Quantium Virtual Internship - Retail Strategy and Analytics

Solution for Task 1

```
## -- Attaching packages ------------------------------------------ tidyverse 1.3.0 --

## v tibble  3.0.3     v dplyr   1.0.2
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.4

## -- Conflicts ------------------------------------------------ tidyverse_conflicts() --
## x dplyr::between()   masks data.table::between()
## x dplyr::filter()    masks stats::filter()
## x dplyr::first()     masks data.table::first()
## x dplyr::lag()       masks stats::lag()
## x dplyr::last()      masks data.table::last()
## x purrr::transpose() masks data.table::transpose()

## Parsed with column specification:
## cols(
##   LYLTY_CARD_NBR = col_double(),
##   LIFESTAGE = col_character(),
##   PREMIUM_CUSTOMER = col_character()
## )
```

Exploratory data analysis

Cleaning the data

```
#### Examine transaction data
str(transactionData)
```

```
## tibble [264,836 x 8] (S3: tbl_df/tbl/data.frame)
## $ DATE : num [1:264836] 43390 43599 43605 43329 43330 ...
## $ STORE_NBR : num [1:264836] 1 1 1 2 2 4 4 4 5 7 ...
## $ LYLTY_CARD_NBR: num [1:264836] 1000 1307 1343 2373 2426 ...
## $ TXN_ID : num [1:264836] 1 348 383 974 1038 ...
## $ PROD_NBR : num [1:264836] 5 66 61 69 108 57 16 24 42 52 ...
## $ PROD_NAME : chr [1:264836] "Natural Chip Compny SeaSalt175g" "CCs Nacho
Cheese 175g" "Smiths Crinkle Cut Chips Chicken 170g" "Smiths Chip Thinly
S/Cream&Onion 175g" ...
## $ PROD_QTY : num [1:264836] 2 3 2 5 3 1 1 1 1 2 ...
## $ TOT_SALES : num [1:264836] 6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
```

We can change the date from integer to an R date format.

```r
#### Convert DATE column to a date format
#### A quick search online tells us that CSV and Excel integer dates begin on 30 Dec 1899

transactionData$DATE <- as.Date(transactionData$DATE, origin = "1899-12-30")
```

We should check that we are looking at the right products by examining PROD_NAME.

```r
#### Examine PROD_NAME
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] "Old El Paso Salsa   Dip Tomato Mild 300g"
##   [7] "Smiths Crinkle Chips Salt & Vinegar 330g"
##   [8] "Grain Waves         Sweet Chilli 210g"
##   [9] "Doritos Corn Chip Mexican Jalapeno 150g"
##  [10] "Grain Waves Sour    Cream&Chives 210G"
##  [11] "Kettle Sensations   Siracha Lime 150g"
##  [12] "Twisties Cheese     270g"
##  [13] "WW Crinkle Cut      Chicken 175g"
##  [14] "Thins Chips Light&  Tangy 175g"
##  [15] "CCs Original 175g"
##  [16] "Burger Rings 220g"
##  [17] "NCC Sour Cream &    Garden Chives 175g"
##  [18] "Doritos Corn Chip Southern Chicken 150g"
##  [19] "Cheezels Cheese Box 125g"
##  [20] "Smiths Crinkle      Original 330g"
##  [21] "Infzns Crn Crnchers Tangy Gcamole 110g"
##  [22] "Kettle Sea Salt     And Vinegar 175g"
##  [23] "Smiths Chip Thinly  Cut Original 175g"
##  [24] "Kettle Original 175g"
##  [25] "Red Rock Deli Thai  Chilli&Lime 150g"
##  [26] "Pringles Sthrn FriedChicken 134g"
##  [27] "Pringles Sweet&Spcy BBQ 134g"
##  [28] "Red Rock Deli SR    Salsa & Mzzrlla 150g"
##  [29] "Thins Chips         Originl saltd 175g"
##  [30] "Red Rock Deli Sp    Salt & Truffle 150G"
##  [31] "Smiths Thinly       Swt Chli&S/Cream175G"
##  [32] "Kettle Chilli 175g"
##  [33] "Doritos Mexicana    170g"
##  [34] "Smiths Crinkle Cut  French OnionDip 150g"
##  [35] "Natural ChipCo      Hony Soy Chckn175g"
##  [36] "Dorito Corn Chp     Supreme 380g"
##  [37] "Twisties Chicken270g"
##  [38] "Smiths Thinly Cut   Roast Chicken 175g"
##  [39] "Smiths Crinkle Cut  Tomato Salsa 150g"
##  [40] "Kettle Mozzarella   Basil & Pesto 175g"
##  [41] "Infuzions Thai SweetChili PotatoMix 110g"
##  [42] "Kettle Sensations   Camembert & Fig 150g"
##  [43] "Smith Crinkle Cut   Mac N Cheese 150g"
```

```
## [44] "Kettle Honey Soy    Chicken 175g"
## [45] "Thins Chips Seasonedchicken 175g"
## [46] "Smiths Crinkle Cut  Salt & Vinegar 170g"
## [47] "Infuzions BBQ Rib   Prawn Crackers 110g"
## [48] "GrnWves Plus Btroot & Chilli Jam 180g"
## [49] "Tyrrells Crisps     Lightly Salted 165g"
## [50] "Kettle Sweet Chilli And Sour Cream 175g"
## [51] "Doritos Salsa       Medium 300g"
## [52] "Kettle 135g Swt Pot Sea Salt"
## [53] "Pringles SourCream  Onion 134g"
## [54] "Doritos Corn Chips  Original 170g"
## [55] "Twisties Cheese     Burger 250g"
## [56] "Old El Paso Salsa   Dip Chnky Tom Ht300g"
## [57] "Cobs Popd Swt/Chlli &Sr/Cream Chips 110g"
## [58] "Woolworths Mild     Salsa 300g"
## [59] "Natural Chip Co     Tmato Hrb&Spce 175g"
## [60] "Smiths Crinkle Cut  Chips Original 170g"
## [61] "Cobs Popd Sea Salt  Chips 110g"
## [62] "Smiths Crinkle Cut  Chips Chs&Onion170g"
## [63] "French Fries Potato Chips 175g"
## [64] "Old El Paso Salsa   Dip Tomato Med 300g"
## [65] "Doritos Corn Chips  Cheese Supreme 170g"
## [66] "Pringles Original   Crisps 134g"
## [67] "RRD Chilli&         Coconut 150g"
## [68] "WW Original Corn    Chips 200g"
## [69] "Thins Potato Chips  Hot & Spicy 175g"
## [70] "Cobs Popd Sour Crm  &Chives Chips 110g"
## [71] "Smiths Crnkle Chip  Orgnl Big Bag 380g"
## [72] "Doritos Corn Chips  Nacho Cheese 170g"
## [73] "Kettle Sensations   BBQ&Maple 150g"
## [74] "WW D/Style Chip     Sea Salt 200g"
## [75] "Pringles Chicken    Salt Crips 134g"
## [76] "WW Original Stacked Chips 160g"
## [77] "Smiths Chip Thinly  CutSalt/Vinegr175g"
## [78] "Cheezels Cheese 330g"
## [79] "Tostitos Lightly    Salted 175g"
## [80] "Thins Chips Salt &  Vinegar 175g"
## [81] "Smiths Crinkle Cut  Chips Barbecue 170g"
## [82] "Cheetos Puffs 165g"
## [83] "RRD Sweet Chilli &  Sour Cream 165g"
## [84] "WW Crinkle Cut      Original 175g"
## [85] "Tostitos Splash Of  Lime 175g"
## [86] "Woolworths Medium   Salsa 300g"
## [87] "Kettle Tortilla ChpsBtroot&Ricotta 150g"
## [88] "CCs Tasty Cheese    175g"
## [89] "Woolworths Cheese   Rings 190g"
## [90] "Tostitos Smoked     Chipotle 175g"
## [91] "Pringles Barbeque   134g"
## [92] "WW Supreme Cheese   Corn Chips 200g"
## [93] "Pringles Mystery    Flavour 134g"
## [94] "Tyrrells Crisps     Ched & Chives 165g"
## [95] "Snbts Whlgrn Crisps Cheddr&Mstrd 90g"
## [96] "Cheetos Chs & Bacon Balls 190g"
## [97] "Pringles Slt Vingar 134g"
```

```
##  [98] "Infuzions SourCream&Herbs Veg Strws 110g"
##  [99] "Kettle Tortilla ChpsFeta&Garlic 150g"
## [100] "Infuzions Mango     Chutny Papadums 70g"
## [101] "RRD Steak &         Chimuchurri 150g"
## [102] "RRD Honey Soy       Chicken 165g"
## [103] "Sunbites Whlegrn    Crisps Frch/Onin 90g"
## [104] "RRD Salt & Vinegar  165g"
## [105] "Doritos Cheese      Supreme 330g"
## [106] "Smiths Crinkle Cut  Snag&Sauce 150g"
## [107] "WW Sour Cream &OnionStacked Chips 160g"
## [108] "RRD Lime & Pepper   165g"
## [109] "Natural ChipCo Sea  Salt & Vinegr 175g"
## [110] "Red Rock Deli Chikn&Garlic Aioli 150g"
## [111] "RRD SR Slow Rst     Pork Belly 150g"
## [112] "RRD Pc Sea Salt     165g"
## [113] "Smith Crinkle Cut   Bolognese 150g"
## [114] "Doritos Salsa Mild  300g"
```

```r
#We have 114 unique products in the Dataset
```

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 summarizing the individual words in the product name.

```r
#### Examine the words in PROD_NAME to see if there are any incorrect entries
#### such as products that are not chips
productWords <- as_tibble(unlist(strsplit(unique(transactionData$PROD_NAME), "\\s+")))
setnames(productWords, 'words')
```

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()`.

```r
#  Remove digits, and special characters, and then sort the distinct words by frequency of occurrence.
#### Removing digits
 productWords <- productWords[!grepl("\\d+",productWords$words),]

#### Removing special characters
productWords<- productWords[!grepl("&",productWords$words),]
#### Let's look at the most common words by counting the number of times a word
productWords %>% group_by(words) %>% summarize(n = n()) %>%
#### sorting them by this frequency in order of highest to lowest frequency
arrange(desc(n))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 171 x 2
##    words        n
##    <chr>    <int>
##  1 Chips       21
##  2 Smiths      16
##  3 Crinkle     14
##  4 Cut         14
##  5 Kettle      13
```

```
##   6 Cheese      12
##   7 Salt        12
##   8 Original    10
##   9 Chip         9
##  10 Doritos      9
## # ... with 161 more rows
```

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

```r
#### Remove salsa products
transactionData <- transactionData[!grepl("salsa", tolower(transactionData$PROD_NAME)), ]
```

Summary Statistics

```r
#### Summarise the data to check for nulls and possible outliers
#summarizing
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
```

The data seems to be in check with no missing values and obvious outliers are only in product quantity and total sales.

```r
#### Filter the dataset to find the outlier
filter(transactionData, PROD_QTY == 200)
```

```
## # A tibble: 2 x 8
##   DATE       STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_QTY
##   <date>         <dbl>          <dbl>  <dbl>    <dbl> <chr>        <dbl>
## 1 2018-08-19       226         226000 226201        4 Dorito C~      200
## 2 2019-05-20       226         226000 226210        4 Dorito C~      200
## # ... with 1 more variable: TOT_SALES <dbl>
```

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

```r
#### Let's see if the customer has had other transactions
filter(transactionData, LYLTY_CARD_NBR == 226000)
```

```
## # A tibble: 2 x 8
##   DATE       STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_QTY
##   <date>         <dbl>          <dbl>  <dbl>    <dbl> <chr>        <dbl>
## 1 2018-08-19       226         226000 226201        4 Dorito C~      200
## 2 2019-05-20       226         226000 226210        4 Dorito C~      200
## # ... with 1 more variable: TOT_SALES <dbl>
```

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.

```r
#### Filter out the customer based on the loyalty card number
transactionData <- filter(transactionData, !LYLTY_CARD_NBR == 226000)
#### Re-examine transaction data
transactionData
```

```
## # A tibble: 246,740 x 8
##    DATE       STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_QTY
##    <date>         <dbl>          <dbl>  <dbl>    <dbl> <chr>        <dbl>
##  1 2018-10-17         1           1000      1        5 Natural ~        2
##  2 2019-05-14         1           1307    348       66 CCs Nach~        3
##  3 2019-05-20         1           1343    383       61 Smiths C~        2
##  4 2018-08-17         2           2373    974       69 Smiths C~        5
##  5 2018-08-18         2           2426   1038      108 Kettle T~        3
##  6 2019-05-16         4           4149   3333       16 Smiths C~        1
##  7 2019-05-16         4           4196   3539       24 Grain Wa~        1
##  8 2018-08-20         5           5026   4525       42 Doritos ~        1
##  9 2018-08-18         7           7150   6900       52 Grain Wa~        2
## 10 2019-05-17         7           7215   7176       16 Smiths C~        1
## # ... with 246,730 more rows, and 1 more variable: TOT_SALES <dbl>
```

That's better. 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.

```r
#### Count the number of transactions by date
transactions_by_day <- group_by(transactionData, DATE) %>% summarise(N = n())
```

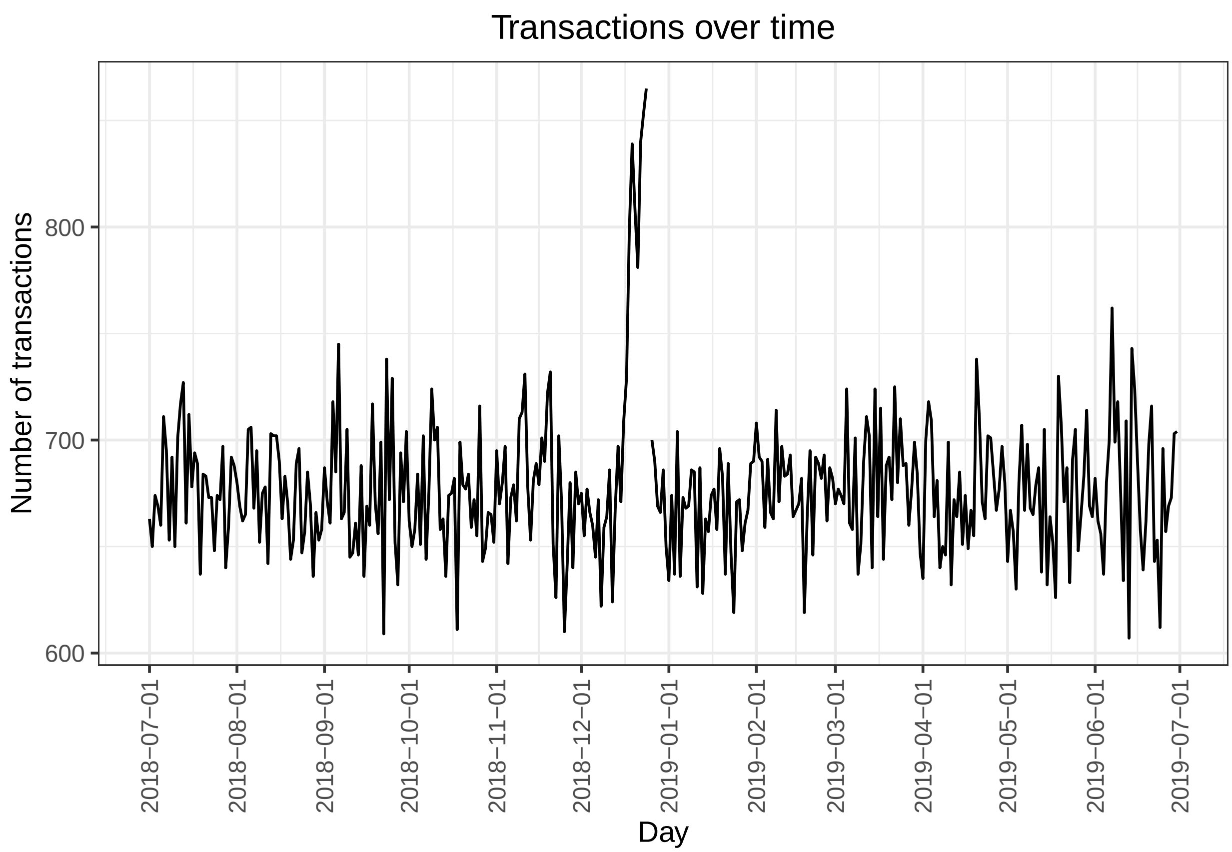
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

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.

```r
#### Create a sequence of dates and join this the count of transactions by date
DATE <- seq.Date(from = as.Date("2018-07-01"), to = as.Date("2019-06-30"), by = "day")
#convert to tibble for join
DATE<- data.frame(DATE)

transactions_by_day <- right_join(transactions_by_day, DATE, by = "DATE")
#find missing day
transactions_by_day[is.na(transactions_by_day$N),]
```

```
## # A tibble: 1 x 2
##   DATE           N
##   <date>     <int>
## 1 2018-12-25    NA
```

```
#### Setting plot themes to format graphs
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
#### Plot transactions over time
ggplot(transactions_by_day, aes(x = DATE, y = N)) +
 geom_line() +
 labs(x = "Day", 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))
```

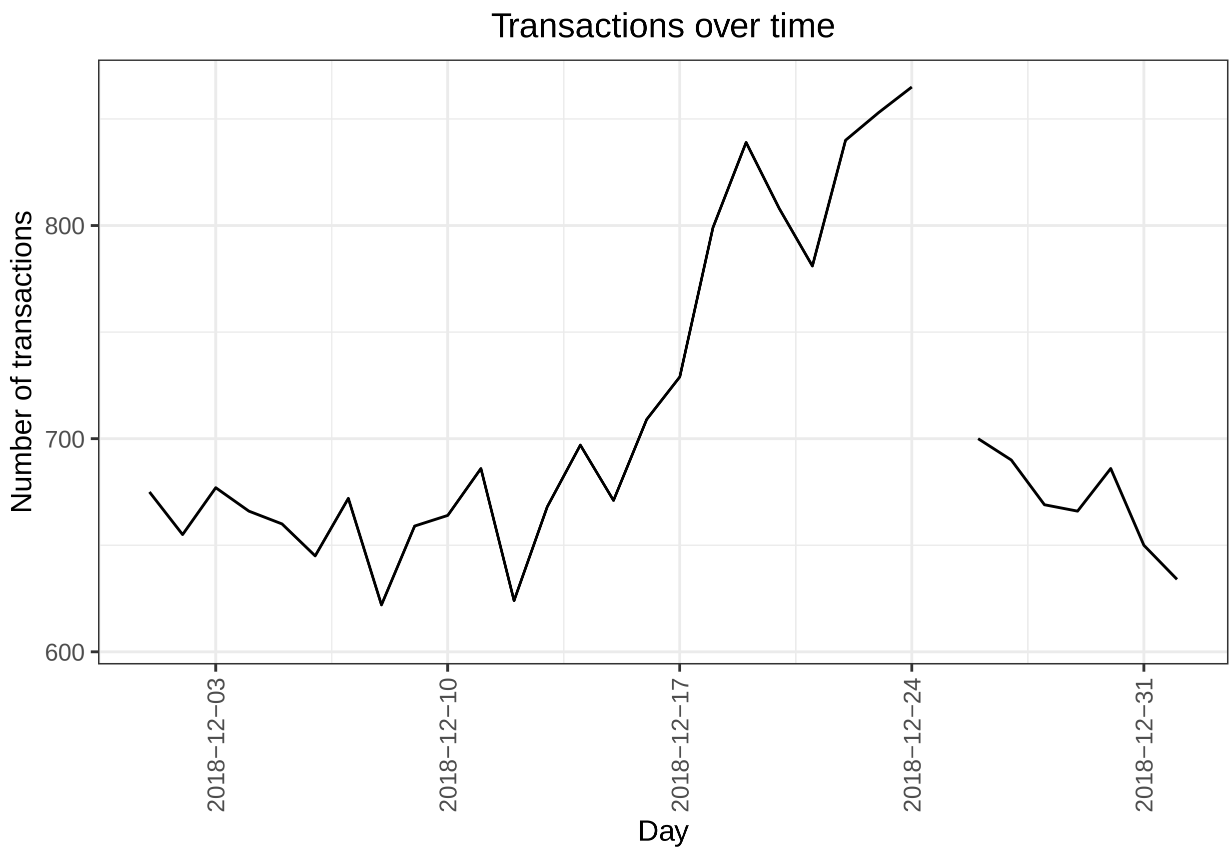


We can see that there is an increase in purchases in December and a break in late December, form the code chunk above we can see that Christmas day data is missing . Let's zoom in on this.

```
#### Filter to December and look at individual days
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
#### Plot transactions over time
ggplot(transactions_by_day, aes(x = DATE, y = N)) +
 geom_line() +
 labs(x = "Day", y = "Number of transactions", title = "Transactions over time") +
```

```
scale_x_date(breaks = "1 week", limits = c(as.Date("2018-12-01"), as.Date("2019-01-01"))) +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```

```
## Warning: Removed 333 row(s) containing missing values (geom_path).
```



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.

```
#### Pack size
#### We can work this out by taking the digits that are in PROD_NAME
transactionData$PACK_SIZE <- parse_number(transactionData$PROD_NAME)
head(transactionData$PACK_SIZE)
```

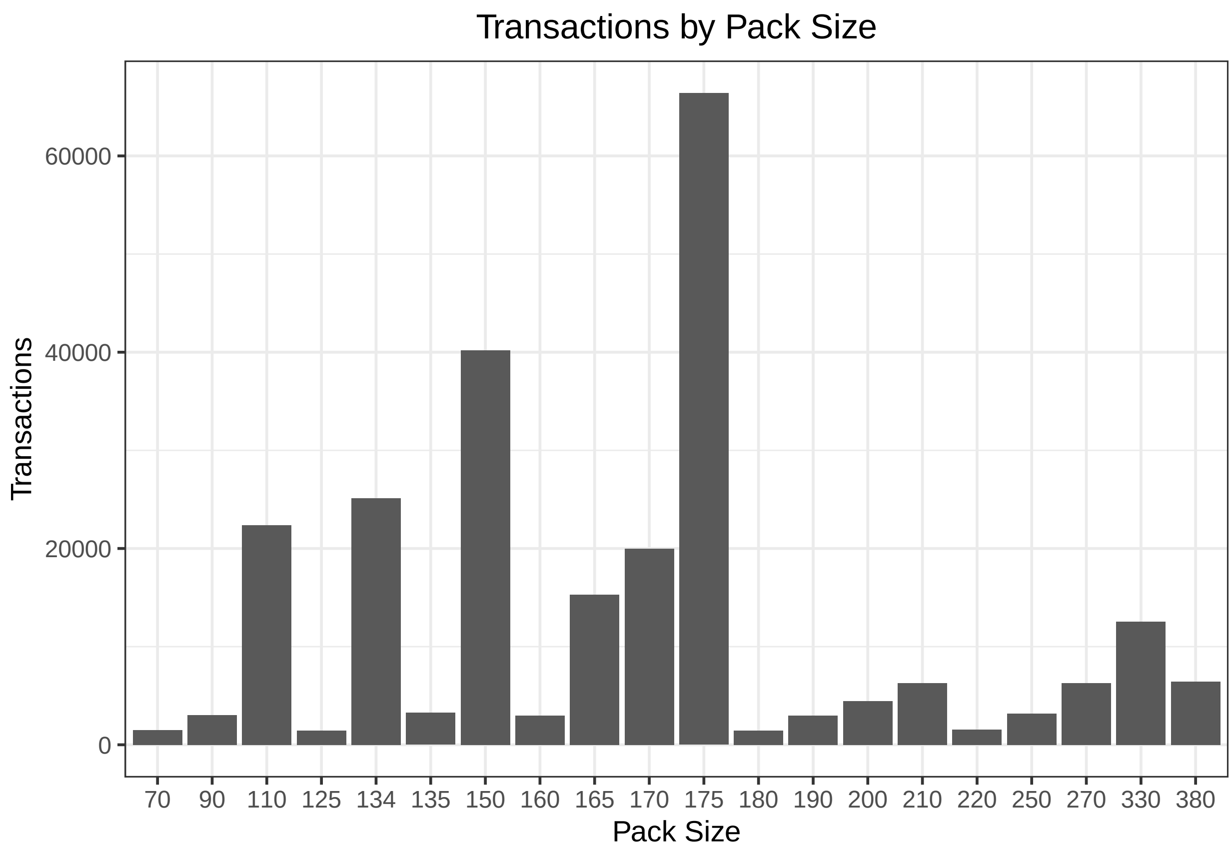
```
## [1] 175 175 170 175 150 330
```

```
#### Let's check if the pack sizes look sensible
summary(transactionData$PACK_SIZE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    70.0   150.0   170.0   175.6   175.0   380.0
```

The largest size is 380g and the smallest size is 70g - seems sensible!

```r
ggplot(transactionData, aes(factor(PACK_SIZE))) +
  geom_bar() +
  ylab("Transactions") +
  xlab("Pack Size") +
  ggtitle("Transactions by Pack Size")
```



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

```r
#### extract first word in PROD_NAME
transactionData <- mutate(transactionData, BRAND = str_extract(PROD_NAME,"^[a-zA-Z]+\\s"), BRAND = str_t


head(transactionData$BRAND)
```

```
## [1] "Natural" "CCs"     "Smiths"  "Smiths"  "Kettle"  "Smiths"
```

```r
#### Checking brands
```

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.

```r
#### Clean brand names
brandNames <- group_by(transactionData, BRAND) %>%
  summarise(N = n()) %>% arrange(desc(N))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```r
#check brandnames
transactionData <-mutate(transactionData, BRAND = dplyr::recode(BRAND, Dorito = "Doritos",
                                      Red = "RRD",
                                      NCC = "Natural",
                                      Infzns = "Infuzions",
                                      Snbts = "Sunbits",
                                      Grnwvs = "Grain"))
#check the new brand names

unique(transactionData$BRAND)
```

```
##  [1] "Natural"    "CCs"        "Smiths"     "Kettle"     "Grain"
##  [6] "Doritos"    "Twisties"   "WW"         "Thins"      "Burger"
## [11] "Cheezels"   "Infuzions"  "RRD"        "Pringles"   "Smith"
## [16] "GrnWves"    "Tyrrells"   "Cobs"       "French"     "Tostitos"
## [21] "Cheetos"    "Woolworths" "Sunbits"    "Sunbites"
```

Examining customer data

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

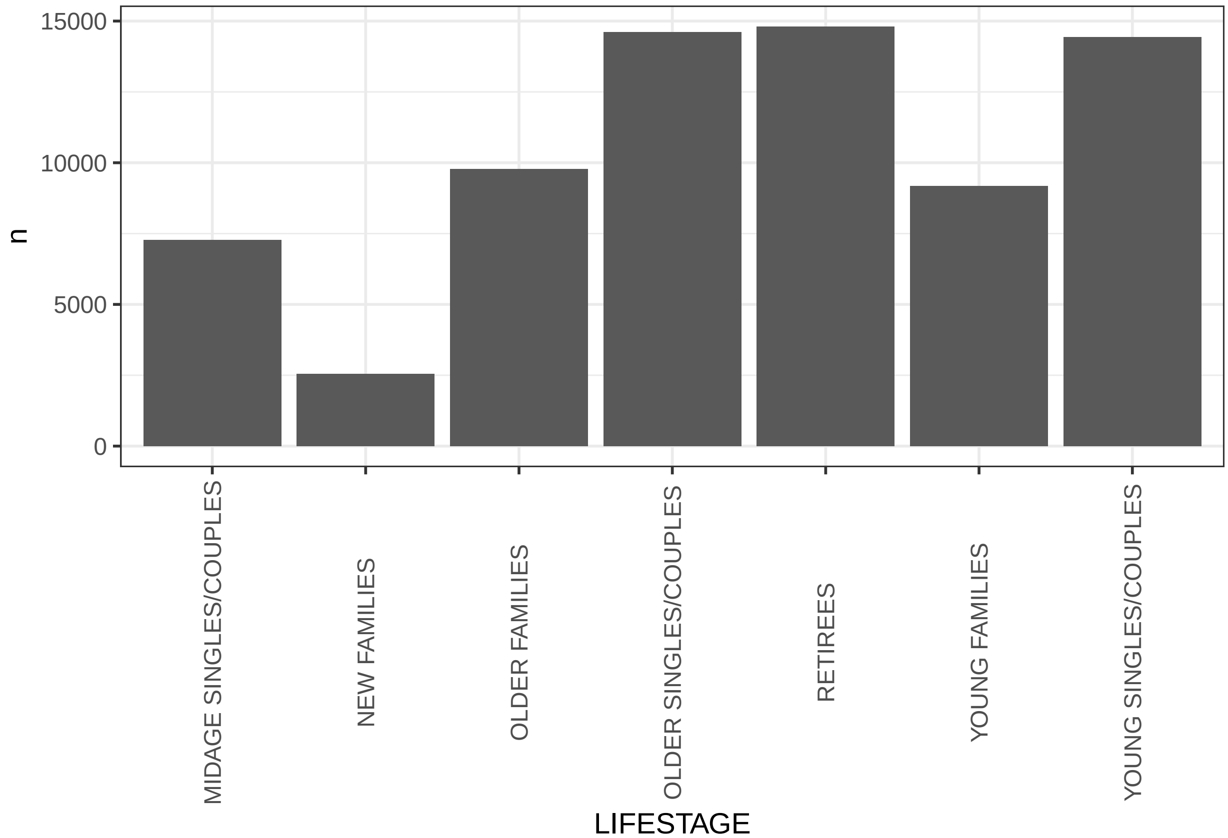
```r
#Examining customer data
head(customerData)
```

```
## # A tibble: 6 x 3
##   LYLTY_CARD_NBR LIFESTAGE              PREMIUM_CUSTOMER
##            <dbl> <chr>                  <chr>
## 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
```

```r
#distribution of lifestage
by_lifestage <- group_by(customerData, LIFESTAGE) %>% summarise(n = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

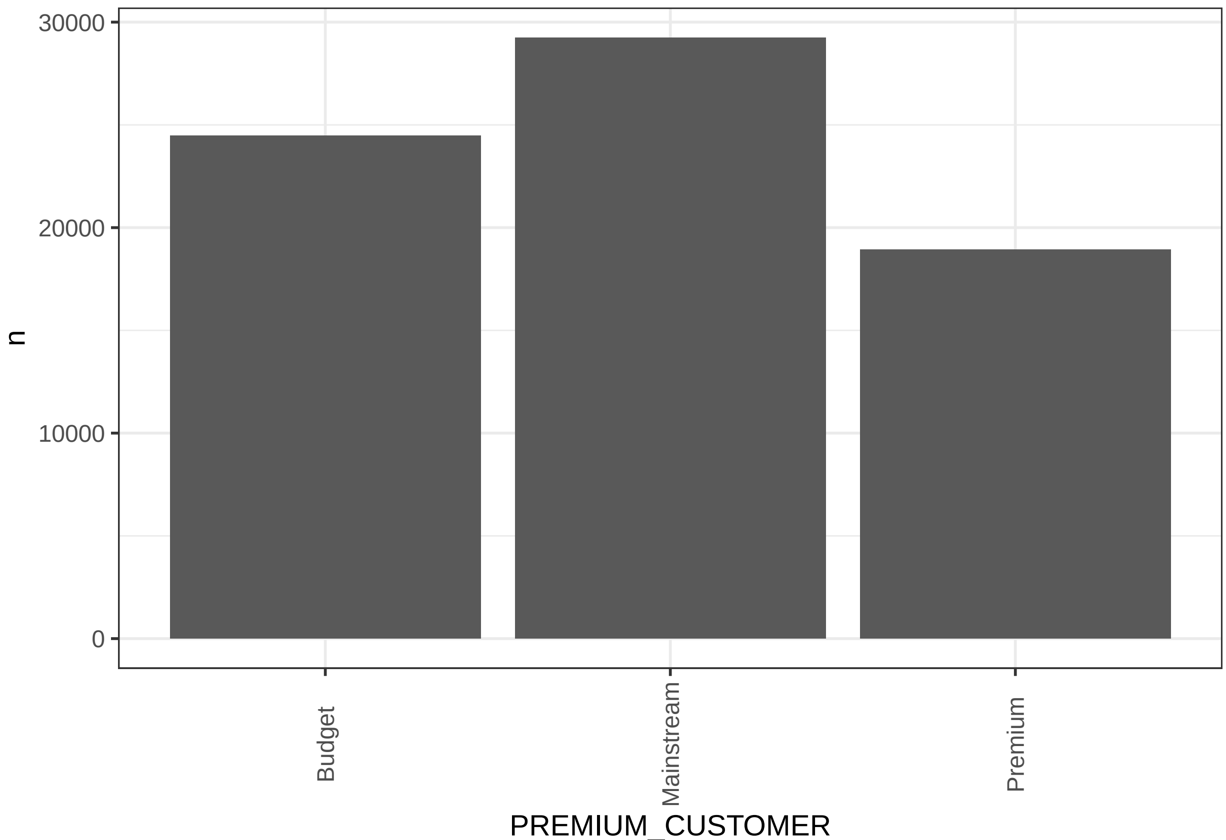
```r
#They are 7 levels of LIFESTAGE
by_lifestage %>%
  ggplot(aes(LIFESTAGE, n))+
  geom_col() +
 theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




```
#by membership
by_membership <- group_by(customerData, PREMIUM_CUSTOMER) %>% summarise(n = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
#They are 7 levels of LIFESTAGE
by_membership %>%
  ggplot(aes(PREMIUM_CUSTOMER, n))+
  geom_col() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




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

data <- mutate(data, price_per_qty = TOT_SALES/PROD_QTY)
```

As the number of rows in `data` is the same as that of `transactionData`, we can be sure that no duplicates were created. This is because we created `data` by setting `all.x = TRUE` (in other words, a left join) which means take all the rows in `transactionData` and find rows with matching values in shared columns and then joining the details in these rows to the `x` or the first mentioned table.

```
sum(is.na(data$LIFESTAGE))
```

```
## [1] 0
```

```
sum(is.na(data$PREMIUM_CUSTOMER))
```

```
## [1] 0
```

Great, there are no nulls! So all our customers in the transaction data has been accounted for in the customer dataset. Note that if you are continuing with Task 2, you may want to retain this dataset which you can write out as a csv

```
fwrite(data, "C:/Users/Omorinola O/Documents/Quantium Virtual Internship/QVI_data.csv")
```

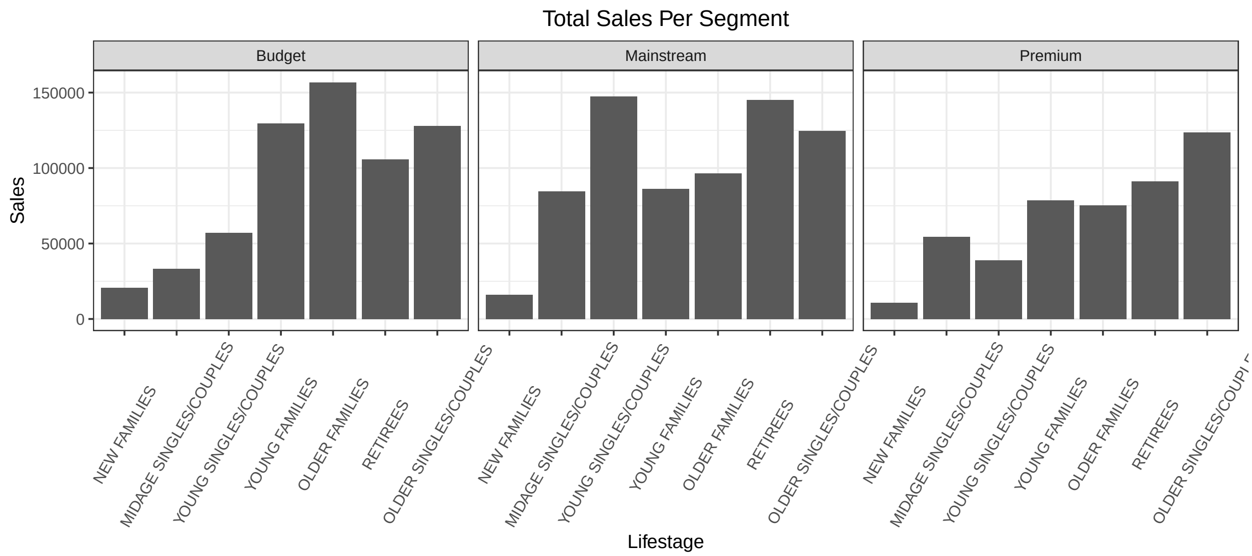
Data exploration is now complete!

Data analysis on customer segments Now that the data is ready for analysis, we can define some metrics of interest to the client: - Who spends the most on chips (total sales), describing customers by life stage and how premium their general purchasing behavior is - How many customers are in each segment - How many chips are bought per customer by segment - What's the average chip price by customer segment , We could also ask our data team for more information. Examples are: - The customer's total spend over the period and total spend for each transaction to understand what proportion of their grocery spend is on chips - Proportion of customers in each customer segment overall to compare against the mix of customers who purchase chips 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.

```
#### Total sales by LIFESTAGE and PREMIUM_CUSTOMER
segments <- group_by(data, LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarize(N = n(),
            sales = sum(TOT_SALES),
            mean_packsize = mean(PACK_SIZE),
            units = sum(PROD_QTY),
            avg_unit = mean(PROD_QTY),
            avg_ppq = mean(price_per_qty)) %>%
  ungroup()
```

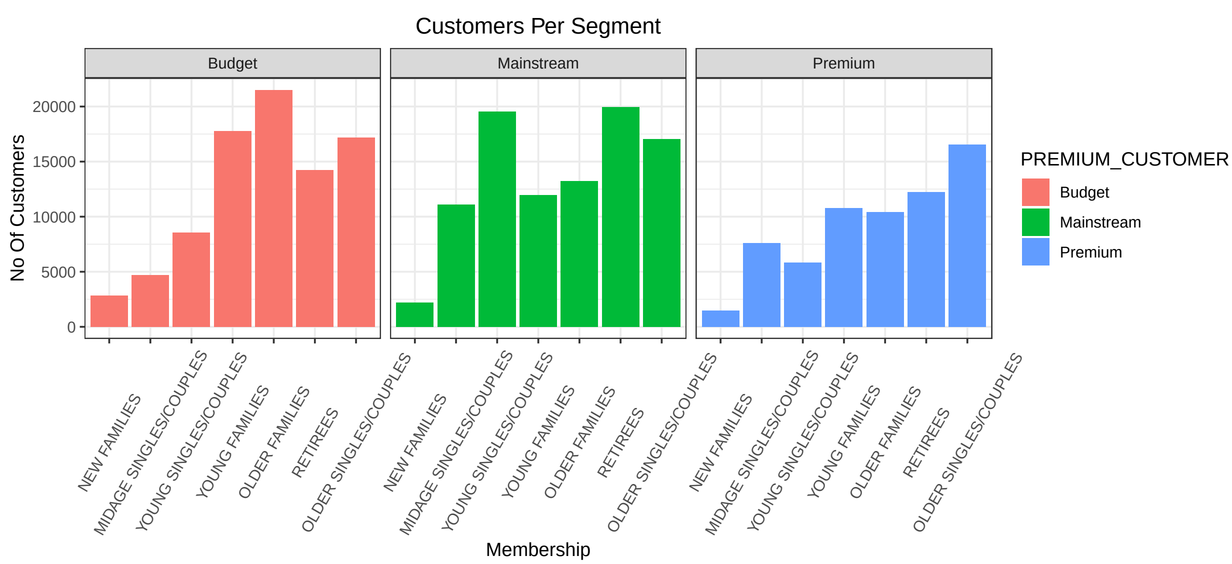
```
## `summarise()` regrouping output by 'LIFESTAGE' (override with `.groups` argument)
```

```
#total vs lifestage
segments %>% ggplot(aes(reorder(LIFESTAGE, sales), sales)) +
  geom_col() +
  facet_wrap(~PREMIUM_CUSTOMER) +
  theme(axis.text.x = element_text(angle = 60, vjust = 0.5)) +
  ggtitle("Total Sales Per Segment") +
  xlab("Lifestage") +
  ylab("Sales")
```



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.

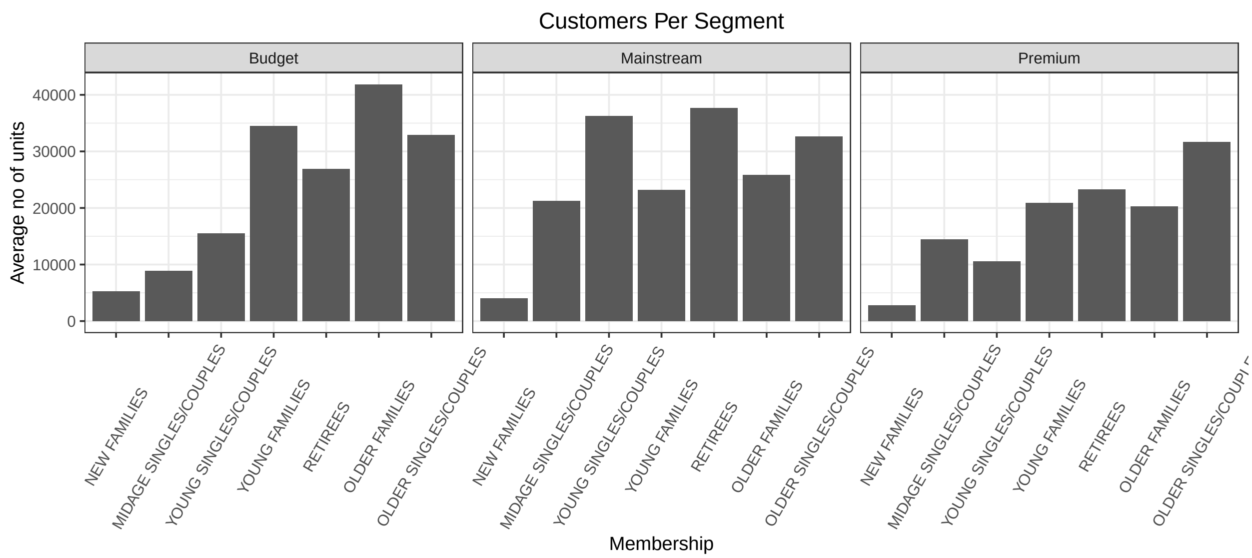
```
#### Number of customers by LIFESTAGE and PREMIUM_CUSTOMER
segments %>% ggplot(aes(reorder(LIFESTAGE, N), N, fill = PREMIUM_CUSTOMER)) +
  geom_col() +
  facet_grid(~PREMIUM_CUSTOMER) +
  theme(axis.text.x = element_text(angle = 60, vjust = 0.5)) +
  ggtitle("Customers Per Segment") +
  xlab("Membership") +
  ylab("No Of Customers")
```



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.

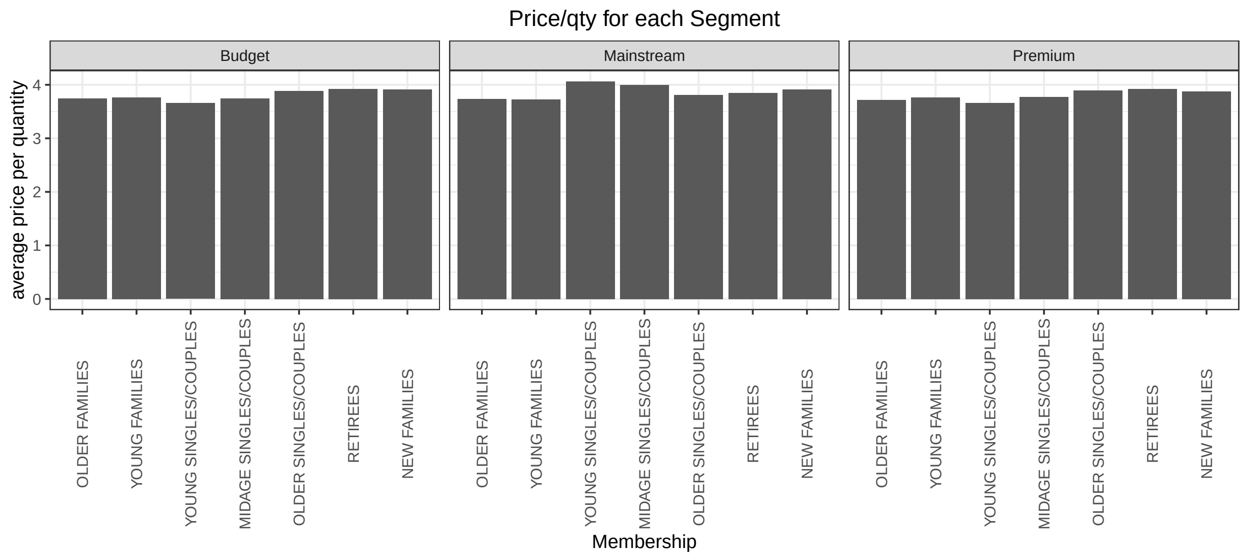
```
#### number of units per customer by LIFESTAGE and PREMIUM_CUSTOMER
segments %>% ggplot(aes(reorder(LIFESTAGE, units), units)) +
  geom_col() +
  facet_wrap(~PREMIUM_CUSTOMER) +
  theme(axis.text.x = element_text(angle = 60, vjust = 0.5)) +
  ggtitle("Customers Per Segment") +
  xlab("Membership") +
  ylab("Average no of units")
```




```
# Over to you! Calculate and plot the average number of units per customer by those  two dimensions.
```

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.

```
#### price per unit by LIFESTAGE and PREMIUM_CUSTOMER
segments %>% ggplot(aes(reorder(LIFESTAGE, avg_ppq), avg_ppq)) +
  geom_col() +
  facet_wrap(~PREMIUM_CUSTOMER) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5)) +
  ggtitle("Price/qty for each Segment") +
  xlab("Membership") +
  ylab("average price per quantity")
```



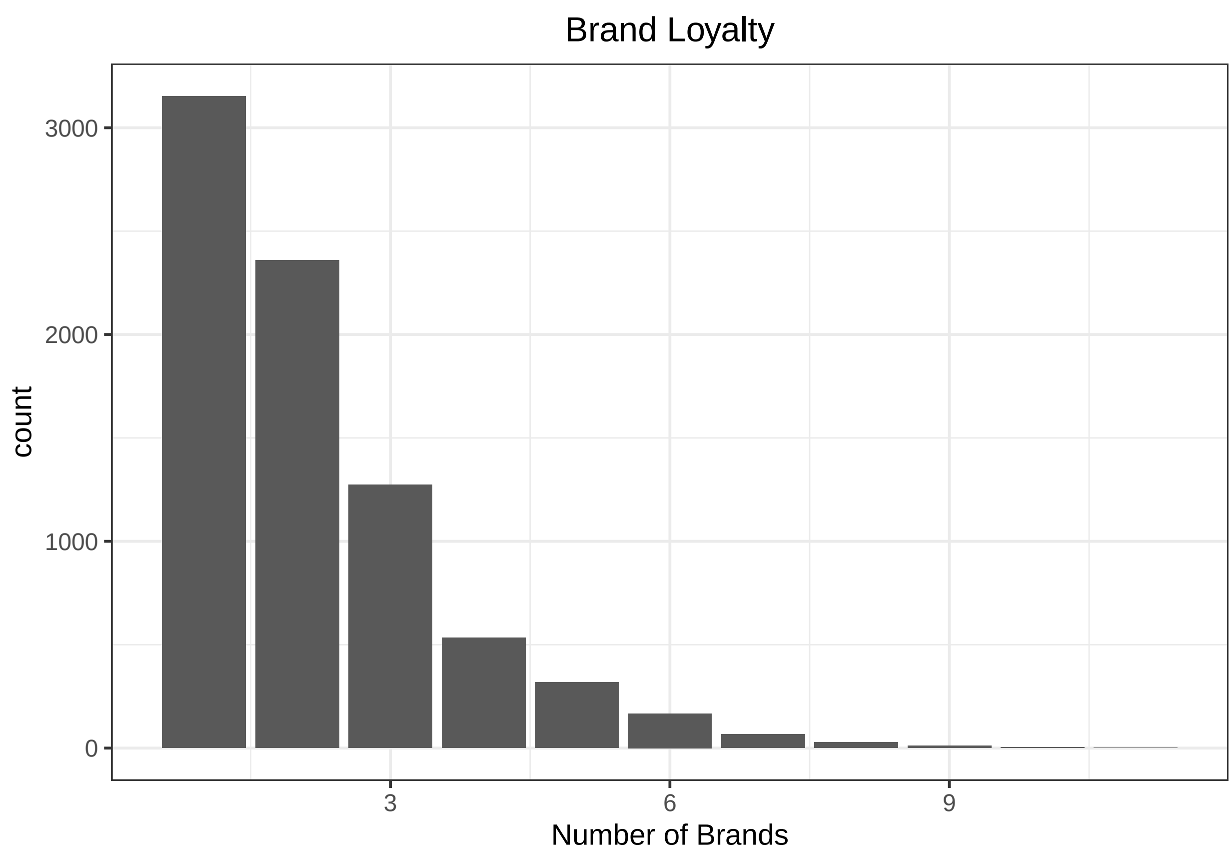
Mainstream mid age and young singles and couples are more willing to pay more per packet of chips compared to their budget and premium counterparts. This may be due to premium shoppers being more likely to buy healthy snacks and when they buy chips, this is mainly for entertainment purposes rather than their own consumption. This is also supported by there being fewer premium mid age and young singles and couples buying chips compared to their mainstream counterparts. As the difference in average price per unit isn't large, we can check if this difference is statistically significant.

The t-test results in a p-value of 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 mid age 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

newdata <- unite(data, segments, LIFESTAGE:PREMIUM_CUSTOMER, sep = "_") %>%
  mutate(segments = as.factor(segments))

#pull segments
segLevels <- levels(newdata$segments)

#create function
filter_dataframe <- function(...){
  filter(newdata, segments == ...)
}
#create a new dataframe for every segment
dataList <- lapply(segLevels, filter_dataframe)



#pull target market data from index 20
dataList[[20]] %>%
  group_by(LYLTY_CARD_NBR) %>%
  summarize(N = n_distinct(BRAND)) %>%
  ggplot(aes(N)) +
```

```
  geom_bar() +
  ggtitle("Brand Loyalty") +
  xlab("Number of Brands")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```
#we can see here that most customers show high brand loyalty
```

The plots above show that most customers in the young-mainstream dataset stick to a only a handful of brands

```
#find frequent Item sets for each
frame <- dataList[[20]] #pull dataframe
tmp <- split(frame$BRAND,  frame$LYLTY_CARD_NBR) #split by lylty card(list)
Trx <- methods::as(tmp, "transactions") #from list to transactions

#extract itemsets
itemsets <- arules::apriori(Trx, parameter = list(
                                  target="frequent itemsets",
                                  supp = 0.01,
                                  minlen = 2))

#extract rules
rules <- arules::apriori(Trx, parameter = list(
```

```
                                    target="rules",
                                    supp = 0.01,
                                    conf = 0.4))
#extract rules
itemsets <- arules::apriori(Trx, parameter = list(
                                    target="frequent itemsets",
                                    supp = 0.01,
                                    conf = 0.4))

#rules by lift
arules::inspect(sort(rules, by='lift', decreasing = T))

#rules by support
arules::inspect(sort(rules, by='support', decreasing = T))

#mainstream youngins  descending
items <- arules::inspect(sort(itemsets, by='support', decreasing = T))


#mainstream yougins multiple items ascending
arules::inspect(sort(itemsets, by='support', decreasing = F)[1:20])
```

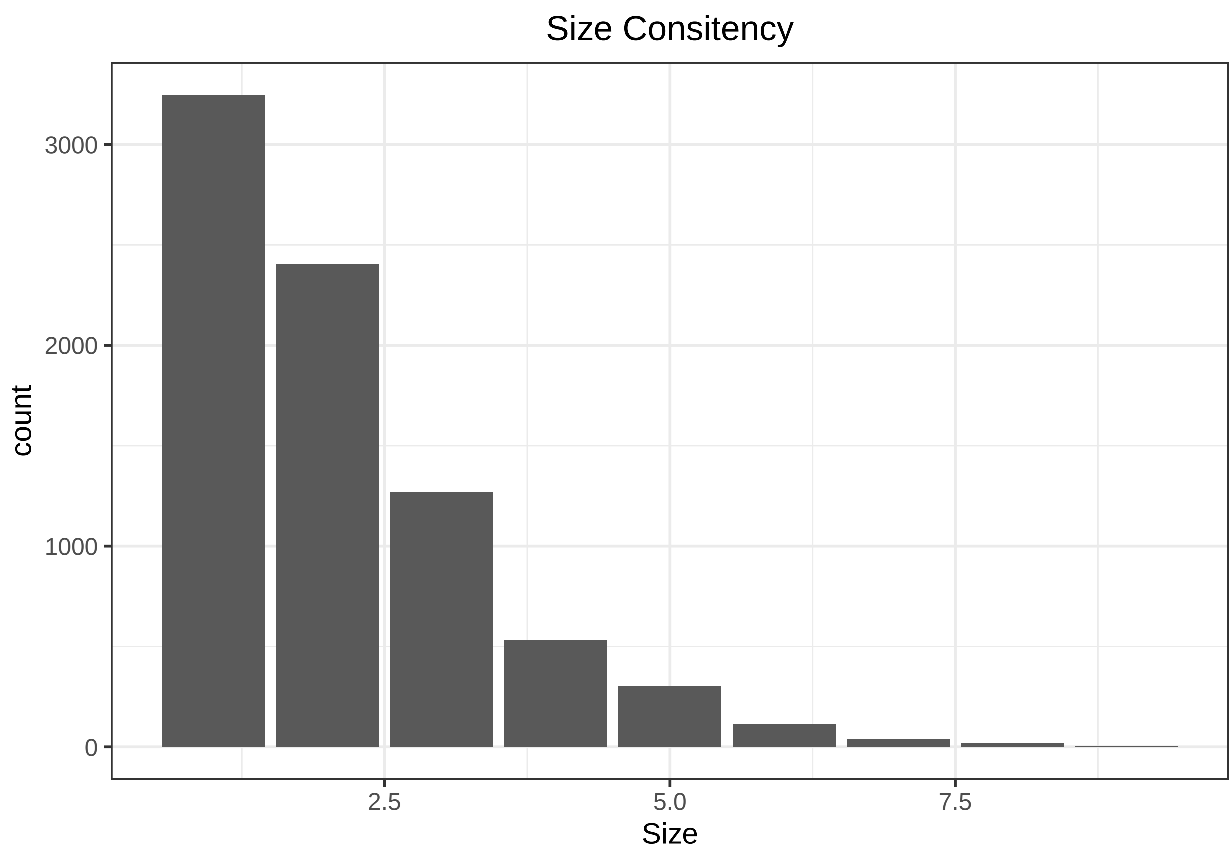
```
#mine frequent Item sets for each segments
storage <- list()

for(i in 1:21){
frame_loop <- dataList[[i]] #pull dataframe

#split by lylty card(list)
tmp_loop <- split(frame_loop$BRAND, frame_loop$LYLTY_CARD_NBR)

#from list to transactions
Trx_loop <- as(tmp_loop, "transactions")

#mine itemsets
 storage[[i]] <- arules::apriori(Trx_loop, parameter = list(
                                    target="frequent itemsets",
                                    supp = 0.01
                                    ))

}
#mined rule for each segment is stored in the "Storage" variable
```

We can see that : most customers in the young-mainstream segment prefer the kettle brand followed by Doritos and Pringles, we can also see that they often purchase kettles and Pringles together. Let's also find out if our target segment tends to buy larger packs of chips.

```
#### Preferred pack size compared to the rest of the population

#pull target market data from index 20
dataList[[20]] %>%
  group_by(LYLTY_CARD_NBR) %>%
  summarize(N = n_distinct(PACK_SIZE)) %>%
```

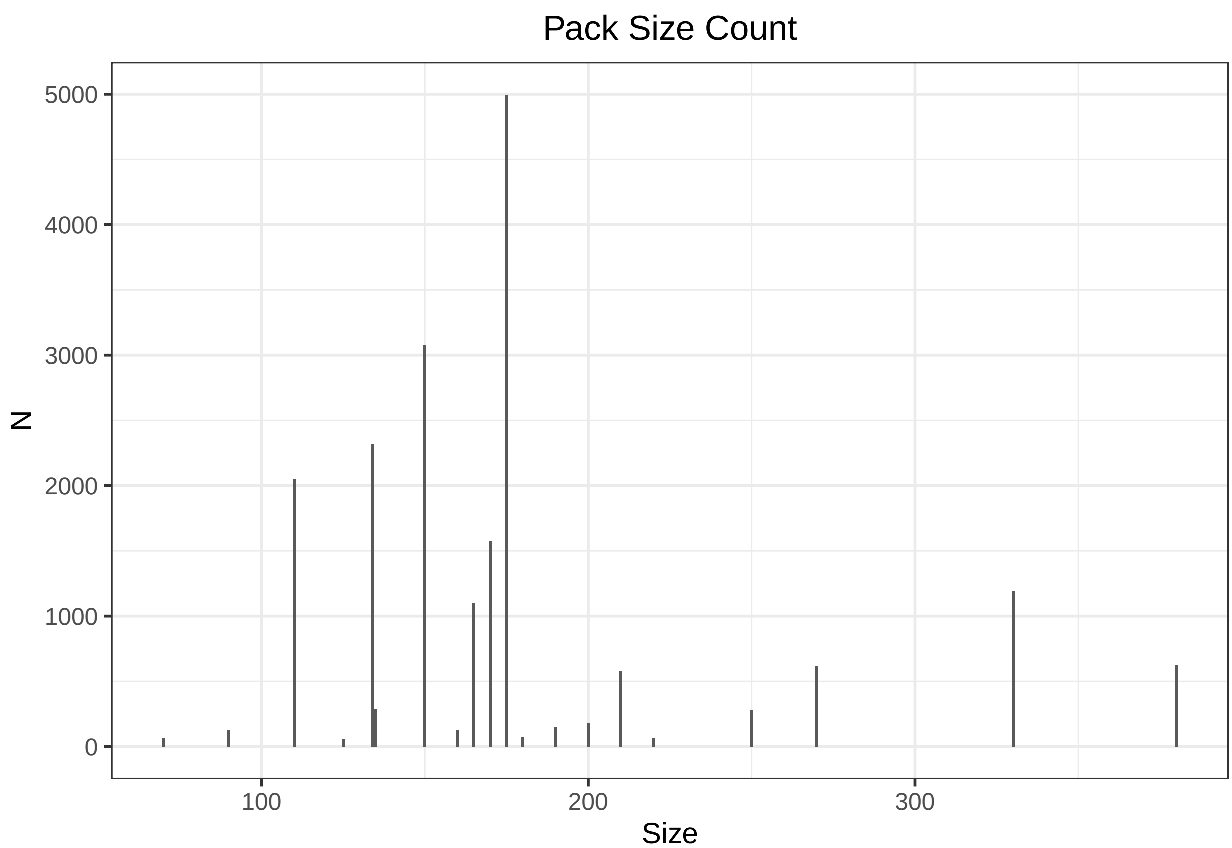
```r
  ggplot(aes(N)) +
  geom_bar() +
  ggtitle("Size Consitency") +
  xlab("Size")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```




```r
#the most common pack size is 175g
dataList[[20]] %>%
  group_by(PACK_SIZE) %>%
  summarize(N = n()) %>%
  ggplot(aes(PACK_SIZE, N)) +
  geom_col() +
  ggtitle("Pack Size Count") +
  xlab("Size")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```



The first plot above represents the affinity for a buying particular pack sizes, by first computing the amount of distinct sizes a customer buys, this revealed that about most of the customers in this segment stick to the 175g pack size.

```r
#create empty lists
rules_size <- list()
itemsets_size <- list()

#loop
for(i in 1:21){
frame <- dataList[[i]] #pull dataframe
tmp_size <- split(frame$PACK_SIZE, frame$LYLTY_CARD_NBR) #split by lylty card(list)
Trx_size <- as(tmp_size, "transactions") #from list to transactions


#extract rules
rules_size[[i]] <- arules::apriori(Trx_size, parameter = list(
                                  target="rules",
                                  supp = 0.01,
                                  conf = 0.4))
#extract rules
itemsets_size[[i]] <- arules::apriori(Trx_size, parameter = list(
                                  target="frequent itemsets",
                                  supp = 0.01,
                                  conf = 0.4))
}
```

```r
#rules by lift
arules::inspect(sort(itemsets_size[[20]], by='support', decreasing = T))


arules::inspect(sort(rules_size[[20]], by='lift', decreasing = T))
```

Most customers tend to buy chips at 175g pack size, and customers that buy 200g also buy 150g. Most customers show brand loyalty and a strong preference for pack sizes, we also established that most sales come young-mainstream customer segment, and as such designated them our target market. We also saw a statistically significant difference in buying patterns among customer segments, this somewhat justifies our choice to target a particular segments.