Stawberries: exploratory data analysis

Yingmai Chen

2023-10-11

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

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.1     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.2     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter()     masks stats::filter()
x dplyr::group_rows() masks kableExtra::group_rows()
x dplyr::lag()        masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(stringr)
```

Read the file

```
strawberry <- read_csv("strawberry_oct4.csv", col_names = TRUE)
```

Data cleaning

```
drop_one_value_col <- function(df){
  drop <- NULL
  for (i in 1:ncol(df)){
    unique_count <- n_distinct(df[, i])
    if (unique_count == 1){
      drop <- c(drop, i)
    }
  }

  if (length(drop) == 0) {
    print("No columns to drop.")
    return(df)
  } else {
    cat("Columns dropped:", colnames(df)[drop], "\n")
    strawberry <- df[, -drop]
    return(strawberry)
  }
}
str <- drop_one_value_col(strawberry)
```

```
[1] "No columns to drop."
```

```
str <- str$col_name
```

```
Warning: Unknown or uninitialised column: `col_name`.
```

```
strawberry <- strawberry |> select(!all_of(str))


vals=strawberry$Value
vals=sub(",","",vals)
vals=sub('"',"",vals)
vals=as.numeric(vals)
```

```
Warning: NAs introduced by coercion
```

```
strawberry["Value"]=vals

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

strawberry_census <- strawberry |> filter((Program=="CENSUS"))
 strawberry_census <- strawberry_census |>
  separate_wider_delim(  cols = `Data Item`,
                         delim = ",",
                         names = c("Fruit",
                                 "temp1",
                                 "temp2",
                                 "temp3"),
                         too_many = "error",
                         too_few = "align_start"
                       )

strawberry_census <- strawberry_census |>
  separate_wider_delim(  cols = temp1,
                         delim = " - ",
                         names = c("crop_type",
                                 "prop_acct"),
                         too_many = "error",
                         too_few = "align_start"
                       )



strawberry_census$crop_type <- str_trim(strawberry_census$crop_type, side = "both")

strawberry_census$temp2 <- str_trim(strawberry_census$temp2, side = "both")

strawberry_census$temp3 <- str_trim(strawberry_census$temp3, side = "both")
strawberry_census <- strawberry_census |> mutate(`Fresh Market` = temp2, .after = temp2)
strawberry_census$`Fresh Market` <- strawberry_census$`Fresh Market` |> str_replace( "^MEA
strawberry_census$`Fresh Market` <- strawberry_census$`Fresh Market` |> str_replace( "^P.*
strawberry_census$`Fresh Market`[is.na(strawberry_census$`Fresh Market`)] <- ""
 strawberry_census$temp2 <- strawberry_census$temp2 |> str_replace("^F.*", "")
strawberry_census$`Fresh Market` <- strawberry_census$`Fresh Market` |> str_replace("^FRES
strawberry_census <- strawberry_census |> mutate(`Process Market` = temp2, .after = temp2)
strawberry_census$`Process Market` <-  strawberry_census$`Process Market` |> str_replace("
strawberry_census$`Process Market`[is.na(strawberry_census$`Process Market`)] <- ""
```

```
strawberry_census$temp2 <- strawberry_census$temp2 |> str_replace("^P.*", "")
strawberry_census$`Process Market` <-  strawberry_census$`Process Market` |> str_replace("
strawberry_census$prop_acct[is.na(strawberry_census$prop_acct)] <- ""
strawberry_census$temp2[is.na(strawberry_census$temp2)] <- ""
strawberry_census$temp3[is.na(strawberry_census$temp3)] <- ""
strawberry_census <- strawberry_census |> unite(temp2, temp3, col = "Metric", sep = "")
strawberry_census$Metric <- strawberry_census$Metric |> str_replace("MEASURED IN ", "")
strawberry_census <- strawberry_census |> relocate(Metric, .before = Domain)

strawberry_census <- strawberry_census |> relocate(`Process Market`, .before = Metric)

strawberry_census <- strawberry_census |> rename(Totals = prop_acct)
```

EDA

Once the data has been cleaned and organized, you must conduct your own EDA. Be sure to include a discussion of your analysis of the chemical information, including citations for data and other information you have used. Visualizations should play a key role in your analysis. Plots should be labeled and captioned.

Dealing with Missing Values, Outliers, and Duplicates

EDA

##data analysis

```
strawberry_census_dollar <- strawberry_census %>%
  filter(!is.na(Value) & (Metric == "$"))

top_10_states_dollar <- strawberry_census_dollar %>%
  group_by(State) %>%
  summarise(avg_value = mean(Value, na.rm = TRUE)) %>%
  arrange(desc(avg_value)) %>%
  top_n(10)
```

```
Selecting by avg_value
```

```
strawberry_census_CWT <- strawberry_census %>%
  filter(!is.na(Value) & (Metric == "CWT"))
```

```r
top_10_states_CWT <- strawberry_census_CWT %>%
  group_by(State) %>%
  summarise(avg_value = mean(Value, na.rm = TRUE)) %>%
  arrange(desc(avg_value)) %>%
  top_n(10)
```

```
Selecting by avg_value
```

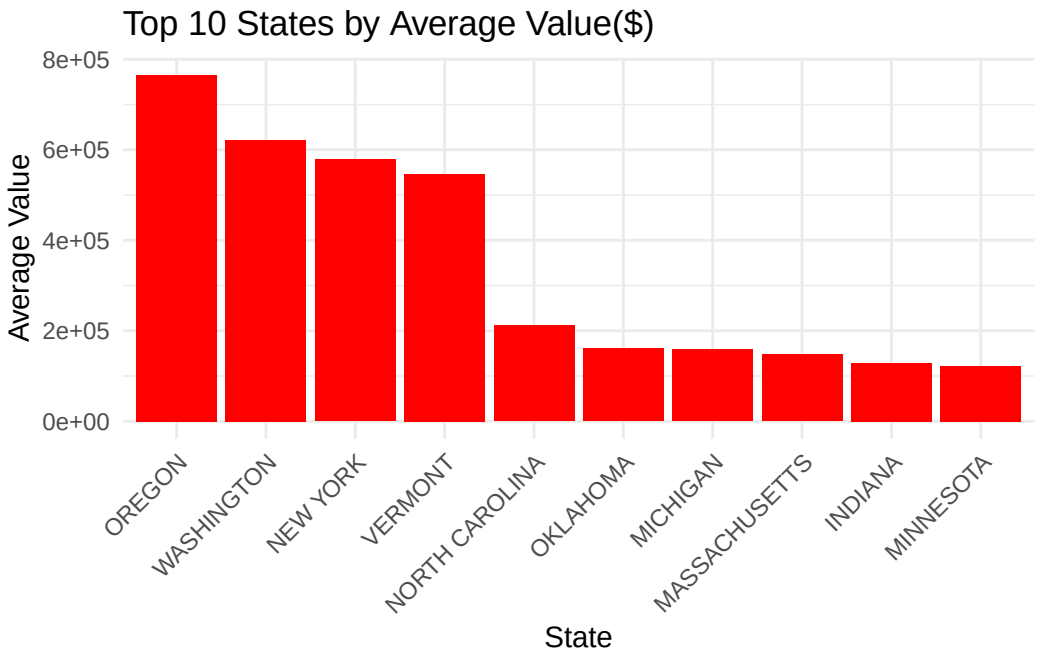
```r
strawberry_census_OWS <- strawberry_census %>%
  filter(!is.na(Value) & (Totals == "OPERATIONS WITH SALES" | 'Fresh Market' == "OPERATION

top_10_states_OWS <- strawberry_census_OWS %>%
  group_by(State) %>%
  summarise(avg_value = mean(Value, na.rm = TRUE)) %>%
  arrange(desc(avg_value)) %>%
  top_n(10)
```

```
Selecting by avg_value
```

```r
df_ows <- data.frame(State = top_10_states_OWS$State, Metric = "OWS", avg_value = top_10_s
df_cwt <- data.frame(State = top_10_states_CWT$State, Metric = "CWT", avg_value = top_10_s
df_dollar <- data.frame(State = top_10_states_dollar$State, Metric = "Dollar", avg_value =
common_states_data <- rbind(df_ows, df_cwt, df_dollar)
common_states <- intersect(top_10_states_OWS$State, intersect(top_10_states_dollar$State,
common_states_data <- common_states_data %>%
  filter(State %in% common_states)
```

1. Top 10 States by Average Value ($)

```r
gg_dollar <- ggplot(top_10_states_dollar, aes(x = reorder(State, -avg_value), y = avg_valu
  geom_bar(stat = "identity", fill = "red") +
  labs(title = "Top 10 States by Average Value($)",
       x = "State",
       y = "Average Value") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
print(gg_dollar)
```

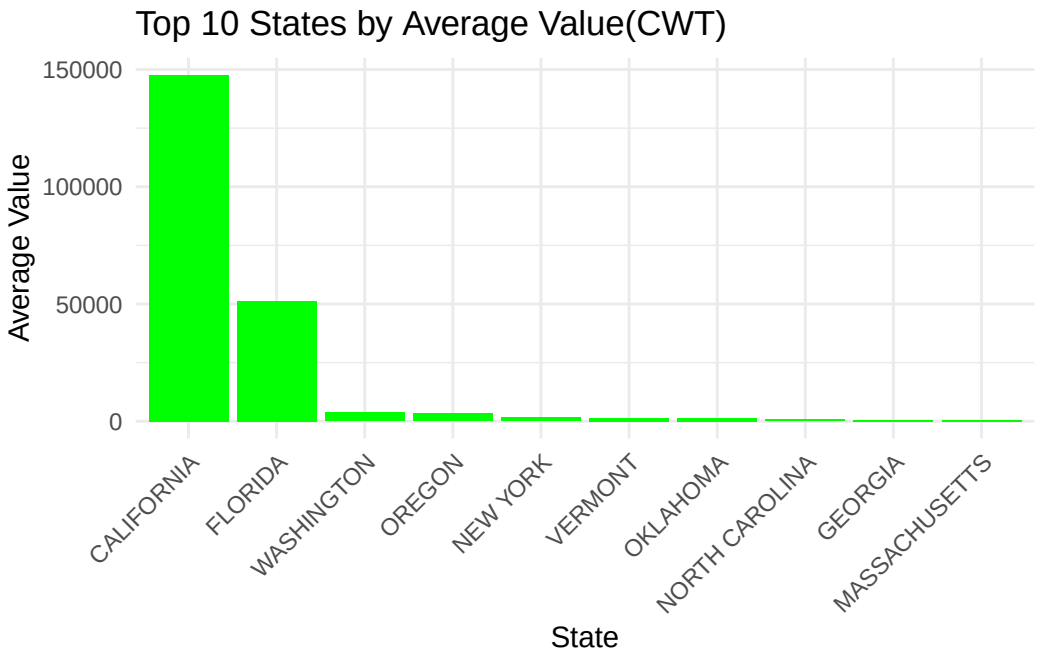



```
cat("The cities with the top 10 ave_sales are($):", top_10_states_dollar$State, "\n")
```

```
The cities with the top 10 ave_sales are($): OREGON WASHINGTON NEW YORK VERMONT NORTH CAROLI
```

2. Top 10 States by Average Value (CWT)

```
gg_cwt <- ggplot(top_10_states_CWT, aes(x = reorder(State, -avg_value), y = avg_value)) +
  geom_bar(stat = "identity", fill = "green") +
  labs(title = "Top 10 States by Average Value(CWT)",
       x = "State",
       y = "Average Value") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
print(gg_cwt)
```

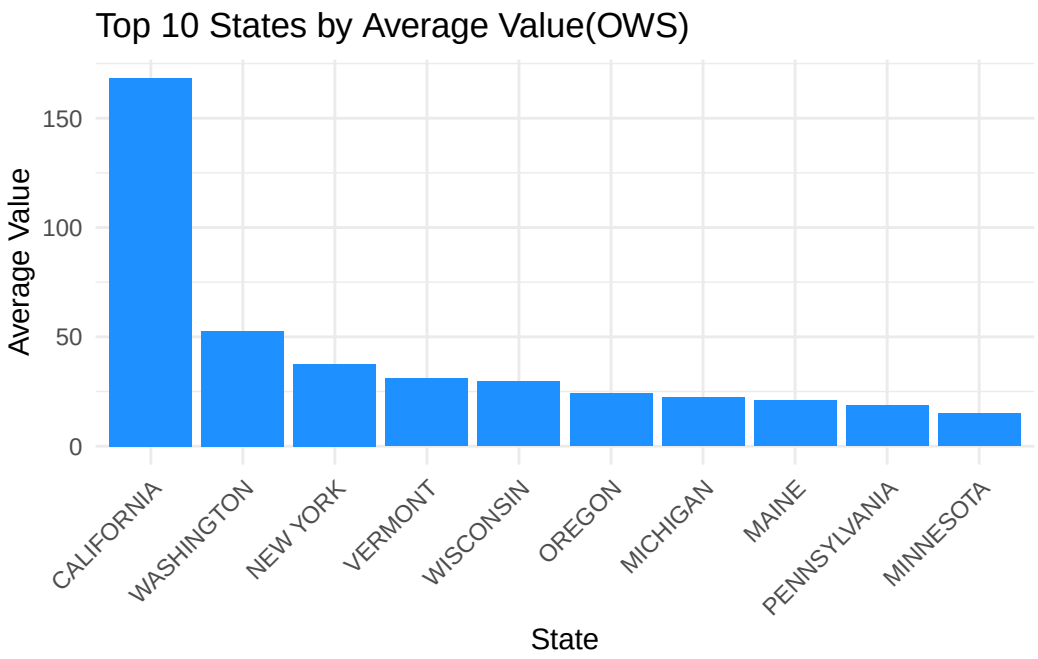



```
cat("The cities with the top 10 ave_sales are(CWT):", top_10_states_CWT$State, "\n")
```

```
The cities with the top 10 ave_sales are(CWT): CALIFORNIA FLORIDA WASHINGTON OREGON NEW YORK
```

3. Top 10 States by Average Value (OWS)

```
gg_ows <- ggplot(top_10_states_OWS, aes(x = reorder(State, -avg_value), y = avg_value)) +
  geom_bar(stat = "identity", fill = "dodgerblue") +
  labs(title = "Top 10 States by Average Value(OWS)",
       x = "State",
       y = "Average Value") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
print(gg_ows)
```

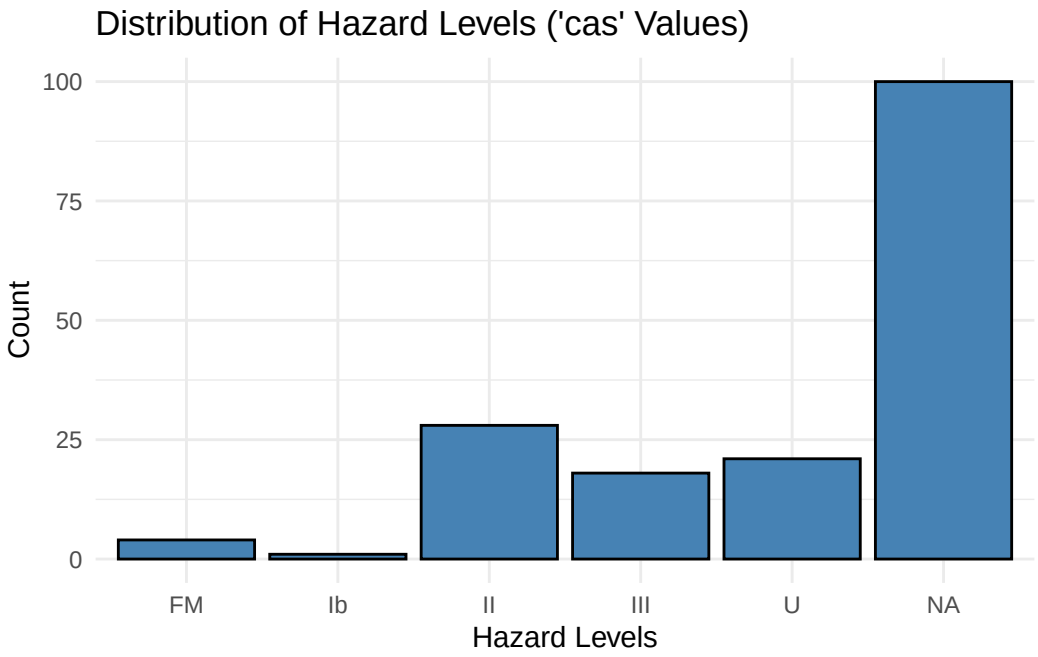



```
cat("The cities with the top 10 ave_sales are(OWS):", top_10_states_OWS$State, "\n")
```

```
The cities with the top 10 ave_sales are(OWS): CALIFORNIA WASHINGTON NEW YORK VERMONT WISCONS
```

5.harzard based on cas value

```
library(ggplot2)

data <- read.csv("merage.csv")

ggplot(data, aes(x=cas)) +
  geom_bar(fill="steelblue", color="black") +
  labs(title="Distribution of Hazard Levels ('cas' Values)", x="Hazard Levels", y="Count")
  theme_minimal()
```


Distribution of Hazard Levels ('cas' Values)
Count
100
75
50
25
0
FM
Ib
II
III
U
NA
Hazard Levels