quantium_task1

Dai Taniguchi

2023-10-17

Before everything, load the required libraries and data sets

```
pacman::p_load(ggplot2, dplyr , readxl ,data.table, ggmosaic, readr)
```

Converting the data set into a data.table which allows me to use data.table-specific functions and syntax

```
transaction_data = read_excel("QVI_transaction_data.xlsx")
purchase_behaviour = read.csv("QVI_purchase_behaviour.csv")
transaction_data = data.table(transaction_data)
purchase_behaviour = data.table(purchase_behaviour)
```

Below is the workflow of this paper

Exploratory data analysis flow

1. **Correct data types**
2. **Remove outliers**
3. **Spot data issues in transaction dates**
4. **Explore pack sizes**
5. **Correct wrongly or differently spelled brand names**
6. **Examine the number of customers and segments**

Customer analysis flow

1. **The proportion of sales by segments**
2. **The proportion of customers by segments**
3. **The number of units sold per customer**
4. **The average price per unit spent by each customer during a transaction.**
5. **Conduct independent T-test**
6. **Identify the target customers that have the greatest impact on sales.**
7. **identify the most popular pack sizes and brands among the target customer base compared to others**

Okay let’s get started

Exploratory data analyis

1. Correct data types

First let's look at the data set. Apparently this data set contains 264,836 transaction data. The format of DATE column is num and value starts with 43282 meaning that the day is 43282 days away from the origin date in excel that is 30 December 1899.

```
str(transaction_data)
```

```
## Classes 'data.table' and 'data.frame':   264836 obs. of  8 variables:
##  $ DATE          : num  43390 43599 43605 43329 43330 ...
##  $ STORE_NBR     : num  1 1 1 2 2 4 4 4 5 7 ...
##  $ LYLTY_CARD_NBR: num  1000 1307 1343 2373 2426 ...
##  $ TXN_ID        : num  1 348 383 974 1038 ...
##  $ PROD_NBR      : num  5 66 61 69 108 57 16 24 42 52 ...
##  $ PROD_NAME     : chr  "Natural Chip        Compny SeaSalt175g" "CCs Nacho Cheese
175g" "Smiths Crinkle Cut  Chips Chicken 170g" "Smiths Chip Thinly  S/Cream&Onion 175
g" ...
##  $ PROD_QTY      : num  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>
```

The DATE column is converted into date format. Others look fine.

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

Remove duplicates, 3 found

```
unique(transaction_data)
```

```
##               DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##      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
##     ---
## 264831: 2019-03-09       272         272319 270088       89
## 264832: 2018-08-13       272         272358 270154       74
## 264833: 2018-11-06       272         272379 270187       51
## 264834: 2018-12-27       272         272379 270188       42
## 264835: 2018-09-22       272         272380 270189       74
##                                        PROD_NAME PROD_QTY TOT_SALES
##      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
##     ---
## 264831:  Kettle Sweet Chilli And Sour Cream 175g        2      10.8
## 264832:            Tostitos Splash Of  Lime 175g        1       4.4
## 264833:                 Doritos Mexicana    170g        2       8.8
## 264834:  Doritos Corn Chip Mexican Jalapeno 150g        2       7.8
## 264835:            Tostitos Splash Of  Lime 175g        2       8.8
```

2. Remove outliers

To find something unusual, look at mean, min, and max values. The max value of PROD_QTY is 200 (purchased 200 packs at one transaction) which is unusual and considered outlier.

There are no nulls as all the summary statistics have a numerical value

```
summary(transaction_data)
```

```
##       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.:  70021   1st Qu.:  67602
##  Median :2018-12-30   Median :130.0   Median : 130358   Median : 135138
##  Mean   :2018-12-30   Mean   :135.1   Mean   : 135550   Mean   : 135158
##  3rd Qu.:2019-03-31   3rd Qu.:203.0   3rd Qu.: 203094   3rd Qu.: 202701
##  Max.   :2019-06-30   Max.   :272.0   Max.   :2373711   Max.   :2415841
##     PROD_NBR       PROD_NAME            PROD_QTY         TOT_SALES
##  Min.   :  1.00   Length:264836      Min.   :  1.000   Min.   :  1.500
##  1st Qu.: 28.00   Class :character   1st Qu.:  2.000   1st Qu.:  5.400
##  Median : 56.00   Mode  :character   Median :  2.000   Median :  7.400
##  Mean   : 56.58                      Mean   :  1.907   Mean   :  7.304
##  3rd Qu.: 85.00                      3rd Qu.:  2.000   3rd Qu.:  9.200
##  Max.   :114.00                      Max.   :200.000   Max.   :650.000
```

Filter the data set to find the outlier. Turns out the same customer did those two transactions at the same store.

```
filter(transaction_data, PROD_QTY == 200)
```

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

Check other past transactions of that customer who caused the outlier. No other transactions were done by the customer except for those two where he purchased 200 packs of chips.

```
transaction_data[LYLTY_CARD_NBR == 226000, ]
```

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

Remove the customer as we don't need outliers. They are likely to be a bug.

```
transaction_data = transaction_data[LYLTY_CARD_NBR != 226000]
```

3. Spot data issues in transaction dates

Try to find other data issues such as missing data or outlier by looking at date and transaction. And this shows only 364 days, not 365 days. it's missing 1 day.

```
transaction_data[,.N, by = DATE]
```

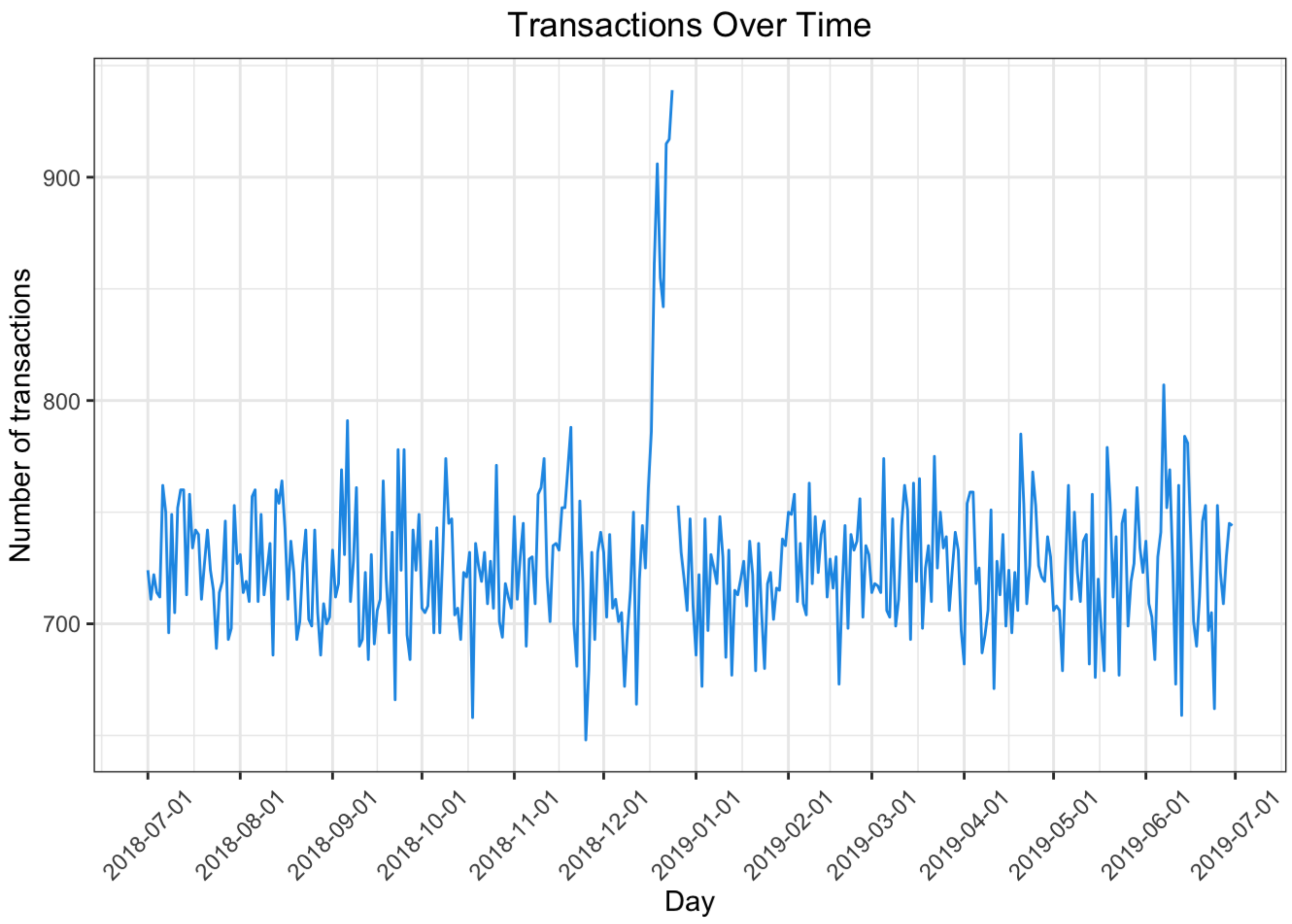
```
##            DATE   N
##   1: 2018-10-17 732
##   2: 2019-05-14 758
##   3: 2019-05-20 754
##   4: 2018-08-17 711
##   5: 2018-08-18 737
##  ---
## 360: 2018-11-21 700
## 361: 2019-05-10 710
## 362: 2018-12-08 672
## 363: 2019-01-30 738
## 364: 2019-02-09 718
```

Count the number of transaction by date

```
#Create a sequence of dates and join this the count of transactions by date
all_dates = data.table(seq(as.Date("2018-07-01"), as.Date("2019-06-30"), by = 'day'))
setnames(all_dates, 'DATE')
#join all_dates and tranasaction_data (left join) by DATE column
transaction_by_day = merge(all_dates, transaction_data[,.N, by = DATE], all.x = TRUE)
```

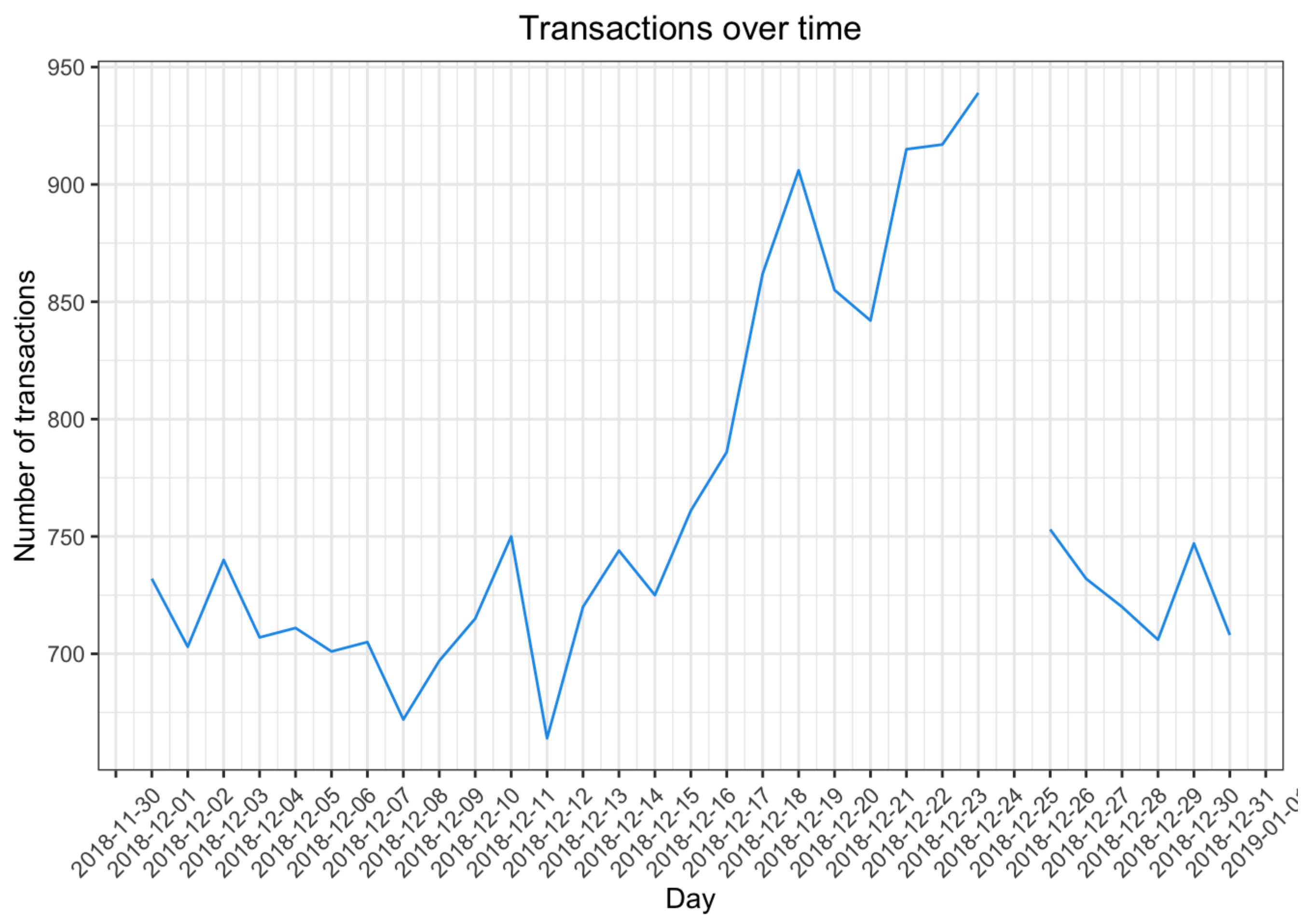
Create a plot to check the missing date. Some thing seemed to happen in December.

```
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))

transaction_by_day %>%
  ggplot(aes(x = DATE, y = N)) +
  geom_line(col = "#1e99e6") +
  labs(x = "Day", y = "Number of transactions", title = "Transactions Over Time") +
  scale_x_date(breaks = "1 month") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5))
```



To find out what happend in December, zoom in. There was not sales on christmas because stores are closed. Okay nice. The issue solved.

```
# there was not sales on christmas because stores are closed. -> issues solved
ggplot(transaction_by_day[month(DATE) == 12, ], aes(x = DATE, y = N)) +
  geom_line(col = "#1e99e6") +
  labs(x = "Day", y = "Number of transactions", title = "Transactions over time") +
  scale_x_date(breaks = "1 day") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5))
```



4. Explore pack sizes

We are going to keep exploring the transaction data. Up next is pack size.

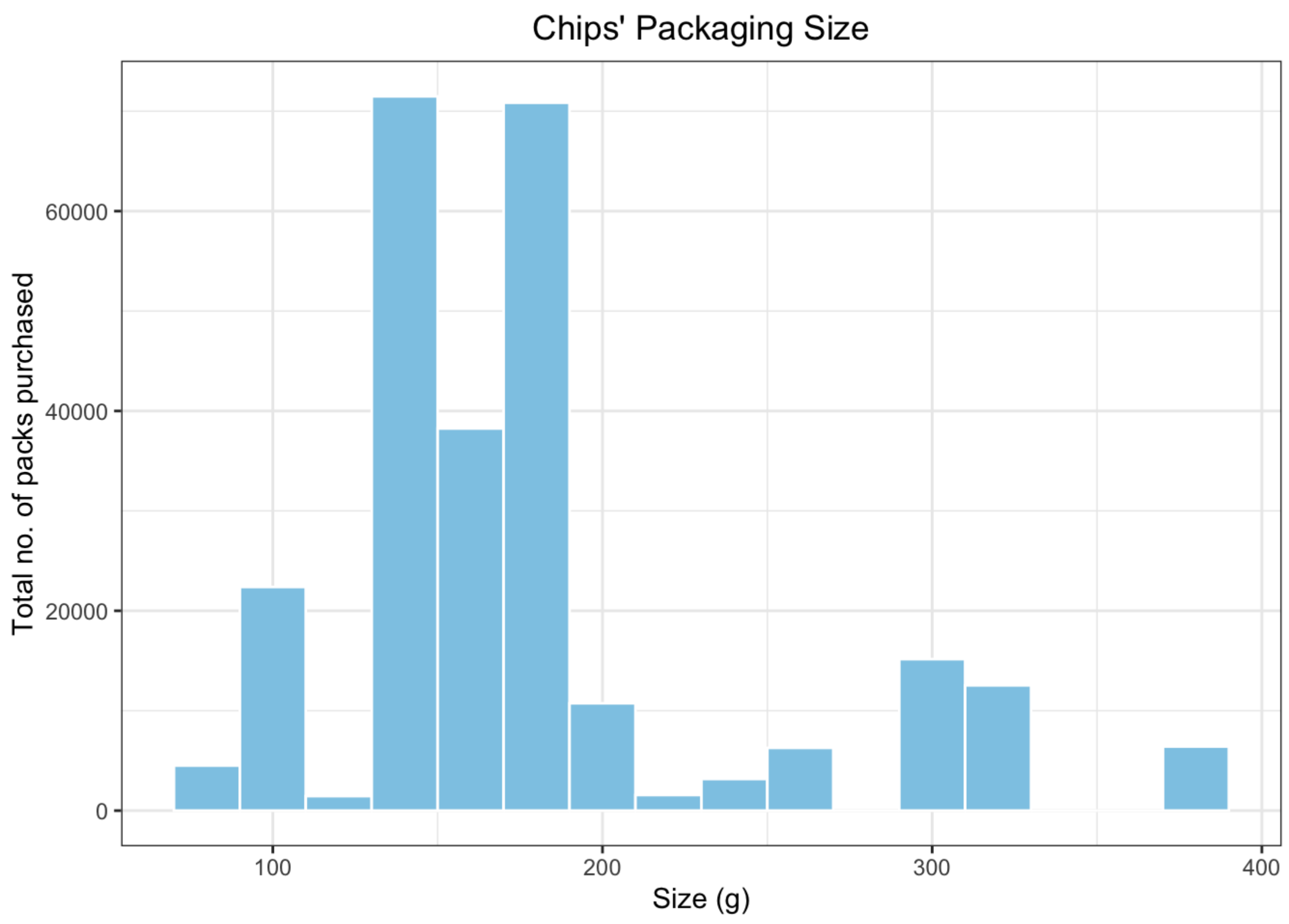
Get the number (pack size) from the product name. There are 21 pack sizes in the data. It’s min 70g, max 380g. Looks fine, not unusual sizes in there.

```
transaction_data = mutate(transaction_data, PACK_SIZE = as.numeric(str_extract(PROD_N
AME, "\\d+")))
summary(transaction_data$PACK_SIZE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    70.0   150.0   170.0   182.4   175.0   380.0
```

Create a bar chart of PACK_SIZE to look at the distribution. Looks pretty fine.

```
options(scipen=999) # turn off scientific notations like 1e+05
ggplot(transaction_data, aes(x = PACK_SIZE)) +
  geom_histogram(binwidth = 20, fill = "#8ecae6", color = "white") +
  labs(title = "Chips' Packaging Size",
       x = "Size (g)", y = "Total no. of packs purchased")
```



```
summary(transaction_data$PACK_SIZE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    70.0   150.0   170.0   182.4   175.0   380.0
```

5. Correct wrongly or differently spelled brand names

First make a column of brand names, and then arrange it in order or occurrence to see brand names and their popularity at once. There are 29 brand names in total but there might be some overlapping.

```
transaction_data = transaction_data %>%
  mutate(BRAND = toupper(substr(PROD_NAME, 1, regexpr(pattern = ' ', PROD_NAME) -
1)))

transaction_data %>%
  group_by(BRAND) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
```

```
## # A tibble: 29 × 2
##    BRAND      count
##    <chr>      <int>
##  1 KETTLE     41288
##  2 SMITHS     28860
##  3 PRINGLES   25102
##  4 DORITOS    24962
##  5 THINS      14075
##  6 RRD        11894
##  7 INFUZIONS  11057
##  8 WW         10320
##  9 COBS        9693
## 10 TOSTITOS    9471
## # ℹ 19 more rows
```

Turns out there are some issues there because of misspelling or different conventions in stores. Correct and unify them to certain names e.g., RED -> RRD, SNBTS -> SUNBITES

```
transaction_data = transaction_data %>%
  mutate(BRAND = case_when(
    BRAND == "RED" ~ "RRD",
    BRAND == "SNBTS" ~ "SUNBITES",
    BRAND == "INFZNS" ~ "INFUZIONS",
    BRAND == "WW" ~ "WOOLWORTHS",
    BRAND == "SMITH" ~ "SMITHS",
    BRAND == "NCC" ~ "NATURAL",
    BRAND == "DORITO" ~ "DORITOS",
    BRAND == "GRAIN" ~ "GRNWVES",
    TRUE ~ BRAND
  ))
```

6. Examine the number of customers and segments

Examine customer data.

This is the data that provides the description of 72637 customers who purchased the chips such as their loyalty card number, life stage and customer type i.e., the price point of products they buy and the types of products they buy.

```
str(purchase_behaviour)
```

```
## 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 F
AMILIES" "OLDER SINGLES/COUPLES" ...
##  $ PREMIUM_CUSTOMER: chr  "Premium" "Mainstream" "Budget" "Mainstream" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
summary(purchase_behaviour)
```

```
##  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                                        
```

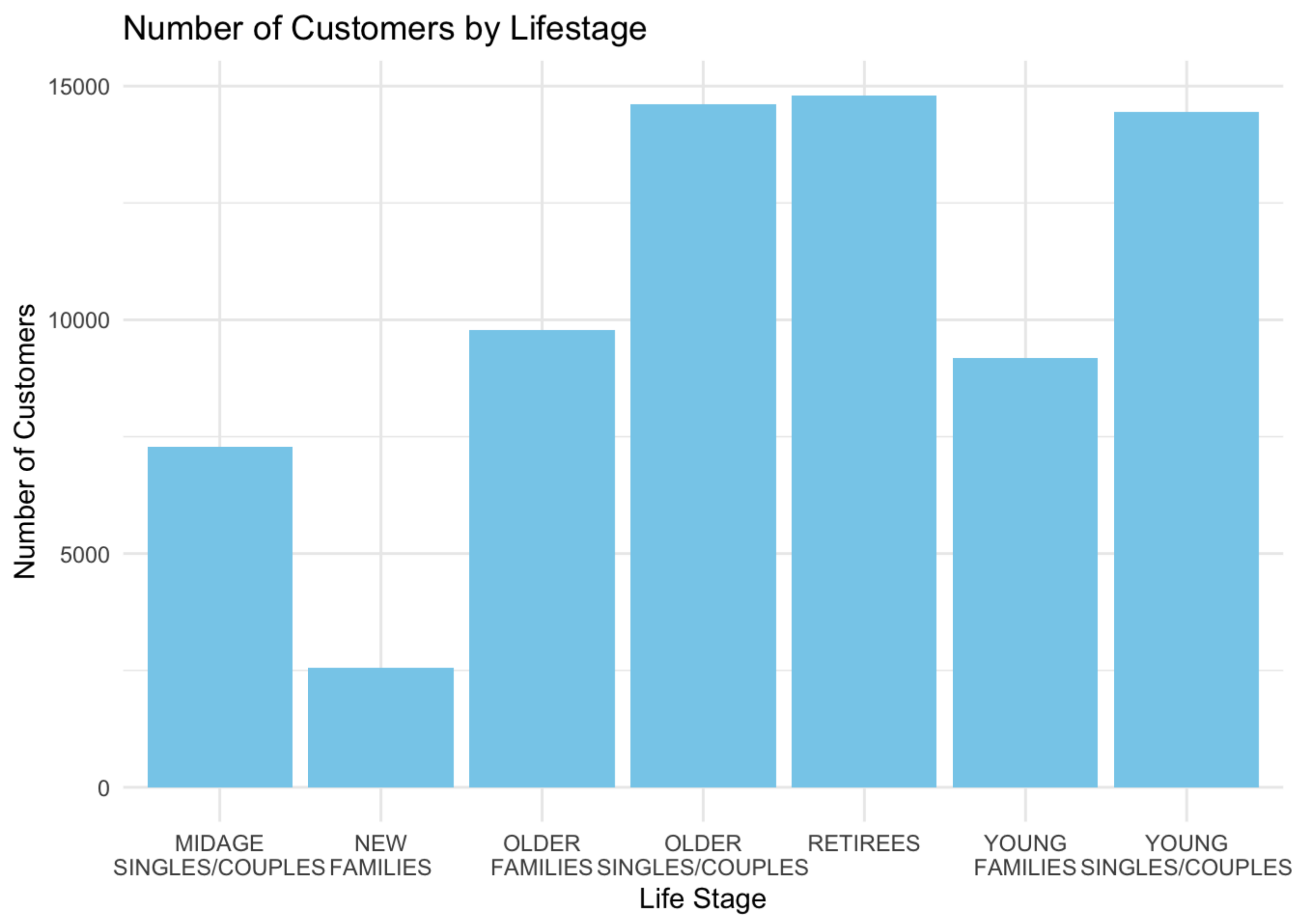
Let’s examine the value of LIFSTAGE and PREMIUM_CUSTOMER

LIFESTAGE: top 3 life stage categories are RETIREES, OLDER SINGLES/COUPLES, and YOUNG SINGLES/COUPLES

```
purchase_behaviour %>%
  group_by(LIFESTAGE) %>%
  summarise(N = n()) %>%
  arrange(desc(N))
```

```
## # A tibble: 7 × 2
##   LIFESTAGE                  N
##   <chr>                  <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
```

```
purchase_behaviour %>%
  group_by(LIFESTAGE) %>%
  summarise(CustomerCount = n_distinct(LYLTY_CARD_NBR)) %>%
  ggplot(aes(x = LIFESTAGE, y = CustomerCount)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "Life Stage", y = "Number of Customers") +
  ggtitle("Number of Customers by Lifestage") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5))+
  theme_minimal()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```

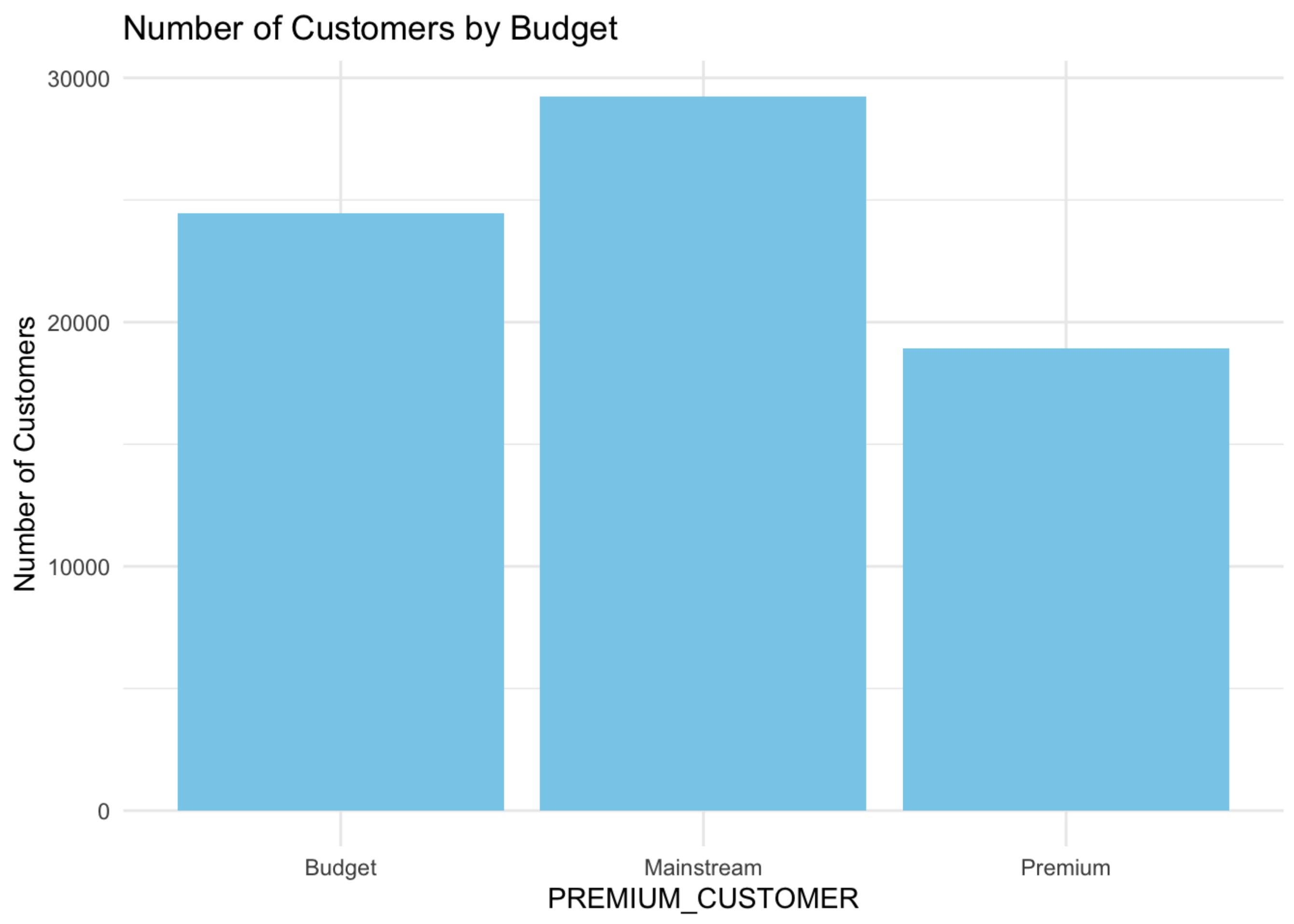


PREMIUM_CUSTOMER: The biggest category is Mainstream, the smallest is

```
purchase_behaviour %>%
  group_by(PREMIUM_CUSTOMER) %>%
  summarise(N = n()) %>%
  arrange(desc(N))
```

```
## # A tibble: 3 × 2
##   PREMIUM_CUSTOMER     N
##   <chr>            <int>
## 1 Mainstream       29245
## 2 Budget           24470
## 3 Premium          18922
```

```
purchase_behaviour %>%
  group_by(PREMIUM_CUSTOMER) %>%
  summarise(CustomerCount = n_distinct(LYLTY_CARD_NBR)) %>%
  ggplot(aes(x = PREMIUM_CUSTOMER, y = CustomerCount)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(x = "PREMIUM_CUSTOMER", y = "Number of Customers") +
  ggtitle("Number of Customers by Budget") +
  theme(axis.text.x = element_text(angle = 45, vjust = 0.5))+
  theme_minimal()+
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```



That’s all for exploratory data analysis. But before moving to customer analysis, I’m merging the transaction and customer data that I cleaned and examined so far.

```
data = merge(transaction_data, purchase_behaviour, all.x = TRUE)
```

Check if some customers are not matched on by checking nulls. All good.

```
colSums(is.na(data))
```

```
##    LYLTY_CARD_NBR             DATE        STORE_NBR           TXN_ID
##                 0                0                0                0
##          PROD_NBR        PROD_NAME         PROD_QTY        TOT_SALES
##                 0                0                0                0
##         PACK_SIZE            BRAND        LIFESTAGE PREMIUM_CUSTOMER
##                 0                0                0                0
```

Save the cleaned data for further analysis

```
summary(data)
```

```
##  LYLTY_CARD_NBR        DATE               STORE_NBR        TXN_ID
##  Min.   :   1000   Min.   :2018-07-01   Min.   :  1.0   Min.   :      1
##  1st Qu.:  70021   1st Qu.:2018-09-30   1st Qu.: 70.0   1st Qu.:  67600
##  Median : 130357   Median :2018-12-30   Median :130.0   Median : 135136
##  Mean   : 135549   Mean   :2018-12-30   Mean   :135.1   Mean   : 135158
##  3rd Qu.: 203094   3rd Qu.:2019-03-31   3rd Qu.:203.0   3rd Qu.: 202700
##  Max.   :2373711   Max.   :2019-06-30   Max.   :272.0   Max.   :2415841
##     PROD_NBR       PROD_NAME           PROD_QTY       TOT_SALES
##  Min.   :  1.00   Length:264834      Min.   :1.000   Min.   : 1.500
##  1st Qu.: 28.00   Class :character   1st Qu.:2.000   1st Qu.: 5.400
##  Median : 56.00   Mode  :character   Median :2.000   Median : 7.400
##  Mean   : 56.58                      Mean   :1.906   Mean   : 7.299
##  3rd Qu.: 85.00                      3rd Qu.:2.000   3rd Qu.: 9.200
##  Max.   :114.00                      Max.   :5.000   Max.   :29.500
##    PACK_SIZE        BRAND             LIFESTAGE          PREMIUM_CUSTOMER
##  Min.   : 70.0   Length:264834      Length:264834      Length:264834
##  1st Qu.:150.0   Class :character   Class :character   Class :character
##  Median :170.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   :182.4
##  3rd Qu.:175.0
##  Max.   :380.0
```

```
write.csv(data, "cleaned_data.csv")
```

Customer analysis

Finally the data set is ready to be analyzed following the various questions according to our interests:

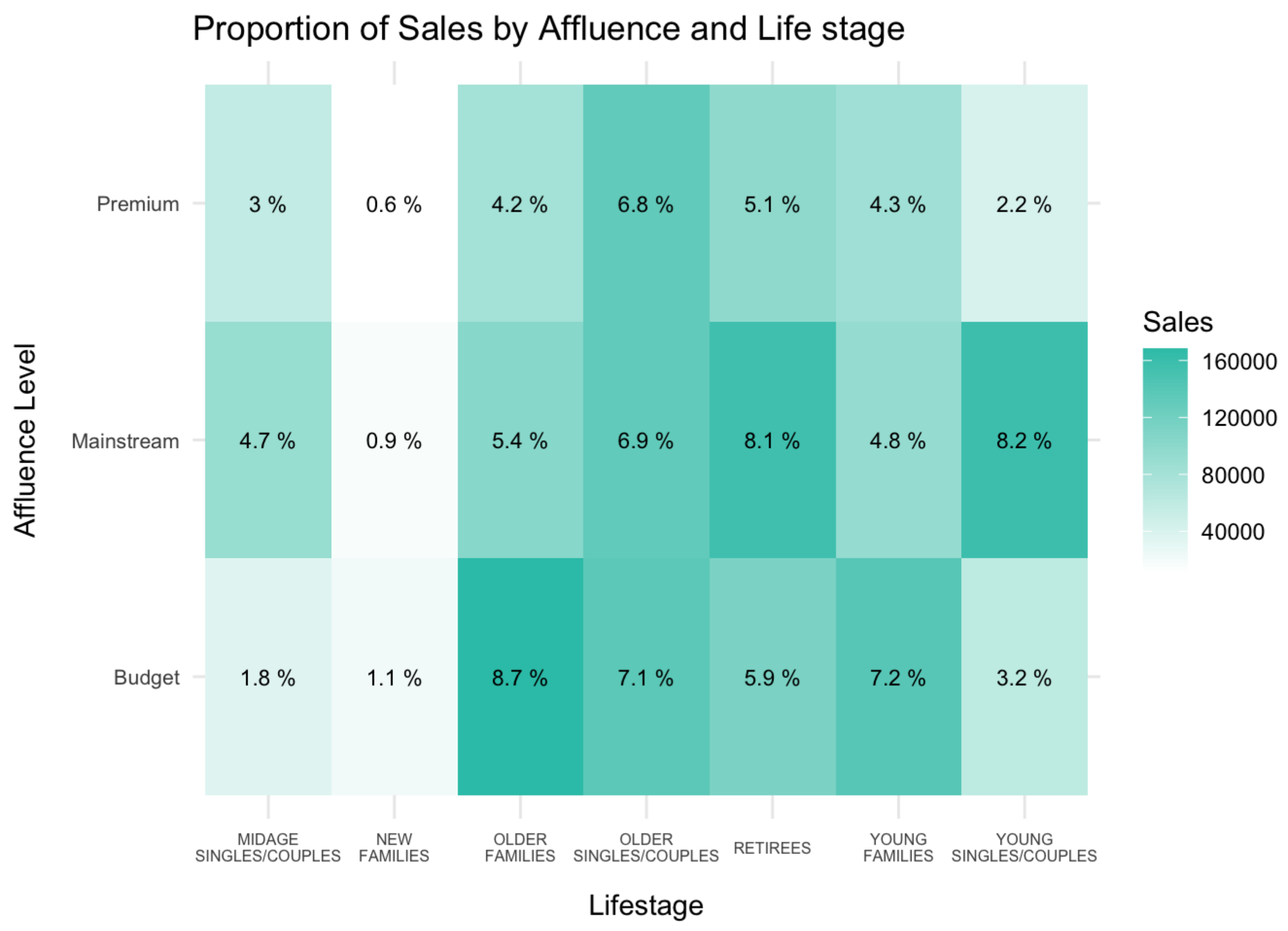
- What is the cause of higher sales by segments?
- The number of customers, or the average spending or unit per customer?
    - What are the customer segments that contribute to the sales the most?
    - How many customers are in each segment?
    - How many chips are bought per customer by segments?
- Who are the target customers who have the greatest impact on sales?
- What brands do the target customers like to buy?

1. The proportion of sales by 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.

We can see from the plot that the sales are mostly due to the budget- older families, mainstream young single/couples, and mainstream - retirees.

```
sales = data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(SALES = sum(TOT_SALES), .groups = "keep")

sales$PERCENT = (sales$SALES /sum(data$TOT_SALES)) * 100

heatmap_sales = ggplot(data = sales) +
  geom_tile(aes(x = LIFESTAGE, y = PREMIUM_CUSTOMER, fill = SALES)) +
  labs(fill = "Sales") +
  labs(x = NULL, y = NULL) +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 0, vjust = 0.5, size = 6),
    axis.text.y = element_text(vjust = 0.5, size = 8),
    axis.title.x = element_text(margin = margin(t = 12)),  # Adjust the top margin fo
r x-axis label
    axis.title.y = element_text(margin = margin(r = 12))
    ) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10)) +
  scale_fill_gradient(low = "white", high = "#2ec4b6")+
  labs(
    x = "Lifestage",
    y = "Affluence Level",
    fill = "Sales",
    title = "Proportion of Sales by Affluence and Life stage")



heatmap_sales +
  geom_text(aes(x = LIFESTAGE, y = PREMIUM_CUSTOMER, label = paste(round(.data[["PERC
ENT"]], 1), "%")), size = 3, color = "black")
```

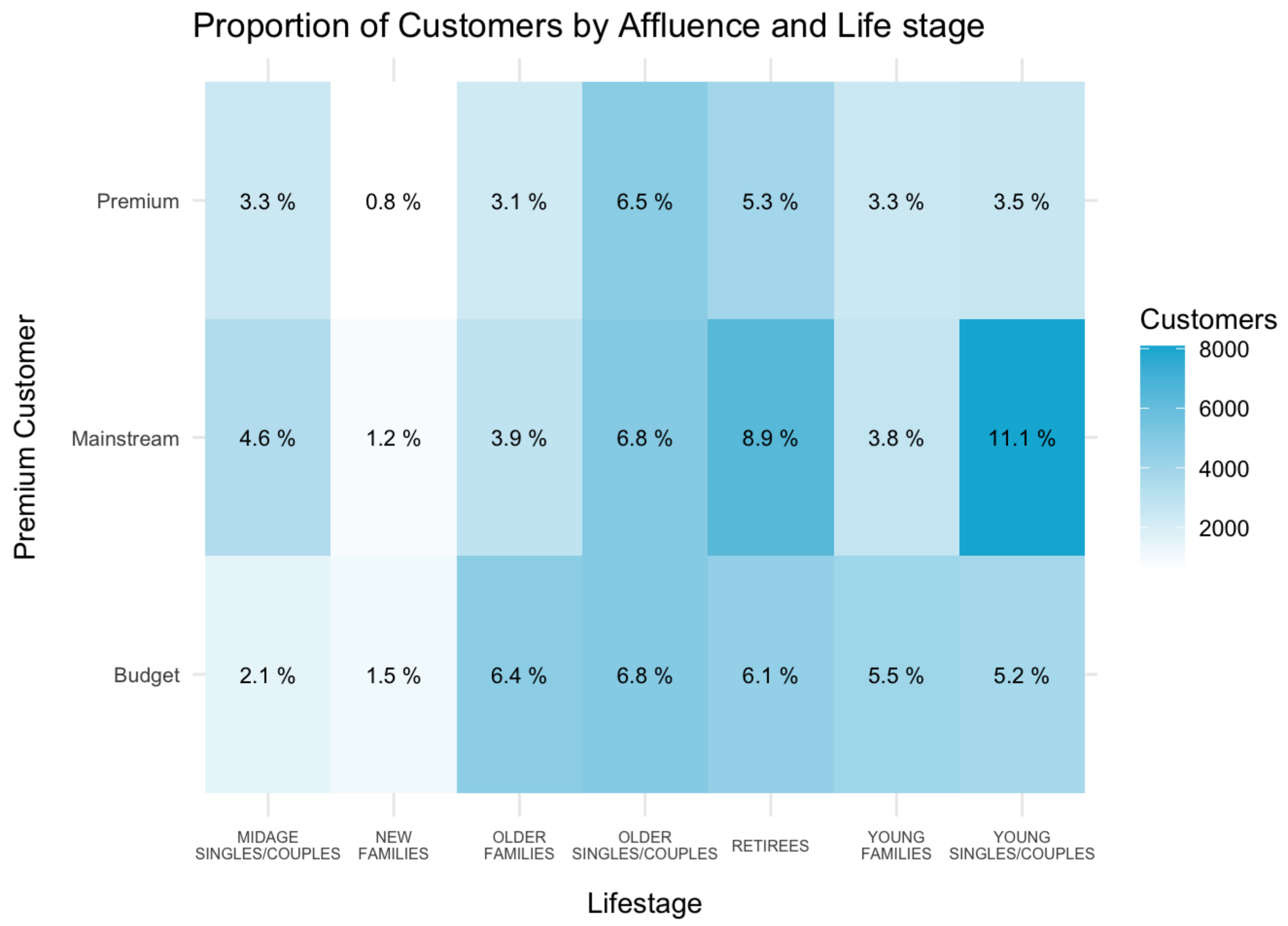


2. The proportion of customers by segments

We will examine the number of customers to see if the highers sales are due to there being more customers who buy chips.

As a result, from here we can see that mainstream- young/single couples and mainstream retirees contribute most to the sales of chips. There might be other factors for the higher sales. We will examine

```
customer_count = data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(CUSTOMER_COUNT = uniqueN(LYLTY_CARD_NBR), .groups = "keep")
sales = merge(sales, customer_count, x.all = TRUE)

sum_customer = sum(sales$CUSTOMER_COUNT)
sales$CUSTOMER_PERCENT = (sales$CUSTOMER_COUNT / sum_customer ) * 100

heatmap_customer = ggplot(data = sales) +
  geom_tile(aes(x = LIFESTAGE, y = PREMIUM_CUSTOMER, fill = CUSTOMER_COUNT)) +
  labs(x = "Lifestage", y = "Premium Customer", fill = "Customers", title = "Proporti
on of Customers by Affluence and Life stage") +
  theme_minimal() +
  theme(
    axis.text.x = element_text(angle = 0, vjust = 0.5, size = 6),
    axis.text.y = element_text(vjust = 0.5, size = 8),
    axis.title.x = element_text(margin = margin(t = 12)),  # Adjust the top margin fo
r x-axis label
    axis.title.y = element_text(margin = margin(r = 12))
    ) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10)) +
  scale_fill_gradient(low = "white", high = "#00b4d8")

heatmap_customer +  geom_text(aes(x = LIFESTAGE, y = PREMIUM_CUSTOMER, label = paste
(round(.data[["CUSTOMER_PERCENT"]], 1), "%")), size = 3, color = "black")
```

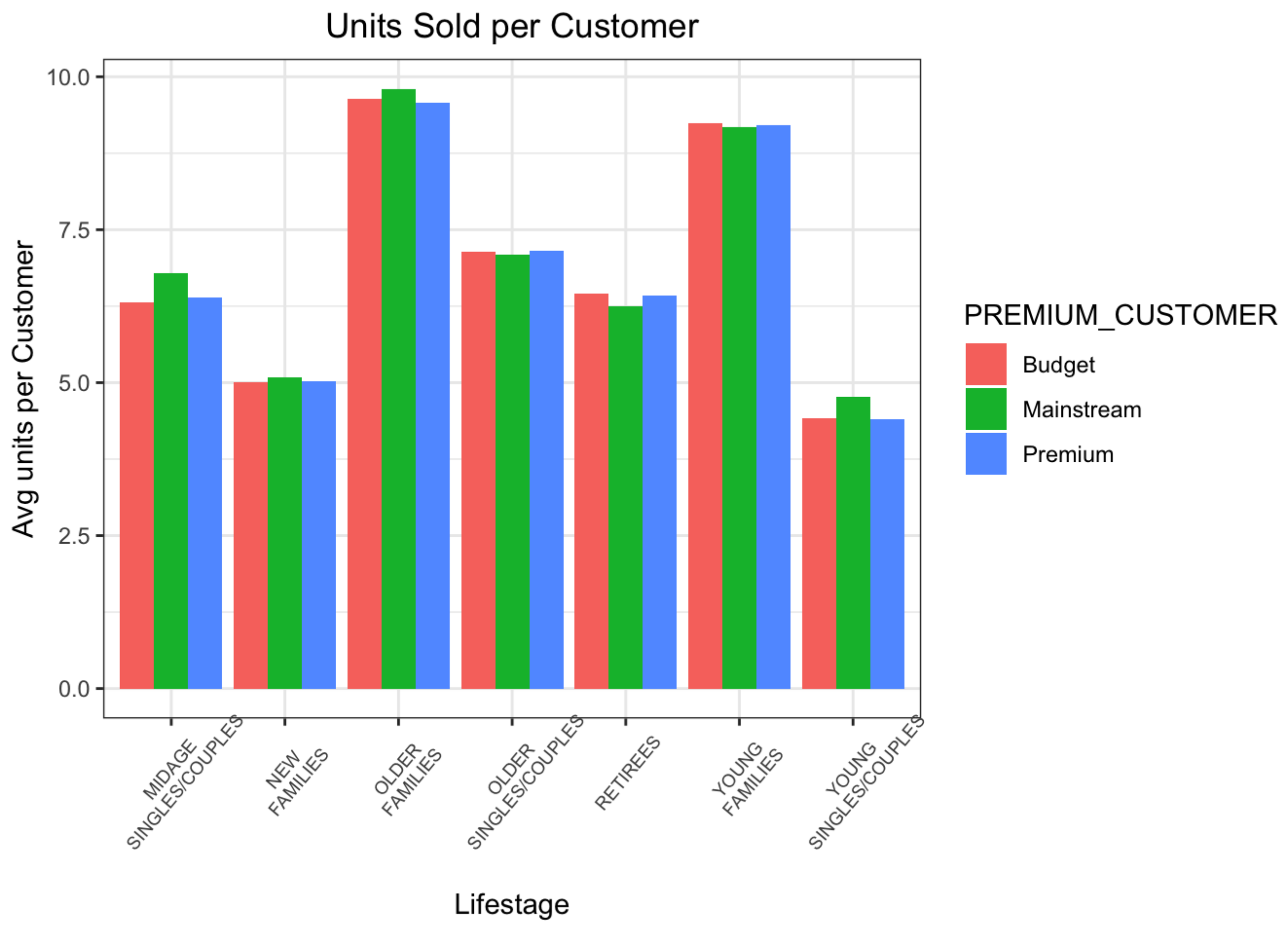


3. The number of units sold per customer

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.

```
avg_unit = data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(AVG_UNIT = sum(PROD_QTY)/uniqueN(LYLTY_CARD_NBR), .groups = "keep")

ggplot(data = avg_unit, aes(weight = AVG_UNIT, x = LIFESTAGE, fill = PREMIUM_CUSTOME
R)) + geom_bar(position = position_dodge()) +
  labs(x = "Lifestage", y = "Avg units per Customer", title = "Units Sold per Custome
r") +
  theme(axis.text.x = element_text(angle = 50, vjust = 0.75, size = 7)) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```

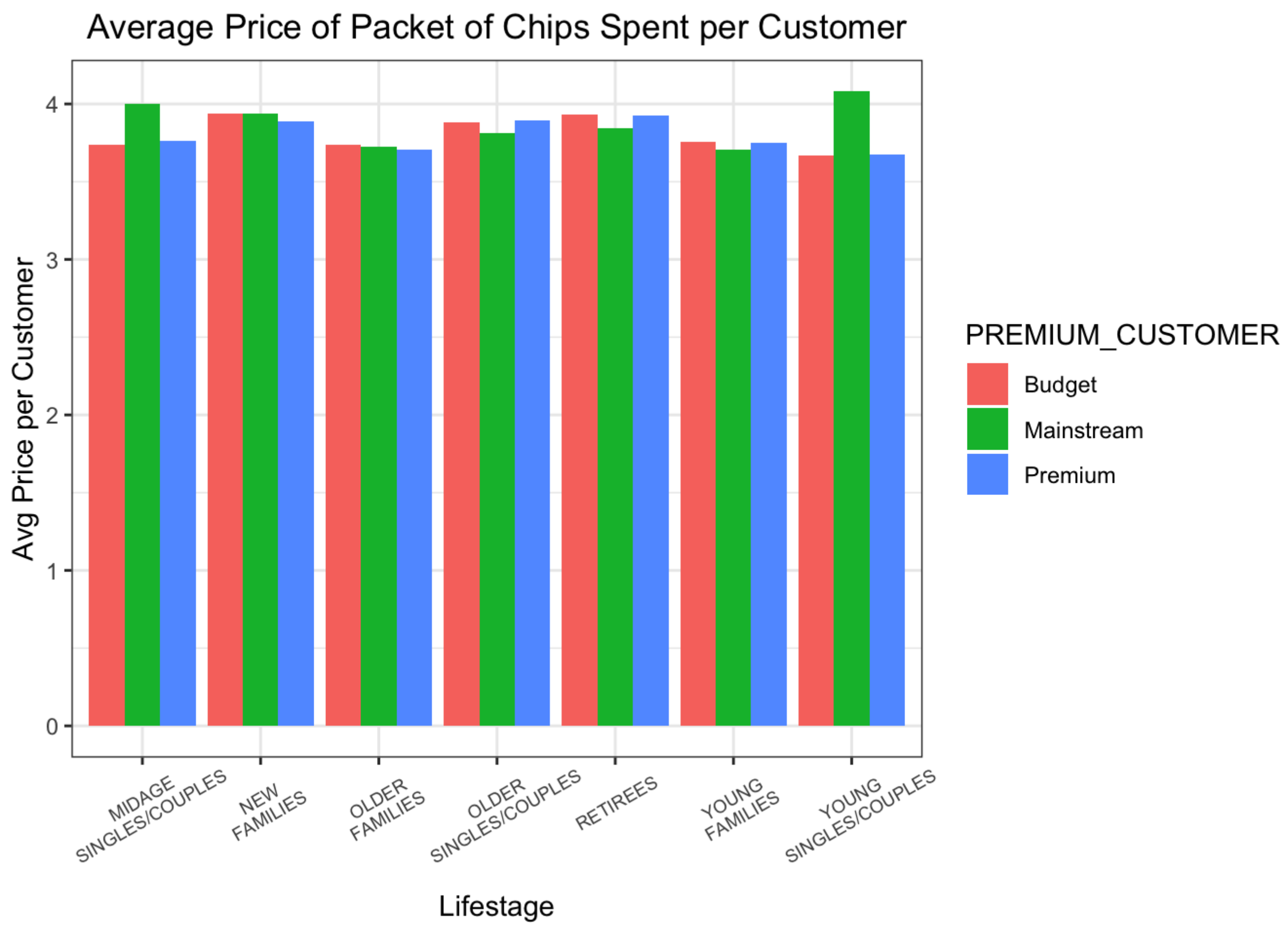


Young families and old families have generally bought more chips in comparison with the midage and retirees

4. The average price per unit spent by each customer during a transaction.

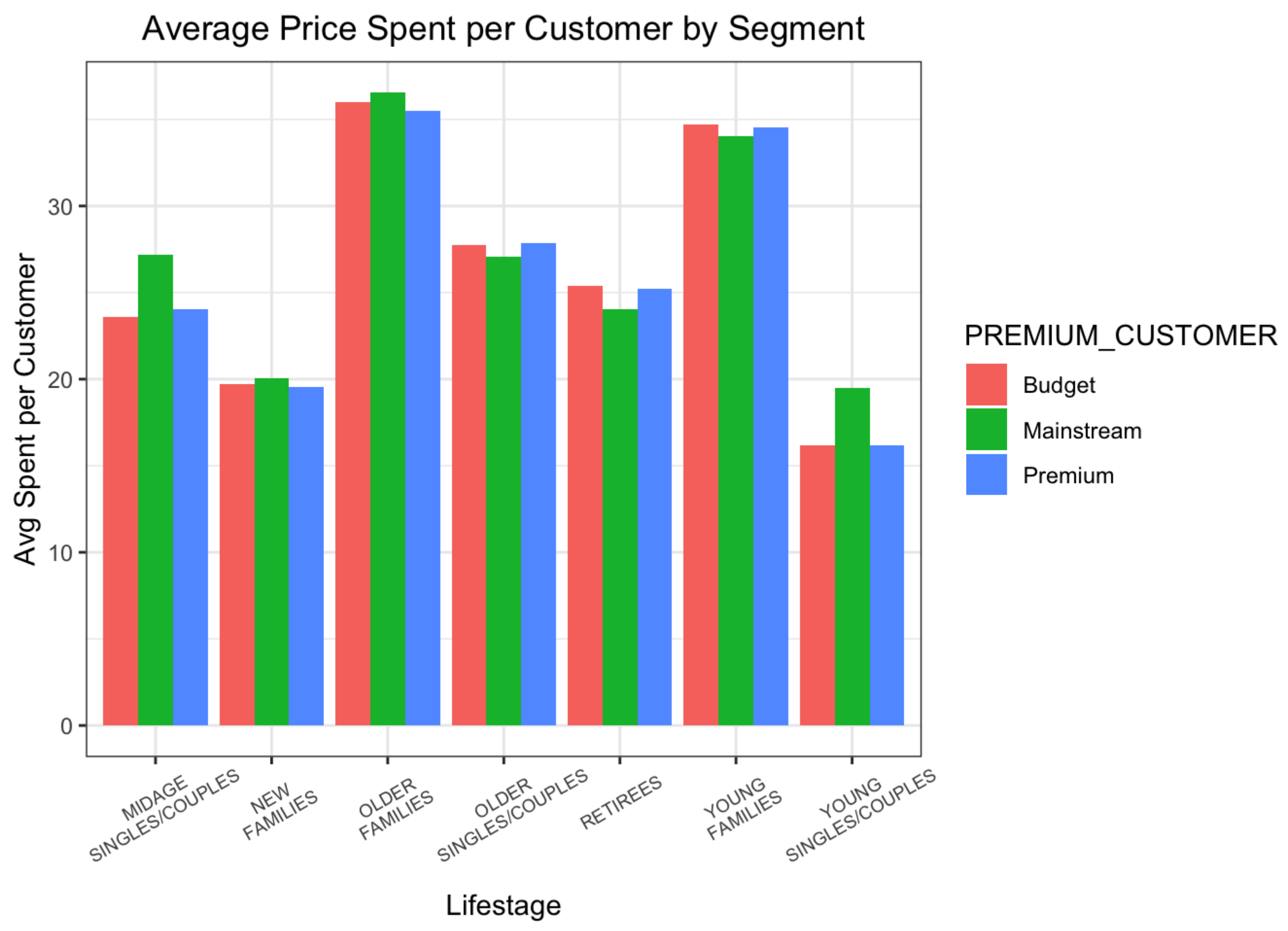
Let's also investigate the average price per unit chips bought for each customer segment as this is also a driver of total sales.

```
avg_price = data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(AVG_PRICE = sum(TOT_SALES)/sum(PROD_QTY), .groups = "keep")

ggplot(data = avg_price, aes(weight = AVG_PRICE, x = LIFESTAGE, fill = PREMIUM_CUSTOM
ER)) + geom_bar(position = position_dodge()) +
  labs(x = "Lifestage", y = "Avg Price per Customer", title = "Average Price of Packe
t of Chips Spent per Customer") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.75, size = 7)) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```



Mainstream midage 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 midage and young singles and couples buying chips compared to their mainstream counterparts.

```
avg_per_cust = data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(AVG_PER_CUST = sum(TOT_SALES)/uniqueN(LYLTY_CARD_NBR), .groups = "keep")

ggplot(data = avg_per_cust, aes(weight = AVG_PER_CUST, x = LIFESTAGE, fill = PREMIUM_
CUSTOMER)) + geom_bar(position = position_dodge()) +
  labs(x = "Lifestage", y = "Avg Spent per Customer", title = "Average Price Spent pe
r Customer by Segment") +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.75, size = 7)) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```



5. Conduct independent T-test

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

```
# get the average price per packet in each transaction (e.g., 5 euros for two packs o
f chips at one purchase-> price is 2.5),
data = data %>%
  mutate(price = TOT_SALES / PROD_QTY)

# 1. get the price from young singles/couples and midage singles/couples with the mai
nstream classification
group1 = data %>%
  filter(LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") & PREMIU
M_CUSTOMER == "Mainstream") %>%
  pull(price)

# 2. get the price form young singles/couples and midage singles/couples other than t
he mainstream classification
group2 = data %>%
  filter(LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") & PREMIU
M_CUSTOMER != "Mainstream") %>%
  pull(price)

#3.
t_test_result = t.test(group1, group2, alternative = "greater")
print(t_test_result)
```

```
##
##  Welch Two Sample t-test
##
## data:  group1 and group2
## t = 40.61, df = 58792, p-value < 0.00000000000000022
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.3429435       Inf
## sample estimates:
## mean of x mean of y
##  4.045586  3.688165
```

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 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.

6. Identify the target customers that have the greatest impact on sales.

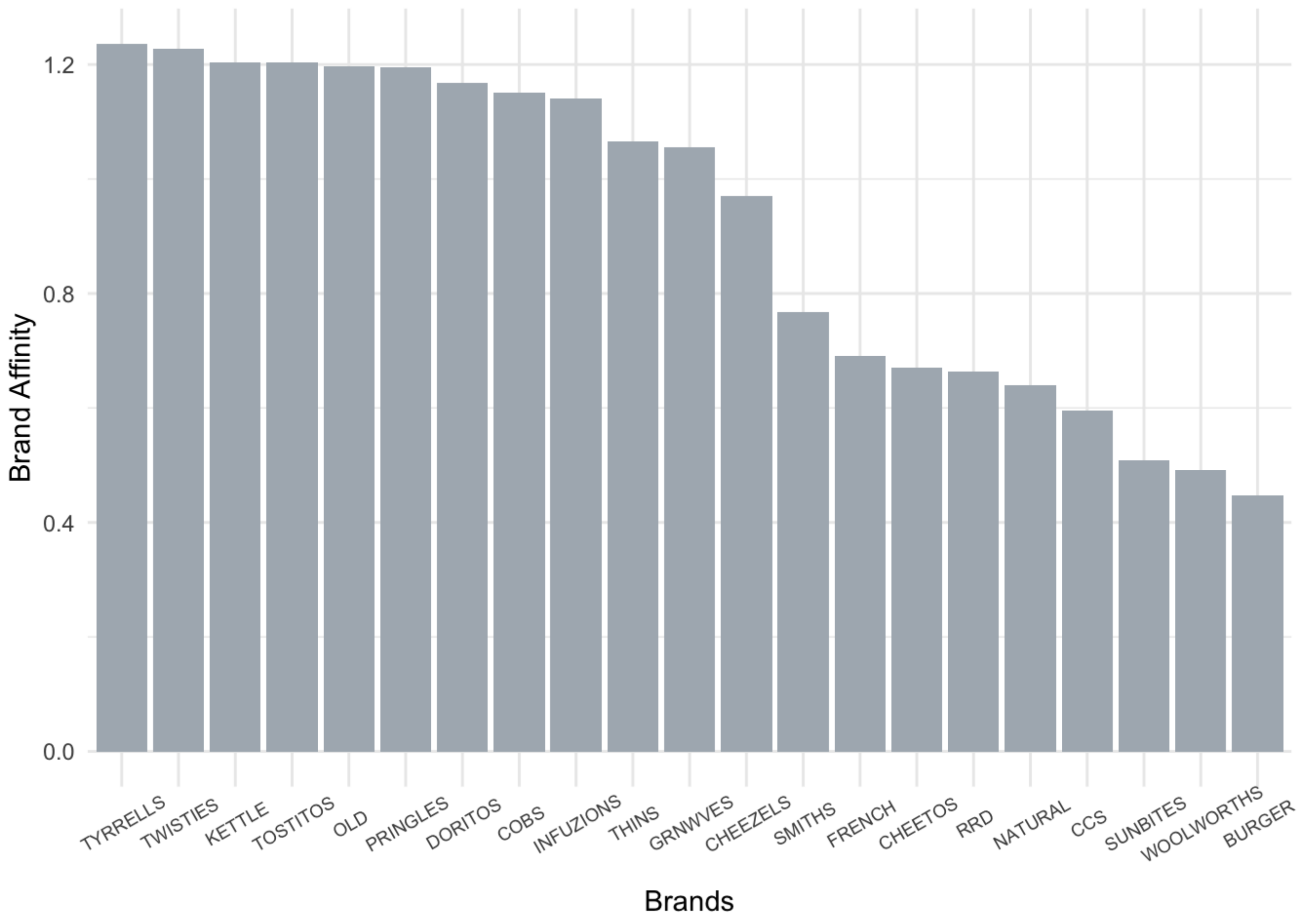
```
# Filter segment1 and other
segment1 <- data %>%
  filter(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream")

other <- data %>%
  filter(!(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream"))

# Calculate quantities (total quantities in segment1 and others)
quantity_segment1 <- sum(segment1$PROD_QTY)
quantity_other <- sum(other$PROD_QTY)

sum(segment1$PROD_QTY)
```

```
## [1] 38632
```

```r
# Calculate quantities by brand (quantity grouped by brands / total quantity of segme
nt1)
# -> spot popular brands among the target segment compared to other

quantity_segment1_by_brand = segment1 %>%
  group_by(BRAND) %>%
  summarise(targetSegment = sum(PROD_QTY)/quantity_segment1)
# Calculate quantities by brand (quantity grouped by brands / total quantity of othe
r)
quantity_other_by_brand = other %>%
  group_by(BRAND) %>%
  summarise(other = sum(PROD_QTY)/quantity_other)

# Merge data frames
brand_proportions = merge(quantity_segment1_by_brand, quantity_other_by_brand)

# Calculate affinityToBrand () (how bigger the proportion of the target segment by br
ands than other)
brand_proportions = brand_proportions %>%
  mutate(affinityToBrand = targetSegment/other)

# Order by affinityToBrand
brand_proportions = arrange(brand_proportions, desc(affinityToBrand))
print(brand_proportions)
```

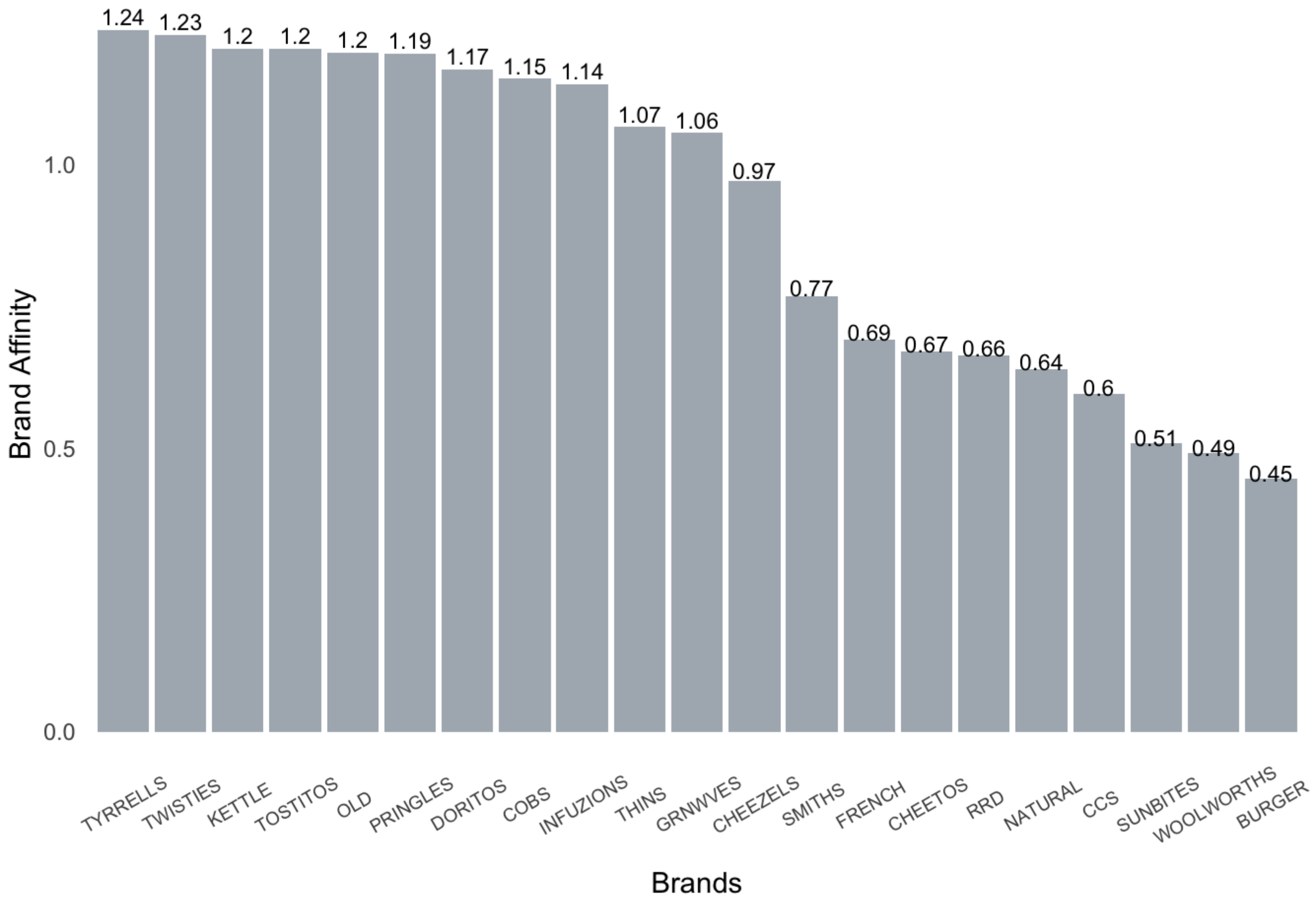
```
##         BRAND targetSegment       other affinityToBrand
## 1    TYRRELLS   0.029586871 0.023933043       1.2362352
## 2    TWISTIES   0.043306068 0.035282734       1.2274011
## 3      KETTLE   0.185649203 0.154216335       1.2038232
## 4    TOSTITOS   0.042581280 0.035377136       1.2036384
## 5         OLD   0.041597639 0.034752796       1.1969581
## 6    PRINGLES   0.111979706 0.093743295       1.1945356
## 7     DORITOS   0.122877407 0.105277499       1.1671764
## 8        COBS   0.041856492 0.036374793       1.1507005
## 9   INFUZIONS   0.060649203 0.053156887       1.1409472
## 10      THINS   0.056611100 0.053083941       1.0664449
## 11    GRNWVES   0.030674053 0.029052204       1.0558253
## 12   CHEEZELS   0.016851315 0.017369961       0.9701412
## 13     SMITHS   0.093419963 0.121714168       0.7675356
## 14     FRENCH   0.003701595 0.005363748       0.6901134
## 15    CHEETOS   0.007532615 0.011240270       0.6701454
## 16        RRD   0.045376890 0.068426405       0.6631488
## 17    NATURAL   0.018378546 0.028741107       0.6394516
## 18        CCS   0.010483537 0.017601675       0.5955988
## 19   SUNBITES   0.005953614 0.011718716       0.5080431
## 20 WOOLWORTHS   0.028189066 0.057428576       0.4908543
## 21     BURGER   0.002743839 0.006144710       0.4465369
```

```r
brand_proportions %>%
  mutate(BRAND = reorder(BRAND, -affinityToBrand)) %>%
  ggplot(aes(x = BRAND, y = affinityToBrand)) +
  geom_bar(stat = "identity", fill = "#adb5bd") +
  labs(x = "Brands", y = "Brand Affinity", title = NULL) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.75, size = 7)) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```



```r
#--------------------
brand_proportions %>%
  mutate(BRAND = reorder(BRAND, -affinityToBrand)) %>%
  ggplot(aes(x = BRAND, y = affinityToBrand)) +
  geom_bar(stat = "identity", fill = "#adb5bd") +
  geom_text(aes(label = paste0(round(affinityToBrand, 2))),
            position = position_stack(vjust = 1.02),
            box.padding = 5,  # Adjust padding
            size = 3) +  # Use ggrepel for label repelling
  labs(x = "Brands", y = "Brand Affinity", title = NULL) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 30, vjust = 0.75, size = 7),
        panel.grid.major = element_blank(),  # Remove major grid lines
        panel.grid.minor = element_blank(),  # Remove minor grid lines
        panel.border = element_blank())+   # Remove panel border
        scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```

```
## Warning in geom_text(aes(label = paste0(round(affinityToBrand, 2))), position =
## position_stack(vjust = 1.02), : Ignoring unknown parameters: `box.padding`
```



```
#------------------------
```

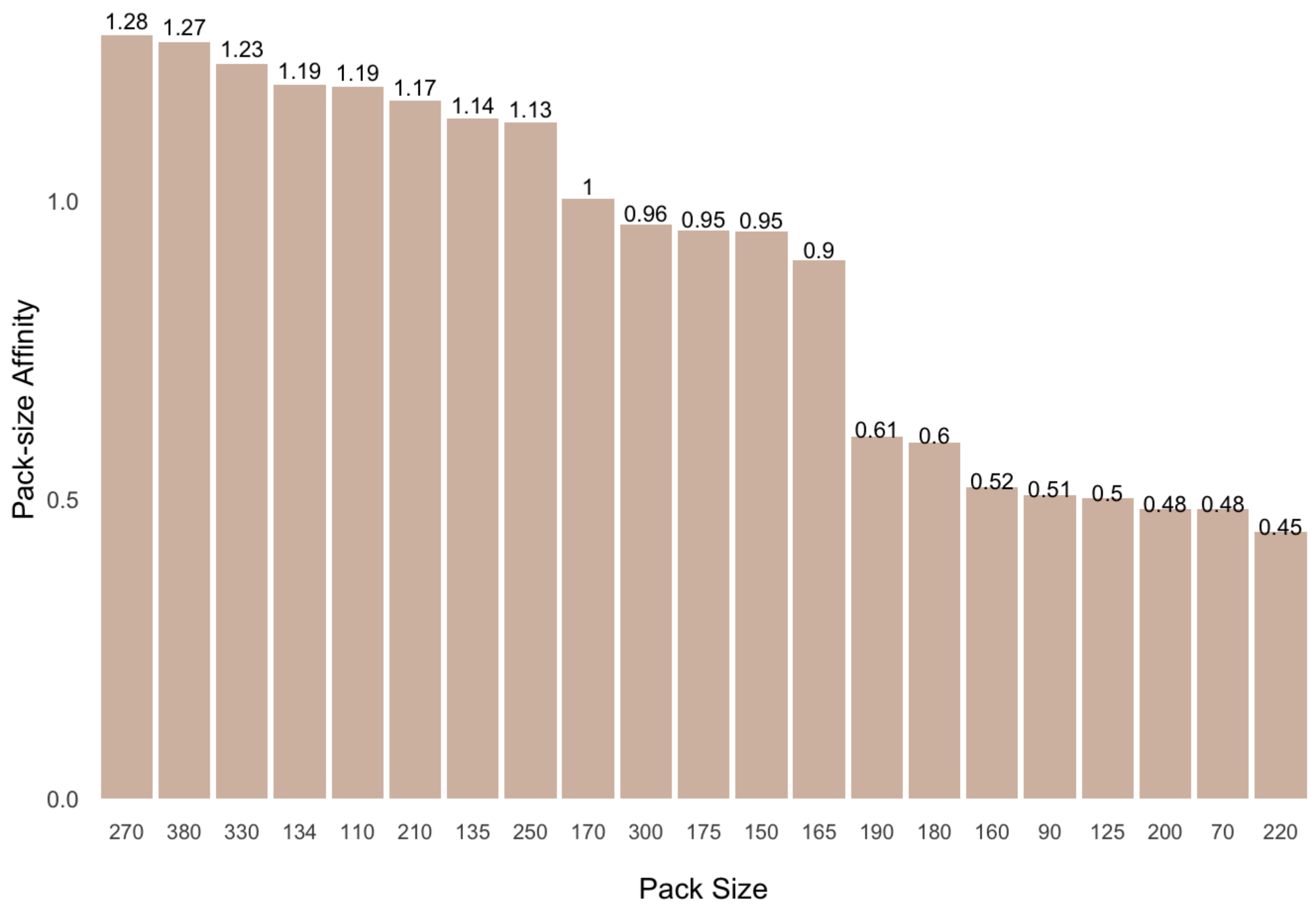
7. identify the most popular pack sizes and brands among the target customer base compared to others

#spot popular pack size among the target segment compared to other

```
quantity_segment1_by_pack <- segment1 %>%
  group_by(PACK_SIZE) %>%
  summarise(targetSegment = sum(PROD_QTY)/quantity_segment1)

quantity_other_by_pack <- other %>%
  group_by(PACK_SIZE) %>%
  summarise(other = sum(PROD_QTY)/quantity_other)

pack_proportions <- merge(quantity_segment1_by_pack, quantity_other_by_pack) %>%
  mutate(affinityToPack = targetSegment/other) %>%
  arrange(desc(affinityToPack))

print(pack_proportions)
```

```
##    PACK_SIZE targetSegment       other affinityToPack
## 1        270   0.029845724 0.023377359      1.2766936
## 2        380   0.030156347 0.023832205      1.2653612
## 3        330   0.057465314 0.046726826      1.2298142
## 4        134   0.111979706 0.093743295      1.1945356
## 5        110   0.099658314 0.083642285      1.1914824
## 6        210   0.027308967 0.023400959      1.1670020
## 7        135   0.013848623 0.012179999      1.1369971
## 8        250   0.013460344 0.011905375      1.1306107
## 9        170   0.075740319 0.075440042      1.0039803
## 10       300   0.054954442 0.057263373      0.9596787
## 11       175   0.239102299 0.251516868      0.9506412
## 12       150   0.155130462 0.163446272      0.9491221
## 13       165   0.052184717 0.058003570      0.8996811
## 14       190   0.007014910 0.011589987      0.6052561
## 15       180   0.003365086 0.005651245      0.5954592
## 16       160   0.006005384 0.011525622      0.5210464
## 17        90   0.005953614 0.011718716      0.5080431
## 18       125   0.002821495 0.005623353      0.5017460
## 19       200   0.008412715 0.017378543      0.4840863
## 20        70   0.002847380 0.005889395      0.4834759
## 21       220   0.002743839 0.006144710      0.4465369
```

```
pack_proportions %>%
  mutate(PACK_SIZE = reorder(PACK_SIZE, -affinityToPack)) %>%
  ggplot(aes(x = PACK_SIZE, y = affinityToPack)) +
  geom_bar(stat = "identity", fill = "#d5bdaf") +
  geom_text(aes(label = paste0(round(affinityToPack, 2))),
            position = position_stack(vjust = 1.02),
            size = 3) +  # Use ggrepel for label repelling
  labs(x = "Pack Size", y = "Pack-size Affinity", title = NULL) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0, vjust = 7, size = 8),
        panel.grid.major = element_blank(),  # Remove major grid lines
        panel.grid.minor = element_blank(),  # Remove minor grid lines
        panel.border = element_blank())+   # Remove panel border
        scale_x_discrete(labels = function(x) str_wrap(x, width = 10))
```



```
#list of items that have pack-size 270 380 330 134 110

data %>%
  group_by(BRAND) %>%
  filter(PACK_SIZE %in% c(270, 380, 330, 134, 110)) %>%
  arrange(PROD_NAME, BRAND)
```

```
## # A tibble: 72,730 × 13
## # Groups:   BRAND [7]
##    LYLTY_CARD_NBR DATE       STORE_NBR TXN_ID PROD_NBR PROD_NAME        PROD_QTY
##             <int> <date>         <dbl>  <dbl>    <dbl> <chr>               <dbl>
##  1           1057 2018-07-06         1     64       23 Cheezels Cheese…        1
##  2           1150 2019-02-23         1    175       23 Cheezels Cheese…        1
##  3           1191 2019-06-17         1    226       23 Cheezels Cheese…        1
##  4           1197 2019-05-12         1    231       23 Cheezels Cheese…        1
##  5           1202 2019-02-11         1    236       23 Cheezels Cheese…        1
##  6           1203 2018-08-24         1    237       23 Cheezels Cheese…        2
##  7           1206 2019-01-26         1    241       23 Cheezels Cheese…        1
##  8           1419 2018-11-07         1    491       23 Cheezels Cheese…        1
##  9           1475 2019-03-11         1    557       23 Cheezels Cheese…        1
## 10           2030 2019-04-02         2    623       23 Cheezels Cheese…        1
## # ℹ 72,720 more rows
## # ℹ 6 more variables: TOT_SALES <dbl>, PACK_SIZE <dbl>, BRAND <chr>,
## #   LIFESTAGE <chr>, PREMIUM_CUSTOMER <chr>, price <dbl>
```