Group2-EDA

Shuting Li, Lauren Marie, Zening Ye, Keliang Xu

10/25/2021

Clean data

import original data

```
strawb <- read_csv("data/Strawberries.csv")
```

```
## Rows: 3021 Columns: 21

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): Program, Period, Geo Level, State, Commodity, Data Item, Domain, D...
## dbl  (3): Year, State ANSI, watershed_code
## lgl  (8): Week Ending, Ag District, Ag District Code, County, County ANSI, Z...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
pesti <- read_csv("data/Pesticides.csv")
```

```
## Rows: 91 Columns: 6

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): Pesticide, Carcinogen, Hormone Disruptor, Neurotoxins, Developmenta...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

data cleaning about strawb.csv

drop all NA columns

```
drop_na_info <- function(df){
  cnames = colnames(df)
  T = NULL
  for(i in 1:ncol(df)){T <- c(T, nrow(unique(df[i])))}
  drop_cols <- cnames[which(T == 1)]
  return(dplyr::select(df, !all_of(drop_cols)))
}

strawb1 <- drop_na_info(strawb)

colnames(strawb1)
```

```
##  [1] "Program"          "Year"             "Period"           "State"
##  [5] "State ANSI"       "Data Item"        "Domain"           "Domain Category"
##  [9] "Value"            "CV (%)"
```

After droping NA columns, we get strawb1 with 10 columns.

Separate 'Data Item' into 4 columns

```
strawb1 %<>% separate(col = 'Data Item', into = c("Strawberries", "Items", "Discription", "Units"),sep

distinct(strawb1, Strawberries)
```

```
## # A tibble: 2 x 1
##   Strawberries
##   <chr>
## 1 STRAWBERRIES
## 2 STRAWBERRIES - YIELD
```

```
distinct(strawb1, Items)
```

```
## # A tibble: 7 x 1
##   Items
##   <chr>
## 1 " ORGANIC - OPERATIONS WITH SALES"
## 2 " ORGANIC - SALES"
## 3 " ORGANIC"
## 4 " MEASURED IN CWT / ACRE"
## 5 " MEASURED IN TONS / ACRE"
## 6 " BEARING - APPLICATIONS"
## 7 " BEARING - TREATED"
```

```
distinct(strawb1, Discription)
```

```
## # A tibble: 12 x 1
##    Discription
##    <chr>
##  1  <NA>
##  2 " MEASURED IN $"
##  3 " MEASURED IN CWT"
##  4 " FRESH MARKET - OPERATIONS WITH SALES"
##  5 " FRESH MARKET - SALES"
##  6 " PROCESSING - OPERATIONS WITH SALES"
##  7 " PROCESSING - SALES"
##  8 " MEASURED IN LB"
##  9 " MEASURED IN LB / ACRE / APPLICATION"
## 10 " MEASURED IN LB / ACRE / YEAR"
## 11 " MEASURED IN NUMBER"
## 12 " MEASURED IN PCT OF AREA BEARING"
```

```
distinct(strawb1, Units)
```

```
## # A tibble: 4 x 1
##   Units
##   <chr>
## 1  <NA>
## 2 " MEASURED IN $"
## 3 " MEASURED IN CWT"
## 4 " AVG"
```

Separate ‘Data Item’ into “Strawberries”, “Items”, “Discription”, “Units”.

Separate ‘Domain’ into 2 columns

```
strawb1 %<>% separate(col = Domain, into = c("dname", "type"), sep = ",", fill = "right")

distinct(strawb1, dname)
```

```
## # A tibble: 4 x 1
##   dname
##   <chr>
## 1 ORGANIC STATUS
## 2 TOTAL
## 3 CHEMICAL
## 4 FERTILIZER
```

```
distinct(strawb1, type)
```

```
## # A tibble: 5 x 1
##   type
##   <chr>
## 1  <NA>
## 2 " FUNGICIDE"
## 3 " HERBICIDE"
## 4 " INSECTICIDE"
## 5 " OTHER"
```

Separate ‘Domain’ into “dname”, “type”.

Separate ‘Domain Category’ into 2 columns

```
strawb1 %<>%
  mutate(Chemicals = `Domain Category`) %>%
  relocate(Chemicals, .after = `Domain Category`)

strawb1 %<>%
  separate(Chemicals, into =c('Title', 'Details'), sep = ":", fill = "right")
```

```
## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1129

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1241

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1351

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1461

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1573

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 1937

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2035

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2131

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2227

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2325

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2418

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2491

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2562

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2633

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2706

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2872

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2906

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2939

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 2972

## Warning in gregexpr(pattern, x, perl = TRUE): PCRE error
##  'UTF-8 error: isolated byte with 0x80 bit set'
##  for element 3007
```

```
distinct(strawb1, Details)
```

```
## # A tibble: 159 x 1
##    Details
##    <chr>
##  1 " (NOP USDA CERTIFIED)"
##  2  <NA>
##  3 " (AZOXYSTROBIN = 128810)"
##  4 " (BACILLUS AMYLOLIQUEFACIENS MBI 600 = 129082)"
##  5 " (BACILLUS AMYLOLIQUEFACIENS STRAIN D747 = 16482)"
##  6 " (BACILLUS PUMILUS = 6485)"
##  7 " (BACILLUS SUBT. GB03 = 129068)"
##  8 " (BACILLUS SUBTILIS = 6479)"
##  9 " (BLAD = 30006)"
## 10 " (BORAX DECAHYDRATE = 11102)"
## # ... with 149 more rows
```

```
strawb1$Details <- str_replace(strawb1$Details, "\\(", "")
strawb1$Details <- str_replace(strawb1$Details, "\\)", "")

strawb1 %<>%
  separate(Details, into = c('Chemical Name', "Number"), sep = "=", fill = "right")

distinct(strawb1, `Chemical Name`)
```

```
## # A tibble: 159 x 1
##    `Chemical Name`
##    <chr>
##  1 " NOP USDA CERTIFIED"
##  2  <NA>
##  3 " AZOXYSTROBIN "
##  4 " BACILLUS AMYLOLIQUEFACIENS MBI 600 "
##  5 " BACILLUS AMYLOLIQUEFACIENS STRAIN D747 "
##  6 " BACILLUS PUMILUS "
##  7 " BACILLUS SUBT. GB03 "
##  8 " BACILLUS SUBTILIS "
##  9 " BLAD "
## 10 " BORAX DECAHYDRATE "
## # ... with 149 more rows
```

```
distinct(strawb1, Number)
```

```
## # A tibble: 153 x 1
##    Number
##    <chr>
##  1  <NA>
##  2 " 128810"
##  3 " 129082"
##  4 " 16482"
##  5 " 6485"
##  6 " 129068"
##  7 " 6479"
##  8 " 30006"
##  9 " 11102"
## 10 " 128008"
## # ... with 143 more rows
```

```
strawb1$`Chemical Name` <- str_trim(strawb1$`Chemical Name`)
```

Replicate 'Domain.Category' to new variable and separate this new variable into 'Title', 'Details'.

And then clean 'Details' and separate it into 'Chemical Name', "Number". Also captalize all words in 'Chemical Name'.

drop useless columns

```
drops <- c("Strawberries", "Domain Category")
strawb1 <- strawb1[ , !(names(strawb1) %in% drops)]
```

Drop "Strawberries", "Domain Category", because it is useless in data analysis.

data cleaning about pesti.csv

drop NA rows in pesti.csv and clean it

```
pesti1 <- pesti %>% rename('Chemical Name' = Pesticide )

pesti1 %<>% filter(!is.na(pesti1))

pesti1$`Chemical Name` <- toupper(pesti1$`Chemical Name`)
```

Drop NA rows in pesti.csv, finally we get pesti1 with 45 rows.

And then rename 'Pesticide' to 'Chemical Name' and captalize all words in 'Chemical Name'.

Define Human Toxins level

```
pesti1 <- pesti1 %>% mutate('Human Toxins' = case_when(
  pesti1$Carcinogen == "known" | pesti1$Neurotoxins== "present" | pesti1$`Developmental or Reproductive
  pesti1$Carcinogen == "possible" & pesti1$`Hormone Disruptor`=="suspected" & is.na(pesti1$Neurotoxins)
  pesti1$Carcinogen == "possible" & is.na(pesti1$`Hormone Disruptor`) & is.na(pesti1$Neurotoxins) & is.
  pesti1$Carcinogen == "probable" & is.na(pesti1$`Hormone Disruptor`) & is.na(pesti1$Neurotoxins) & is.
  is.na(pesti1$Carcinogen) & pesti1$`Hormone Disruptor`=="suspected" & is.na(pesti1$Neurotoxins) & is.n
))
```

Use columns 'carcinogen', 'Neurotoxins', 'Developmental or Reproductive Toxins' to define human toxins level.

High toxic for human: carcinogen = known or Neurotoxins = present or `Developmental or Reproductive Toxins` = present.

Moderate toxic for human: carcinogen = probable/possible and `Hormone Disruptor` = suspect.

Slight toxic for human: carcinogen= possible/possible or `Hormone Disruptor` = suspect, only one happens.

wrangling two datasets

```
strawbPesti <- inner_join(strawb1, pesti1,by="Chemical Name")

# Write the dataset into csv.
write_csv(strawbPesti, "strawbPesti.csv")
```

Combine two dataset into strawbPesti.csv by key column 'Chemical Name'. Only keep rows with known pesticides.

Data Visualization

```
pacman::p_load('ggplot2', 'plotly',"dplyr")
```

clean strawbPesti.csv

```
distinct(strawbPesti,State)
```

```
## # A tibble: 4 x 1
##   State
##   <chr>
## 1 CALIFORNIA
## 2 FLORIDA
## 3 OREGON
## 4 WASHINGTON
```

```
strawbPesti <- arrange(strawbPesti,`Chemical Name`)
#colnames(strawbPesti)
strawbPesti1 <- strawbPesti[,-13]
strawbPesti2 <- strawbPesti1[,c(2,4,7,12,13,20)]
strawbPesti3 <- filter(strawbPesti2, Discription==" MEASURED IN LB")
strawbPesti4 <- filter(strawbPesti3, Value !="(D)")
strawbPesti4$Value <- as.numeric(sub(",", "", strawbPesti4$Value, fixed = TRUE))
```

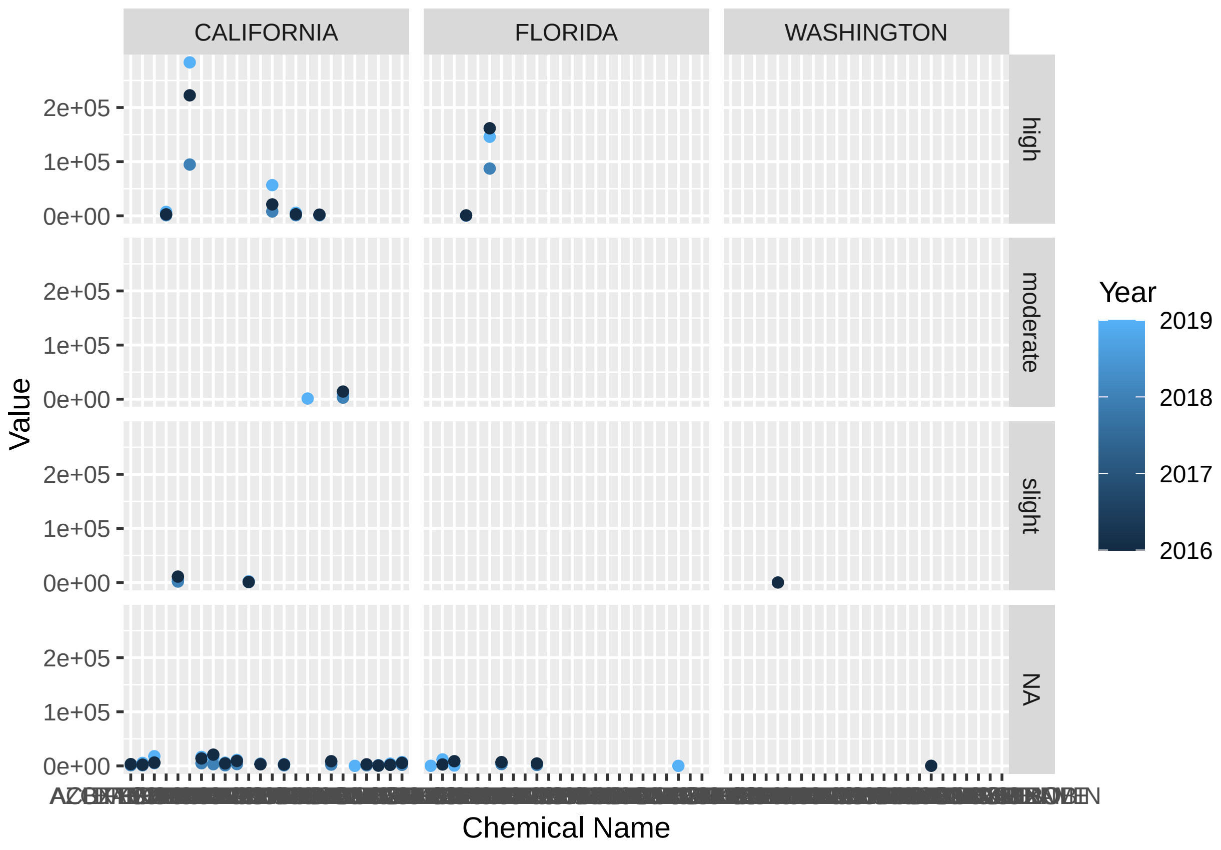
Only choose columns we will use: 'Year', 'State', 'Discription', 'Chemical Name', 'Value', 'Human Toxins'. Which is dataset "strawbPesti2".

We can see from strawbPesti2, values are different depend on measured methods for each chemical type in each state, so we can choose one method to go deep, finally we chose "MEASURED IN LB". Then we get dataset "strawbPesti3".

Then we drop no meaning rows of value. And change value into numeric variable. Finally we get dataset "strawbPesti4".

show all states chemical usage value

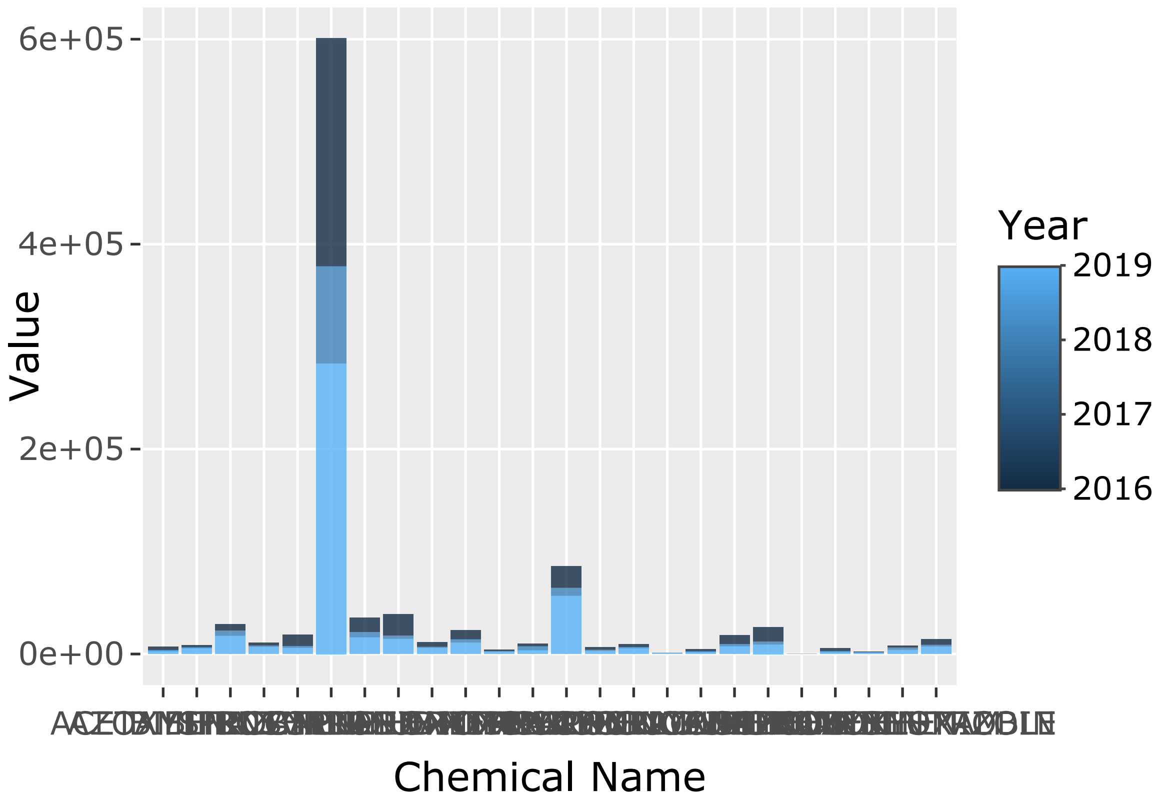
```
ggplot(strawbPesti4) +
  geom_point(aes( x= `Chemical Name`, y= Value,col=Year)) +
  facet_grid(`Human Toxins`~State)
```



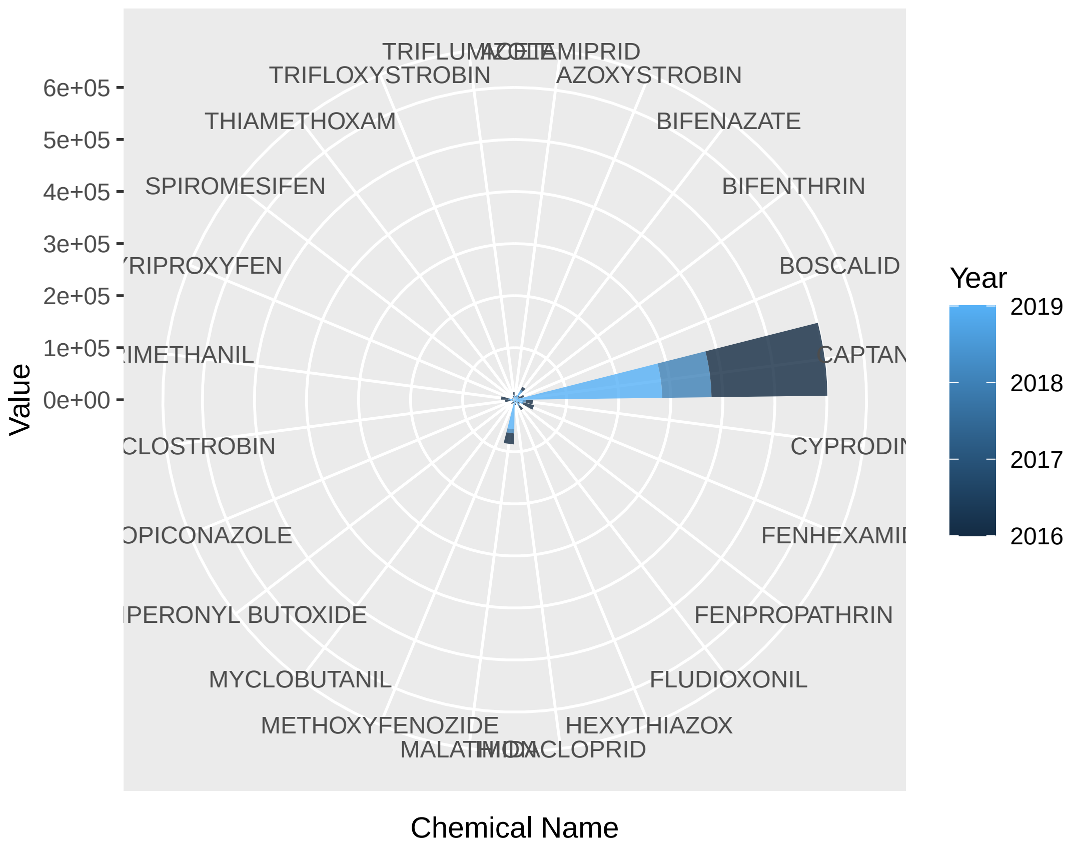
Chemical usage in California

```r
cali <- filter(strawbPesti4,State=="CALIFORNIA")

# bar plot
calibar <- ggplot(cali)+
  geom_bar(mapping = aes(x=`Chemical Name`,y=Value,fill=`Year`), alpha = 4/5,stat = "identity")
ggplotly(calibar)
```



```
# polar plot
calibar + coord_polar()
```



```
## we can see which peti used more often in CALI throughout three years
```