Quantium Virtual Internship - Retail Strategy and Analytics - Task 1

Yussuf Ali

2024-09-07

```
# Install required packages if not already installed
#install.packages("data.table")
#install.packages("ggplot2")
#install.packages("readr")
#install.packages("readxl")

# Load required libraries
library(data.table)
library(ggplot2)
library(readr)
library(readxl)

getwd()

#filePath <- "C:/Users/youke/Desktop/R_prog/Data_Analytics_internship/"
```

```
# Load the data using fread with the correct file path
transactionData <- fread("QVI_transaction_data.csv")
# Check if the data is loaded successfully
str(transactionData)
```

```
## Classes 'data.table' and 'data.frame':   264836 obs. of  8 variables:
##  $ DATE       : int  43390 43599 43605 43329 43330 43604 43601 43601 43332 43330 ...
##  $ STORE_NBR     : int  1 1 1 2 2 4 4 4 5 7 ...
##  $ LYLTY_CARD_NBR: int  1000 1307 1343 2373 2426 4074 4149 4196 5026 7150 ...
##  $ TXN_ID        : int  1 348 383 974 1038 2982 3333 3539 4525 6900 ...
##  $ PROD_NBR      : int  5 66 61 69 108 57 16 24 42 52 ...
##  $ PROD_NAME     : chr  "Natural Chip      Compny SeaSalt175g" "CCs Nacho Cheese    175g" "Smith
##  $ PROD_QTY      : int  2 3 2 5 3 1 1 1 1 2 ...
##  $ TOT_SALES     : num  6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
head(transactionData, n=10)
```

```
##      DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##     <int>     <int>          <int>  <int>    <int>
##  1: 43390         1           1000      1        5
##  2: 43599         1           1307    348       66
```

```
##  3: 43605         1           1343    383       61
##  4: 43329         2           2373    974       69
##  5: 43330         2           2426   1038      108
##  6: 43604         4           4074   2982       57
##  7: 43601         4           4149   3333       16
##  8: 43601         4           4196   3539       24
##  9: 43332         5           5026   4525       42
## 10: 43330         7           7150   6900       52
##                                    PROD_NAME PROD_QTY TOT_SALES
##                                       <char>    <int>     <num>
##  1:   Natural Chip        Compny SeaSalt175g        2       6.0
##  2:                 CCs Nacho Cheese    175g        3       6.3
##  3:   Smiths Crinkle Cut  Chips Chicken 170g        2       2.9
##  4:   Smiths Chip Thinly  S/Cream&Onion 175g        5      15.0
##  5: Kettle Tortilla ChpsHny&Jlpno Chili 150g        3      13.8
##  6: Old El Paso Salsa   Dip Tomato Mild 300g        1       5.1
##  7: Smiths Crinkle Chips Salt & Vinegar 330g        1       5.7
##  8:    Grain Waves         Sweet Chilli 210g        1       3.6
##  9:  Doritos Corn Chip Mexican Jalapeno 150g        1       3.9
## 10:    Grain Waves Sour    Cream&Chives 210G        2       7.2
```

```
customerData <- fread("QVI_purchase_behaviour.csv")

str(customerData)
```

```
## Classes 'data.table' and 'data.frame':   72637 obs. of  3 variables:
##  $ LYLTY_CARD_NBR  : int  1000 1002 1003 1004 1005 1007 1009 1010 1011 1012 ...
##  $ LIFESTAGE     : chr  "YOUNG SINGLES/COUPLES" "YOUNG SINGLES/COUPLES" "YOUNG FAMILIES" "OLD
##  $ PREMIUM_CUSTOMER: chr  "Premium" "Mainstream" "Budget" "Mainstream" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
head(customerData, n=10)
```

```
##     LYLTY_CARD_NBR              LIFESTAGE PREMIUM_CUSTOMER
##              <int>                 <char>           <char>
##  1:           1000  YOUNG SINGLES/COUPLES          Premium
##  2:           1002  YOUNG SINGLES/COUPLES       Mainstream
##  3:           1003         YOUNG FAMILIES           Budget
##  4:           1004  OLDER SINGLES/COUPLES       Mainstream
##  5:           1005 MIDAGE SINGLES/COUPLES       Mainstream
##  6:           1007  YOUNG SINGLES/COUPLES           Budget
##  7:           1009           NEW FAMILIES          Premium
##  8:           1010  YOUNG SINGLES/COUPLES       Mainstream
##  9:           1011  OLDER SINGLES/COUPLES       Mainstream
## 10:           1012         OLDER FAMILIES       Mainstream
```

Exploratory data analysis

Examining transaction data

```
str(transactionData)
```

```
## Classes 'data.table' and 'data.frame':   264836 obs. of  8 variables:
##  $ DATE       : int  43390 43599 43605 43329 43330 43604 43601 43601 43332 43330 ...
##  $ STORE_NBR     : int  1 1 1 2 2 4 4 4 5 7 ...
##  $ LYLTY_CARD_NBR: int  1000 1307 1343 2373 2426 4074 4149 4196 5026 7150 ...
##  $ TXN_ID        : int  1 348 383 974 1038 2982 3333 3539 4525 6900 ...
##  $ PROD_NBR      : int  5 66 61 69 108 57 16 24 42 52 ...
##  $ PROD_NAME    : chr  "Natural Chip      Compny SeaSalt175g" "CCs Nacho Cheese    175g" "Smith
##  $ PROD_QTY      : int  2 3 2 5 3 1 1 1 1 2 ...
##  $ TOT_SALES     : num  6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Convert DATE column in transactionData to date format
transactionData$DATE <- as.Date(transactionData$DATE, origin = "1899-12-30")

# Check the conversion
str(transactionData$DATE)
```

```
##  Date[1:264836], format: "2018-10-17" "2019-05-14" "2019-05-20" "2018-08-
17" "2018-08-18" ...
```

```
# Check for missing values in each column of the transaction data
missing_values <- colSums(is.na(transactionData))
print(missing_values)
```

```
##           DATE      STORE_NBR LYLTY_CARD_NBR         TXN_ID       PROD_NBR
##              0              0              0              0              0
##      PROD_NAME       PROD_QTY      TOT_SALES
##              0              0              0
```

```
# Generate a summary of the PROD_NAME column
summary(transactionData$PROD_NAME)
```

```
##    Length     Class      Mode
##    264836 character character
```

```
# View the unique product names to identify non-chip entries
length(unique(transactionData$PROD_NAME))
```

```
## [1] 114
```

```
#### Examine PROD_NAME
transactionData[, .N, PROD_NAME][order(-N)] # sort from highest to lowest
```

```
##                                     PROD_NAME     N
##                                         <char> <int>
##   1:   Kettle Mozzarella   Basil & Pesto 175g  3304
```

```
##   2: Kettle Tortilla ChpsHny&Jlpno Chili 150g  3296
##   3: Cobs Popd Swt/Chlli &Sr/Cream Chips 110g  3269
##   4:   Tyrrells Crisps     Ched & Chives 165g  3268
##   5:           Cobs Popd Sea Salt  Chips 110g  3265
##  ---
## 110:                 RRD Pc Sea Salt     165g  1431
## 111:           Woolworths Medium   Salsa 300g  1430
## 112:   NCC Sour Cream &    Garden Chives 175g  1419
## 113:           French Fries Potato Chips 175g  1418
## 114:        WW Crinkle Cut      Original 175g  1410
```

Looks like we are definitely looking at potato chips but how can we check that these are all chips? We can do some basic text analysis by summarising the individual words in the product name.

```
# Split product names into individual words
productWords <- data.table(unlist(strsplit(unique(transactionData$PROD_NAME), "
 ↪  ")))
setnames(productWords, 'words')

# View the first few words extracted
head(productWords)
```

```
##      words
##     <char>
## 1: Natural
## 2:    Chip
## 3:
## 4:
## 5:
## 6:
```

As we are only interested in words that will tell us if the product is chips or not, let's remove all words with digits and special characters such as '&' from our set of product words. We can do this using grepl().

```
# Remove empty strings and spaces
productWords <- productWords[words != ""]

# Remove words with digits and special characters
productWords <- productWords[!grepl("\\d", words)]
productWords <- productWords[!grepl("[^a-zA-Z]", words)]

# Count the frequency of each word
wordFrequency <- productWords[, .N, by = words][order(-N)]

# Display the most common words
print(wordFrequency)
```

```
##          words     N
##         <char> <int>
##   1:     Chips    21
##   2:    Smiths    16
##   3:   Crinkle    14
```

```
##   4:       Cut    14
##   5:    Kettle    13
##  ---
## 164:       Rst     1
## 165:      Pork     1
## 166:     Belly     1
## 167:        Pc     1
## 168: Bolognese     1
```

There are salsa products in the dataset but we are only interested in the chips category, so let's remove these.

```
# Remove salsa products from the dataset
transactionData[, SALSA := grepl("salsa", tolower(PROD_NAME))]

# Keep only entries where SALSA is FALSE
transactionData <- transactionData[SALSA == FALSE, ][, SALSA := NULL]
```

```
# Verify the removal of salsa products by checking unique product names
unique(transactionData$PROD_NAME)
```

```
##   [1] "Natural Chip        Compny SeaSalt175g"
##   [2] "CCs Nacho Cheese    175g"
##   [3] "Smiths Crinkle Cut  Chips Chicken 170g"
##   [4] "Smiths Chip Thinly  S/Cream&Onion 175g"
##   [5] "Kettle Tortilla ChpsHny&Jlpno Chili 150g"
##   [6] "Smiths Crinkle Chips Salt & Vinegar 330g"
##   [7] "Grain Waves         Sweet Chilli 210g"
##   [8] "Doritos Corn Chip Mexican Jalapeno 150g"
##   [9] "Grain Waves Sour    Cream&Chives 210G"
##  [10] "Kettle Sensations   Siracha Lime 150g"
##  [11] "Twisties Cheese     270g"
##  [12] "WW Crinkle Cut      Chicken 175g"
##  [13] "Thins Chips Light&  Tangy 175g"
##  [14] "CCs Original 175g"
##  [15] "Burger Rings 220g"
##  [16] "NCC Sour Cream &    Garden Chives 175g"
##  [17] "Doritos Corn Chip Southern Chicken 150g"
##  [18] "Cheezels Cheese Box 125g"
##  [19] "Smiths Crinkle      Original 330g"
##  [20] "Infzns Crn Crnchers Tangy Gcamole 110g"
##  [21] "Kettle Sea Salt     And Vinegar 175g"
##  [22] "Smiths Chip Thinly  Cut Original 175g"
##  [23] "Kettle Original 175g"
##  [24] "Red Rock Deli Thai  Chilli&Lime 150g"
##  [25] "Pringles Sthrn FriedChicken 134g"
##  [26] "Pringles Sweet&Spcy BBQ 134g"
##  [27] "Thins Chips         Originl saltd 175g"
##  [28] "Red Rock Deli Sp    Salt & Truffle 150G"
##  [29] "Smiths Thinly       Swt Chli&S/Cream175G"
##  [30] "Kettle Chilli 175g"
##  [31] "Doritos Mexicana    170g"
##  [32] "Smiths Crinkle Cut  French OnionDip 150g"
```

```
##  [33] "Natural ChipCo      Hony Soy Chckn175g"
##  [34] "Dorito Corn Chp     Supreme 380g"
##  [35] "Twisties Chicken270g"
##  [36] "Smiths Thinly Cut   Roast Chicken 175g"
##  [37] "Kettle Mozzarella   Basil & Pesto 175g"
##  [38] "Infuzions Thai SweetChili PotatoMix 110g"
##  [39] "Kettle Sensations   Camembert & Fig 150g"
##  [40] "Smith Crinkle Cut   Mac N Cheese 150g"
##  [41] "Kettle Honey Soy    Chicken 175g"
##  [42] "Thins Chips Seasonedchicken 175g"
##  [43] "Smiths Crinkle Cut  Salt & Vinegar 170g"
##  [44] "Infuzions BBQ Rib   Prawn Crackers 110g"
##  [45] "GrnWves Plus Btroot & Chilli Jam 180g"
##  [46] "Tyrrells Crisps     Lightly Salted 165g"
##  [47] "Kettle Sweet Chilli And Sour Cream 175g"
##  [48] "Kettle 135g Swt Pot Sea Salt"
##  [49] "Pringles SourCream  Onion 134g"
##  [50] "Doritos Corn Chips  Original 170g"
##  [51] "Twisties Cheese     Burger 250g"
##  [52] "Cobs Popd Swt/Chlli &Sr/Cream Chips 110g"
##  [53] "Natural Chip Co     Tmato Hrb&Spce 175g"
##  [54] "Smiths Crinkle Cut  Chips Original 170g"
##  [55] "Cobs Popd Sea Salt  Chips 110g"
##  [56] "Smiths Crinkle Cut  Chips Chs&Onion170g"
##  [57] "French Fries Potato Chips 175g"
##  [58] "Doritos Corn Chips  Cheese Supreme 170g"
##  [59] "Pringles Original   Crisps 134g"
##  [60] "RRD Chilli&         Coconut 150g"
##  [61] "WW Original Corn    Chips 200g"
##  [62] "Thins Potato Chips  Hot & Spicy 175g"
##  [63] "Cobs Popd Sour Crm  &Chives Chips 110g"
##  [64] "Smiths Crnkle Chip  Orgnl Big Bag 380g"
##  [65] "Doritos Corn Chips  Nacho Cheese 170g"
##  [66] "Kettle Sensations   BBQ&Maple 150g"
##  [67] "WW D/Style Chip     Sea Salt 200g"
##  [68] "Pringles Chicken    Salt Crips 134g"
##  [69] "WW Original Stacked Chips 160g"
##  [70] "Smiths Chip Thinly  CutSalt/Vinegr175g"
##  [71] "Cheezels Cheese 330g"
##  [72] "Tostitos Lightly    Salted 175g"
##  [73] "Thins Chips Salt &  Vinegar 175g"
##  [74] "Smiths Crinkle Cut  Chips Barbecue 170g"
##  [75] "Cheetos Puffs 165g"
##  [76] "RRD Sweet Chilli &  Sour Cream 165g"
##  [77] "WW Crinkle Cut      Original 175g"
##  [78] "Tostitos Splash Of  Lime 175g"
##  [79] "Kettle Tortilla ChpsBtroot&Ricotta 150g"
##  [80] "CCs Tasty Cheese    175g"
##  [81] "Woolworths Cheese   Rings 190g"
##  [82] "Tostitos Smoked     Chipotle 175g"
##  [83] "Pringles Barbeque   134g"
##  [84] "WW Supreme Cheese   Corn Chips 200g"
##  [85] "Pringles Mystery    Flavour 134g"
##  [86] "Tyrrells Crisps     Ched & Chives 165g"
```

```
##  [87] "Snbts Whlgrn Crisps Cheddr&Mstrd 90g"
##  [88] "Cheetos Chs & Bacon Balls 190g"
##  [89] "Pringles Slt Vingar 134g"
##  [90] "Infuzions SourCream&Herbs Veg Strws 110g"
##  [91] "Kettle Tortilla ChpsFeta&Garlic 150g"
##  [92] "Infuzions Mango     Chutny Papadums 70g"
##  [93] "RRD Steak &         Chimuchurri 150g"
##  [94] "RRD Honey Soy       Chicken 165g"
##  [95] "Sunbites Whlegrn    Crisps Frch/Onin 90g"
##  [96] "RRD Salt & Vinegar  165g"
##  [97] "Doritos Cheese      Supreme 330g"
##  [98] "Smiths Crinkle Cut  Snag&Sauce 150g"
##  [99] "WW Sour Cream &OnionStacked Chips 160g"
## [100] "RRD Lime & Pepper   165g"
## [101] "Natural ChipCo Sea  Salt & Vinegr 175g"
## [102] "Red Rock Deli Chikn&Garlic Aioli 150g"
## [103] "RRD SR Slow Rst     Pork Belly 150g"
## [104] "RRD Pc Sea Salt     165g"
## [105] "Smith Crinkle Cut   Bolognese 150g"
```

Summarise the data to check for nulls and possible outliers

```
summary(transactionData)
```

```
##       DATE              STORE_NBR     LYLTY_CARD_NBR        TXN_ID
##  Min.   :2018-07-01   Min.   :  1.0   Min.   :   1000   Min.   :      1
##  1st Qu.:2018-09-30   1st Qu.: 70.0   1st Qu.:  70015   1st Qu.:  67569
##  Median :2018-12-30   Median :130.0   Median : 130367   Median : 135183
##  Mean   :2018-12-30   Mean   :135.1   Mean   : 135531   Mean   : 135131
##  3rd Qu.:2019-03-31   3rd Qu.:203.0   3rd Qu.: 203084   3rd Qu.: 202654
##  Max.   :2019-06-30   Max.   :272.0   Max.   :2373711   Max.   :2415841
##     PROD_NBR       PROD_NAME            PROD_QTY         TOT_SALES
##  Min.   :  1.00   Length:246742      Min.   :  1.000   Min.   :  1.700
##  1st Qu.: 26.00   Class :character   1st Qu.:  2.000   1st Qu.:  5.800
##  Median : 53.00   Mode  :character   Median :  2.000   Median :  7.400
##  Mean   : 56.35                      Mean   :  1.908   Mean   :  7.321
##  3rd Qu.: 87.00                      3rd Qu.:  2.000   3rd Qu.:  8.800
##  Max.   :114.00                      Max.   :200.000   Max.   :650.000
```

There are no nulls in the columns but product quantity appears to have an outlier which we should investigate further. Let's investigate further the case where 200 packets of chips are bought in one transaction.

```
# Filter the dataset to find the outlier where 200 packets of chips are bought
outlier_transactions <- transactionData[PROD_QTY == 200]
print(outlier_transactions)
```

```
##          DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##        <Date>     <int>          <int>  <int>    <int>
## 1: 2018-08-19       226         226000 226201        4
## 2: 2019-05-20       226         226000 226210        4
##                          PROD_NAME PROD_QTY TOT_SALES
##                             <char>    <int>     <num>
```

```
## 1: Dorito Corn Chp     Supreme 380g      200       650
## 2: Dorito Corn Chp     Supreme 380g      200       650
```

There are two transactions where 200 packets of chips are bought in one transaction and both of these transactions where by the same customer.

```
# Find the customer who bought 200 packets of chips
outlier_customer <- outlier_transactions$LYLTY_CARD_NBR

# Use a filter to see what other transactions that customer made
customer_transactions <- transactionData[LYLTY_CARD_NBR %in% outlier_customer]
print(customer_transactions)
```

```
##          DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##        <Date>     <int>          <int>  <int>    <int>
## 1: 2018-08-19       226         226000 226201        4
## 2: 2019-05-20       226         226000 226210        4
##                          PROD_NAME PROD_QTY TOT_SALES
##                             <char>    <int>     <num>
## 1: Dorito Corn Chp     Supreme 380g      200       650
## 2: Dorito Corn Chp     Supreme 380g      200       650
```

It looks like this customer has only had the two transactions over the year and is not an ordinary retail customer. The customer might be buying chips for commercial purposes instead. We'll remove this loyalty card number from further analysis.

```
# Remove the outliers
transactionData <- transactionData[PROD_QTY != 200]

# Verify the removal
summary(transactionData$PROD_QTY)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.000   1.906   2.000   5.000
```

```
# Summarize the data to check for nulls and possible outliers
summary(transactionData)
```

```
##       DATE              STORE_NBR     LYLTY_CARD_NBR        TXN_ID
##  Min.   :2018-07-01   Min.   :  1.0   Min.   :   1000   Min.   :      1
##  1st Qu.:2018-09-30   1st Qu.: 70.0   1st Qu.:  70015   1st Qu.:  67569
##  Median :2018-12-30   Median :130.0   Median : 130367   Median : 135182
##  Mean   :2018-12-30   Mean   :135.1   Mean   : 135530   Mean   : 135130
##  3rd Qu.:2019-03-31   3rd Qu.:203.0   3rd Qu.: 203083   3rd Qu.: 202652
##  Max.   :2019-06-30   Max.   :272.0   Max.   :2373711   Max.   :2415841
##     PROD_NBR       PROD_NAME            PROD_QTY       TOT_SALES
##  Min.   :  1.00   Length:246740      Min.   :1.000   Min.   : 1.700
##  1st Qu.: 26.00   Class :character   1st Qu.:2.000   1st Qu.: 5.800
##  Median : 53.00   Mode  :character   Median :2.000   Median : 7.400
##  Mean   : 56.35                      Mean   :1.906   Mean   : 7.316
##  3rd Qu.: 87.00                      3rd Qu.:2.000   3rd Qu.: 8.800
##  Max.   :114.00                      Max.   :5.000   Max.   :29.500
```

Now, let's look at the number of transaction lines over time to see if there are any obvious data issues such as missing data.

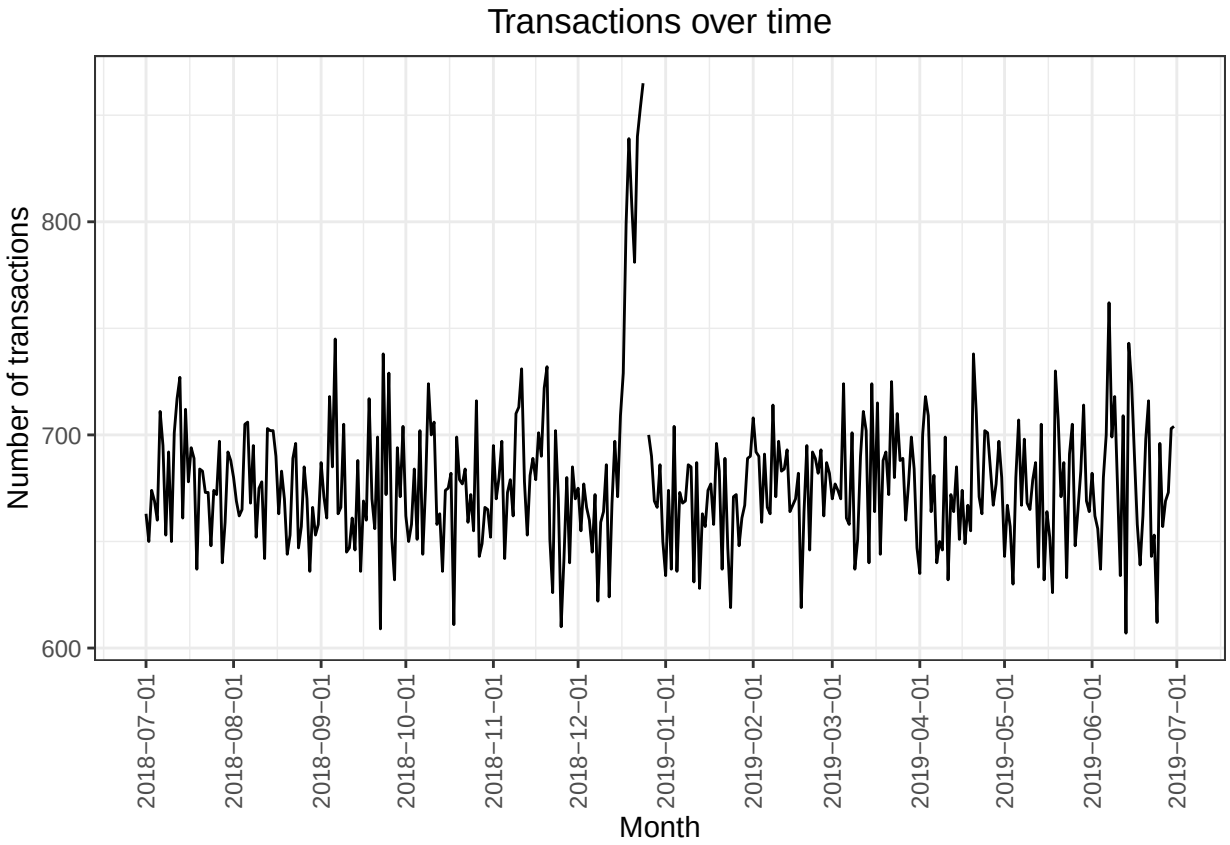
```
# Count the number of transactions by date
transactionData[, .N, by = DATE]
```

```
##            DATE     N
##          <Date> <int>
##   1: 2018-10-17   682
##   2: 2019-05-14   705
##   3: 2019-05-20   707
##   4: 2018-08-17   663
##   5: 2018-08-18   683
##  ---
## 360: 2018-12-08   622
## 361: 2019-01-30   689
## 362: 2019-02-09   671
## 363: 2018-08-31   658
## 364: 2019-02-12   684
```

There's only 364 rows, meaning only 364 dates which indicates a missing date. Let's create a sequence of dates from 1 Jul 2018 to 30 Jun 2019 and use this to create a chart of number of transactions over time to find the missing date.

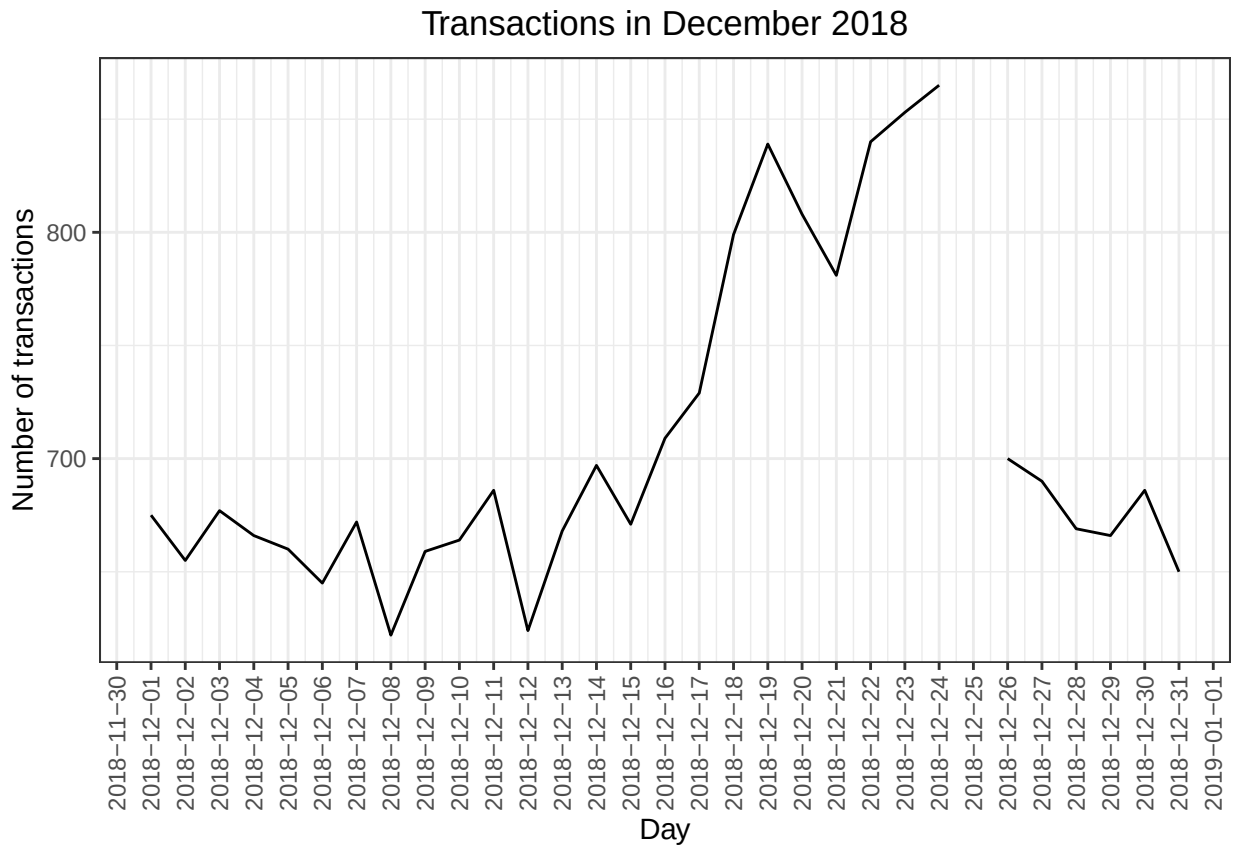
```
# Create a sequence of dates from 1 Jul 2018 to 30 Jun 2019
date_sequence <- data.table(DATE = seq(as.Date("2018-07-01"),
 ↪  as.Date("2019-06-30"), by = "day"))

# Join this sequence with the transaction count data
transactions_by_day <- merge(date_sequence, transactionData[, .N, by = DATE],
 ↪  by = "DATE", all.x = TRUE)
```

```
# Plot the number of transactions over time
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))

ggplot(transactions_by_day, aes(x = DATE, y = N)) +
  geom_line() +
  labs(x = "Month", y = "Number of transactions", title = "Transactions over
   ↪  time") +
  scale_x_date(breaks = "1 month") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



We can see that there is an increase in purchases in December and a break in late December. Let's zoom in on this.

```
# Plot transactions over time for December 2018
ggplot(transactions_by_day[month(DATE) == 12, ], aes(x = DATE, y = N)) +
  geom_line() +
  labs(x = "Day", y = "Number of transactions", title = "Transactions in
   ↪  December 2018") +
  scale_x_date(breaks = "1 day") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



We can see that the increase in sales occurs in the lead-up to Christmas and that there are zero sales on Christmas day itself. This is due to shops being closed on Christmas day. Now that we are satisfied that the data no longer has outliers, we can move on to creating other features such as brand of chips or pack size from PROD_NAME. We will start with pack size.

```
# Extract Pack Size
transactionData[, PACK_SIZE := parse_number(PROD_NAME)]

# Verify the Pack Sizes
transactionData[, .N, PACK_SIZE][order(-N)]
```

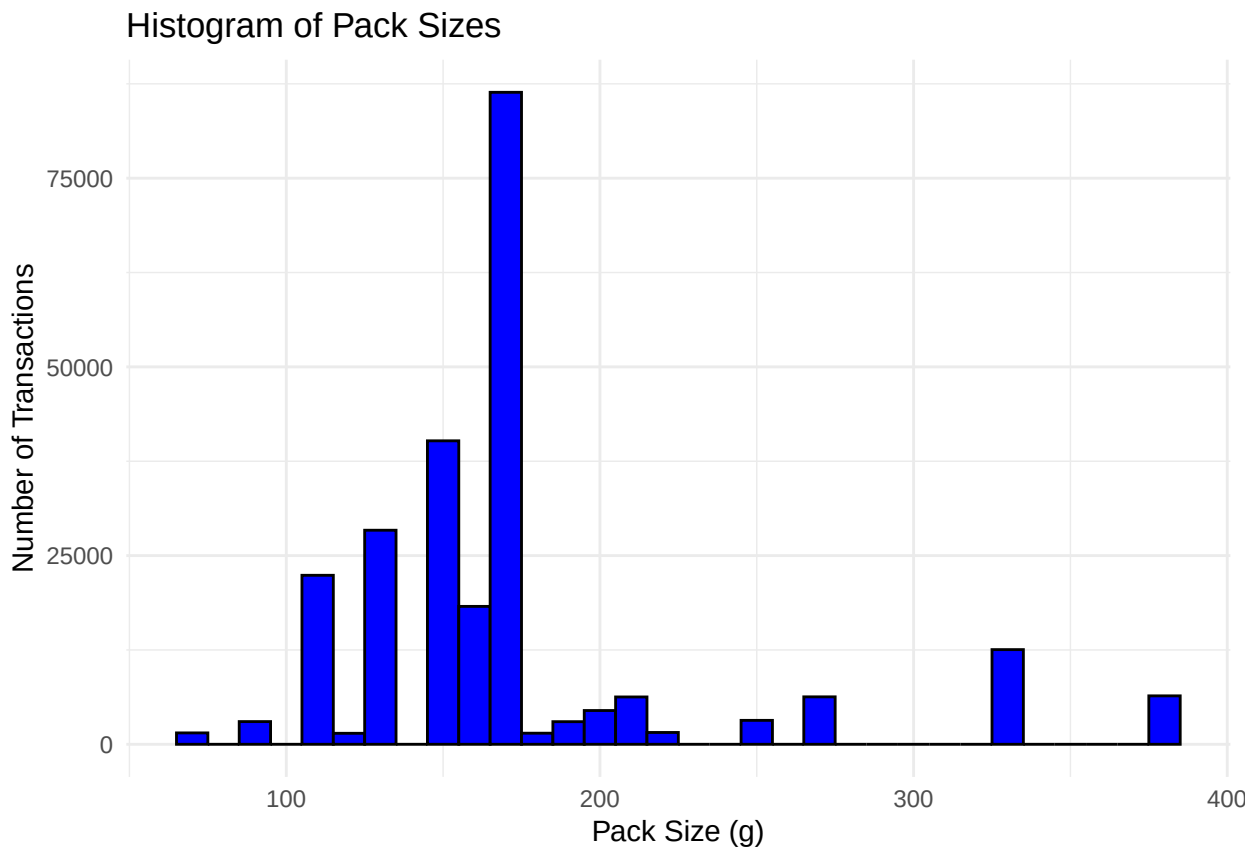
```
##     PACK_SIZE     N
##         <num> <int>
##  1:       175 66390
##  2:       150 40203
##  3:       134 25102
##  4:       110 22387
##  5:       170 19983
##  6:       165 15297
##  7:       330 12540
##  8:       380  6416
##  9:       270  6285
## 10:       210  6272
## 11:       200  4473
## 12:       135  3257
## 13:       250  3169
```

```
## 14:        90  3008
## 15:       190  2995
## 16:       160  2970
## 17:       220  1564
## 18:        70  1507
## 19:       180  1468
## 20:       125  1454
##     PACK_SIZE     N
```

```
#Let's check the output of the first few rows to see if we have indeed picked
 ↪  out pack size
transactionData
```

```
##                DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##              <Date>     <int>          <int>  <int>    <int>
##      1: 2018-10-17         1           1000      1        5
##      2: 2019-05-14         1           1307    348       66
##      3: 2019-05-20         1           1343    383       61
##      4: 2018-08-17         2           2373    974       69
##      5: 2018-08-18         2           2426   1038      108
##     ---
## 246736: 2019-03-09       272         272319 270088       89
## 246737: 2018-08-13       272         272358 270154       74
## 246738: 2018-11-06       272         272379 270187       51
## 246739: 2018-12-27       272         272379 270188       42
## 246740: 2018-09-22       272         272380 270189       74
##                                       PROD_NAME PROD_QTY TOT_SALES PACK_SIZE
##                                          <char>    <int>     <num>     <num>
##      1:   Natural Chip        Compny SeaSalt175g        2       6.0       175
##      2:                CCs Nacho Cheese    175g        3       6.3       175
##      3:   Smiths Crinkle Cut  Chips Chicken 170g        2       2.9       170
##      4:   Smiths Chip Thinly  S/Cream&Onion 175g        5      15.0       175
##      5: Kettle Tortilla ChpsHny&Jlpno Chili 150g        3      13.8       150
##     ---
## 246736:  Kettle Sweet Chilli And Sour Cream 175g        2      10.8       175
## 246737:            Tostitos Splash Of  Lime 175g        1       4.4       175
## 246738:                 Doritos Mexicana    170g        2       8.8       170
## 246739:  Doritos Corn Chip Mexican Jalapeno 150g        2       7.8       150
## 246740:            Tostitos Splash Of  Lime 175g        2       8.8       175
```

```
#Plot a Histogram of Pack Sizes

ggplot(transactionData, aes(x = PACK_SIZE)) +
  geom_histogram(binwidth = 10, fill = "blue", color = "black") +
  labs(title = "Histogram of Pack Sizes", x = "Pack Size (g)", y = "Number of
   ↪  Transactions") +
  theme_minimal()
```




```
#hist(transactionData[, PACK_SIZE])
```

Pack sizes created look reasonable and now to create brands, we can use the first word in PROD_NAME to work out the brand name.

```
library(stringr)
```

```
# Extract the brand name from the PROD_NAME column
transactionData[, BRAND := toupper(word(PROD_NAME, 1))]
#Verify the brand names
transactionData[, .N, BRAND][order(-N)]
```

```
##          BRAND     N
##         <char> <int>
##  1:     KETTLE 41288
##  2:     SMITHS 27390
##  3:   PRINGLES 25102
##  4:    DORITOS 22041
##  5:      THINS 14075
##  6:        RRD 11894
##  7:  INFUZIONS 11057
##  8:         WW 10320
##  9:       COBS  9693
## 10:   TOSTITOS  9471
```

```
## 11:   TWISTIES  9454
## 12:   TYRRELLS  6442
## 13:      GRAIN  6272
## 14:    NATURAL  6050
## 15:   CHEEZELS  4603
## 16:        CCS  4551
## 17:        RED  4427
## 18:     DORITO  3183
## 19:     INFZNS  3144
## 20:      SMITH  2963
## 21:    CHEETOS  2927
## 22:      SNBTS  1576
## 23:     BURGER  1564
## 24: WOOLWORTHS  1516
## 25:    GRNWVES  1468
## 26:   SUNBITES  1432
## 27:        NCC  1419
## 28:     FRENCH  1418
##          BRAND     N
```

Some of the brand names look like they are of the same brands - such as RED and RRD, which are both Red Rock Deli chips. Let's combine these together.

```
#### Clean brand names
transactionData[BRAND == "RED", BRAND := "RRD"]
transactionData[BRAND == "SNBTS", BRAND := "SUNBITES"]
transactionData[BRAND == "INFZNS", BRAND := "INFUZIONS"]
transactionData[BRAND == "WW", BRAND := "WOOLWORTHS"]
transactionData[BRAND == "SMITH", BRAND := "SMITHS"]
transactionData[BRAND == "NCC", BRAND := "NATURAL"]
transactionData[BRAND == "DORITO", BRAND := "DORITOS"]
transactionData[BRAND == "GRAIN", BRAND := "GRNWVES"]

#### Check again
transactionData[, .N, by = BRAND][order(BRAND)]
```

```
##          BRAND     N
##         <char> <int>
##  1:     BURGER  1564
##  2:        CCS  4551
##  3:    CHEETOS  2927
##  4:   CHEEZELS  4603
##  5:       COBS  9693
##  6:    DORITOS 25224
##  7:     FRENCH  1418
##  8:    GRNWVES  7740
##  9:  INFUZIONS 14201
## 10:     KETTLE 41288
## 11:    NATURAL  7469
## 12:   PRINGLES 25102
## 13:        RRD 16321
## 14:     SMITHS 30353
## 15:   SUNBITES  3008
```

```
## 16:      THINS 14075
## 17:   TOSTITOS  9471
## 18:   TWISTIES  9454
## 19:   TYRRELLS  6442
## 20: WOOLWORTHS 11836
##          BRAND     N
```

Examining customer data

Now that we are happy with the transaction dataset, let's have a look at the customer dataset.

```
str(customerData)
```

```
## Classes 'data.table' and 'data.frame':   72637 obs. of  3 variables:
##  $ LYLTY_CARD_NBR  : int  1000 1002 1003 1004 1005 1007 1009 1010 1011 1012 ...
##  $ LIFESTAGE       : chr  "YOUNG SINGLES/COUPLES" "YOUNG SINGLES/COUPLES" "YOUNG FAMILIES" "OLD
##  $ PREMIUM_CUSTOMER: chr  "Premium" "Mainstream" "Budget" "Mainstream" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Summary of the entire dataset
summary(customerData)
```

```
##  LYLTY_CARD_NBR     LIFESTAGE         PREMIUM_CUSTOMER
##  Min.   :   1000   Length:72637       Length:72637
##  1st Qu.:  66202   Class :character   Class :character
##  Median : 134040   Mode  :character   Mode  :character
##  Mean   : 136186
##  3rd Qu.: 203375
##  Max.   :2373711
```

Let's have a closer look at the LIFESTAGE and PREMIUM_CUSTOMER columns.

```
#### Examining the values of lifestage and premium_customer
customerData[, .N, LIFESTAGE][order(-N)]
```

```
##                 LIFESTAGE     N
##                    <char> <int>
## 1:               RETIREES 14805
## 2:  OLDER SINGLES/COUPLES 14609
## 3:  YOUNG SINGLES/COUPLES 14441
## 4:         OLDER FAMILIES  9780
## 5:         YOUNG FAMILIES  9178
## 6: MIDAGE SINGLES/COUPLES  7275
## 7:           NEW FAMILIES  2549
```

```
customerData[, .N, PREMIUM_CUSTOMER][order(-N)]
```

```
##    PREMIUM_CUSTOMER     N
##               <char> <int>
## 1:       Mainstream 29245
## 2:           Budget 24470
## 3:          Premium 18922
```

As there do not seem to be any issues with the customer data, we can now go ahead and join the transaction and customer data sets together.

```
#### Merge transaction data to customer data
data <- merge(transactionData, customerData, all.x = TRUE)
```

Let's also check if some customers were not matched on by checking for nulls.

```
# Checking for missing customer details after the merge
data[is.na(LIFESTAGE) | is.na(PREMIUM_CUSTOMER)]
```

```
## Key: <LYLTY_CARD_NBR>
## Empty data.table (0 rows and 12 cols): LYLTY_CARD_NBR,DATE,STORE_NBR,TXN_ID,PROD_NBR,PROD_NA
```

There are no nulls! So all the customers in the transaction data has been accounted for in the customer dataset.

Retain this dataset for Task 2 and write out as a csv.

```
#Save the merged dataset as CSV for Task 2
fwrite(data, paste0("QVI_data.csv"))
```

Data analysis on customer segments

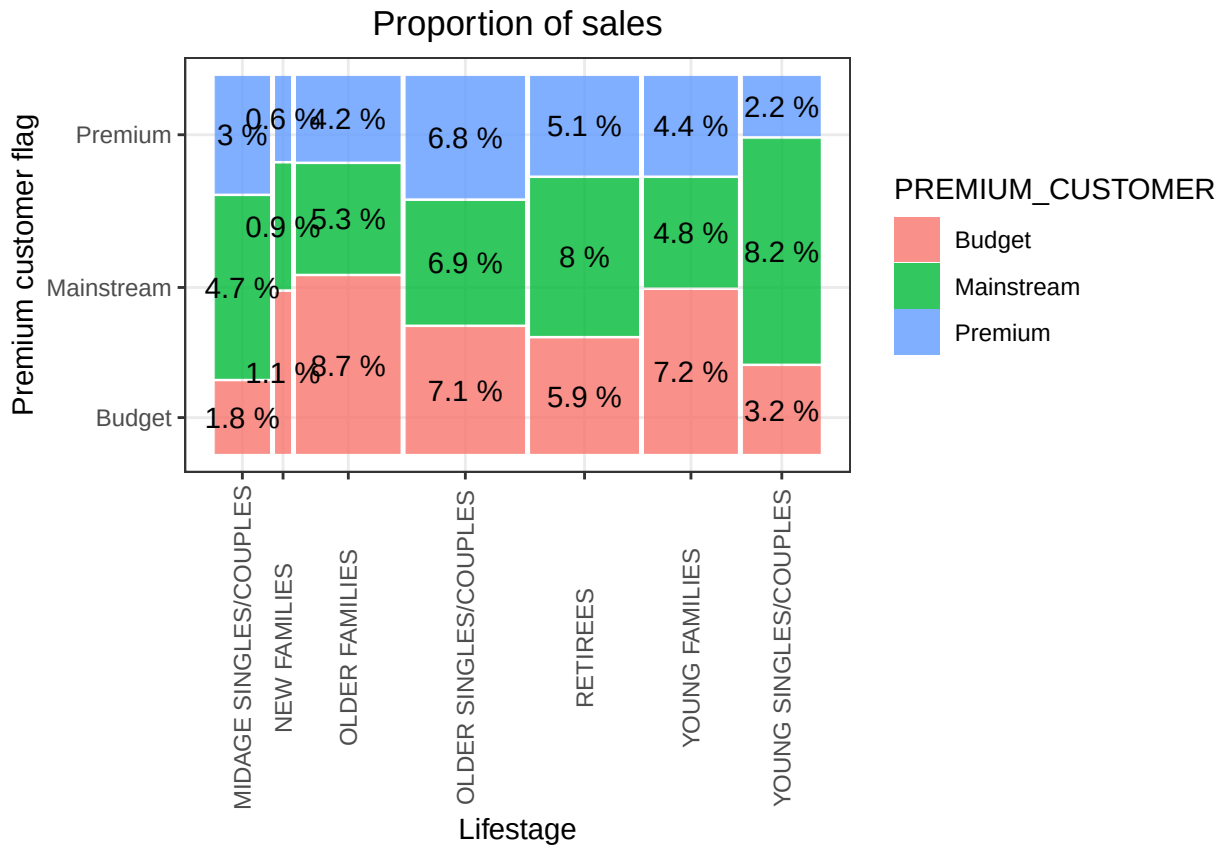
Let's start with calculating total sales by LIFESTAGE and PREMIUM_CUSTOMER and plotting the split by these segments to describe which customer segment contribute most to chip sales.

```
#install.packages("ggmosaic")
library(ggmosaic)
```

```
#### Total sales by LIFESTAGE and PREMIUM_CUSTOMER
sales = data[, .(SALES = sum(TOT_SALES)), .(LIFESTAGE, PREMIUM_CUSTOMER)]

#### Create plot
p = ggplot(data = sales) + geom_mosaic(aes(weight = SALES, x =
 ↪  product(PREMIUM_CUSTOMER, LIFESTAGE), fill = PREMIUM_CUSTOMER)) + labs(x =
 ↪  "Lifestage", y = "Premium customer flag", title = "Proportion of sales") +
 ↪  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))

#### Plot and label with proportion of sales
p + geom_text(data = ggplot_build(p)$data[[1]], aes(x = (xmin + xmax)/2 , y =
 ↪  (ymin + ymax)/2, label = as.character(paste(round(.wt/sum(.wt),3)*100,
 ↪  '%'))))
```

```
## Warning: The `scale_name` argument of `continuous_scale()` is deprecated as of ggplot2
## 3.5.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

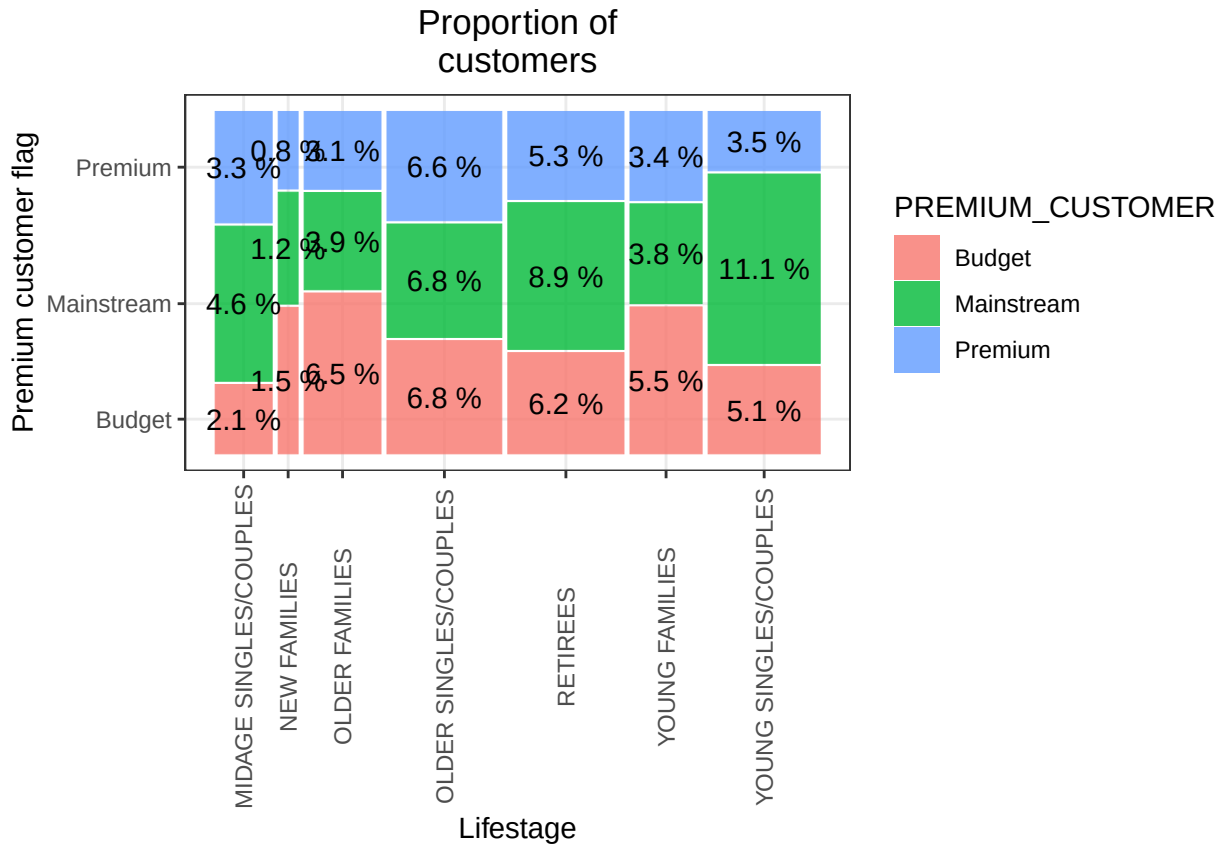
```
## Warning: The `trans` argument of `continuous_scale()` is deprecated as of ggplot2 3.5.0.
## i Please use the `transform` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## Warning: `unite_()` was deprecated in tidyr 1.2.0.
## i Please use `unite()` instead.
## i The deprecated feature was likely used in the ggmosaic package.
##   Please report the issue at <https://github.com/haleyjeppson/ggmosaic>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Sales are coming mainly from Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees Let's see if the higher sales are due to there being more customers who buy chips.

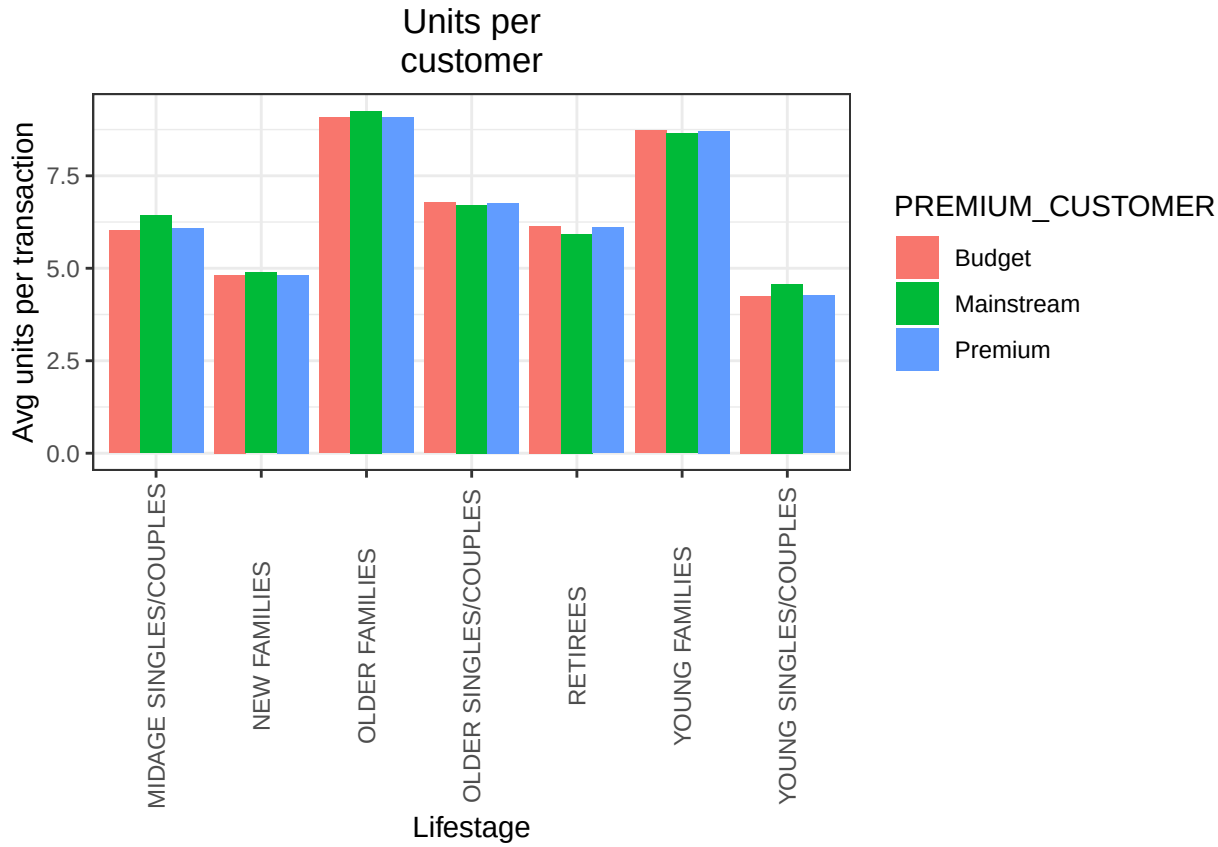
```
#### Number of customers by LIFESTAGE and PREMIUM_CUSTOMER
customers = data[, .(CUSTOMERS = uniqueN(LYLTY_CARD_NBR)), .(LIFESTAGE,
 ↪  PREMIUM_CUSTOMER)][order(-CUSTOMERS)]


#### Create plot
p = ggplot(data = customers) + geom_mosaic(aes(weight = CUSTOMERS, x =
 ↪  product(PREMIUM_CUSTOMER,
LIFESTAGE), fill = PREMIUM_CUSTOMER)) +
```

```
labs(x = "Lifestage", y = "Premium customer flag", title = "Proportion of
customers") +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
#### Plot and label with proportion of customers
p + geom_text(data = ggplot_build(p)$data[[1]], aes(x = (xmin + xmax)/2 , y =
(ymin + ymax)/2, label = as.character(paste(round(.wt/sum(.wt),3)*100,
'%'))))
```



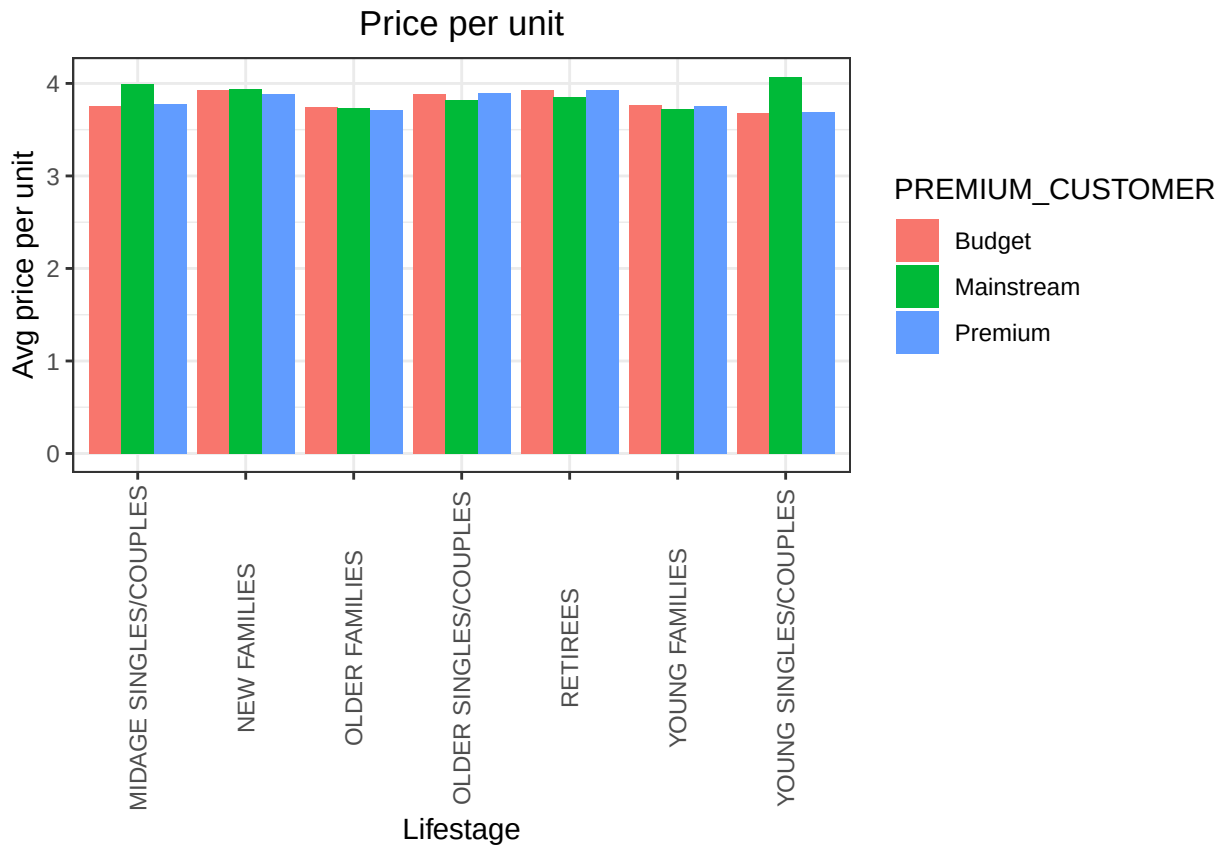
There are more Mainstream - young singles/couples and Mainstream - retirees who buy chips. This contributes to there being more sales to these customer segments but this is not a major driver for the Budget - Older families segment. Higher sales may also be driven by more units of chips being bought per customer. Let's have a look at this next.

```
# Average number of units per customer by LIFESTAGE and PREMIUM_CUSTOMER
avg_units <- data[, .(AVG = sum(PROD_QTY)/uniqueN(LYLTY_CARD_NBR)),
 ↪  .(LIFESTAGE, PREMIUM_CUSTOMER)][order(-AVG)]

#### Create plot
ggplot(data = avg_units, aes(weight = AVG, x = LIFESTAGE, fill =
 ↪  PREMIUM_CUSTOMER)) +
geom_bar(position = position_dodge()) +
labs(x = "Lifestage", y = "Avg units per transaction", title = "Units per
customer") +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



Older families and young families in general buy more chips per customer. Let's also investigate the average price per unit chips bought for each customer segment as this is also a driver of total sales.

```
#### Average price per unit by LIFESTAGE and PREMIUM_CUSTOMER
avg_price <- data[, .(AVG = sum(TOT_SALES)/sum(PROD_QTY)), .(LIFESTAGE,
 ↪  PREMIUM_CUSTOMER)][order(-AVG)]

#### Create plot
ggplot(data = avg_price, aes(weight = AVG, x = LIFESTAGE, fill =
PREMIUM_CUSTOMER)) +
geom_bar(position = position_dodge()) +
labs(x = "Lifestage", y = "Avg price per unit", title = "Price per unit") +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



As the difference in average price per unit isn't large, we can check if this difference is statistically different.

```
#### Perform an independent t-test between mainstream vs premium and budget
 ↪  midage and young singles and couples

pricePerUnit <- data[, price := TOT_SALES/PROD_QTY]

t.test(data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES")
 ↪  & PREMIUM_CUSTOMER == "Mainstream", price],

       data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES")
 ↪  & PREMIUM_CUSTOMER != "Mainstream", price],
       alternative = "greater")
```

```
##
##  Welch Two Sample t-test
##
## data:  data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") & PREMIUM_CU
## t = 37.624, df = 54791, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.3187234       Inf
## sample estimates:
## mean of x mean of y
##  4.039786  3.706491
```

The t-test results in a p-value < 2.2e-16, i.e. the unit price for mainstream, young and mid-age singles and

couples are significantly higher than that of budget or premium, young and midage singles and couples.

Deep dive into specific customer segments for insights

We have found quite a few interesting insights that we can dive deeper into.

We might want to target customer segments that contribute the most to sales to retain them or further increase sales. Let's look at Mainstream - young singles/couples. For instance, let's find out if they tend to buy a particular brand of chips.

```
#### Deep dive into Mainstream, young singles/couples
segment1 <- data[LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER ==
 ↪  "Mainstream",]

other <- data[!(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER ==
 ↪  "Mainstream"),]

#### Brand affinity compared to the rest of the population
quantity_segment1 <- segment1[, sum(PROD_QTY)]

#### Brand affinity compared to the rest of the population
quantity_segment1 < segment1[, sum(PROD_QTY)]
```

```
## [1] FALSE
```

```
quantity_other <- other[, sum(PROD_QTY)]

quantity_segment1_by_brand <- segment1[, .(targetSegment =
 ↪  sum(PROD_QTY)/quantity_segment1), by = BRAND]

quantity_other_by_brand <- other[, .(other = sum(PROD_QTY)/quantity_other), by
 ↪  = BRAND]

brand_proportions <- merge(quantity_segment1_by_brand,
 ↪  quantity_other_by_brand)[, affinityToBrand := targetSegment/other]

brand_proportions[order(-affinityToBrand)]
```

```
##          BRAND targetSegment       other affinityToBrand
##         <char>         <num>       <num>           <num>
##  1:   TYRRELLS   0.031552795 0.025692464       1.2280953
##  2:   TWISTIES   0.046183575 0.037876520       1.2193194
##  3:    DORITOS   0.122760524 0.101074684       1.2145526
##  4:     KETTLE   0.197984817 0.165553442       1.1958967
##  5:   TOSTITOS   0.045410628 0.037977861       1.1957131
##  6:   PRINGLES   0.119420290 0.100634769       1.1866703
##  7:       COBS   0.044637681 0.039048861       1.1431238
##  8:  INFUZIONS   0.064679089 0.057064679       1.1334347
##  9:      THINS   0.060372671 0.056986370       1.0594230
## 10:    GRNWVES   0.032712215 0.031187957       1.0488733
## 11:   CHEEZELS   0.017971014 0.018646902       0.9637534
## 12:     SMITHS   0.096369910 0.124583692       0.7735355
```

```
## 13:     FRENCH   0.003947550 0.005758060       0.6855694
## 14:    CHEETOS   0.008033126 0.012066591       0.6657329
## 15:        RRD   0.043809524 0.067493678       0.6490908
## 16:    NATURAL   0.019599724 0.030853989       0.6352412
## 17:        CCS   0.011180124 0.018895650       0.5916771
## 18:   SUNBITES   0.006349206 0.012580210       0.5046980
## 19: WOOLWORTHS   0.024099379 0.049427188       0.4875733
## 20:     BURGER   0.002926156 0.006596434       0.4435967
##          BRAND targetSegment       other affinityToBrand
```

We can see that: • Mainstream young singles/couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population. • Mainstream young singles/couples are 56% less likely to purchase Burger Rings compared to the rest of the population.

Let's also find out if our target segment tends to buy larger packs of chips.

```
quantity_segment1_by_pack <- segment1[, .(targetSegment =
 ↪  sum(PROD_QTY)/quantity_segment1), by = PACK_SIZE]

quantity_other_by_pack <- other[, .(other = sum(PROD_QTY)/quantity_other), by =
 ↪  PACK_SIZE]

pack_proportions <- merge(quantity_segment1_by_pack, quantity_other_by_pack)[,
 ↪  affinityToPack := targetSegment/other]

pack_proportions[order(-affinityToPack)]
```

```
##     PACK_SIZE targetSegment       other affinityToPack
##         <num>         <num>       <num>          <num>
##  1:       270   0.031828847 0.025095929      1.2682873
##  2:       380   0.032160110 0.025584213      1.2570295
##  3:       330   0.061283644 0.050161917      1.2217166
##  4:       134   0.119420290 0.100634769      1.1866703
##  5:       110   0.106280193 0.089791190      1.1836372
##  6:       210   0.029123533 0.025121265      1.1593180
##  7:       135   0.014768806 0.013075403      1.1295106
##  8:       250   0.014354727 0.012780590      1.1231662
##  9:       170   0.080772947 0.080985964      0.9973697
## 10:       150   0.157598344 0.163420656      0.9643722
## 11:       175   0.254989648 0.270006956      0.9443818
## 12:       165   0.055652174 0.062267662      0.8937572
## 13:       190   0.007481021 0.012442016      0.6012708
## 14:       180   0.003588682 0.006066692      0.5915385
## 15:       160   0.006404417 0.012372920      0.5176157
## 16:        90   0.006349206 0.012580210      0.5046980
## 17:       125   0.003008972 0.006036750      0.4984423
## 18:       200   0.008971705 0.018656115      0.4808989
## 19:        70   0.003036577 0.006322350      0.4802924
## 20:       220   0.002926156 0.006596434      0.4435967
##     PACK_SIZE targetSegment       other affinityToPack
```

It looks like Mainstream young singles/couples are 27% more likely to purchase a 270g pack of chips compared to the rest of the population but let's dive into what brands sell this pack size.

```
data[PACK_SIZE == 270, unique(PROD_NAME)]
```

```
## [1] "Twisties Cheese     270g" "Twisties Chicken270g"
```

Twisties are the only brand offering 270g packs and so this may instead be reflecting a higher likelihood of purchasing Twisties.

Conclusion

In conclusion, **Mainstream Young Singles/Couples** prefer specific brands like **Tyrrells, Twisties, and Doritos** over others, indicating brand loyalty in this segment. Specifically, Mainstream young singles and couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population. They also tend to purchase slightly **larger pack sizes** compared to the rest of the population, likely due to their preference for bulk buying, possibly for social or entertainment purposes. These insights suggest targeting this group with brand-specific promotions and larger pack options could increase sales retention and growth within this segment.