QVI_case

Zhe GUAN

2024-06-19

Preparation for datasets

Two datasets are provided from forage link1 and forage link2.

LINK1 is transfromed to csv format by EXCEL first.

Settings

```
options(repos = "https://cran.rstudio.com/")
install.packages("tidyverse")
```

```
##
## The downloaded binary packages are in
##  /var/folders/pt/0nf2m3pj1f3b373wsyfc6glh0000gn/T//RtmprDNDLG/downloaded_packages
```

```
install.packages("dplyr")
```

```
##
## The downloaded binary packages are in
##  /var/folders/pt/0nf2m3pj1f3b373wsyfc6glh0000gn/T//RtmprDNDLG/downloaded_packages
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(lubridate)

transaction_data <- read_csv("/Users/zheguan/CWR_fig/QVI_case/QVI_transaction_data.csv")
```

```
## Rows: 264836 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): PROD_NAME
## dbl (7): DATE, STORE_NBR, LYLTY_CARD_NBR, TXN_ID, PROD_NBR, PROD_QTY, TOT_SALES
##
## 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.
```

```
purchanse_data <- read_csv("/Users/zheguan/CWR_fig/QVI_case/QVI_purchase_behaviour.csv")
```

```
## Rows: 72637 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): LIFESTAGE, PREMIUM_CUSTOMER
## dbl (1): LYLTY_CARD_NBR
##
## 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.
```

Review the data

LYLTY_CARD_NBR is Primary Key for two tables.

Clean the transaction data

change the decimal to date

```
transaction_data$DATE <- as_date(transaction_data$DATE,origin = "1899/12/30")
```

Check missing values for each colomn:

```
transaction_data %>%
  summarise(across(everything(),  ~ sum(is.na(.))))
```

```
## # A tibble: 1 x 8
##    DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_QTY TOT_SALES
##   <int>     <int>          <int>  <int>    <int>     <int>    <int>     <int>
## 1     0         0              0      0        0         0        0         0
```

No missing value for table transaction_data.

check if there is inconsistency for product price for single products with the same product name.

add a new coloum which indicates the unit_price(single_price) for different products.

```
test <- transaction_data %>%
  mutate(single_price = round(TOT_SALES/PROD_QTY,3)) %>%
  group_by(PROD_NAME,single_price) %>%
  summarise(n()) %>%
  arrange()
```

```
## `summarise()` has grouped output by 'PROD_NAME'. You can override using the
## `.groups` argument.
```

```
tibble(test)
```

```
## # A tibble: 134 x 3
##    PROD_NAME                              single_price `n()`
##    <chr>                                         <dbl> <int>
##  1 Burger Rings 220g                               2.3  1564
##  2 CCs Nacho Cheese    175g                        2.1  1498
##  3 CCs Original 175g                               2.1  1514
##  4 CCs Tasty Cheese    175g                        2.1  1539
##  5 Cheetos Chs & Bacon Balls 190g                  3.3  1479
##  6 Cheetos Puffs 165g                              2.8  1448
##  7 Cheezels Cheese 330g                            5.7  3149
##  8 Cheezels Cheese Box 125g                        2.1  1454
##  9 Cobs Popd Sea Salt  Chips 110g                  3.8  3265
## 10 Cobs Popd Sour Crm  &Chives Chips 110g          3.8  3159
## # i 124 more rows
```

We found that there are some discripency that some records have different unit/single price but belong to same categories/product. we checked them separately.

We first define a function which will return the mode value for different products transactions.

```
get_mode <- function(x) {
  uniqx <- unique(x)
  uniqx[which.max(tabulate(match(x, uniqx)))]
}
```

then check the mode prices for different products:

```
transaction_data_test <- transaction_data %>%
    mutate(single_price = round(TOT_SALES/PROD_QTY,3))
price_modes <- transaction_data_test %>%
  group_by(PROD_NAME) %>%
  summarise(price_mode = get_mode(single_price))
View(price_modes)
```

Here are some products which are sold without normal price, so we need to report them for data specialists to check the specific cases.

They are "Dorito Corn Chp Supreme 380g", "Doritos Corn Chips Cheese Supreme 170g", "Grain Waves Sweet Chilli 210g", "Grain Waves Sour Cream&Chives 210G","Infuzions BBQ Rib Prawn Crackers 110g","Kettle Chilli 175g","Kettle Original 175g","Kettle Sensations Camembert & Fig 150g", "Kettle Sweet Chilli And Sour Cream 175g", "Kettle Tortilla ChpsFeta&Garlic 150g", "Old El Paso Salsa Dip Chnky Tom Ht300g", "Old El Paso Salsa Dip Tomato Mild 300g", "Pringles Original Crisps 134g", "Pringles Sthrn FriedChicken 134g", "RRD SR Slow Rst Pork Belly 150g", "Red Rock Deli Thai Chilli&Lime 150g", "Smiths Crinkle Cut Chips Chicken 170g,"Thins Chips Salt & Vinegar 175g", "Tyrrells Crisps Ched & Chives 165g". We need to report these records for further checks to ensure they are normal cases.

Based on business questions, we only focus on the general cases where products were sold with original price. So we filter transaction_data_test based on the calculated mode:

```
transaction_data_test_filtered <- transaction_data_test %>%
  left_join(price_modes, by = "PROD_NAME") %>%
  filter(single_price >= price_mode)
```

Double Check:

```
test <- transaction_data_test_filtered %>%
  group_by(PROD_NAME,single_price) %>%
  summarise(n()) %>%
  arrange()
```

```
## `summarise()` has grouped output by 'PROD_NAME'. You can override using the
## `.groups` argument.
```

```
tibble(test)
```

```
## # A tibble: 114 x 3
##    PROD_NAME                              single_price `n()`
##    <chr>                                         <dbl> <int>
##  1 Burger Rings 220g                               2.3  1564
##  2 CCs Nacho Cheese    175g                        2.1  1498
##  3 CCs Original 175g                               2.1  1514
##  4 CCs Tasty Cheese    175g                        2.1  1539
##  5 Cheetos Chs & Bacon Balls 190g                  3.3  1479
##  6 Cheetos Puffs 165g                              2.8  1448
##  7 Cheezels Cheese 330g                            5.7  3149
##  8 Cheezels Cheese Box 125g                        2.1  1454
##  9 Cobs Popd Sea Salt  Chips 110g                  3.8  3265
## 10 Cobs Popd Sour Crm  &Chives Chips 110g          3.8  3159
## # i 104 more rows
```

Clean the purchase data

check the missing values:

```
purchanse_data %>% summarise(across(everything(), ~ sum(is.na(.))))
```

```
## # A tibble: 1 x 3
##   LYLTY_CARD_NBR LIFESTAGE PREMIUM_CUSTOMER
##            <int>     <int>            <int>
## 1              0         0                0
```

check duplicate values for card number:

```
purchanse_data %>%
  group_by(LYLTY_CARD_NBR) %>%
  summarise(n())
```

```
## # A tibble: 72,637 x 2
##    LYLTY_CARD_NBR `n()`
##             <dbl> <int>
##  1           1000     1
##  2           1002     1
##  3           1003     1
##  4           1004     1
##  5           1005     1
##  6           1007     1
##  7           1009     1
##  8           1010     1
##  9           1011     1
## 10           1012     1
## # i 72,627 more rows
```

or

```
purchanse_data %>%
  group_by(LYLTY_CARD_NBR) %>%
  summarise(number=n()) %>%
  filter(number != 1)
```

```
## # A tibble: 0 x 2
## # i 2 variables: LYLTY_CARD_NBR <dbl>, number <int>
```

we found that the card number type in the purchanse_data table is different from transaction table, we need to transfer one of them.

```
purchase_data_cleaned <- purchanse_data %>%
  mutate(LYLTY_CARD_NBR = as.integer(LYLTY_CARD_NBR))
View(purchase_data_cleaned)
```

link the purchase table and transction table

```
total_table <- transaction_data_test_filtered %>%
  left_join(purchase_data_cleaned, by = "LYLTY_CARD_NBR")
View(total_table)
```

define the metrics

By investigate the database, we found that we can compare different patterns from different levels: total_sales/total_quantity vs. lifestage/premium_customer/date.

first lifestage versus total sales and quantities.

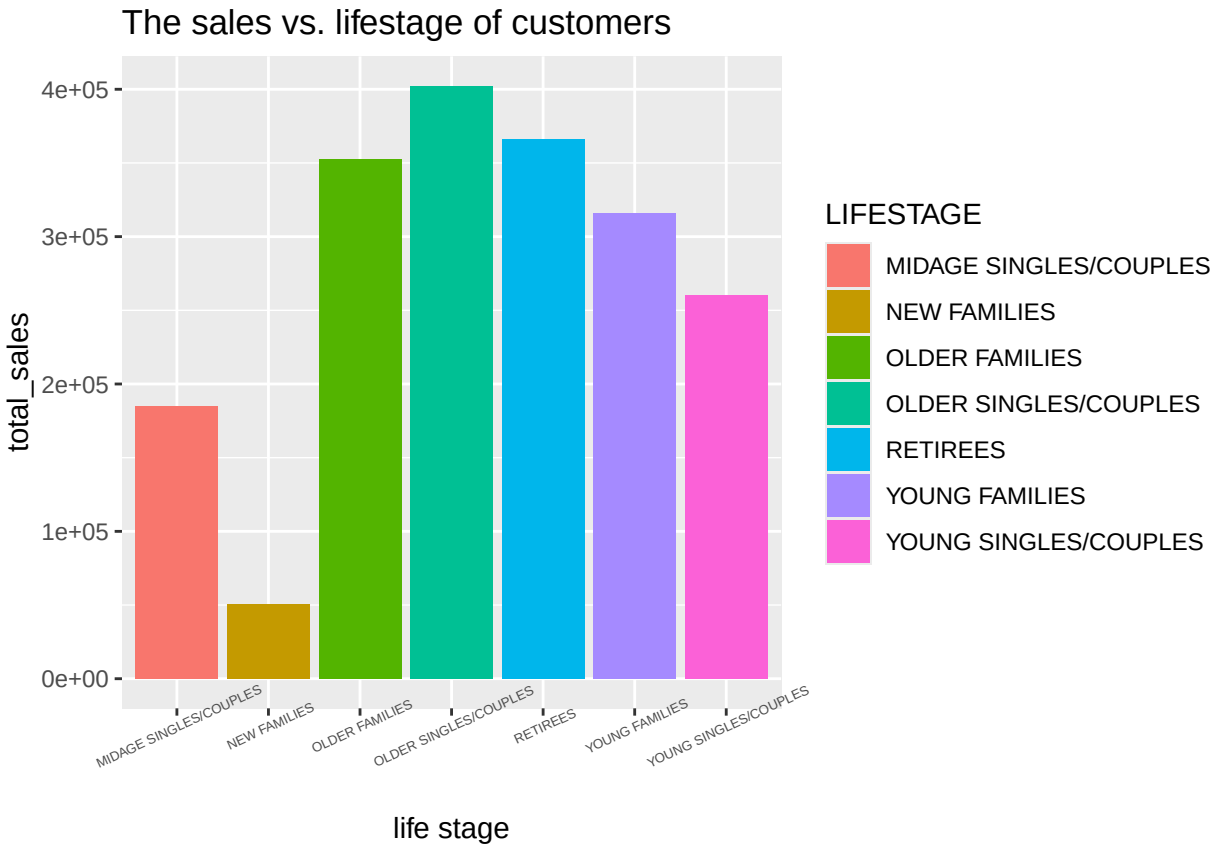
```
summary_table_lifestage <- total_table %>%
  group_by(LIFESTAGE) %>%
  summarise(total_sales = sum(TOT_SALES),total_quantity = sum(PROD_QTY),average_sales=sum(TOT_SALES)/n()
tibble(summary_table_lifestage)
```

```
## # A tibble: 7 x 4
##   LIFESTAGE              total_sales total_quantity average_sales
##   <chr>                        <dbl>          <dbl>         <dbl>
## 1 MIDAGE SINGLES/COUPLES     184664.          47691          7.36
## 2 NEW FAMILIES                50391.          12834          7.29
## 3 OLDER FAMILIES             352370.          94559          7.25
## 4 OLDER SINGLES/COUPLES      402249.         104144          7.39
## 5 RETIREES                   366310.          94113          7.37
## 6 YOUNG FAMILIES             316029.          84512          7.25
## 7 YOUNG SINGLES/COUPLES      260321.          66605          7.16
```

in the first step, we can find the **New family** generally tends to buy less chips compared to other kinds of families and **Older families** and **Retirees** tend to buy more chips compared to others. We can recommend stakeholders to focus on these two kinds of groups to organise possible promotional campaigns. Need to mention that Midage Singles and couples are also possible customers with higher average spendings on chips.

More details can be viewed in plot:

```
ggplot(data = summary_table_lifestage, aes(x = LIFESTAGE, y = total_sales,  fill = LIFESTAGE ) ) + geom_
```



similarily, we can make more plots for different ranks of customers, and in this case, we will focus on the average sales:

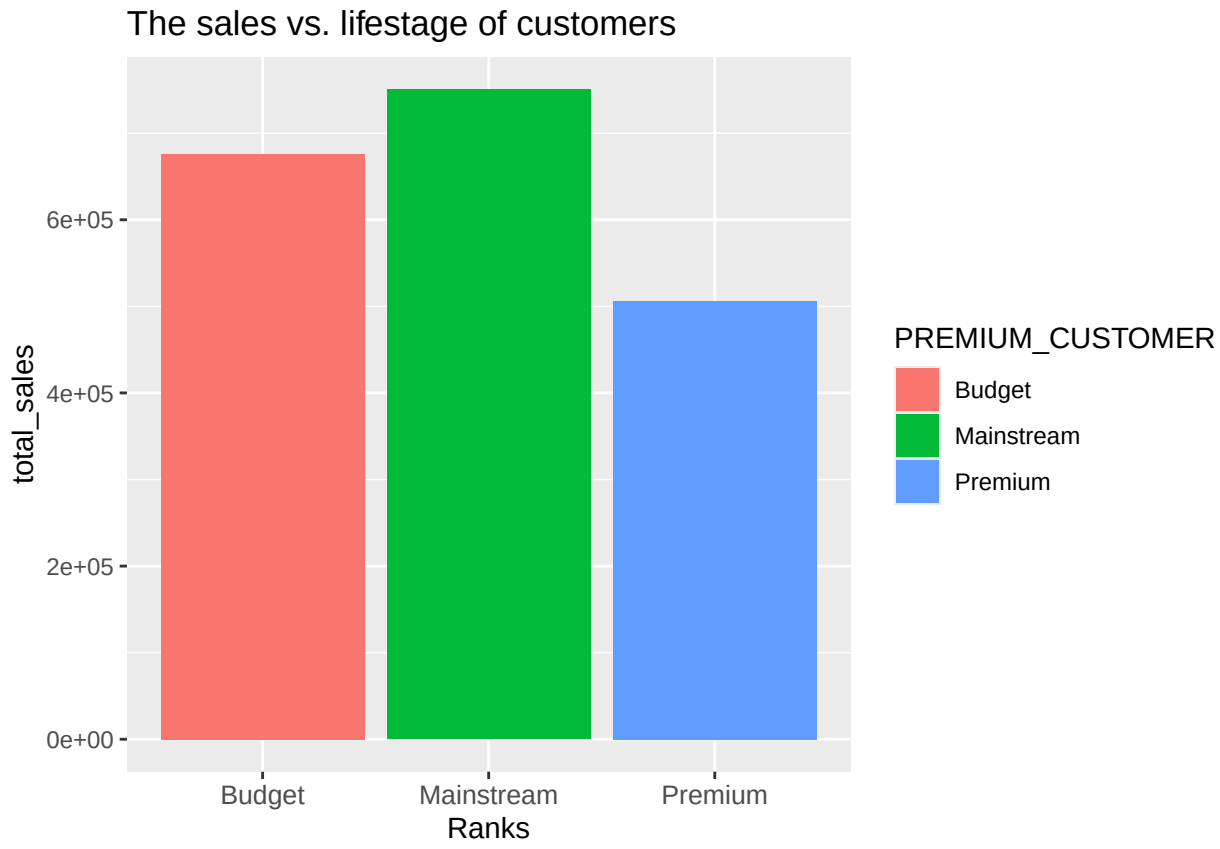
```
summary_table_premium <- total_table %>%
  group_by(PREMIUM_CUSTOMER) %>%
```

```
  summarise(total_sales = sum(TOT_SALES),total_quantity = sum(PROD_QTY), average_sales = sum(TOT_SALES),
tibble(summary_table_premium)
```

```
## # A tibble: 3 x 4
##   PREMIUM_CUSTOMER total_sales total_quantity average_sales
##   <chr>                 <dbl>          <dbl>         <dbl>
## 1 Budget               675960.         177813          7.26
## 2 Mainstream           750420.         193859          7.36
## 3 Premium              505955.         132786          7.26
```

More details can be viewed in plot:

```
ggplot(data = summary_table_premium, aes(x = PREMIUM_CUSTOMER, y = total_sales,  fill = PREMIUM_CUSTOMER
```



just look at the total sales, **Mainstream** contributes most and next one is Budget. Budget is the next one, and Premium contribute least. However, if we look at the average sales, there is no big difference between Budget and Premium. So we can suggest stakeholders focus on the Mainstream market.

Next business question is : Which chip kind is the most popular?

First, we grouped the dataset based on product_name/numbers:

```
total_table_chips <- total_table %>%
  group_by(PROD_NBR,PREMIUM_CUSTOMER,PROD_NAME) %>%
  summarise(total_quantity=sum(PROD_QTY),total_sales=sum(TOT_SALES),average_sales=sum(TOT_SALES)/sum(PR
```

```
## `summarise()` has grouped output by 'PROD_NBR', 'PREMIUM_CUSTOMER'. You can
## override using the `.groups` argument.
```

```
tibble(total_table_chips)
```

```
## # A tibble: 342 x 6
##    PROD_NBR PREMIUM_CUSTOMER PROD_NAME  total_quantity total_sales average_sales
##       <dbl> <chr>            <chr>               <dbl>       <dbl>         <dbl>
##  1        1 Budget           Smiths Cr~           1061       3077.          2.90
##  2        1 Mainstream       Smiths Cr~            999       2897.          2.90
##  3        1 Premium          Smiths Cr~            742       2152.          2.90
##  4        2 Budget           Cobs Popd~           2001       7604.          3.80
##  5        2 Mainstream       Cobs Popd~           2451       9314.          3.80
##  6        2 Premium          Cobs Popd~           1586       6027.          3.80
##  7        3 Budget           Kettle Se~           2084       9586.          4.60
##  8        3 Mainstream       Kettle Se~           2461      11321.          4.60
##  9        3 Premium          Kettle Se~           1607       7392.          4.60
## 10        4 Budget           Dorito Co~           1978      12857           6.5
## # i 332 more rows
```

Then check them in three different premium of customers:

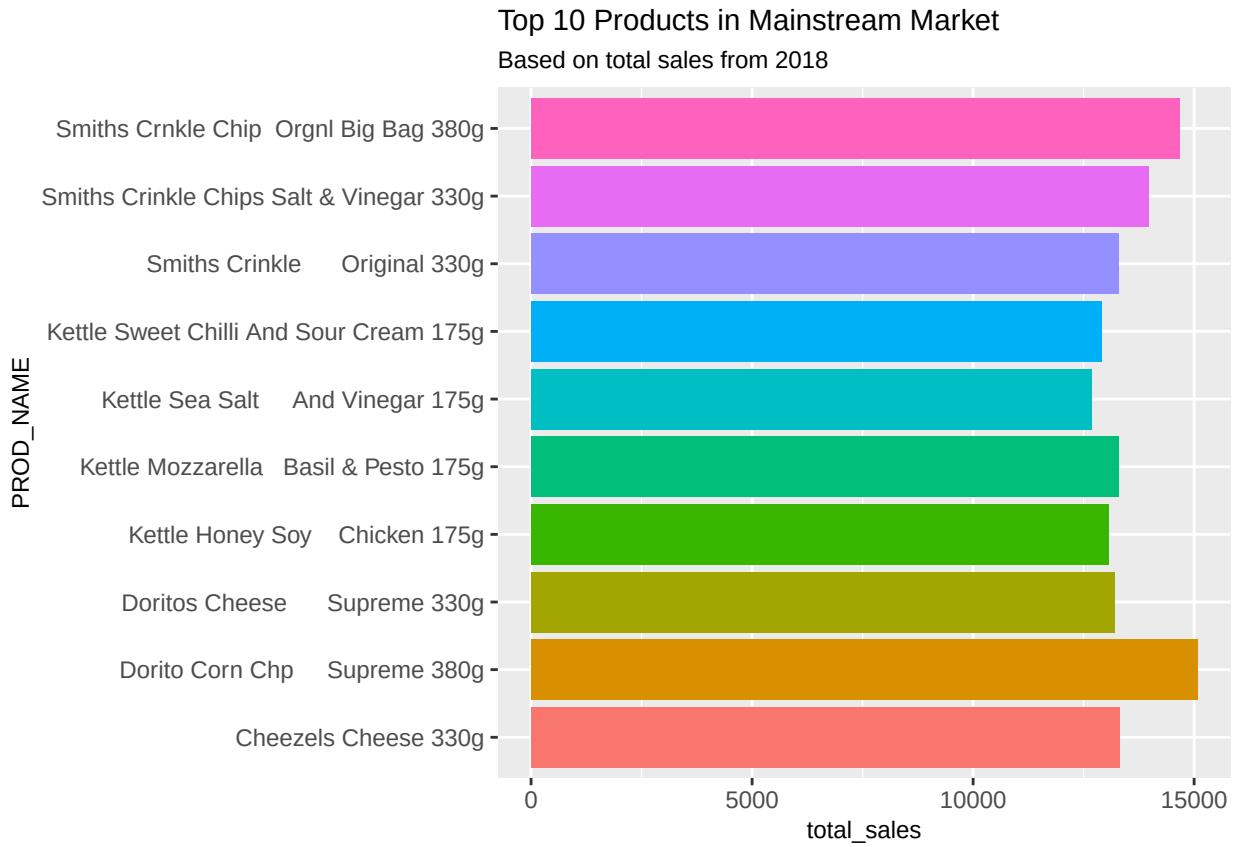
for MainStream:

```
total_table_chips_mainstream <- total_table_chips %>%
  filter(PREMIUM_CUSTOMER == "Mainstream") %>%
  arrange(desc(total_sales)) %>% head(10)
tibble(total_table_chips_mainstream)
```

```
## # A tibble: 10 x 6
##    PROD_NBR PREMIUM_CUSTOMER PROD_NAME  total_quantity total_sales average_sales
##       <dbl> <chr>            <chr>               <dbl>       <dbl>         <dbl>
##  1        4 Mainstream       Dorito Co~           2318      15067           6.5
##  2       14 Mainstream       Smiths Cr~           2487      14673.          5.90
##  3       16 Mainstream       Smiths Cr~           2450      13965.          5.70
##  4       23 Mainstream       Cheezels ~           2337      13321.          5.70
##  5      102 Mainstream       Kettle Mo~           2463      13300.          5.40
##  6        7 Mainstream       Smiths Cr~           2330      13281.          5.70
##  7       20 Mainstream       Doritos C~           2314      13190.          5.70
##  8       88 Mainstream       Kettle Ho~           2421      13073.          5.40
##  9       89 Mainstream       Kettle Sw~           2392      12917.          5.40
## 10       32 Mainstream       Kettle Se~           2348      12679.          5.40
```

```
ggplot(total_table_chips_mainstream) + geom_col(mapping = aes(x=total_sales,y = PROD_NAME ,fill = PROD_
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

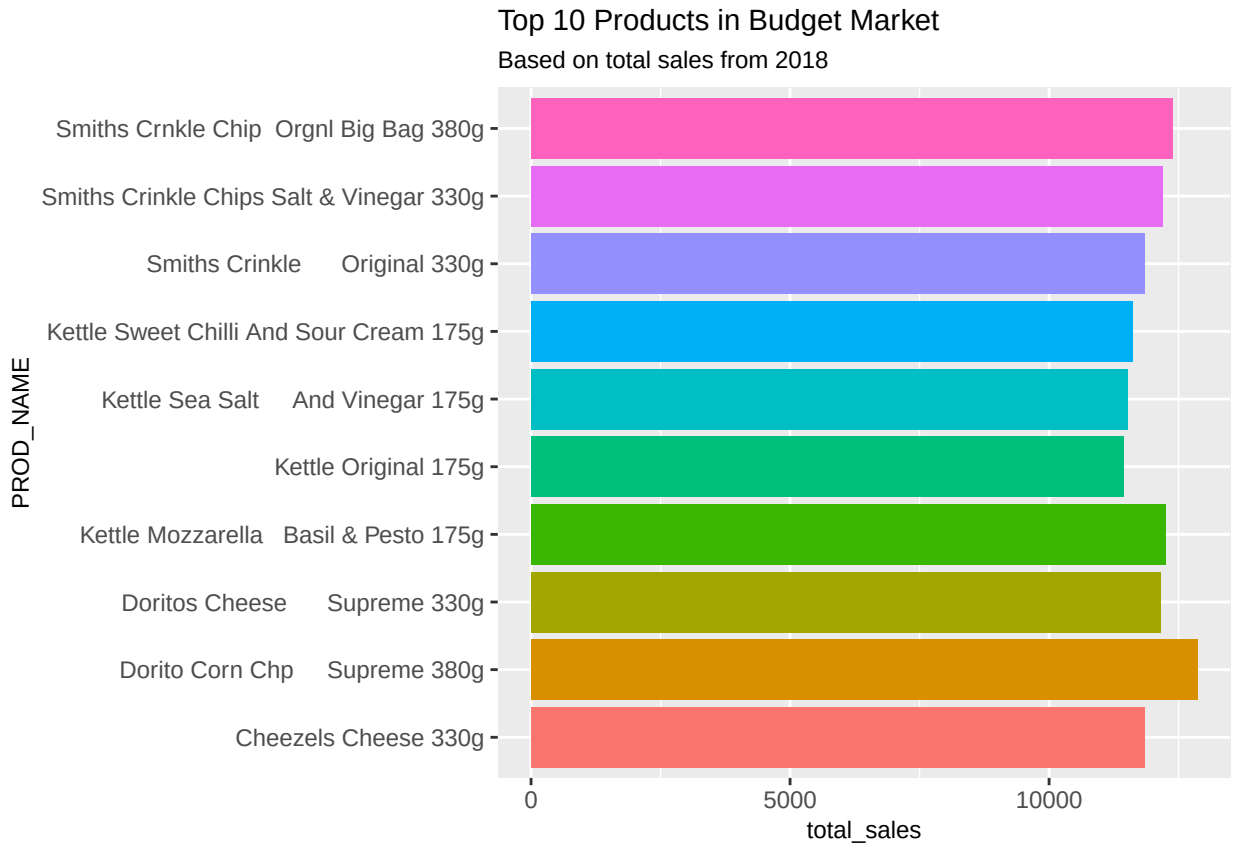


For Budget:

```
total_table_chips_budget <- total_table_chips %>%
  filter(PREMIUM_CUSTOMER == "Budget") %>%
  arrange(desc(total_sales)) %>% head(10)
tibble(total_table_chips_budget)
```

```
## # A tibble: 10 x 6
##    PROD_NBR PREMIUM_CUSTOMER PROD_NAME  total_quantity total_sales average_sales
##       <dbl> <chr>            <chr>               <dbl>       <dbl>         <dbl>
##  1        4 Budget           Dorito Co~           1978      12857           6.5
##  2       14 Budget           Smiths Cr~           2099      12384.          5.90
##  3      102 Budget           Kettle Mo~           2267      12242.          5.40
##  4       16 Budget           Smiths Cr~           2139      12192.          5.70
##  5       20 Budget           Doritos C~           2132      12152.          5.70
##  6       23 Budget           Cheezels ~           2079      11850.          5.70
##  7        7 Budget           Smiths Cr~           2079      11850.          5.70
##  8       89 Budget           Kettle Sw~           2152      11621.          5.40
##  9       32 Budget           Kettle Se~           2132      11513.          5.40
## 10       46 Budget           Kettle Or~           2118      11437.          5.40
```

```
ggplot(total_table_chips_budget) + geom_col(mapping = aes(x=total_sales,y = PROD_NAME ,fill = PROD_NAME)
```

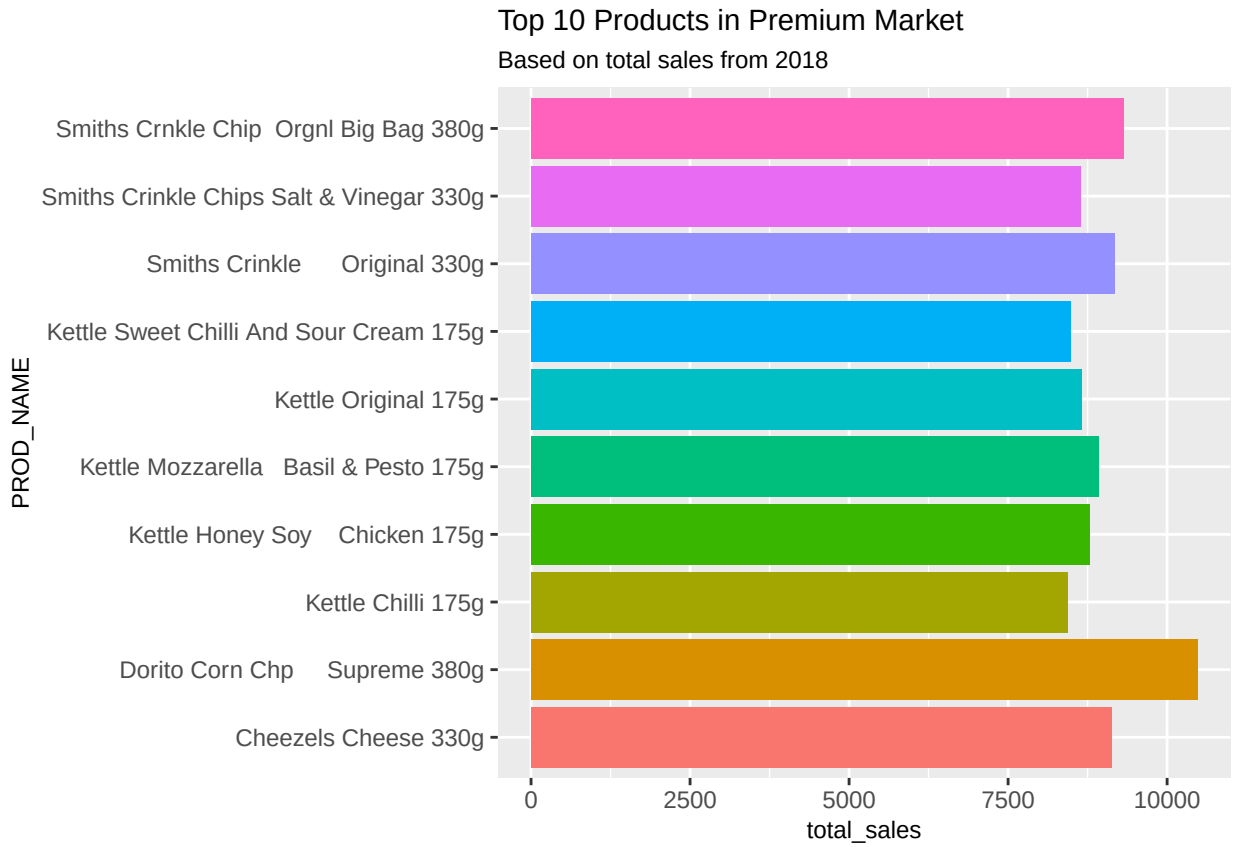


For Premium:

```
total_table_chips_premium <- total_table_chips %>%
  filter(PREMIUM_CUSTOMER == "Premium") %>%
  arrange(desc(total_sales)) %>%
  head(10)
tibble(total_table_chips_premium)
```

```
## # A tibble: 10 x 6
##    PROD_NBR PREMIUM_CUSTOMER PROD_NAME  total_quantity total_sales average_sales
##       <dbl> <chr>            <chr>               <dbl>       <dbl>         <dbl>
##  1        4 Premium          Dorito Co~           1611      10472.          6.5
##  2       14 Premium          Smiths Cr~           1578       9310.          5.9
##  3        7 Premium          Smiths Cr~           1609       9171.          5.70
##  4       23 Premium          Cheezels ~           1601       9126.          5.70
##  5      102 Premium          Kettle Mo~           1651       8915.          5.40
##  6       88 Premium          Kettle Ho~           1625       8775.          5.40
##  7       46 Premium          Kettle Or~           1604       8662.          5.40
##  8       16 Premium          Smiths Cr~           1517       8647.          5.70
##  9       89 Premium          Kettle Sw~           1571       8483.          5.40
## 10       36 Premium          Kettle Ch~           1563       8440.          5.40
```

```
ggplot(total_table_chips_premium) + geom_col(mapping = aes(x=total_sales,y = PROD_NAME ,fill = PROD_NAM
```



We can found the most popular product in three groups is **Dorto Corn Chp Supreme 380g**. Perhaps its some characters make it stand out. And other products can be improved based on it.

We also investigate the Old Singles/Couples have their own preferences because they contribute most in total sales.

```
total_table_chips_life <- total_table %>%
  group_by(PROD_NBR,LIFESTAGE,PROD_NAME) %>%
  summarise(total_quantity=sum(PROD_QTY),total_sales=sum(TOT_SALES),average_sales=sum(TOT_SALES)/sum(PR
```

```
## `summarise()` has grouped output by 'PROD_NBR', 'LIFESTAGE'. You can override
## using the `.groups` argument.
```

```
tibble(total_table_chips_life)
```

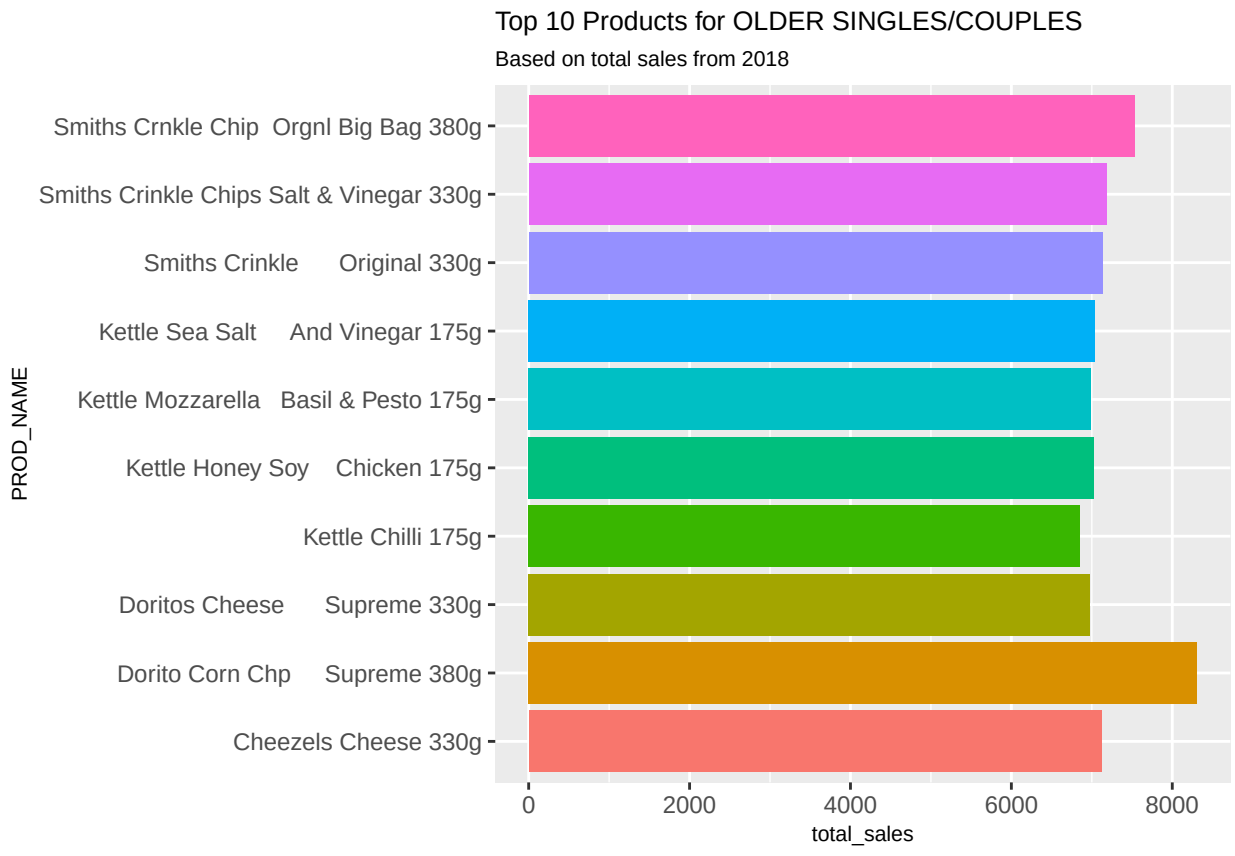
```
## # A tibble: 798 x 6
##    PROD_NBR LIFESTAGE         PROD_NAME total_quantity total_sales average_sales
##       <dbl> <chr>             <chr>              <dbl>       <dbl>         <dbl>
##  1        1 MIDAGE SINGLES/C~ Smiths C~            220        638.          2.90
##  2        1 NEW FAMILIES      Smiths C~             63        183.          2.9
##  3        1 OLDER FAMILIES    Smiths C~            687       1992.          2.90
##  4        1 OLDER SINGLES/CO~ Smiths C~            535       1551.          2.90
##  5        1 RETIREES          Smiths C~            447       1296.          2.90
##  6        1 YOUNG FAMILIES    Smiths C~            528       1531.          2.90
##  7        1 YOUNG SINGLES/CO~ Smiths C~            322        934.          2.90
```

```
##  8        2 MIDAGE SINGLES/C~ Cobs Pop~            623       2367.          3.80
##  9        2 NEW FAMILIES      Cobs Pop~            185        703.          3.80
## 10        2 OLDER FAMILIES    Cobs Pop~           1090       4142.          3.80
## # i 788 more rows
```

```
total_table_chips_premium <- total_table_chips_life %>%
  filter(LIFESTAGE == "OLDER SINGLES/COUPLES") %>%
  arrange(desc(total_sales)) %>%
  head(10)
tibble(total_table_chips_premium)
```

```
## # A tibble: 10 x 6
##    PROD_NBR LIFESTAGE         PROD_NAME total_quantity total_sales average_sales
##       <dbl> <chr>             <chr>              <dbl>       <dbl>         <dbl>
##  1        4 OLDER SINGLES/CO~ Dorito C~           1278       8307           6.5
##  2       14 OLDER SINGLES/CO~ Smiths C~           1276       7528.          5.90
##  3       16 OLDER SINGLES/CO~ Smiths C~           1260       7182.          5.70
##  4        7 OLDER SINGLES/CO~ Smiths C~           1251       7131.          5.70
##  5       23 OLDER SINGLES/CO~ Cheezels~           1249       7119.          5.70
##  6       32 OLDER SINGLES/CO~ Kettle S~           1303       7036.          5.40
##  7       88 OLDER SINGLES/CO~ Kettle H~           1301       7025.          5.40
##  8      102 OLDER SINGLES/CO~ Kettle M~           1294       6988.          5.40
##  9       20 OLDER SINGLES/CO~ Doritos ~           1224       6977.          5.70
## 10       36 OLDER SINGLES/CO~ Kettle C~           1269       6853.          5.40
```

```
ggplot(total_table_chips_premium) + geom_col(mapping = aes(x=total_sales,y = PROD_NAME ,fill = PROD_NAM
```

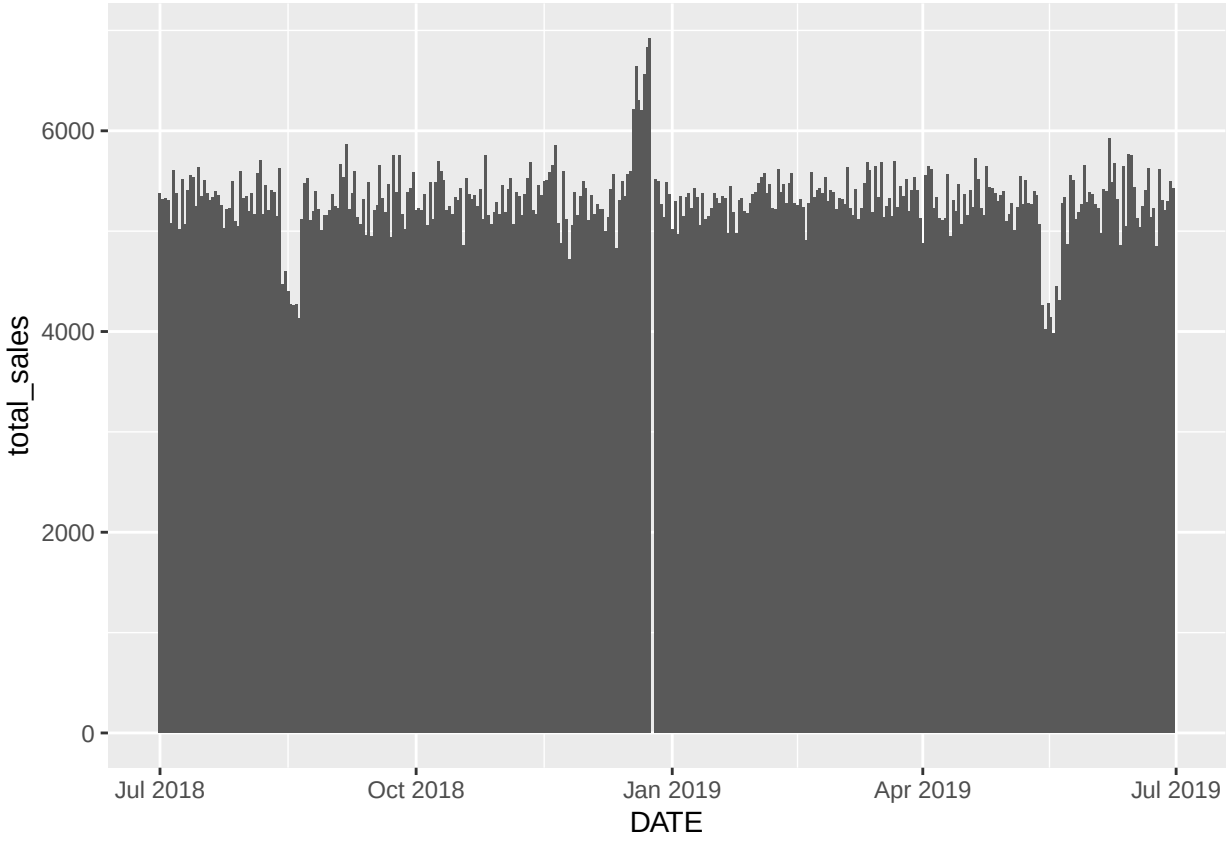


the most popular product is still **Dorito Corn Chp Supreme 380g**.

We can do further check to check the correlations between the most popular product/groups and sales and need more time for that.

How the sales change with the date?

```
total_table_date <- total_table %>%
  group_by(DATE) %>%
  summarise(total_quantity=sum(PROD_QTY),total_sales=sum(TOT_SALES),average_sales=sum(TOT_SALES)/sum(PR
ggplot(total_table_date) + geom_col(mapping = aes(x=DATE,y=total_sales))
```



We can easily found that nealy November and December, the chips sales increased a lot, but during the **mid-May** to **mid-June**, the sales would experieced regular drop.

We suggest stakeholders to prepare/organize possible sales promotion in Nov. to Dec.

Summary and suggestions

- **Older families** and **Retirees** contribute most in total chips sales. New families purchase least but also could be a possible market to dip.
- For different ranks of customers, **Mainstream** is the most important part, and we recommend to focus on Mainstream market. Although total sales of customers with "Budget" is larger than that of customers with "Premium", the average sales is pretty close.

- Although there are some fluctuations for top 10 popular chips, the most popular product in three groups is **Dorto Corn Chp Supreme 380g** without doubt. Perhaps its some special characters make it stand out. And other products can be improved based on it.
- The data also reveals that the sales is increased during **November** to **December** and the lowest vales appears in **May**. We recommend to focus on the time point to prepare promotion activities.

NOTES: * Although different metrics used(total_sales/total_quantity/average_sales) showed similar patterns, more details of products/dates could be studied based on correlations among different groups.

- The sales characteristics is based on data between 2018 and 2019, more data should be reviewed in case some variations.