - the variance of "Loan & Guarantee participations Cumulative Commitments (US$ Thousands)" for the corresponding IFC region.

   The case uses the function `var()` to get the values of 'lg_var'.

Note that the missing values(NA) should be removed when computing the summary data. To remove the missing values, the function `mean()`, `quantile()`, `var()` should set `na.rm` as `TRUE`.

The data frame shown is the summary data frame.

```
summary_data<-finance_data%>%
  group_by(IFC)%>%
  # group the rows when summarize by the sepecies of IFC
  summarise(ifc_mn=mean(IFC_CC,na.rm=TRUE),
            #summarise the mean of IFC_CC and remove the NA values
            ifc_21q=quantile(IFC_CC,0.21,na.rm=TRUE),
            # get the 0.21-quantile value of IFC for corresponding IFC region
            ifc_var=var(IFC_CC,na.rm=TRUE), # get variance
            lg_mn=mean(Loan_Guarantee_CC,na.rm=TRUE), # get the mean values
            lg_21q=quantile(Loan_Guarantee_CC,0.21,na.rm=TRUE),
            # get the 0.21-quantile values
            lg_var=var(Loan_Guarantee_CC,na.rm = TRUE)) # get the variance values
summary_data
```

```
## # A tibble: 7 x 7
##   IFC                            ifc_mn ifc_21q ifc_var  lg_mn lg_21q  lg_var
##   <chr>                           <dbl>   <dbl>   <dbl>  <dbl>  <dbl>   <dbl>
## 1 East Asia and the Pacific      1481717.  19678. 6.32e12 4.18e5      0 5.71e11
## 2 Europe and Central Asia        1280212. 122080. 6.51e12 3.06e5      0 5.92e11
## 3 Latin America and the Caribbean 1791922.  76417. 1.06e13 7.06e5      0 2.48e12
## 4 Middle East and North Africa   1114380. 206004. 2.34e12 2.06e5      0 7.84e10
## 5 South Asia                     2854665. 168250  2.28e13 2.89e5      0 3.18e11
## 6 Sub-Saharan Africa              640224.  26152. 2.61e12 5.91e4      0 2.49e10
## 7 Worldwide                      1307271.   1233. 1.55e13 3.89e4      0 1.19e10
```

A.7

In this part, the case would create a plot to display the "IFC Cumulative Commitments"("IFC_CC") and "Loan & Guarantee participations Cumulative Commitments" as functions of the years, for two different countries Argentina and Brazil, respectively.

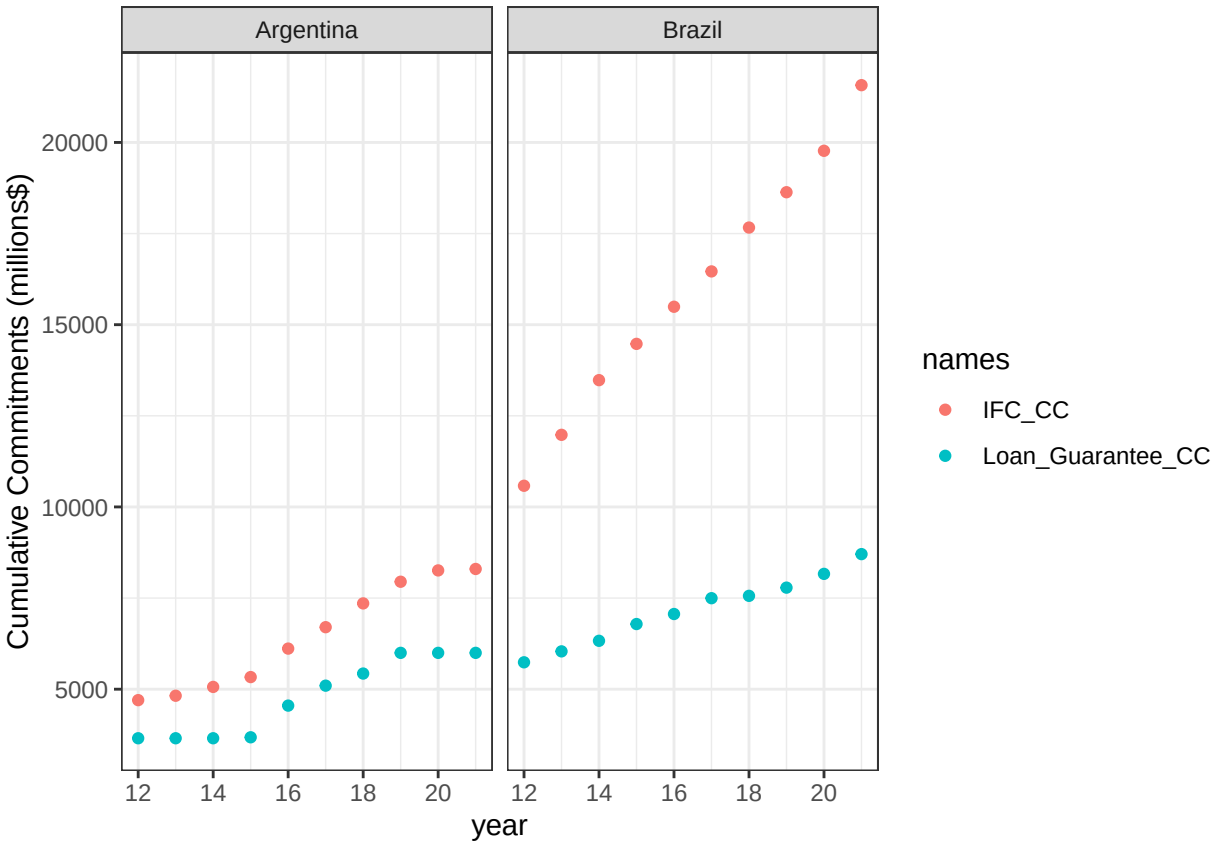
Firstly, the case creates a new data frame to get the subset of the "finance_data" by using the function `filter()`, `select()`, `mutate()` and `pivot_longer()`.

```
subset_finance<-finance_data%>%
  filter(Country=="Argentina"|Country=="Brazil")%>%
  # get the Argentina and Brazil country
  select(Country,year,IFC_CC,Loan_Guarantee_CC)%>%
  # get the required columns
  mutate(year=year-2000,IFC_CC=IFC_CC/1000,Loan_Guarantee_CC=Loan_Guarantee_CC/1000)%>%
  #create a new column to get the last two digits of the years.
  pivot_longer(cols=c(IFC_CC,Loan_Guarantee_CC),names_to="names",values_to="values")
  # narrow the data to create a plot
subset_finance%>%head(10)
```

```
## # A tibble: 10 x 4
##    Country    year names             values
##    <chr>     <dbl> <chr>              <dbl>
##  1 Argentina    15 IFC_CC              5332.
##  2 Argentina    15 Loan_Guarantee_CC   3680.
##  3 Brazil       15 IFC_CC             14473.
##  4 Brazil       15 Loan_Guarantee_CC   6787.
##  5 Argentina    14 IFC_CC              5062.
##  6 Argentina    14 Loan_Guarantee_CC   3654.
##  7 Brazil       14 IFC_CC             13479.
##  8 Brazil       14 Loan_Guarantee_CC   6327.
##  9 Argentina    13 IFC_CC              4820.
## 10 Argentina    13 Loan_Guarantee_CC   3654.
```

Finally, create the plot which has two panels.Using the function `facet_wrap()` to create two panels which are grouped by "Country".The Country includes "Argentina" and "Brazil".

```
ggplot(subset_finance,aes(x=year,y=values,color=names))+ #create the plot
  theme_bw()+
  xlab("year")+
  ylab("Cumulative Commitments (millions$)")+
  geom_point()+ # create the point for the plot
  facet_wrap(~Country) # create two panels grouped by Country
```



A.8

1. Firstly,the case creates a function called `impute_by_quantile()` which takes as input a vector numerical values, which may include some "NA"s, and replaces any missing values ("NA"s) with the 0.9-quantile of the vector.
2. Next, the case applies the function `impute_by_quantile()` to each of the columns 'IFC_CC', 'Loan_Guarantee_CC', and 'IFC_ratio' in your data frame "finance_data". This aims to replace the missing values (NA) with the 0.9 quantile of the corresponding column, within the data frame "finance_data".

```
# 1. create a function
impute_by_quantile<-function(sample){
  re<-quantile(sample,0,9,na.rm=TRUE) # compute the 0.9-quantile value.
  impute_na<-function(x){ # create a function to replace the NA with 0.9-quantile value
    if(is.na(x)){ # if the x is NA
      return(re) # return the 0.9-quantile value
    }else{
      return(x)
    }
  }
  return(map_dbl(.x=sample,.f=~impute_na(.x)))
}

#2. apply the function to data
finance_data<-finance_data%>%
```

```
  mutate(IFC_CC=impute_by_quantile(IFC_CC),
         Loan_Guarantee_CC=impute_by_quantile(Loan_Guarantee_CC),
         IFC_ratio=impute_by_quantile(IFC_ratio))
finance_data%>%head(10)
```

```
##                          IFC     IFC_CC                          Country
## 1  East Asia and the Pacific  316463.25                         Cambodia
## 2  East Asia and the Pacific 8199672.61                            China
## 3  East Asia and the Pacific   52493.22                             Fiji
## 4  East Asia and the Pacific 4068991.86                        Indonesia
## 5  East Asia and the Pacific    1798.00                         Kiribati
## 6  East Asia and the Pacific  868449.18                Korea, Republic of
## 7  East Asia and the Pacific   66026.45 Lao People's Democratic Republic
## 8  East Asia and the Pacific  154868.40                         Malaysia
## 9  East Asia and the Pacific  335725.36                         Mongolia
## 10 East Asia and the Pacific   87140.66                          Myanmar
##    Loan_Guarantee_CC day month year IFC_ratio
## 1             155000   6    30 2015 0.6712363
## 2            1830109   6    30 2015 0.8175325
## 3               2500   6    30 2015 0.9545399
## 4            2512055   6    30 2015 0.6182895
## 5                  0   6    30 2015 1.0000000
## 6             195700   6    30 2015 0.8160972
## 7                  0   6    30 2015 1.0000000
## 8               5389   6    30 2015 0.9663728
## 9              31000   6    30 2015 0.9154681
## 10                 0   6    30 2015 1.0000000
```

3. Next, using the function `summarise()` displays a data frame of three columns ('IFC_CC', 'Loan_Guarantee_CC', and 'IFC_ratio') and 1 row. The 'IFC_CC' column should contain a single number representing the mean of the 'IFC_CC' column of your data frame 'finance_data'. The 'Loan_Guarantee_CC' column should contain a single number representing the mean of the 'Loan_Guarantee_CC' column of the data frame "finance_data". The 'IFC_ratio' column should contain a single number representing the mean of the 'IFC_ratio' column of the data frame "finance_data".

```
# 3. display three columns
finance_data%>%
  summarise(IFC_CC=mean(IFC_CC), # get the mean of the 'IFC_CC' column
            Loan_Guarantee_CC=mean(Loan_Guarantee_CC),
            # get the mean of the 'Loan_Guarantee_CC' column
            IFC_ratio=mean(IFC_ratio)) #get the mean of the 'IFC_ratio' column
```

```
##    IFC_CC Loan_Guarantee_CC IFC_ratio
## 1 1288574          301330.2 0.8876765
```