Stawberries 2

Gary Wang

2024-09-30

Preparing data for analysis

Acquire, explore, clean & structure, EDA

Data cleaning and organization

"An introduction to data cleaning with R" by Edwin de Jonge and Mark van der Loo

"Problems, Methods, and Challenges in Comprehensive Data Cleansing" by Heiko Müller and Johann-Christoph Freytag

Strawberries

Questions

- Where they are grown? By whom?
- Are they really loaded with carcinogenic poisons?
- Are they really good for your health? Bad for your health?
- Are organic strawberries carriers of deadly diseases?
- When I go to the market should I buy conventional or organic strawberries?
- Do Strawberry farmers make money?
- How do the strawberries I buy get to my market?

The data

The data set for this assignment has been selected from:

[USDA_NASS_strawb_2024SEP25 The data have been stored on NASS here: USDA_NASS_strawb_2024SEP

and has been stored on the blackboard as strawberries25_v3.csv.

read and explore the data

Set-up

```
library(knitr)
library(kableExtra)
library(tidyverse)
```

Read the data and take a first look

```
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.
```

```
glimpse(strawberry)
```

```
Rows: 12,669
Columns: 21
$ Program            <chr> "CENSUS", "CENSUS", "CENSUS", "CENSUS", "CENSUS", "~
$ Year               <dbl> 2022, 2022, 2022, 2022, 2022, 2022, 2022, 2022, 202~
$ Period             <chr> "YEAR", "YEAR", "YEAR", "YEAR", "YEAR", "YEAR", "YE~
$ `Week Ending`      <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ `Geo Level`        <chr> "COUNTY", "COUNTY", "COUNTY", "COUNTY", "COUNTY", "~
$ State              <chr> "ALABAMA", "ALABAMA", "ALABAMA", "ALABAMA", "ALABAM~
$ `State ANSI`       <chr> "01", "01", "01", "01", "01", "01", "01", "01", "01~
```

```
$ `Ag District`      <chr> "BLACK BELT", "BLACK BELT", "BLACK BELT", "BLACK BE~
$ `Ag District Code` <dbl> 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40,~
$ County             <chr> "BULLOCK", "BULLOCK", "BULLOCK", "BULLOCK", "BULLOC~
$ `County ANSI`      <chr> "011", "011", "011", "011", "011", "011", "101", "1~
$ `Zip Code`         <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ Region             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ watershed_code     <chr> "00000000", "00000000", "00000000", "00000000", "00~
$ Watershed          <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
$ Commodity          <chr> "STRAWBERRIES", "STRAWBERRIES", "STRAWBERRIES", "ST~
$ `Data Item`        <chr> "STRAWBERRIES - ACRES BEARING", "STRAWBERRIES - ACR~
$ Domain             <chr> "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL~
$ `Domain Category`  <chr> "NOT SPECIFIED", "NOT SPECIFIED", "NOT SPECIFIED", ~
$ Value              <chr> "(D)", "3", "(D)", "1", "6", "5", "(D)", "(D)", "2"~
$ `CV (%)`           <chr> "(D)", "15.7", "(D)", "(L)", "52.7", "47.6", "(D)",~
```

I have 12699 rows and 21 columns.

All I can see from the glimpse is I have date, location, values and coefficients of variation.

Examine the data. How is it organized?

```
## Is every line associated with a state?

state_all <- strawberry |> distinct(State)

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

## every row is associated with a state

if(sum(state_all1$n) == dim(strawberry)[1]){print("Yes every row in the data is associated wi
```

```
[1] "Yes every row in the data is associated with a state."
```

```
## rm(state_all, state_all1)
```

remove columns with a single value in all rows

```
#|label: function def - drop 1-item columns

drop_one_value_col <- function(df){   ## takes whole dataframe
```

```
drop <- NULL

## test each column for a single value
for(i in 1:dim(df)[2]){
if((df |> distinct(df[,i]) |> count()) == 1){
drop = c(drop, i)
} }

## report the result -- names of columns dropped
## consider using the column content for labels
## or headers

if(is.null(drop)){return("none")}else{

   print("Columns dropped:")
   print(colnames(df)[drop])
   strawberry <- df[, -1*drop]
   }
}


## use the function

strawberry <- drop_one_value_col(strawberry)
```

```
[1] "Columns dropped:"
[1] "Week Ending"    "Zip Code"       "Region"         "watershed_code"
[5] "Watershed"      "Commodity"
```

```
drop_one_value_col(strawberry)
```

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

To get better look at the data, look at California.

```
calif <- strawberry |>  filter(State=="CALIFORNIA")

## look at the unique values in the "Program" column

## in the consol
```

```
## unique(calif$Program)

## and look at the data selection widget on
##      https://quickstats.nass.usda.gov

## You can see that CENSUS AND SURVEY are the two sources
## of data. (Why? What's the differences?).  So, let's see
## they differ.

calif_census <- calif |> filter(Program=="CENSUS")

calif_survey  <- calif |>  filter(Program=="SURVEY")

###


##calif_survey <- strawberry |> select(Year, Period, `Data Item`, Value)
```

Explore California to understand the census and survey

```
## no assignment -- just exploring

drop_one_value_col(calif_census)
```

```
[1] "Columns dropped:"
[1] "Program"    "Period"     "State"      "State ANSI"
```

```
drop_one_value_col(calif_survey)
```

```
[1] "Columns dropped:"
[1] "Program"          "Geo Level"        "State"            "State ANSI"
[5] "Ag District"      "Ag District Code" "County"           "County ANSI"
[9] "CV (%)"
```

Conclusions from California data exploration.

Now return to the entire data set.

take the lessons learned by examinging the California data

Two strategies – columns first, rows first

Split the census data from the survey data. drop single value columns

separate composite columns

`Data Item` into (fruit, category, item)

```
#|label: split Data Item

  strawberry <- strawberry |>
  separate_wider_delim(  cols = `Data Item`,
                         delim = ",",
                         names = c("Fruit",
                                 "Category",
                                 "Item",
                                 "Metric"),
                         too_many = "error",
                         too_few = "align_start"
                       )

## Use too_many and too_few to set up the separation operation.
```

There is a problem you have to fix – a leading space.

```
#|label: fix the leading space

 # note
strawberry$Category[1]
```

```
[1] NA
```

```
# strawberry$Item[2]
# strawberry$Metric[6]
# strawberry$Domain[1]
##
```

```
## trim white space

strawberry$Category <- str_trim(strawberry$Category, side = "both")
strawberry$Item <- str_trim(strawberry$Item, side = "both")
strawberry$Metric <- str_trim(strawberry$Metric, side = "both")
```

now exam the Fruit column – find hidden sub-columns

```
unique(strawberry$Fruit)
```

```
 [1] "STRAWBERRIES - ACRES BEARING"
 [2] "STRAWBERRIES - ACRES GROWN"
 [3] "STRAWBERRIES - ACRES NON-BEARING"
 [4] "STRAWBERRIES - OPERATIONS WITH AREA BEARING"
 [5] "STRAWBERRIES - OPERATIONS WITH AREA GROWN"
 [6] "STRAWBERRIES - OPERATIONS WITH AREA NON-BEARING"
 [7] "STRAWBERRIES"
 [8] "STRAWBERRIES - PRICE RECEIVED"
 [9] "STRAWBERRIES - ACRES HARVESTED"
[10] "STRAWBERRIES - ACRES PLANTED"
[11] "STRAWBERRIES - PRODUCTION"
[12] "STRAWBERRIES - YIELD"
[13] "STRAWBERRIES - APPLICATIONS"
[14] "STRAWBERRIES - TREATED"
```

```
## generate a list of rows with the production and price information

spr <- which((strawberry$Fruit=="STRAWBERRIES - PRODUCTION") | (strawberry$Fruit=="STRAWBERRI


strw_prod_price <- strawberry |> slice(spr)


## this has the census data, too

strw_chem <- strawberry |> slice(-1*spr)  ## too soon
```

now examine the rest of the columns

Which ones need to be split?

split sales and chemicals into two dataframes

(do this last after separating rows into separate data frames) (THEN rename the columns to correspond the analysis being done with the data frames)

```
#|label: split srawberry into census and survey pieces

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

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

nrow(strawberry) == (nrow(strw_b_chem) + nrow(strw_b_sales))
```

```
[1] TRUE
```

```
## Move marketing-related rows in strw_b_chem
## to strw_b_sales
```

plots

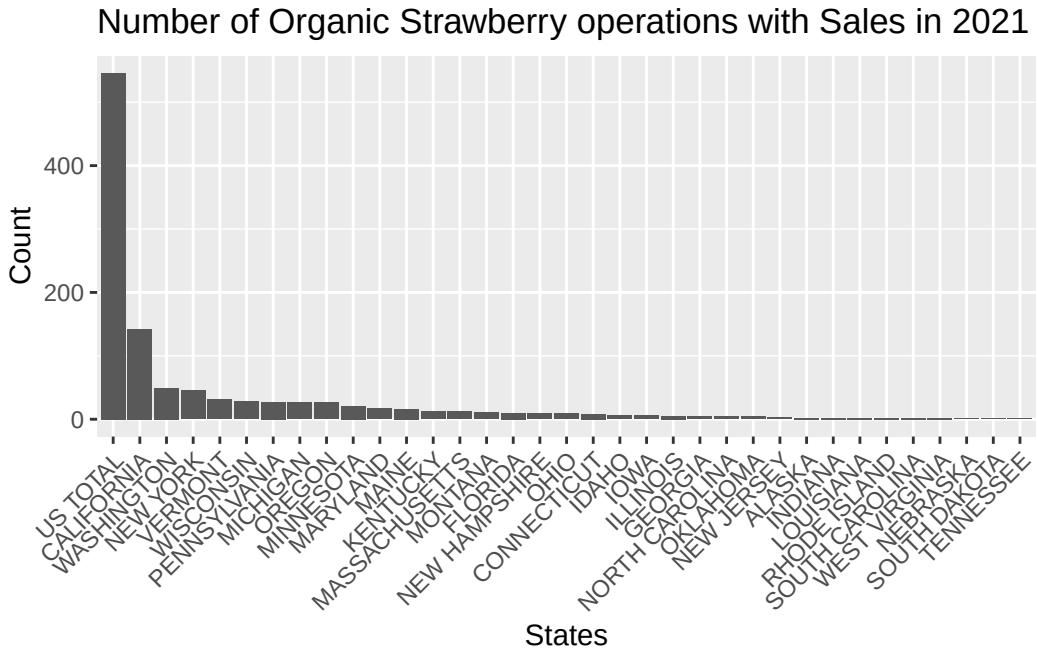
```
#|label: plot 1

plot1_data <- strawberry |>
  select(c(Year, State, Category, Value)) |>
  filter((Year == 2021) & (Category == "ORGANIC - OPERATIONS WITH SALES"))

plot1_data$Value <- as.numeric(plot1_data$Value)

plot1_data <- plot1_data |> arrange(desc(Value))

ggplot(plot1_data, aes(x=reorder(State, -Value), y=Value)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x=element_text(angle=45,hjust=1)) +
  labs(x = "States", y = "Count",
title ="Number of Organic Strawberry operations with Sales in 2021")
```



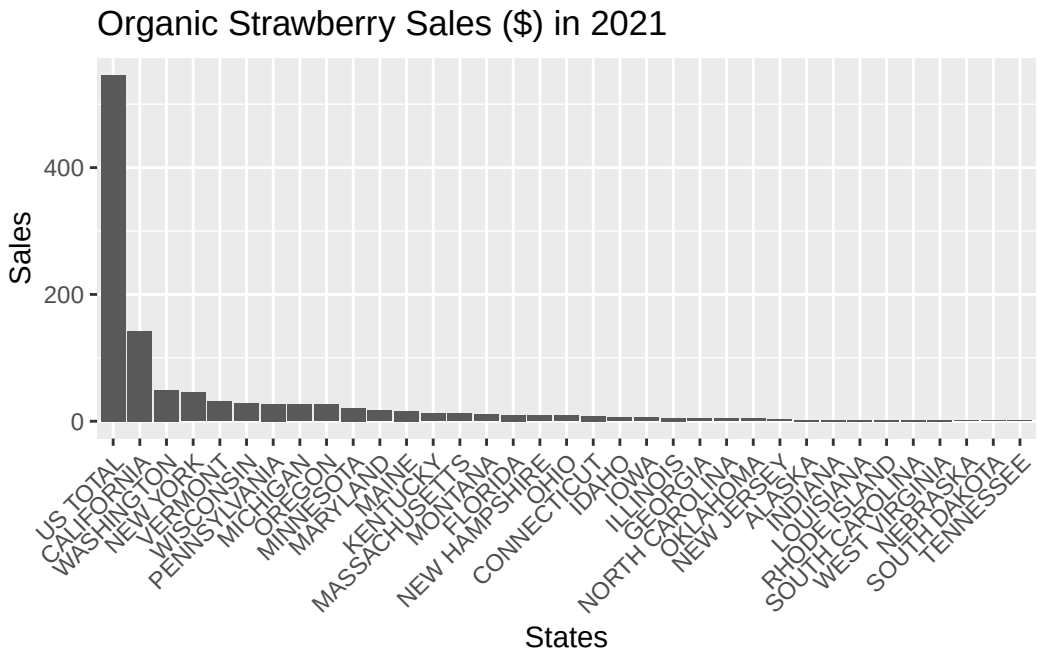

```
## plot 2

plot2_data <- strawberry |>
  select(c(Year, State, Category, Item, Value)) |>
  filter((Year == 2021) &
           (Category == "ORGANIC - SALES") &
           (Item == "MEASURED IN $") &
           (Value != "(D)"))


plot2_data$Value <- as.numeric(gsub(",", "", plot2_data$Value))

plot2_data <- plot1_data |> arrange(desc(Value))

ggplot(plot2_data, aes(x=reorder(State, -Value), y=Value)) +
  geom_bar(stat = "identity") +
  theme(axis.text.x=element_text(angle=45,hjust=1)) +
  labs(x = "States", y = "Sales",
title ="Organic Strawberry Sales ($) in 2021")
```



Further Cleaning

```
# remove "STRAWBERRIES" and any remaining dashes or spaces from the 'Fruit' column
strawberry$Fruit <- str_replace_all(strawberry$Fruit, "STRAWBERRIES|- ", "") %>%
  str_trim()

head(strawberry$Fruit)
```

```
[1] "ACRES BEARING"                    "ACRES GROWN"
[3] "ACRES NON-BEARING"                "OPERATIONS WITH AREA BEARING"
[5] "OPERATIONS WITH AREA GROWN"       "OPERATIONS WITH AREA NON-BEARING"
```

```
# separate "TOTAL" from the "Domain" column
strawberry$Total <- ifelse(str_detect(strawberry$Domain, "TOTAL"),
                           "TOTAL", NA)

# replace "TOTAL" in the "Domain" column with NA
strawberry$Domain <- ifelse(str_detect(strawberry$Domain,
                                       "TOTAL"), NA, strawberry$Domain)
```

```r
# separate "SURVEY" from the "Program" column
strawberry$Survey <- ifelse(str_detect(strawberry$Program,
                                       "SURVEY"), "SURVEY", NA)

# replace "SURVEY" in the "Program" column with NA
strawberry$Program <- ifelse(str_detect(strawberry$Program,
                                        "SURVEY"), NA,
                             strawberry$Program)

# rearrange columns so "Survey" is next to "Program" and "Total" is next to "Domain"
strawberry <- strawberry %>%
  relocate(Survey, .after = Program) %>%
  relocate(Total, .after = Domain)

# rename the "Program" column to "Census"
strawberry <- strawberry %>%
  rename(Census = Program)
```

```r
# replace non-numeric characters with NA in "Value" and "CV (%)" columns
strawberry$Value <- ifelse(str_detect(
  strawberry$Value, "^[0-9]+(\\.[0-9]+)?$"),
  strawberry$Value, NA)
strawberry$`CV (%)` <- ifelse(str_detect(
  strawberry$`CV (%)`, "^[0-9]+(\\.[0-9]+)?$"),
  strawberry$`CV (%)`, NA)
```

```r
strawberry <- strawberry %>%
  mutate(
    # extract the use
    Use = str_extract(`Domain Category`, "(?<=, )[A-Za-z]+"),

    # extract the name
    Name = str_extract(`Domain Category`, "(?<=\\()[^=]+"),

    # extract the code
    Code = str_extract(`Domain Category`, "(?<=\\= )[0-9]+")
  ) %>%

  # replace the original ones with NA
  mutate(`Domain Category` = ifelse(!is.na(Use) |
                                    !is.na(Name) |
                                    !is.na(Code), NA,
```

```
                                    `Domain Category`))

# We extracted the chemical use, name, and code from the "Domain Category" column. After that

write.csv(strawberry, "cleaned_strawberries.csv", row.names = FALSE)
```