Strawberries Dataset EDA

Suheng Yao

2024-09-30

Read and Explore the Data

```
library(knitr)
library(kableExtra)
library(tidyverse)
library(ggplot2)
library(tidyr)
library(stringr)
library(dplyr)
```

We first read in the data and have an overview of the data:

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

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`       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
$ `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`      <dbl> 11, 11, 11, 11, 11, 11, 101, 101, 101, 101, 119, 11~
$ `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     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
$ 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)",~
```

We got 12699 rows and 21 columns.

Data Cleaning

We can check for NA values and the columns with exact same values:

```
# Identify columns with only NA values and remove them
na_num <- colSums(is.na(strawberry))
print(na_num)
```

```
         Program             Year           Period      Week Ending
               0                0                0            12669
       Geo Level            State       State ANSI      Ag District
               0                0              264             5359
Ag District Code           County      County ANSI         Zip Code
            5359             5359             5385            12669
          Region   watershed_code        Watershed        Commodity
           12669                0            12669                0
       Data Item           Domain  Domain Category            Value
               0                0                0                0
          CV (%)
            3965
```

```
strawberry <- strawberry[, na_num < nrow(strawberry)]
```

Since Week Ending, Zip Code, Region and Watershed contain only NA values, those four columns have to be removed.

```r
# Identify columns with exact same values and remove them
col_same <- sapply(strawberry, function(x) length(unique(x)) == 1)
print(col_same)
```

```
         Program             Year           Period        Geo Level
           FALSE            FALSE            FALSE            FALSE
           State       State ANSI      Ag District Ag District Code
           FALSE            FALSE            FALSE            FALSE
          County      County ANSI   watershed_code        Commodity
           FALSE            FALSE             TRUE             TRUE
       Data Item           Domain  Domain Category            Value
           FALSE            FALSE            FALSE            FALSE
          CV (%)
           FALSE
```

```r
strawberry <- strawberry[, !col_same]
```

Since watershed_code and Commodity are the columns with the same values, they also have to be removed.

```r
glimpse(strawberry)
```

```
Rows: 12,669
Columns: 15
$ 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~
$ `Geo Level`        <chr> "COUNTY", "COUNTY", "COUNTY", "COUNTY", "COUNTY", "~
$ State              <chr> "ALABAMA", "ALABAMA", "ALABAMA", "ALABAMA", "ALABAM~
$ `State ANSI`       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
$ `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`      <dbl> 11, 11, 11, 11, 11, 11, 101, 101, 101, 101, 119, 11~
$ `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)",~
```

Now, there are 12669 rows and 15 variables.

Since the "State ANSI", "County ANSI", "Value" and "CV (%)" are all numeric values, but they were treated as character values in this data, it is necessary to convert them into numeric value:

```
# convert two columns to numeric
strawberry$`State ANSI` <- as.numeric(strawberry$`State ANSI`)
strawberry$`County ANSI` <- as.numeric(strawberry$`County ANSI`)
strawberry$Value <- as.numeric(strawberry$Value)
strawberry <- strawberry %>%
  mutate(`CV (%)` = case_when(
    `CV (%)` == "(H)" ~ 99.5,
    `CV (%)` == "(L)" ~ 0.05,
    `CV (%)` == "(D)" ~ NA_real_,
    TRUE ~ as.numeric(`CV (%)`) # convert remaining values to numeric values
  ))
strawberry$`CV (%)` <- as.numeric(strawberry$`CV (%)`)
```

Since the "Geo Level", "Ag District", "State" and "County" are all good indicators of the group, it is better to convert them into factor data type:

```
# convert four columns to factor type
cols_convert <- c("Geo Level", "State", "Ag District")
strawberry[cols_convert] <- lapply(strawberry[cols_convert], as.factor)
```

Examine The Structure of the Data

```
## Check if every row is associated with a state

state_all <- strawberry |> distinct(State)

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

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

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

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

Separate Composite Columns

Since variable "Data Item" contains too much information, we plan to separate the information in this column into a few new columns:

```
# Step1: Split the Data Item variable first based on commas
strawberry <- strawberry %>%
  separate(`Data Item`, into = c("Fruit", "Category1", "Category2"),
           sep = ",", fill = "right", extra = "merge")

# Step2: Split the Fruit column based on hyphen
strawberry <- strawberry %>%
  separate(Fruit, into = c("Fruit", "Operation"),
           sep = "-", fill = "right", extra = "merge")

# Step3: Separate Category1 column based on hyphen
strawberry <- strawberry %>%
  separate(Category1, into = c("Product Type", "Product Details"),
           sep = "-", fill = "right", extra = "merge")

# Step4: Separate Product Type column into Product Type and Measurement
strawberry <- strawberry %>%
  mutate(Measurement = ifelse(str_detect(
    `Product Type`, "MEASURED IN"),
    `Product Type`, NA_character_),
    `Product Type` = ifelse(str_detect(
      `Product Type`, "MEASURED IN"),
      NA_character_, `Product Type`))

# Step5: Separate Category2 column based on hyphen
strawberry <- strawberry %>%
  separate(Category2, into = c("Product Type1", "Product Details1"),
           sep = "-", fill = "right", extra = "merge")

# Step6: Split the Product Type1 column to create another new column Measurement1
strawberry <- strawberry %>%
  separate(`Product Type1`, into = c("Product Type1", "Measurement1"),
           sep = ",", fill = "right", extra = "merge")
```

```r
# Step7: Put the measurement information in Product Type1 into Measurement column
strawberry <- strawberry %>%
  mutate(Measurement = ifelse(str_detect(
    `Product Type1`, "MEASURED IN| AVG"),
    `Product Type1`, Measurement),
    `Product Type1` = ifelse(str_detect(
      `Product Type1`, "MEASURED IN| AVG"),
      NA_character_, `Product Type1`))

# Step8: Split the Product Details1 column based on comma
strawberry <- strawberry %>%
  separate(`Product Details1`, into = c("Product Details1", "Measurement2"),
           sep = ",", fill = "right", extra = "merge")

# Step9: Combine measurement2, measurement1 and measurement
strawberry <- strawberry %>%
  mutate(
    Measurement = coalesce(Measurement, Measurement1, Measurement2)
  ) %>%
  select(-Measurement1, -Measurement2)

# Step10: Combine Product Type, Product Type1, Product Details, Product Details1
strawberry <- strawberry %>%
  mutate(
    `Product Type` = coalesce(`Product Type`, `Product Type1`)
  ) %>%
  select(-`Product Type1`)

strawberry <- strawberry %>%
  mutate(
    `Product Details` = coalesce(`Product Details`, `Product Details1`)
  ) %>%
  select(-`Product Details1`)

# Step11: Trim the leading space
strawberry$Fruit <- str_trim(strawberry$Fruit, side = "both")
strawberry$Operation <- str_trim(strawberry$Operation, side = "both")
strawberry$`Product Type` <- str_trim(strawberry$`Product Type`,
                                      side = "both")
strawberry$`Product Details` <- str_trim(strawberry$`Product Details`,
                                         side = "both")
strawberry$Measurement <- str_trim(strawberry$Measurement,
```

```
                                        side = "both")
```

The “Data Item” column has been splitted into “Fruit”, “Operation”, “Product Type”, “Product Details” and “Measurement”.

Now, since the Domain and Domain Category columns have redundant and complex information, we consider split and combine the information in those two columns:

```
# Step1: Split the Domain Category based on comma
strawberry <- strawberry %>%
  separate(`Domain Category`, into = c("Domain1", "Category1"),
           sep = ",", fill = "right", extra = "merge")

# Step2: Focus on Domain1 column first, split it based on colon
strawberry <- strawberry %>%
  separate(Domain1, into = c("Domain2", "Category2"),
           sep = ":", fill = "right", extra = "merge")

# Step3: Combine the Domain and Domain2 column
strawberry <- strawberry %>%
  mutate(
    Domain = coalesce(Domain, Domain2)
  ) %>%
  select(-Domain2)

# Step4: Split Category2 column into three column
strawberry <- strawberry %>%
  mutate(
    Acre_Info = case_when(
      # Contains numbers --> Acre info
      grepl("\\d+\\.?\\d+", Category2) ~ Category2,
      TRUE ~ NA_character_
    ),
    Organic_Status = case_when(
      # "NOP USDA CERTIFIED" --> Organic Status
      grepl("\\(NOP USDA CERTIFIED\\)", Category2) ~ Category2,
      TRUE ~ NA_character_
    ),
    Fertilizer_Status = case_when(
      # Rest are Fertilizer info
      !grepl("\\d+\\.?\\d+", Category2) &
        !grepl("\\(NOP USDA CERTIFIED\\)", Category2)
```

```
      ~ Category2,
      TRUE ~ NA_character_
    )
  ) %>%
  select(-Category2)

# Step5: Split Category1 column into two columns based on colon
strawberry <- strawberry %>%
  separate(Category1, into = c("Domain1", "Category1"),
           sep = ":", fill = "right", extra = "merge")

# Step6: Combine the Domain1 column with Domain column
strawberry <- strawberry %>%
  mutate(
    Domain = coalesce(Domain, Domain1)
  ) %>%
  select(-Domain1)

# Step7: Split the Category1 column into two new columns
strawberry <- strawberry %>%
  # Remove the parentheses in each entry
  mutate(Category1 = gsub("\\(|\\)", "", Category1)) %>%
  separate(Category1,
           into = c("Chemical_Info", "Chemical_Product_Number"),
           sep = "=", extra = "merge") %>%
  # Put the new columns next to Domain Column
  relocate(Chemical_Info, Chemical_Product_Number, .after = Domain)

# Step8: Trim the leading space in all newly created columns
strawberry$Domain <- str_trim(strawberry$Domain, side = "both")

strawberry$Chemical_Info <- str_trim(strawberry$Chemical_Info, side = "both")

strawberry$Chemical_Product_Number <-
  str_trim(strawberry$Chemical_Product_Number,
           side = "both")

strawberry$Acre_Info <- str_trim(strawberry$Acre_Info, side = "both")

strawberry$Organic_Status <- str_trim(strawberry$Organic_Status, side = "both")

strawberry$Fertilizer_Status<- str_trim(strawberry$Fertilizer_Status, side = "both")
```

Output the final CSV file

```
write.csv(strawberry, "strawberry_cleaned.csv")
```