HW5

Songyu Tang

Fall 2024

read and explore the data

Set-up

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.

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"
```

```
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`       <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~
## $ `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)",~
```

separate composite columns

Data Item into two columns

```
#|label: split Data Item

  strawberry <- strawberry |>
  separate_wider_delim(  cols = `Data Item`,
                         delim = "-",
                         names = c("column1",
                                   "column2"),
                         too_many = "merge",
                         too_few = "align_start"
                       )
```

```
strawberry <- strawberry |>
  separate_wider_delim(  cols = `column1`,
                         delim = ",",
                         names = c("Fruit",
                                   "Category"),
                         too_many = "merge",
                         too_few = "align_start"
                       )
strawberry$Fruit <- str_trim(strawberry$Fruit, side = "both")
strawberry$Category <- str_trim(strawberry$Category, side = "both")
strawberry$column2 <- str_trim(strawberry$column2, side = "both")
strawberry <- drop_one_value_col(strawberry)
```

```
## [1] "Columns dropped:"
## [1] "Fruit"
```

```
unique(strawberry$Category)
```

```
##  [1] NA                       "ORGANIC"                "ORGANIC, FRESH MARKET"
##  [4] "ORGANIC, PROCESSING"    "FRESH MARKET"           "PROCESSING"
##  [7] "FRESH MARKET, UTILIZED" "NOT SOLD"               "PROCESSING, UTILIZED"
## [10] "UTILIZED"               "BEARING"
```

Next, we want to set string in the Category into different columns. According to the standard from nass, we put fresh market and processing into “Marketing Channels”, put organic into “Method”, put bearing into “Class”, put Utilized into “utilization”, put not sold into “Measurement”.

```
#|label: Clean data in the Category
strawberry <- strawberry %>%
  mutate(Marketing_channels = ifelse(str_detect(Category, "FRESH MARKET|PROCESSING"),
                                     str_extract(Category, "FRESH MARKET|PROCESSING"), NA)) %>%
  mutate(Utilizations = ifelse(str_detect(Category, "UTILIZED"),
                               str_extract(Category, "UTILIZED"), NA)) %>%
  mutate(Method = ifelse(str_detect(Category, "ORGANIC"),
                         str_extract(Category, "ORGANIC"), NA)) %>%
  mutate(Class = ifelse(str_detect(Category, "BEARING"),
                        str_extract(Category, "BEARING"), NA)) %>%
  mutate(Measurement = ifelse(str_detect(Category, "NOT SOLD"),
                              str_extract(Category, "NOT SOLD"), NA)) %>%
  select(1:match("County ANSI", names(.)), Class, Method, Marketing_channels, Utilizations, Measurement
  select( -Category)
```

Then, we have to clean data in "column2".

```
strawberry <- strawberry %>%
  mutate(
    Measurement1 = ifelse(str_detect(column2, ","),
                          str_split_fixed(column2, ",", 2)[, 1],
                          column2),
    Metric1 = ifelse(str_detect(column2, ","),
                     str_split_fixed(column2, ",", 2)[, 2],
                     NA)
  ) %>%
  select(1:match("Measurement", names(.)), Measurement1, Metric1, everything()) %>%
  select( -column2)
```

Finally, we are going to classify all the string in their positions

```
strawberry <- strawberry %>%
  mutate(Metric = str_extract(Metric1, "(?<=,|^)[^,]*MEASURED IN[^,]*(?=,|$)")) %>%
  mutate(Remark = str_remove_all(Metric1, "(?<=,|^)[^,]*MEASURED IN[^,]*(,)?")) %>%
  mutate(Remark = str_trim(str_replace_all(Remark, "^,|,$|,,", ""))) %>%
  select(1:match("Measurement1", names(.)), Metric, Remark, everything())  %>%
  select(-Metric1)
strawberry <- strawberry %>%
  mutate(Category = ifelse(is.na(Measurement),
                           Measurement1,
                           paste(Measurement, Measurement1, sep = " ")
                           )) %>%
  select(1:match("Utilizations", names(.)), Category, everything()) %>%
  select(-Measurement,-Measurement1)
strawberry$Metric <- str_trim(strawberry$Metric, side = "both")
strawberry$Category <- str_trim(strawberry$Category, side = "both")
strawberry <- strawberry %>% mutate(across(Remark, ~ na_if(., "")))
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~
## $ `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~
## $ Class               <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Method              <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Marketing_channels  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Utilizations        <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Category            <chr> "ACRES BEARING", "ACRES GROWN", "ACRES NON-BEARING"~
## $ Metric              <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Remark              <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ 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)",~
```

Seperate Domain and Domain Category

In both of sections, we find that expect TOTAL in Domain and NOT SPECIFIED in 'Domain Category', other string in two sections have a high similarity, like AREA GROWN in 'Domain' is the same the front character of 'AREA GROWN: (0.1 TO 0.9 ACRES)' in 'Domain Category'. Therefore, we want to split 'Domain Category' and delete the '()' and the same character in 'Domain'

```
strawberry <- strawberry %>%
  mutate(`Domain Category` = ifelse(str_detect(`Domain Category`, ":"),
                                    str_split_fixed(`Domain Category`, ":", 2)[, 2],
                                    `Domain Category`)) %>%
  mutate(`Domain Category` = str_replace_all(`Domain Category`, "[\\(\\)]", ""))
strawberry$`Domain Category` <- str_trim(strawberry$`Domain Category`, side = "both")
```

Then, we want separate data in 'Domain Category' into the specific chemical and the numbers

```
strawberry <- strawberry %>%
  mutate(Chemical_Number = ifelse(str_detect(`Domain Category`, "="),
                                  str_split_fixed(`Domain Category`, "=", 2)[, 2],
                                  NA)) %>%
  mutate(`Domain Category` = ifelse(str_detect(`Domain Category`, "="),
                                    str_split_fixed(`Domain Category`, "=", 2)[, 1],
                                    `Domain Category`)) %>%
  select(1:match("Domain Category", names(.)), Chemical_Number, everything())
strawberry$`Domain Category` <- str_trim(strawberry$`Domain Category`, side = "both")
strawberry$Chemical_Number <- str_trim(strawberry$Chemical_Number, side = "both")
```

Finally, delete all 'CHEMICAL' in the 'Domain'

```
strawberry <- strawberry %>%
  mutate(Domain = str_replace(Domain, "CHEMICAL, ", ""))
glimpse(strawberry)
```

```
## Rows: 12,669
## Columns: 22
## $ 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`       <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~
## $ Class              <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Method             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Marketing_channels <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Utilizations       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Category           <chr> "ACRES BEARING", "ACRES GROWN", "ACRES NON-BEARING"~
## $ Metric             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Remark             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Domain             <chr> "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL~
## $ `Domain Category`  <chr> "NOT SPECIFIED", "NOT SPECIFIED", "NOT SPECIFIED", ~
## $ Chemical_Number    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Value              <chr> "(D)", "3", "(D)", "1", "6", "5", "(D)", "(D)", "2"~
## $ `CV (%)`           <chr> "(D)", "15.7", "(D)", "(L)", "52.7", "47.6", "(D)",~
```

Transfer data type into correct one

In 'strawberry', we find that 'Value' and 'CV(%)' columns are both string type, which is not correct for the numeric. With EDA, (D) means that data exists but not provided because of privacy, with some (D) we can just use sum to estimate every single area, some may need some investigation. In this case, we have to analyze different (D) under different circumstance. In this case, I just transfer it into NA Also, we transfer (L) into 0.05, (H) into 99.95, and (Z) into 0.0005 based on the Quick Stats Glossary.

```
strawberry <- strawberry %>%
  mutate(Value = ifelse(Value == "(D)", NA, Value)) %>%
  mutate(Value = ifelse(Value == "(NA)", NA, Value)) %>%
  mutate(Value = ifelse(Value == "(Z)", "0.0005", Value)) %>%
  mutate(Value = str_replace_all(Value,",","")) %>%
  mutate(Value = as.numeric(Value))
strawberry <- strawberry %>%
  mutate(`CV (%)` = ifelse(`CV (%)` == "(D)", NA, `CV (%)`)) %>%
  mutate(`CV (%)` = ifelse(`CV (%)` == "(NA)", NA, `CV (%)`)) %>%
  mutate(`CV (%)` = ifelse(`CV (%)` == "(L)", "0.05", `CV (%)`)) %>%
  mutate(`CV (%)` = ifelse(`CV (%)` == "(H)", "99.95", `CV (%)`)) %>%
  mutate(`CV (%)` = str_replace_all(`CV (%)`,",","")) %>%
  mutate(`CV (%)` = as.numeric(`CV (%)`))
glimpse(strawberry)
```

```
## Rows: 12,669
## Columns: 22
## $ 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`       <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~
## $ Class              <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Method             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Marketing_channels <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Utilizations       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Category           <chr> "ACRES BEARING", "ACRES GROWN", "ACRES NON-BEARING"~
## $ Metric             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Remark             <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Domain             <chr> "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL", "TOTAL~
## $ `Domain Category`  <chr> "NOT SPECIFIED", "NOT SPECIFIED", "NOT SPECIFIED", ~
## $ Chemical_Number    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ Value              <dbl> NA, 3, NA, 1, 6, 5, NA, NA, 2, 2, NA, NA, 2, 2, 1, ~
## $ `CV (%)`           <dbl> NA, 15.70, NA, 0.05, 52.70, 47.60, NA, NA, 55.70, 5~
```

Separate the data in certain category

```
#First we split the table in 'census' and 'survey'
strawberry_census <- strawberry %>% filter(Program == "CENSUS")
strawberry_survey <- strawberry %>% filter(Program == "SURVEY")
strawberry_census <- drop_one_value_col(strawberry_census)
```

```
## [1] "Columns dropped:"
## [1] "Program"         "Period"          "Class"           "Utilizations"
## [5] "Remark"          "Chemical_Number"
```

```
strawberry_survey <- drop_one_value_col(strawberry_survey)
```

```
## [1] "Columns dropped:"
## [1] "Program"          "Ag District"      "Ag District Code" "County"
## [5] "County ANSI"      "Method"           "CV (%)"
```

```
#Second we split the census table into organic and non-organic
strawberry_organic <- strawberry_census %>% filter(Domain == "ORGANIC STATUS")
strawberry_organic <- drop_one_value_col(strawberry_organic)
```

```
## [1] "Columns dropped:"
## [1] "Ag District"      "Ag District Code" "County"           "County ANSI"
## [5] "Method"           "Domain"           "Domain Category"
```

```
strawberry_non_organic <- strawberry_census %>% filter(!Domain == "ORGANIC STATUS")
strawberry_non_organic <- drop_one_value_col(strawberry_non_organic)
```

```
## [1] "Columns dropped:"
## [1] "Year"               "Method"             "Marketing_channels"
## [4] "Metric"
```

```r
#Third we split the survey table into chemical and non-chemical
strawberry_chemical <- strawberry_survey %>% filter(!Domain == "TOTAL")
strawberry_chemical <- drop_one_value_col(strawberry_chemical)
```

```
## [1] "Columns dropped:"
## [1] "Period"             "Geo Level"          "Marketing_channels"
## [4] "Utilizations"
```

```r
strawberry_non_chemical <- strawberry_survey %>% filter(Domain == "TOTAL")
strawberry_non_chemical <- drop_one_value_col(strawberry_non_chemical)
```

```
## [1] "Columns dropped:"
## [1] "Class"           "Domain"          "Domain Category" "Chemical_Number"
```

```r
#Now we have four table for organic, non-organic, chemical and non-chemical
```

Estimate the NA in Value and CV(%)

In this section, we are going to use linear regression to estimated the the NA in both columns.

```r
#First, we estimate the NA in the strawberry_organic table.
#Value
strawberry_organic1 <- strawberry_organic %>% mutate(original_order = row_number())
organic_with_value <- strawberry_organic1 %>% filter(!is.na(Value))
organic_missing_value <- strawberry_organic1 %>% filter(is.na(Value))
organicvaluemodel <- lm(log(Value) ~ factor(Year) + State + Category, data = organic_with_value)
organic_missing_value <- organic_missing_value %>%
  mutate(Value = round(exp(predict(organicvaluemodel, newdata = organic_missing_value)),1))
organic_filled <- bind_rows(organic_with_value, organic_missing_value)
organic_filled <- organic_filled %>% arrange(original_order)
#CV
organic_with_CV <- organic_filled %>% filter(!is.na(`CV (%)`))
organic_missing_CV <- organic_filled %>% filter(is.na(`CV (%)`))
organicCVmodel <- lm(`CV (%)` ~ factor(Year) + State + Category, data = organic_with_CV)
organic_missing_CV <- organic_missing_CV %>%
  mutate(`CV (%)` = round(predict(organicCVmodel, newdata = organic_missing_CV),2))
organic_full <- bind_rows(organic_with_CV, organic_missing_CV)
#In the original frame, there exists (L) and (H) to express the abnormal value so we just keep value gr
strawberry_organic <- organic_full %>% arrange(original_order) %>% select(-original_order) %>% mutate(V
```

```r
#Next, we estimate the NA in the strawberry_non_organic table.
#Value
strawberry_non_organic1 <- strawberry_non_organic %>% mutate(original_order = row_number())
non_organic_with_value <- strawberry_non_organic1 %>% filter(!is.na(Value))
non_organic_missing_value <- strawberry_non_organic1 %>% filter(is.na(Value))
non_organicvaluemodel <- lm(Value ~  State + Category + Domain + `Domain Category`, data = non_organic_
non_organic_missing_value <- non_organic_missing_value %>%
  mutate(Value = round(predict(non_organicvaluemodel, newdata = non_organic_missing_value),1))
non_organic_filled <- bind_rows(non_organic_with_value, non_organic_missing_value)
non_organic_filled <- non_organic_filled %>% arrange(original_order)
```

```
#CV
non_organic_with_CV <- non_organic_filled %>% filter(!is.na(`CV (%)`))
non_organic_missing_CV <- non_organic_filled %>% filter(is.na(`CV (%)`))
non_organicCVmodel <- lm(`CV (%)` ~ State + Category + Domain + `Domain Category`, data = non_organic_wi
non_organic_missing_CV <- non_organic_missing_CV %>%
  mutate(`CV (%)` = round(predict(non_organicCVmodel, newdata = non_organic_missing_CV),2))
non_organic_full <- bind_rows(non_organic_with_CV, non_organic_missing_CV)
strawberry_non_organic <- organic_full %>% arrange(original_order) %>% select(-original_order) %>% muta
```

```
#Then, we estimate the NA in the strawberry_chemical table. There is no CV columns in chemical table.
strawberry_chemical1 <- strawberry_chemical %>% mutate(original_order = row_number())
chemical_with_value <- strawberry_chemical1 %>% filter(!is.na(Value))
chemical_missing_value <- strawberry_chemical1 %>% filter(is.na(Value))
chemicalvaluemodel <- lm(log(Value) ~ factor(Year) + State + Category + Domain, data = chemical_with_val
chemical_missing_value <- chemical_missing_value %>%
  mutate(Value = round(exp(predict(chemicalvaluemodel, newdata = chemical_missing_value)),1))
chemical_filled <- bind_rows(chemical_with_value, chemical_missing_value)
strawberry_chemical <- chemical_filled %>% arrange(original_order) %>% select(-original_order) %>% muta
```

```
#Finally, we estimate the NA in the strawberry_non_chemical table. There is also no CV columns in non-c
strawberry_non_chemical1 <- strawberry_non_chemical %>% mutate(original_order = row_number())
non_chemical_with_value <- strawberry_non_chemical1 %>% filter(!is.na(Value))
non_chemical_missing_value <- strawberry_non_chemical1 %>% filter(is.na(Value))
non_chemicalvaluemodel <- lm(log(Value+1) ~ factor(Year) + State + Category + Period, data = non_chemica
non_chemical_missing_value <- non_chemical_missing_value %>%
  mutate(Value = round(exp(predict(non_chemicalvaluemodel, newdata = non_chemical_missing_value))-1,1))
non_chemical_filled <- bind_rows(non_chemical_with_value, non_chemical_missing_value)
strawberry_non_chemical <- non_chemical_filled %>% arrange(original_order) %>% select(-original_order)
```

Write in four new csvs

As we have done data cleaning, we want to get a new csv that contains all the data we have cleaned

```
write.csv(strawberry_organic, "Strawberry_organic.csv", row.names = FALSE)
write.csv(strawberry_non_organic, "Strawberry_non_organic.csv", row.names = FALSE)
write.csv(strawberry_chemical, "Strawberry_chemical.csv", row.names = FALSE)
write.csv(strawberry_non_chemical, "Strawberry_non_chemical.csv", row.names = FALSE)
```