MA615 Unit 1 Final Assignment

Rong Li

2020/10/19

1 Summary

In this assignment, we use the data set berries from USDA. We focus on one kind of berries–strawberries. Finish cleaning and organizing the data. Visualize and explore the data. This report can be divided into 2 parts: Data cleaning and EDA.

- Data cleaning: We select the data of strawberries from the raw data. Seperate some columns and eliminate the redundancy.
- EDA: We plot the boxplots for values of every year and of every states. Then we conduct the pca for values.

Besides, we deploy a shiny app Shiny_strawberries for data display.

2 Data cleaning

In this part, we obtain the data set strawberries that we will conduct EDA for later.

2.1 Deal with the raw data set berries

First, we explore the data set berries. Find out the redundent columns and delete them. Then display the data set.

```
## read the data
ag_data <- read_csv("/Users/amelia/Documents/mssp/MA615/Hw_berries/Berries_strawberries/data/berries.cs
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   Year = col_double(),
##   `Week Ending` = col_logical(),
##   `Ag District` = col_logical(),
##   `Ag District Code` = col_logical(),
##   County = col_logical(),
##   `County ANSI` = col_logical(),
##   `Zip Code` = col_logical(),
##   Region = col_logical(),
##   Watershed = col_logical(),
##   `CV (%)` = col_logical()
## )
```

```
## See spec(...) for full column specifications.
```

```
## look at number of unique values in each column
aa <- summarize_all(ag_data, n_distinct)
```

```
## make a list of the columns with only one unique value
bb <- which(aa[1,] == 1)

## list the 1-unique valu column names
colnames(ag_data)[bb]
```

```
##  [1] "Program"          "Week Ending"      "Geo Level"        "Ag District"
##  [5] "Ag District Code" "County"           "County ANSI"      "Zip Code"
##  [9] "Region"           "watershed_code"   "Watershed"        "CV (%)"
```

```
## remove the 1-unique columns from the dataset
ag_data <- select(ag_data, -all_of(bb))
aa <- select(aa, -all_of(bb))

## State name and the State ANSI code are (sort of) redundant
## Just keep the name
ag_data <- select(ag_data, -4)
aa <- select(aa, -4)

## Display the head of ag_data
head(ag_data)
```

```
## # A tibble: 6 x 8
##    Year Period  State  Commodity `Data Item`       Domain `Domain Categor~ Value
##   <dbl> <chr>   <chr>  <chr>     <chr>             <chr>  <chr>            <chr>
## 1  2019 MARKET~ CALIF~ BLUEBERR~ BLUEBERRIES, TAM~ TOTAL  NOT SPECIFIED    2.85
## 2  2019 MARKET~ CALIF~ BLUEBERR~ BLUEBERRIES, TAM~ TOTAL  NOT SPECIFIED    3.56
## 3  2019 MARKET~ CALIF~ BLUEBERR~ BLUEBERRIES, TAM~ TOTAL  NOT SPECIFIED    0.29
## 4  2019 MARKET~ CALIF~ RASPBERR~ RASPBERRIES - PR~ TOTAL  NOT SPECIFIED    2.69
## 5  2019 MARKET~ CALIF~ RASPBERR~ RASPBERRIES, FRE~ TOTAL  NOT SPECIFIED    (D)
## 6  2019 MARKET~ CALIF~ RASPBERR~ RASPBERRIES, PRO~ TOTAL  NOT SPECIFIED    (D)
```

2.2 Get the raw data of strawberries

Now, we focus on one kind of berries–strawberries.

```
## Strawberries
sberry <- filter(ag_data, (Commodity=="STRAWBERRIES") & (Period=="YEAR"))
sberry <- select(sberry, -c(Period, Commodity))
```

2.3 Separate the column ‘Data Item’ into 3 columns “type”, “meas” and “what”.

```
#### Does every Data Item begin with "STRAWBERRIES, "
sum(str_detect(sberry$`Data Item`, "^STRAWBERRIES, ")) == length(sberry$`Data Item`)
```

```
## [1] FALSE
```

```
## Seperate the Data Item
sberry1 <- subset(sberry, str_detect(sberry$`Data Item`, "^STRAWBERRIES, ") == "TRUE")
sberry2 <- subset(sberry, str_detect(sberry$`Data Item`, "^STRAWBERRIES, ") == "FALSE")
sberry1 <- separate(sberry1, `Data Item`, c("B","type", "meas", "what"), sep = ",")
sberry2 <- separate(sberry2, `Data Item`, c("B","todo"), sep = "-")
sberry2 <- separate(sberry2, `todo`, c("type","meas", "what"), sep = ",")
sberry <- rbind(sberry1, sberry2)
sberry <- select(sberry, -B)
```

2.4 Seperate the column “type” into 3 columns “type”, “lab1” and “lab2”.

```
## Seperate the type
sberry1 <- subset(sberry, str_detect(sberry$`type`, " - ") == "TRUE")
sberry1 <- separate(sberry1, type,c("type", "lab1", "lab2"), " - ")
sberry1 <- separate(sberry1, type,c("b1", "type"), " ")

sberry2 <- subset(sberry, type == " FRESH MARKET")
sberry2$type <- "FRESH MARKET"

sberry3 <- subset(sberry, type == " PROCESSING")
sberry3$type <- "PROCESSING"

sberry4 <- subset(sberry, str_detect(sberry$`type`, " - ") == "FALSE" &
                    type != " FRESH MARKET" & type != " PROCESSING")
sberry4 <- separate(sberry4, type,c("b1", "lab1", "lab2"), " ")

sberry <- plyr::rbind.fill(sberry1, sberry2, sberry3, sberry4)

sberry[is.na(sberry)] <- ""

sberry <- select(sberry, -b1)
## OK now Data Item has been split into parts
```

2.5 Separate the column “Domain” into 2 columns “D_left” and “D_right”.

```
## onto Domain
sberry <- separate(sberry, Domain, c("D_left", "D_right"), sep = ", ")

sberry[is.na(sberry)] <- ""
```

2.6 Separate the column “Domain Category” into 4 columns “DC_left_l”, “DC_left_r”, “DC_right_l”, “DC_right_r”.

```
## And now Domain Category
sberry <- separate(sberry, `Domain Category`, c("DC_left", "DC_right"), sep = ", ")

## unique(sberry$DC_left)
## unique(sberry$DC_right)

## looks like DC_left combines labels
sberry$DC_right[which(str_detect(sberry$DC_left,"138831"))]="INSECTICIDE: (CYFLUMETOFEN = 138831)"
sberry$DC_left[which(str_detect(sberry$DC_left,"138831"))]="CHEMICAL"


## work on DC_left first

sberry <- separate(sberry, DC_left, c("DC_left_l", "DC_left_r"), sep = ": ")

## sberry$DC_left_l %>% unique()
## sberry$DC_left_r %>% unique()

## now work on DC_right
```

```
sberry <- separate(sberry, DC_right, c("DC_right_l", "DC_right_r"), sep = ": ")


sberry[is.na(sberry)] <- ""
```

2.7 Eliminate the redundancy in data set strawberries

a) remove column “DC_left_l”; b) remove column “DC_right_l”; c)create new column “label” = “lab1” + “lab2”; d)create new column “Chemical”; e)select the columns

```
## fine and remove redundant columns

## a) Test for D_left, DC_left_l
## paste(sberry$D_left,sberry$DC_left_l) %>% unique
## returns -- "CHEMICAL CHEMICAL"     "FERTILIZER FERTILIZER" "TOTAL NOT SPECIFIED"

## remove column bberry$DC_left_l
sberry <- select(sberry, -DC_left_l)

## b) Test for D_right, DC_right_l
# sum(sberry$D_right == sberry$DC_right_l)
# [1] 3220
# sberry$DC_left_r %>% unique()
# [1] ""            "(NITROGEN)"  "(PHOSPHATE)" "(POTASH)"    "(SULFUR)"

## remove column DC_right_l
sberry <- select(sberry, -DC_right_l)


## c) Test for lab1, lab2
# paste(sberry$lab1, sberry$lab2) %>% unique()
# [1] "APPLICATIONS "   "TREATED "        "PRODUCTION "     " "
# [5] "ACRES HARVESTED" "ACRES PLANTED"   "YIELD "

## create new column label
sberry <- mutate(sberry, label = paste(lab1,lab2))


## d) test for necessity of "chemical" in col D_left
# paste(sberry$D_left, sberry$D_right) %>% unique()
# [1] "CHEMICAL FUNGICIDE"   "CHEMICAL HERBICIDE"   "CHEMICAL INSECTICIDE"
# [4] "CHEMICAL OTHER"       "FERTILIZER "          "TOTAL "

## remove "Chemical" and joint the columns
sberry$D_left[which(sberry$D_left == "CHEMICAL")] <- ""
sberry <- mutate(sberry, Chemical=paste(D_left, D_right))
sberry <- select(sberry, -c(D_left, D_right))

## e) select the columns
sberry <- select(sberry, Year, State, type, what, meas, label, DC_left_r, DC_right_r, Chemical, Value )
```

2.8 Check for overlaps in "what" & "meas" and create a new column "units"

```
###  Now the problem is that we have entries in both the "what" and "meas" columns
##  that begin  "MEASURED IN"
##  how many are there

## in the column "what"
cnt_1 <- str_detect(sberry$what, "MEASURED IN")
sum(cnt_1)
```

```
## [1] 59
```

```
## in the column "meas"

cnt_2 <- str_detect(sberry$meas, "MEASURED IN")
sum(cnt_2)
```

```
## [1] 3093
```

```
## We want to put them all in the same column
## So, we will separate them from their current column and put them into
## two columns -- then we will test to make sure there aren't any overlaps
## and then merge the two columns

## we're going to use PURRR.  We need a simple function that takes a logical
## variable and a second variable.  It returns the second variable if the logical
## variable is true and returns a blank if it is false

f1 <- function(a,b){
  if(a){
    return(b)
  }else{
    return("")
  }
}

#########################################################
## now let's separate the "MEASURED IN" entries in the meas column
## form an index of the entries to be separated out

index_meas <- str_detect(sberry$meas, "MEASURED IN")

sberry <- mutate(sberry, m_in_1 = unlist(map2(index_meas, sberry$meas, f1)))

sberry <- mutate(sberry, meas = str_replace(sberry$meas, "MEASURED IN.*$", ""))

## Check
cnt_3 <- str_detect(sberry$meas, "MEASURED IN")
sum(cnt_3)
```

```
## [1] 0
```

```
#########################
## Now we will do the same thing with the
## "what" column
```

```
### index of cells to be isolated
index_what <- str_detect(sberry$what, "MEASURED IN")
sum(index_what)
```

```
## [1] 59
```

```
### create a column of the isolated cells
sberry <- mutate(sberry, m_in_2 = unlist(map2(index_what, sberry$what, f1)))

###  eliminate the isolated cells from the original column
sberry <- mutate(sberry, what = str_replace(sberry$what, "MEASURED IN.*$", ""))

### test that theere are no more "MEASURED IN" cells in the original column
cnt_what <- str_detect(sberry$what, "MEASURED IN")
sum(cnt_what)
```

```
## [1] 0
```

```
### Check for overlaps
sberry <- mutate(sberry, units = str_trim(paste(m_in_1, m_in_2)))

unique(sberry$units)
```

```
##  [1] "MEASURED IN LB"                      "MEASURED IN LB / ACRE / APPLICATION"
##  [3] "MEASURED IN LB / ACRE / YEAR"        "MEASURED IN NUMBER"
##  [5] "MEASURED IN PCT OF AREA BEARING"     "MEASURED IN $"
##  [7] "MEASURED IN CWT"                     "MEASURED IN TONS"
##  [9] ""                                    "MEASURED IN CWT / ACRE"
```

2.9 Rename and merge the columns with the same meaning

```
## rename the columns
sberry <- dplyr::rename(sberry, Avg = what, Marketing = meas, Harvest = label,
                               Chem_family = DC_left_r, Materials = DC_right_r,
                               Measures = units)

## select the columns
sberry <- select(sberry, Year, State, type, Marketing,
                   Measures, Avg, Harvest, Chem_family,
                   Materials, Chemical, Value )

unique(str_trim(paste(sberry$Marketing, sberry$Harvest)))
```

```
## [1] "APPLICATIONS"           "TREATED"                "PRODUCTION"
## [4] "UTILIZED - PRODUCTION" "ACRES HARVESTED"        "ACRES PLANTED"
## [7] "YIELD"
```

```
###  "Marketing" and "Harvest" belong in one column
sberry <- mutate(sberry, production = str_trim(paste(Marketing, Harvest)))

sberry <- select(sberry, Year, State, type, production, Measures,
                   Avg, Chem_family, Materials, Chemical, Value)


## "Chem_family" and "Chemical" belong in one column
sberry <- mutate(sberry, Chemical = str_trim(paste(Chem_family, Chemical)))
sberry <- select(sberry, Year, State, type, production, Avg, Measures,
                   Materials, Chemical, Value)

## numeric the column "Value"
sberry$Value <- as.numeric(str_replace_all(sberry$Value,c(','='')))
head(sberry)
```

```
##   Year      State    type   production Avg       Measures
## 1 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
## 2 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
## 3 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
## 4 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
## 5 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
## 6 2019 CALIFORNIA BEARING APPLICATIONS     MEASURED IN LB
##                                         Materials  Chemical Value
## 1                          (AZOXYSTROBIN = 128810) FUNGICIDE  5500
## 2    (BACILLUS AMYLOLIQUEFACIENS MBI 600 = 129082) FUNGICIDE    NA
## 3 (BACILLUS AMYLOLIQUEFACIENS STRAIN D747 = 16482) FUNGICIDE    NA
## 4                        (BACILLUS PUMILUS = 6485) FUNGICIDE    NA
## 5                   (BACILLUS SUBT. GB03 = 129068) FUNGICIDE    NA
## 6                       (BACILLUS SUBTILIS = 6479) FUNGICIDE    NA
```

```
write.csv(sberry, file = "/Users/amelia/Documents/mssp/MA615/Hw_berries/strawberries.csv", row.names = 
```

When we finish this part, we get a new csv file named “strawberries.csv” which is cleaned and organized.

3 EDA

3.1 Calculate the total values of strawberries every state and every year

```
## load the data
sberry <- read.csv("/Users/amelia/Documents/mssp/MA615/Hw_berries/Berries_strawberries/data/strawberrie
df <- filter(sberry, is.na(Value) == "FALSE")

## Calculate the total values of strawberries every state
t1 <- summarize(group_by(df, State), total=sum(Value))
t1
```

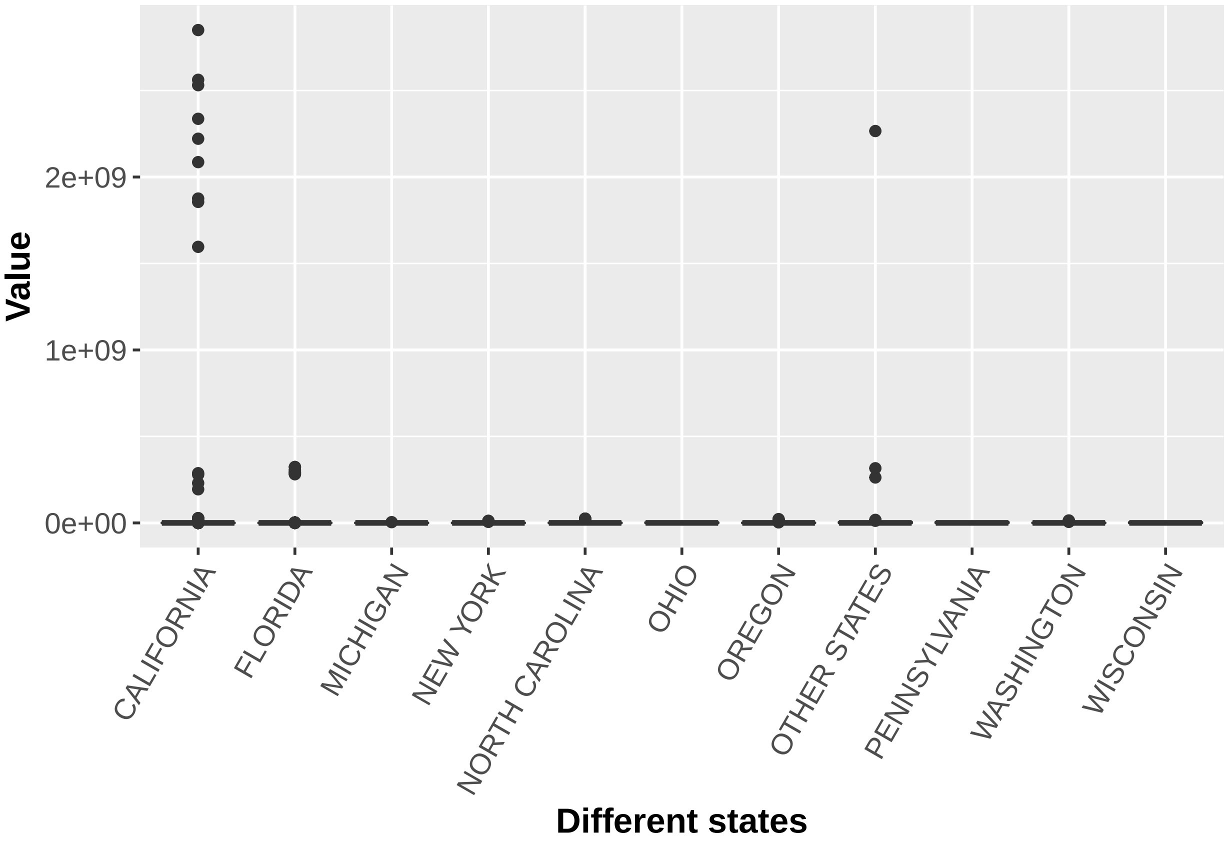
```
##         total
## 1 27108618147
```

```
## Calculate the total values of strawberries every state
t2 <- summarize(group_by(df, Year), total=sum(Value))
t2
```

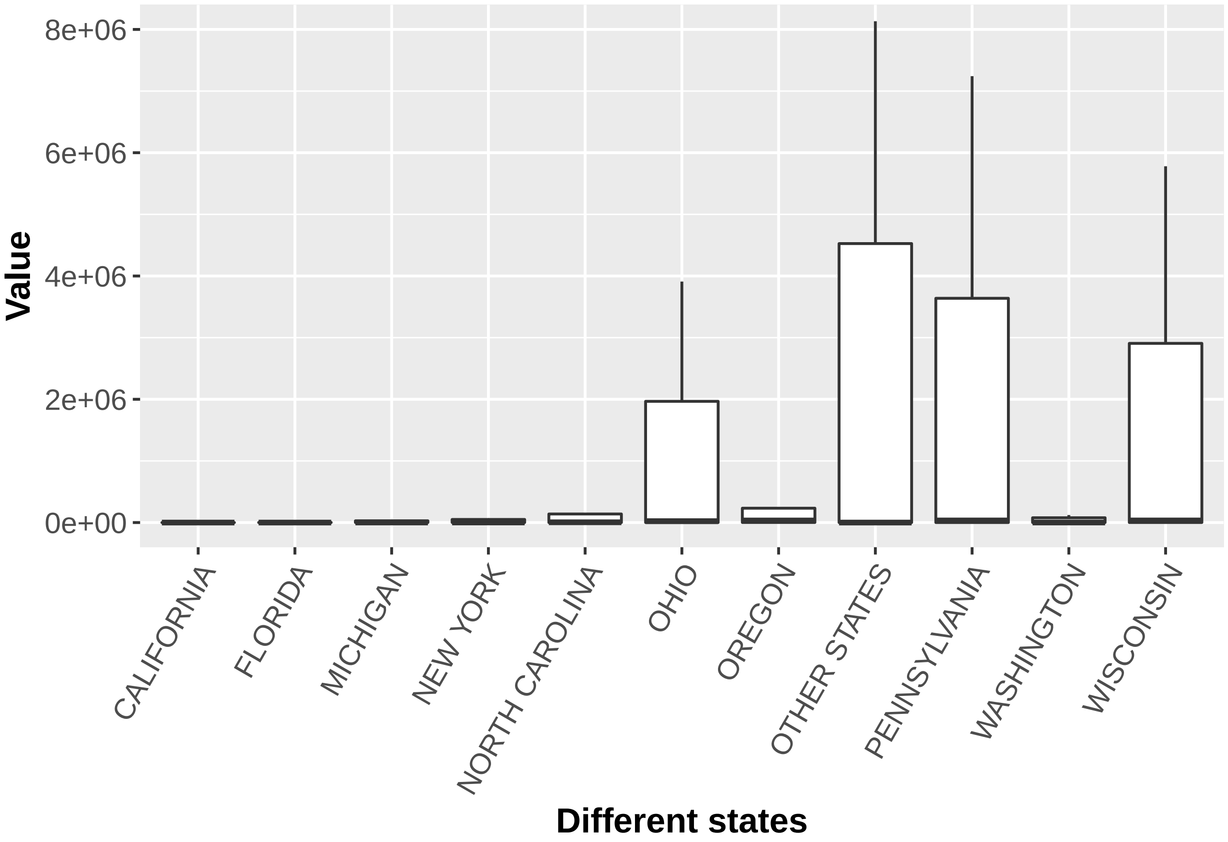
```
##         total
## 1 27108618147
```

3.2 Plot the boxplots of different states and different years

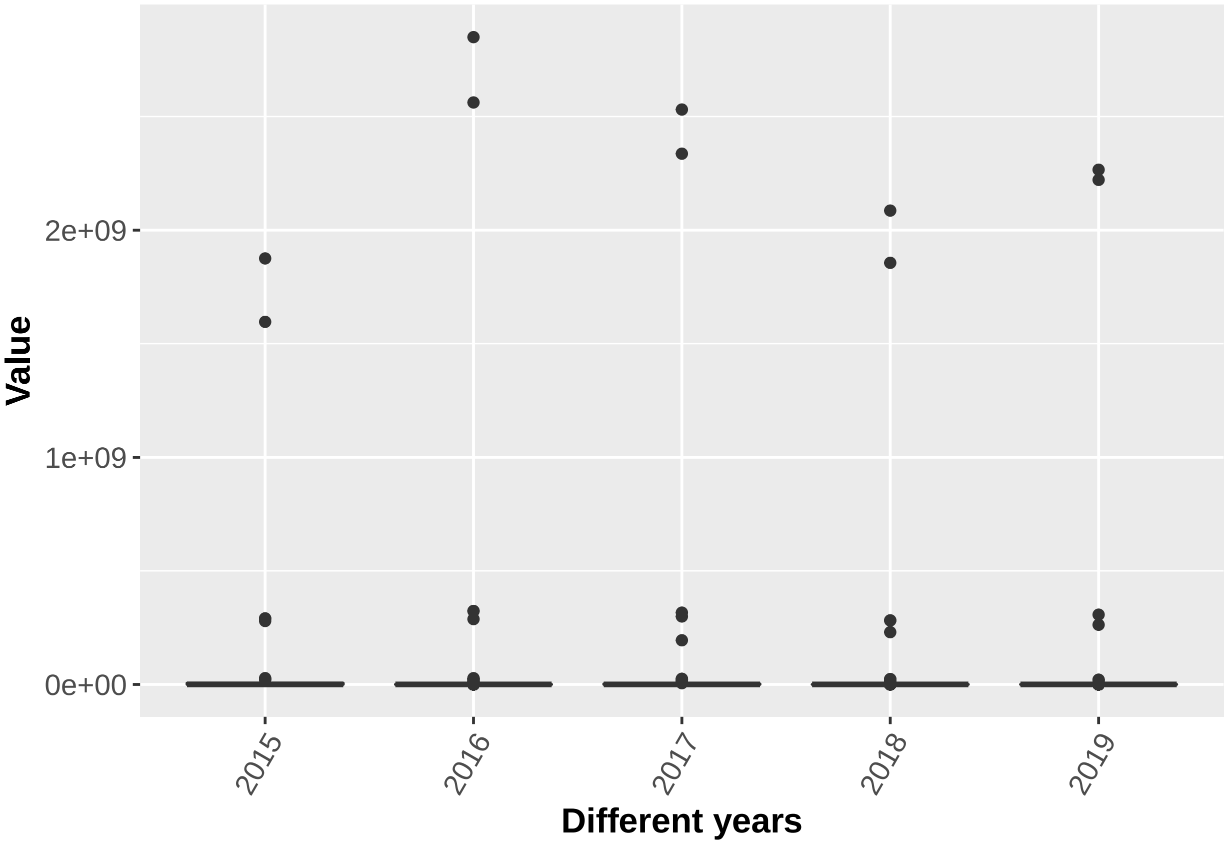
```
## boxplot of different states
bp1 <- ggplot(df, aes(x = State, y = Value))
bp1 <- bp1 + geom_boxplot() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        axis.text = element_text(size = 11),
        axis.title = element_text(size = 13, face = "bold")) +
  labs(x = "Different states")
bp1
```



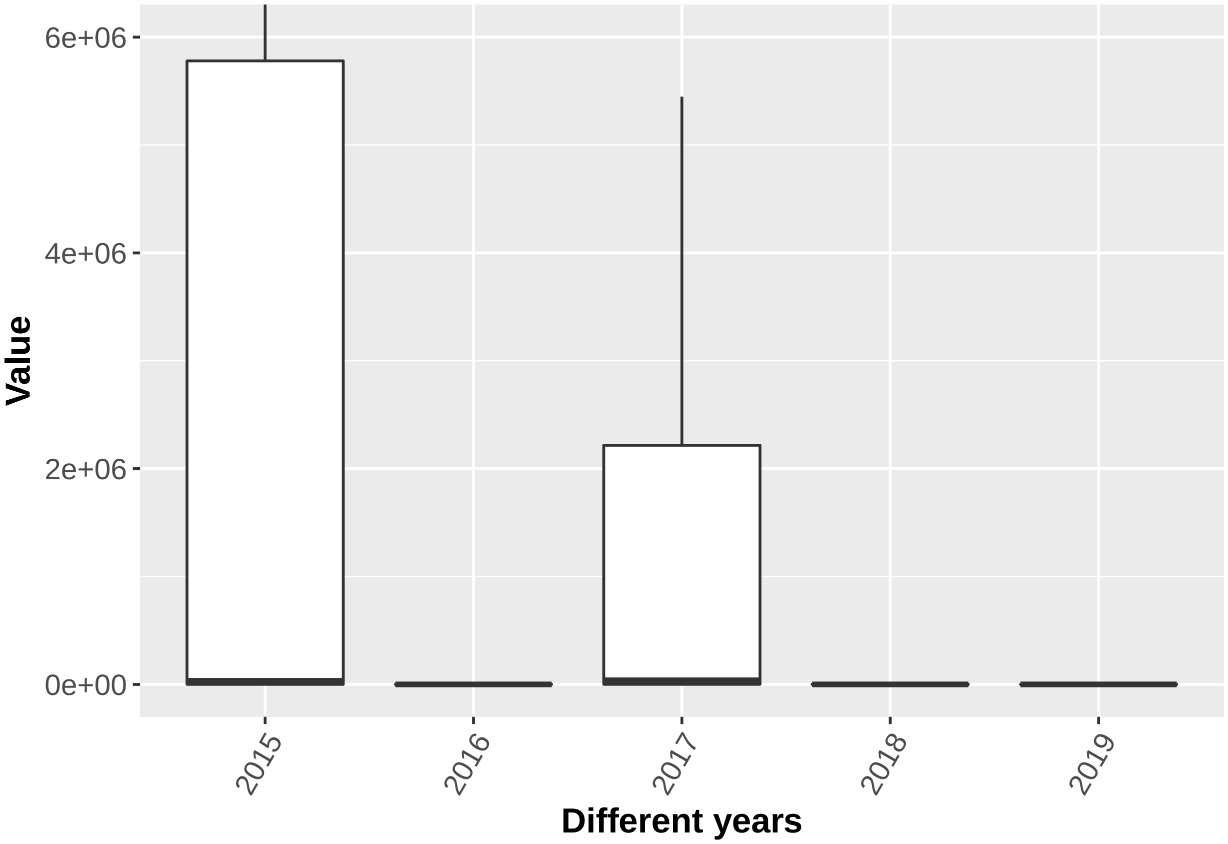
```
## excluding outliers
bp2 <- ggplot(df, aes(x = State, y = Value))
bp2 <- bp2 + geom_boxplot(outlier.colour = NA) +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        axis.text = element_text(size = 11),
        axis.title = element_text(size = 13, face = "bold")) +
  coord_cartesian(ylim = c(0, 8e+6)) +
  labs(x = "Different states")
bp2
```



```
## boxplot of different years
bp3 <- ggplot(df, aes(x = factor(Year), y = Value))
bp3 <- bp3 + geom_boxplot() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        axis.text = element_text(size = 11),
        axis.title = element_text(size = 13, face = "bold")) +
  labs(x = "Different years")
bp3
```



```
## excluding outliers
bp4 <- ggplot(df, aes(x = factor(Year), y = Value))
bp4 <- bp4 + geom_boxplot(outlier.colour = NA) +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        axis.text = element_text(size = 11),
        axis.title = element_text(size = 13, face = "bold")) +
  coord_cartesian(ylim = c(0, 6e+6)) +
  labs(x = "Different years")
bp4
```



3.2 Prepare the data for PCA

Here we use Principal Component Analysis (PCA) to analyze the relationships between different chemimcals. There are 8 kinds of chemicals: (NITROGEN) FERTILIZER, (PHOSPHATE) FERTILIZER, (POTASH) FERTILIZER, (SULFUR) FERTILIZER, (TOTAL) FUNGICIDE, (TOTAL) HERBICIDE, (TOTAL) INSECTICIDE and (TOTAL) OTHER.
Since the FERRTILIZER data only collected in 2015 and 2019, we will choose these two years 2018 and 2019 only. Besides, for missing value, we use mean value to replace.

```
## a) prepare data for pca
# load data
df_value <- filter(sberry,
                Year%in%c('2019','2018'),
                Measures=='MEASURED IN LB',
                Materials%in%c('(TOTAL)','')
                )

# in the rear of the data, it lacks a row of (SULFUR) FERTILIZER, just add it.
df_value <- rbind(df_value, df_value[nrow(df_value), ])
df_value$Value[nrow(df_value)] <- NA
df_value$Chemical[nrow(df_value)] <- '(SULFUR) FERTILIZER'

# arrange data
df_value$MC <- paste(df_value$Materials, df_value$Chemical)
df_value <- arrange(df_value, df_value$Year, df_value$State, df_value$MC)

# handle missing value
```

```
for(i in unique(df_value$MC)){
  m <- df_value$Value[df_value$Value!=' (D)' &
                        !is.na(df_value$Value) &
                        df_value$MC==i] %>%
    str_replace_all(c(','='')) %>%
    as.numeric() %>%
    mean()
  df_value$Value[(df_value$Value==' (D)' |
                    is.na(df_value$Value)) &
                   df_value$MC==i] <- m

}

# transform for PCA
j <- 1
for(i in unique(df_value$MC)){
  if(j==1) df_pca <- df_value$Value[df_value$MC==i]
  else df_pca <- cbind(df_pca, df_value$Value[df_value$MC==i])
  j <- 0
}
colnames(df_pca) <- unique(df_value$MC)
head(df_pca)
```

```
##       (NITROGEN) FERTILIZER  (PHOSPHATE) FERTILIZER  (POTASH) FERTILIZER
## [1,]               10676000                 5745000             10583000
## [2,]                 136000                   42000               173000
## [3,]                 668000                  493000               430000
## [4,]                 320000                   86000               389000
##       (SULFUR) FERTILIZER (TOTAL) FUNGICIDE (TOTAL) HERBICIDE
## [1,]              2637000            466900              7200
## [2,]              1347000            128400              5900
## [3,]                57000           1233500             10200
## [4,]              1347000            286900               300
##      (TOTAL) INSECTICIDE (TOTAL) OTHER
## [1,]              134400       3586800
## [2,]              102700       3803867
## [3,]              212600       7698900
## [4,]                5100        125900
```

```
## the data is ready.
```

3.2 Conduct PCA

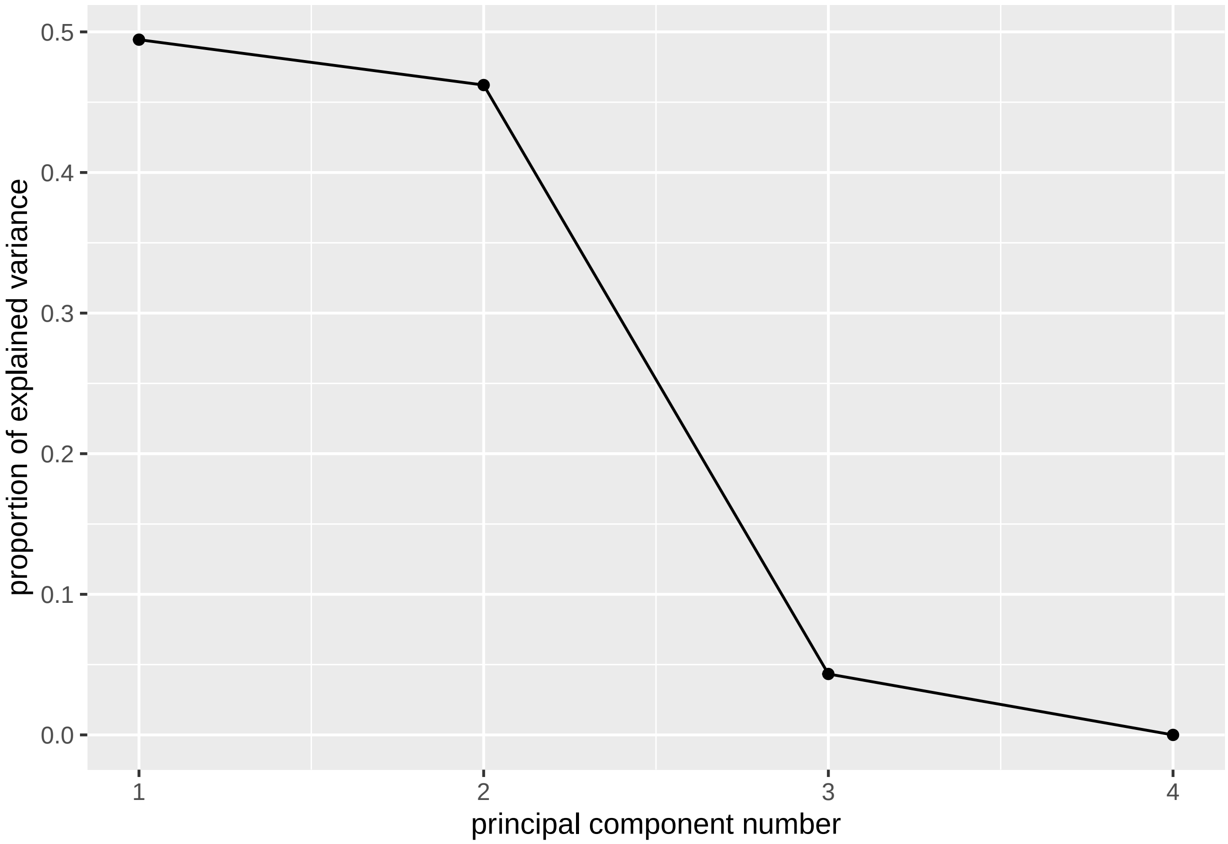
```
# start pca
pca <- prcomp(df_pca, center = T, scale. = T)
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3       PC4
## Standard deviation     1.9889 1.9229 0.58878 1.882e-16
## Proportion of Variance 0.4945 0.4622 0.04333 0.000e+00
## Cumulative Proportion  0.4945 0.9567 1.00000 1.000e+00
```

```
print(pca)
```

```
## Standard deviations (1, .., p=4):
## [1] 1.988906e+00 1.922911e+00 5.887812e-01 1.882050e-16
##
## Rotation (n x k) = (8 x 4):
##                                  PC1       PC2        PC3         PC4
##  (NITROGEN) FERTILIZER  -0.2068521 0.4720540  0.1399554  0.63273166
##  (PHOSPHATE) FERTILIZER -0.1913560 0.4788289  0.1459033 -0.64299113
##  (POTASH) FERTILIZER    -0.2187249 0.4665964  0.1287299 -0.21951687
##  (SULFUR) FERTILIZER    -0.4282683 0.2605169 -0.2604161  0.24336213
## (TOTAL) FUNGICIDE        0.4220915 0.1517651  0.7784517  0.12888862
## (TOTAL) HERBICIDE        0.3779983 0.3216774 -0.3879665  0.01214803
## (TOTAL) INSECTICIDE      0.4033952 0.3032953 -0.2158807  0.19152271
## (TOTAL) OTHER            0.4533973 0.2093073 -0.2676328 -0.15924657
```

```
ggbiplot::ggscreeplot(pca)
```



As we can see, the first 4 principal component is important.

4 Reference

[1] National Agricultural Statistics Service