Strawberry Analysis - MA615

Jordan Stout

2024-10-10

In this document I will be explaining the process of cleaning open-source USDA data on Strawberry production in the US from 2018 to 2024. The cleaned data is exported as 4 separate CSV files: survey_data and census_data includes data on strawberry production that was collected through surveys and the census respectively, chemicals includes all the data regarding chemical use in strawberry production, and census_organic_data includes all data involving organic strawberry sales as per the census.

First we will load the data

```
strawberry <- read_csv("strawberries25_v3.csv", col_names = TRUE)
```

```
## Rows: 12669 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (15): Program, Period, Geo Level, State, State ANSI, Ag District, County...
## dbl  (2): Year, Ag District Code
## lgl  (4): Week Ending, Zip Code, Region, Watershed
##
## 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.
```

Now we check to make sure that every row is associated with a state.

```
state_all <- strawberry |> distinct(State)

state_all1 <- strawberry |> group_by(State) |> count()
```

Some columns of the data only have one value which is useless to us. To assist us in data cleaning we create a function 'drop_one_value_col()' which will remove such columns.

```
drop_one_value_col <- function(df){
  drop <- NULL
  for(i in 1:dim(df)[2]){
    if((df |> distinct(df[,i]) |> count()) == 1){
    drop = c(drop, i)
    }
  }
  if(is.null(drop)){
    return("none")}
  else{
     strawberry <- df[, -1*drop]
  }
```

```
}
strawberry <- drop_one_value_col(strawberry)
drop_one_value_col(strawberry)
```

```
## [1] "none"
```

Start by removing bad values from 'Value' and 'CV (%)' columns and make 'Value' column numeric type

```
strawberry$Value[strawberry$Value == "(D)"] <- NA
strawberry$Value[strawberry$Value == "(Z)"] <- NA

strawberry$`CV (%)`[strawberry$`CV (%)` == "(D)"] <- NA
strawberry$`CV (%)`[strawberry$`CV (%)` == "(L)"] <- NA
strawberry$`CV (%)`[strawberry$`CV (%)` == "(H)"] <- NA
```

We will only keep values of 'Geo Level' that include NATIONAL AND STATE becasue the rest is useless to us.

```
strawberry <- strawberry |>
  filter(`Geo Level`== "NATIONAL" | `Geo Level`== "STATE")
```

We will create two new data frames. 'census' will contain all the data where the program was the census, likewise 'survey' will include those where the program was survey.

```
census <- strawberry |>
  filter(Program=="CENSUS")

census <- census |>
  drop_one_value_col()

survey <- strawberry |>
  filter(Program == "SURVEY")

survey <- survey |>
  drop_one_value_col()

nrow(strawberry) == (nrow(survey) + nrow(census))
```

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

Clean and isolate the chemicals used by creating two new columns: 'chem_name' and 'chem_num'

```
survey <- survey |>
  mutate(`Domain Category` = gsub(".*: \\(([^=]+) = ([0-9]+)\\)", "\\1,\\2", `Domain Category`)) |>
  separate(`Domain Category`, into = c("chem_name", "chem_num"), sep = ",") |>
  mutate(chem_name = trimws(chem_name), chem_num = as.numeric(trimws(chem_num)))
```

Extract 'use' from 'domain' column

```
survey <- survey |>
  separate(Domain, into = c("Domain", "use"), sep = ",", extra = "merge")
```

Use stringr + regexs to sort out the 'Data Item' column

```
survey <- survey |>
  mutate(measurement = str_extract(`Data Item`, "(?<=MEASURED\\s).*")) |>
  mutate(`Data Item` = str_remove(`Data Item`, "MEASURED.*")) |>
  separate(`Data Item`, into = c("Data Item", "category"), sep = "[,-]", extra = "merge", fill = "right
  mutate(measurement = gsub("\\bIN\\b", "", measurement)) |>
  mutate(measurement = trimws(measurement))
```

Remove 'Data Item' column because it has no more data that we need as well as remove unwanted commas

```
survey <- survey |>
  select(-`Data Item`)
survey$category <- gsub(",", "", survey$category)
```

Separate chemicals and total

```
total <- survey |>
  filter(Domain == "TOTAL")

chemicals <- survey |>
  filter(Domain == "CHEMICAL")

total <- drop_one_value_col(total)
chemicals <- drop_one_value_col(chemicals)
```

Organize 'Data Item' column in Census data

```
census <- census |>
  separate_wider_delim(cols = `Data Item`, delim = " - ", names = c("strawberries", "Category"), too_ma
```

Make a new dataframe out of the census specific data that includes only organic strawberry sales

```
census <- census |>
  separate_wider_delim(cols = strawberries, delim = ", ", names = c("strawberries", "ORGANIC", "organic
                       too_few = "align_start") |>
  drop_one_value_col()

census_organic <- census |>
  filter(ORGANIC == "ORGANIC")

census <- census[(is.na(census$ORGANIC)),]

census <- census |>
  drop_one_value_col()
```

Now we can split the category column in census into two columns. Measure and bearing to organize the bearing type.

```
census <- census |>
  separate_wider_delim(cols = `Category`, delim = " ", names = c("COL1", "COL2"), too_many = "merge", to

census$COL2 <- str_replace(census$COL2,"WITH ","")

census <- census |>
  rename(Measure = COL1, Bearing_type= COL2)
```

Split up 'Domain Category' column in the census dataframe

```
census <- census |>
  rename(size_bracket = `Domain Category`)

census$size_bracket <- str_replace(census$size_bracket, "NOT SPECIFIED", "TOTAL")

census$size_bracket <- str_replace(census$size_bracket, "AREA GROWN: ", "")
```

Now onto the organic census data. First lets drop all the one value columns

```
census_organic <- census_organic |>
  drop_one_value_col()
```

Now we can clean "Category" column similar to how we cleaned it with the survey data. By splitting the columns at the word measure.

```
census_organic <- census_organic |>
  separate(Category, into = c("Category", "measurement"), sep = " MEASURED ", extra = "merge", fill = "
```

Get rid of commas and 'IN' from the new column it is just units again.

```
census_organic$Category <- gsub(",", "", census_organic$Category)
census_organic$measurement <- gsub("IN", "", census_organic$measurement, ignore.case = TRUE)
```

Change values in 'Value' column to numeric type

```
census <- census |>
  mutate(Value = as.numeric(gsub(",", "", Value)))
```

This is a function that calculates ratios for each state by dividing total acres grown by total bearing acres, then applies those ratios to the total acres grown in the same size bracket where the value is marked as NA.

```
fix_bearing_na <- function(df) {
  df_ratio <- df
  states <- unique(df$State)
  for (state in states) {
    state_data <- df |>
      filter(State == state)

    grown_value <- state_data |>
      filter(Bearing_type == "GROWN", Domain == "TOTAL", size_bracket == "TOTAL") |>
```

```
      pull(Value)

    bearing_value <- state_data |>
      filter(Bearing_type == "BEARING", Domain == "TOTAL", size_bracket == "TOTAL") |>
      pull(Value)

    if (length(grown_value) > 0 && length(bearing_value) > 0 && grown_value > 0) {
      percentage <- bearing_value / grown_value}
    else {
      next
    }

    df_ratio <- df_ratio |>
      mutate(Value = ifelse(is.na(Value) & Bearing_type == "BEARING" & State == state, percentage * Valu
  }
  return(df_ratio)
}
```

Same function but for non-bearing acres

```
fix_nonbearing_na <- function(df) {
  df_ratio <- df

  states <- unique(df$State)

  for (state in states) {
    state_data <- df |>
      filter(State == state)

    grown_value <- state_data |>
      filter(Bearing_type == "GROWN", Domain == "TOTAL", size_bracket == "TOTAL") |>
      pull(Value)

    non_bearing_value <- state_data |>
      filter(Bearing_type == "NON-BEARING", Domain == "TOTAL", size_bracket == "TOTAL") |>
      pull(Value)

    if (length(grown_value) > 0 && length(non_bearing_value) > 0 && grown_value > 0) {
      percentage <- non_bearing_value / grown_value}
    else{
      next
    }

    df_ratio <- df_ratio |>
      mutate(Value = ifelse(is.na(Value) & Bearing_type == "NON-BEARING" & State == state, percentage *
  }
  return(df_ratio)
}
```

Apply both these functions

```
census <- fix_bearing_na(census)
census <- fix_nonbearing_na(census)
```

Write CSVs

```
# write.csv(survey, "survey_data.csv", row.names = FALSE)
# write.csv(census, "census_data.csv", row.names = FALSE)
# write.csv(chemicals, "chemicals.csv", row.names = FALSE)
# write.csv(census_organic, "census_organic_data.csv", row.names = FALSE)
```

Chemical Analysis

I like to believe that the EPA is not sentencing me to death by eating strawberries, so I will be using a dataset found through the USDA Pesticide Data Program which is a national pesticide residue monitoring program.

```
usda_chem_original<-read.csv("usda_chem_sample_data.csv")
library(tidyverse)
```

Do some initial cleaning. We have been working with Florda and California data, but we will include all samples whose origin is the US. We will subset the data to include the sample ID, pesticide names, the concentration found in the sample (ppm), and the established EPA tolerance level for that pesticide (ppm).

```
usda_chem<-as.data.frame(usda_chem_original)

usda_chem <- usda_chem |>
  filter(Country == "") |>
  select(Pesticide.Name, EPA.Tolerance..ppm., Concentration, Sample.ID) |>
  mutate(EPA.Tolerance..ppm. = as.numeric(EPA.Tolerance..ppm.))
usda_chem<-na.omit(usda_chem)
```

Now we will do some analysis to see if any samples exceed that EPA tolerance level.

```
usda_chem <- usda_chem |>
  mutate(uh_oh = Concentration > EPA.Tolerance..ppm.)

is_true<-usda_chem |>
  filter(uh_oh == TRUE)
is_true
```

```
##   Pesticide.Name EPA.Tolerance..ppm. Concentration          Sample.ID uh_oh
## 1   Pyriproxyfen                 0.3          0.37 OH1811260207SZWA1P  TRUE
```

We can see that there was one instance of a detected pesticide being about the EPA tolerance level: Pyriproxyfen. As per the EPAs Human Health Draft Risk Assessment, Pyriproxyfen has an RfD of 0.01mg/kg/day. This means that for every kilogram a persons weight, they are allowed to consume 0.01 mg/day. Let's see

how many strawberries I would have to eat to experience the adverse effects of Pyriproxyfen using me as an example.

Safe Exposure = RfD x Body Weight = 0.01mg/kg/day x 65kg = 0.7mg/day

Kilograms of Strawberries = Safe Exposure / Concentration Found = 0.7mg/day / 0.37mg/kg = 1.89kg

Number of Strawberries = 1.89kg / 0.02kg = 95 Strawberries

If you find me eating 95 strawberries, please know I have bigger issues than Pyriproxyfen toxicity.

Let's talk about Chlorpyrifos, widely considered one of the most dangerous pesticides used in strawberry production. It has been linked to Reduced IQ, Attention disorders, Autism, Neurodevelopmental disorders, and Death. Chlorpyrifos was banned in the United States in March 2021 (6 years after the original proposition to ban the substance). It was later reinstated its use in August 2022 stating the EPA had failed to follow proper procedures.

We will now analyze the instances of Chlorpyrifos being detected in samples of US grown strawberries.

```
chlor <- usda_chem |>
  filter(Pesticide.Name == "Chlorpyrifos")
chlor
```

```
##   Pesticide.Name EPA.Tolerance..ppm. Concentration         Sample.ID uh_oh
## 1   Chlorpyrifos                 0.2         0.008 TX1906100105SZWA1 FALSE
```

As we can see, out of 2141 samples taken across the US there was one instance of Chlorpyrifos detection where is was recorded at 4% of the EPA defined tolerance level.

Malathion is, like Chlorpyrifos, considered an extremely dangerous Organophosphate used in strawberry production. Let's see how much gets into our food.

```
mala <- usda_chem |>
  filter(Pesticide.Name == "Malathion")
head(mala)
```

```
##   Pesticide.Name EPA.Tolerance..ppm. Concentration          Sample.ID uh_oh
## 1      Malathion                   8        0.0120 CA1810150001SZWA1P FALSE
## 2      Malathion                   8        0.0084 CA1810150004SZWA1P FALSE
## 3      Malathion                   8        0.0052 CA1810150151SZWA1P FALSE
## 4      Malathion                   8        0.0038  CA1810150504SZWA1 FALSE
## 5      Malathion                   8        0.0050  CA1810150554SZWA1 FALSE
## 6      Malathion                   8        0.0120  CA1811140516SZWA1 FALSE
```

```
maxi<-max(mala$Concentration)
paste("Maximum Malathion Concentration:",maxi)
```

```
## [1] "Maximum Malathion Concentration: 0.091"
```

In 2% of the samples Malathion was detected and the maximum concentration found in those samples was 0.091ppm. This is merely 1.1% of the EPA declared tolerance level.

Finally we will explore Carbendazim, a fungicide associated with reproductive toxicity and potential endocrine disruption.

```
carb <- usda_chem |>
  filter(Pesticide.Name == "Carbendazim (MBC)")
head(carb)
```

```
##      Pesticide.Name EPA.Tolerance..ppm. Concentration          Sample.ID uh_oh
## 1 Carbendazim (MBC)                   7        0.0096 CA1810150001SZWA1P FALSE
## 2 Carbendazim (MBC)                   7        0.0120  CA1810150504SZWA1 FALSE
## 3 Carbendazim (MBC)                   7        0.0021 CA1811140561SZWA1P FALSE
## 4 Carbendazim (MBC)                   7        0.0250  CA1812170450SZWA1 FALSE
## 5 Carbendazim (MBC)                   7        0.0250  CO1810220008SZWA1 FALSE
## 6 Carbendazim (MBC)                   7        0.0310  CO1810220010SZWA1 FALSE
```

```
max(carb$Concentration)
```

```
## [1] 0.044
```

Again we can see that only 0.4% of samples contained traces of Carbendazim and the maximum concentration found was 0.5% of that allowed by the EPA.

Jon Neimann helped me with the final functions

https://pmc.ncbi.nlm.nih.gov/articles/PMC10217756/ https://www.sciencedirect.com/science/article/pii/S2352364615300055