# Quantium Virtual Internship Retail Strategy and Analytics Task 1

Maulana Akbar Dwijaya

28-09-2021

## Task 1

Below is my report for the analysis of Julia's data.

### Data Checks

#### Loading Libraries

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.0.5
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
library(ggmosaic)
```

```
## Warning: package 'ggmosaic' was built under R version 4.0.5
```

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.0.5
```

```
filePath <- ""
transactionData <- data.table(read_excel("QVI_transaction_data.xlsx", sheet = "in"))
customerData <- data.table(fread(paste0(filePath,"QVI_purchase_behaviour.csv")))
str(transactionData)
```

#### Ensuring Data is in Correct Format

Check to see data is in right format

```r
#### Examine transaction data
str(transactionData)
# We can see that all columns are in reasonable formats for analysis except for DATE. From online resea
```

We can see date is stored as an integer. Let's cast the date column from integer to Date

```r
# Map date column to R date object
transactionData$DATE <- as.Date(transactionData$DATE , origin = "1899-12-30")

# Verify transformed date column
str(transactionData)
```

```
## Classes 'data.table' and 'data.frame':   264836 obs. of  8 variables:
##  $ DATE          : Date, format: "2018-10-17" "2019-05-14" ...
##  $ 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 C
##  $ 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>
```

**Determining Non Chip Rows** Determine which transaction are non chip transactions

```r
transactionData[, .N, PROD_NAME]
```

```
##                                     PROD_NAME    N
##   1:   Natural Chip        Compny SeaSalt175g 1468
##   2:                 CCs Nacho Cheese    175g 1498
##   3:   Smiths Crinkle Cut  Chips Chicken 170g 1484
##   4:   Smiths Chip Thinly  S/Cream&Onion 175g 1473
##   5: Kettle Tortilla ChpsHny&Jlpno Chili 150g 3296
##  ---
## 110:    Red Rock Deli Chikn&Garlic Aioli 150g 1434
## 111:      RRD SR Slow Rst     Pork Belly 150g 1526
## 112:                 RRD Pc Sea Salt     165g 1431
## 113:       Smith Crinkle Cut   Bolognese 150g 1451
## 114:                 Doritos Salsa Mild  300g 1472
```

These transactions definitely contain chip products. However, to be sure they are all chips, we can map this column to a set of unique products, split them up into component words and then sort by frequency.

```r
# Get list of unique words in PROD_NAME col to subsequently analyse if chips or not
productWords <- data.table(unlist(strsplit(unique(transactionData$PROD_NAME), " ")))
setnames(productWords, 'words')

# Remove any entries not containing strictly alphabetical chars
productWords <- productWords[!grepl('[^[:alpha:]]', productWords$words )]
print(productWords)
```

```
##          words
##   1:   Natural
##   2:      Chip
##   3:
##   4:
##   5:
##  ---
## 667: Bolognese
## 668:   Doritos
## 669:     Salsa
## 670:      Mild
## 671:
```

```
# Sort by words frequency
head(sort(table(productWords$words), decreasing = T), 30)
```

```
##
##                Chips    Smiths   Crinkle       Cut    Kettle    Cheese      Salt
##       234         21        16        14        14        13        12        12
##  Original       Chip   Doritos     Salsa      Corn  Pringles       RRD   Chicken
##        10          9         9         9         8         8         8         7
##        WW        Sea      Sour    Chilli    Crisps    Thinly     Thins   Vinegar
##         7          6         6         5         5         5         5         5
##     Cream       Deli Infuzions   Natural       Red      Rock
##         4          4         4         4         4         4
```

**Removing Non-Chip Entries** Let's remove all salsa transactions

```
# Remove rows pertaining to salsa
transactionData[, SALSA := grepl("salsa", tolower(transactionData$PROD_NAME))]
transactionData <- transactionData[SALSA == FALSE, ][, SALSA := NULL]
```

```
# Summary reports nulls
summary(transactionData)
```

**Checking for Nulls and Outliers**

```
##       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
```

```r
# Check prod_qty
sort(table(transactionData$PROD_QTY), decreasing = T)
```

```
##
##      2      1      5      3      4    200
## 220070  25476    415    408    371      2
```

```r
print(transactionData[PROD_QTY > 226201])
```

```
## Empty data.table (0 rows and 8 cols): DATE,STORE_NBR,LYLTY_CARD_NBR,TXN_ID,PROD_NBR,PROD_NAME...
```

It can be seen that there are no nulls indicated in any rows. It can also be seen that there is a transaction involving 200 items. This is an outlier and should be removed. All other transactions involve product quantity of 5 or less and thus are congruent.

The following is an investigation to see if the outlier was responsible for any other transactions that are reasonable

```r
# Check prod_qty
sort(table(transactionData$PROD_QTY), decreasing = T)
```

```
##
##      2      1      5      3      4    200
## 220070  25476    415    408    371      2
```

```r
# Check PROD_QTY==200 transactions
print(transactionData[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
```

```r
# Check if customer had any other transactions
print(transactionData[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
```

```r
# Remove commerical customer from dataset
transactionData <- transactionData[LYLTY_CARD_NBR != 226000]
```

The customer who made the transactions involving product quantities of 200 was not responsible for any other transactions. It's likely they were buying for commercial purposes and can be ignored by removing his transactions from dataset

```r
# Check if any values are empty or null
missingData <- transactionData[apply(transactionData, 1, function(x) any(!nzchar(x)) || any(is.na(x))),]
print(missingData)
```

```
## Empty data.table (0 rows and 8 cols): DATE,STORE_NBR,LYLTY_CARD_NBR,TXN_ID,PROD_NBR,PROD_NAME...
```

There are no empty strings or null values in data. Further cleaning would include checking if any strings existed with only whitespace

```r
numDates <- length(unique(transactionData$DATE))
print(numDates)
```

### Check For Missing Dates

```
## [1] 364
```

There are only 364 dates present, indicating one is missing. Let's find the missing one and add it in

```r
partialYear <- as.Date(unique(transactionData$DATE) , origin = "1899-12-30")
fullYear <- seq(as.Date("2018/7/1"), by = "day", length.out = 365)

missingDate <- fullYear[!(fullYear %in% partialYear)]

print(missingDate)
```

```
## [1] "2018-12-25"
```

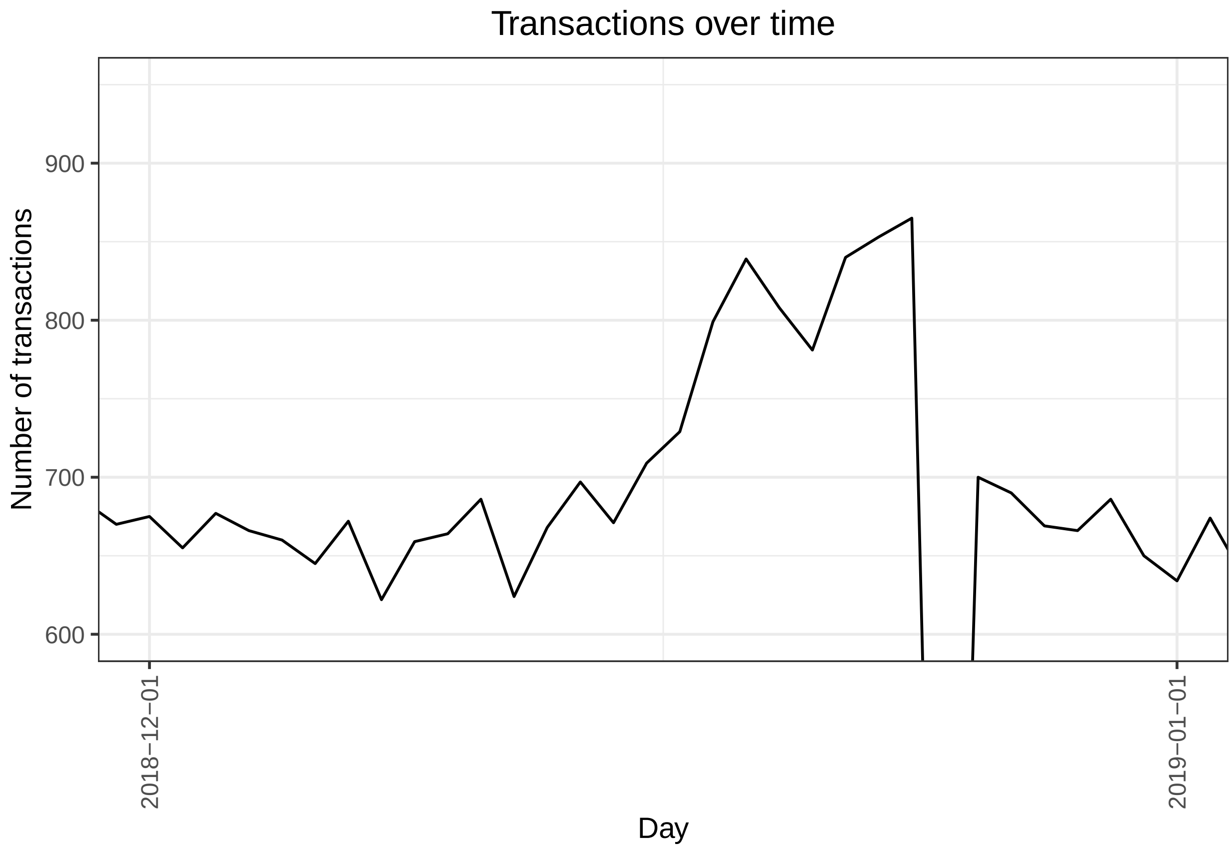
```r
transactionsByDay <- data.table(table(c(as.Date(transactionData$DATE, origin = "1899-12-30"), missingDa
setnames(transactionsByDay, c('day', 'count'))
transactionsByDay$day <- as.Date(transactionsByDay$day)

str(transactionsByDay)
```

```
## Classes 'data.table' and 'data.frame':   365 obs. of  2 variables:
##  $ day  : Date, format: "2018-07-01" "2018-07-02" ...
##  $ count: int  663 650 674 669 660 711 695 653 692 650 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

We can see that the missing date is Christmas day.

```r
#### Setting plot themes to format graphs
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
#### Plot transactions over time
ggplot(transactionsByDay, aes(x = transactionsByDay$day, y = transactionsByDay$count)) +
 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)) +
 coord_cartesian(xlim = c(as.Date('2018-12-01'),as.Date('2019-01-01')), ylim=c(600, 950))
```




We can see that there is an increase in sales leading up to Christmas and then a dip afterwards. No sales on christmas day as not trading.

**Check if Packet Sizes are Reasonable** Get packsizes

```
transactionData[, PACK_SIZE := parse_number(PROD_NAME)]

# .N refers to number of instances, below is a shorthand way of counting instances by column=PACK_SIZE
packSizes <- transactionData[, .N, PACK_SIZE][order(PACK_SIZE)]

# Order by frequency to see largest pack size
print(packSizes[order(N)])
```

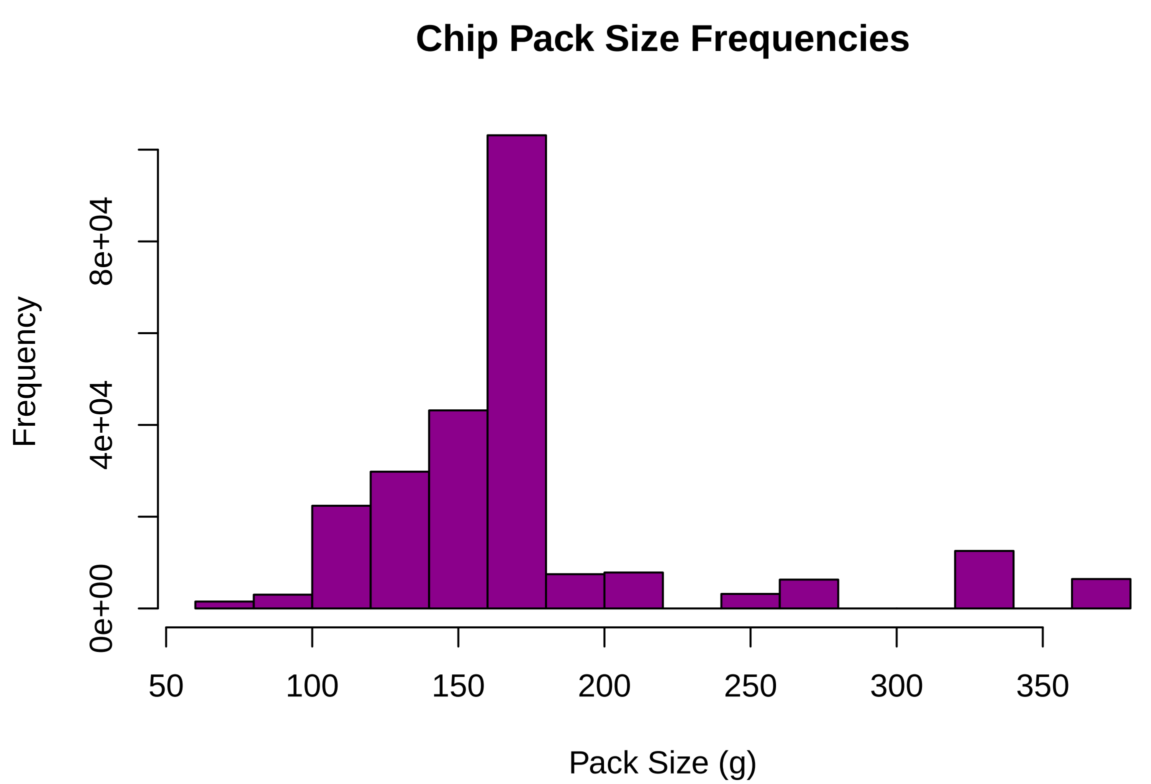
```
##     PACK_SIZE    N
##  1:       125 1454
##  2:       180 1468
##  3:        70 1507
##  4:       220 1564
##  5:       160 2970
##  6:       190 2995
##  7:        90 3008
##  8:       250 3169
##  9:       135 3257
## 10:       200 4473
```

```
## 11:         210   6272
## 12:         270   6285
## 13:         380   6416
## 14:         330  12540
## 15:         165  15297
## 16:         170  19983
## 17:         110  22387
## 18:         134  25102
## 19:         150  40203
## 20:         175  66390
```

We can see that the min is 70g and max 380g which is quite reasonable for chip packets. 175g is also the most frequently bought pack size, it also happens to be in the middle of both extremes.

Plotting histogram for pack size frequencies

```
hist(transactionData$PACK_SIZE,
     main="Chip Pack Size Frequencies",
        xlab="Pack Size (g)",
        ylab="Frequency",
        # xlim=c(50,100),
        col="darkmagenta")
```




#### Reduce PROD_NAME to Unique Brand Add column for brand

```r
transactionData[, BRAND := tstrsplit(PROD_NAME, " ", fixed=TRUE)[1]]
print(transactionData[, .N, BRAND][order(BRAND)])
```

```
##          BRAND     N
##  1:     Burger  1564
##  2:        CCs  4551
##  3:    Cheetos  2927
##  4:   Cheezels  4603
##  5:       Cobs  9693
##  6:     Dorito  3183
##  7:    Doritos 22041
##  8:     French  1418
##  9:      Grain  6272
## 10:    GrnWves  1468
## 11:   Infuzions 11057
## 12:     Infzns  3144
## 13:     Kettle 41288
## 14:        NCC  1419
## 15:    Natural  6050
## 16:   Pringles 25102
## 17:        RRD 11894
## 18:        Red  4427
## 19:      Smith  2963
## 20:     Smiths 27390
## 21:      Snbts  1576
## 22:   Sunbites  1432
## 23:      Thins 14075
## 24:   Tostitos  9471
## 25:   Twisties  9454
## 26:   Tyrrells  6442
## 27:         WW 10320
## 28: Woolworths  1516
##          BRAND     N
```

It can be seen that there are 7 brands represented in multiple forms. These will be merged. They are mapped below: * RRD, Red -> RRD * Sunbites, Snbts -> Sunbites * GrnWves, Grain -> GrnWves * WW, Woolworths -> Woolworths * Smith, Smiths -> Smiths * Infuzions, Infzns -> Infuzions * Dorito, Doritos -> Doritos

```r
transactionData[BRAND == "Snbts", BRAND := "Sunbites"]
transactionData[BRAND == "Grain", BRAND := "GrnWves"]
transactionData[BRAND == "WW", BRAND := "Woolworths"]
transactionData[BRAND == "Smith", BRAND := "Smiths"]
transactionData[BRAND == "Infzns", BRAND := "Infuzions"]
transactionData[BRAND == "Dorito", BRAND := "Doritos"]
transactionData[BRAND == "Red", BRAND := "RRD"]


# Confirm mappings were successful
print(transactionData[, .N, BRAND][order(BRAND)])
```

```
##          BRAND     N
```

```
##  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:        NCC  1419
## 12:    Natural  6050
## 13:   Pringles 25102
## 14:        RRD 16321
## 15:     Smiths 30353
## 16:   Sunbites  3008
## 17:      Thins 14075
## 18:   Tostitos  9471
## 19:   Twisties  9454
## 20:   Tyrrells  6442
## 21: Woolworths 11836
##          BRAND     N
```

This all looks good.

### Exploring Customer Data

Let's now explore customer data.

```
# Check data format
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" "OLDER SI
##  $ PREMIUM_CUSTOMER: chr  "Premium" "Mainstream" "Budget" "Mainstream" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Get data summary
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
```

```
# See set of unique values and which dominate
print(customerData[,.N,LIFESTAGE][order(N, decreasing = TRUE)])
```

```
##                LIFESTAGE     N
## 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
```

```r
print(customerData[,.N,PREMIUM_CUSTOMER][order(N, decreasing = TRUE)])
```

```
##    PREMIUM_CUSTOMER     N
## 1:       Mainstream 29245
## 2:           Budget 24470
## 3:          Premium 18922
```

```r
# Check for any missing entries
print(customerData[is.null(PREMIUM_CUSTOMER), .N] )
```

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

Most customers (who have a loyalty card) are retirees. Interestingly, older and young couples have loyalty cards in similar number to retirees and significantly more than familes.

In accordance with expectations, most customers are Mainstream, followed by budget and then premium.

Merge customer data with transaction data

```r
data <- merge(transactionData, customerData, all.x = TRUE)
print(data)
```

```
##         LYLTY_CARD_NBR       DATE STORE_NBR TXN_ID PROD_NBR
##      1:           1000 2018-10-17         1      1        5
##      2:           1002 2018-09-16         1      2       58
##      3:           1003 2019-03-07         1      3       52
##      4:           1003 2019-03-08         1      4      106
##      5:           1004 2018-11-02         1      5       96
##     ---
## 246736:        2370651 2018-08-03        88 240350        4
## 246737:        2370701 2018-12-08        88 240378       24
## 246738:        2370751 2018-10-01        88 240394       60
## 246739:        2370961 2018-10-24        88 240480       70
## 246740:        2373711 2018-12-14        88 241815       16
##                                        PROD_NAME PROD_QTY TOT_SALES PACK_SIZE
##      1:   Natural Chip        Compny SeaSalt175g        2       6.0       175
##      2:    Red Rock Deli Chikn&Garlic Aioli 150g        1       2.7       150
##      3:    Grain Waves Sour    Cream&Chives 210G        1       3.6       210
##      4:   Natural ChipCo      Hony Soy Chckn175g        1       3.0       175
##      5:           WW Original Stacked Chips 160g        1       1.9       160
##     ---
## 246736:         Dorito Corn Chp     Supreme 380g        2      13.0       380
## 246737:    Grain Waves          Sweet Chilli 210g        2       7.2       210
## 246738:     Kettle Tortilla ChpsFeta&Garlic 150g        2       9.2       150
```

```
## 246739:  Tyrrells Crisps     Lightly Salted 165g        2       8.4       165
## 246740: Smiths Crinkle Chips Salt & Vinegar 330g        2      11.4       330
##               BRAND              LIFESTAGE PREMIUM_CUSTOMER
##      1:    Natural  YOUNG SINGLES/COUPLES          Premium
##      2:        RRD  YOUNG SINGLES/COUPLES       Mainstream
##      3:    GrnWves         YOUNG FAMILIES           Budget
##      4:    Natural         YOUNG FAMILIES           Budget
##      5: Woolworths  OLDER SINGLES/COUPLES       Mainstream
##     ---
## 246736:    Doritos MIDAGE SINGLES/COUPLES       Mainstream
## 246737:    GrnWves         YOUNG FAMILIES       Mainstream
## 246738:     Kettle         YOUNG FAMILIES          Premium
## 246739:   Tyrrells         OLDER FAMILIES           Budget
## 246740:     Smiths  YOUNG SINGLES/COUPLES       Mainstream
```

Check there are no entries missing loyalty numbers

```
print(data[is.null(PREMIUM_CUSTOMER), .N])
```

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

All transactions have corresponding customers.

Write out to file

```
fwrite(data, paste0(filePath,"QVI_data.csv"))
```

## Data Analysis

### Metrics

Metrics to investigate: * Who spends the most on chips (total sales), describing customers by lifestage and how premium their general purchasing behaviour 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

### Total Sales by Customer Segment

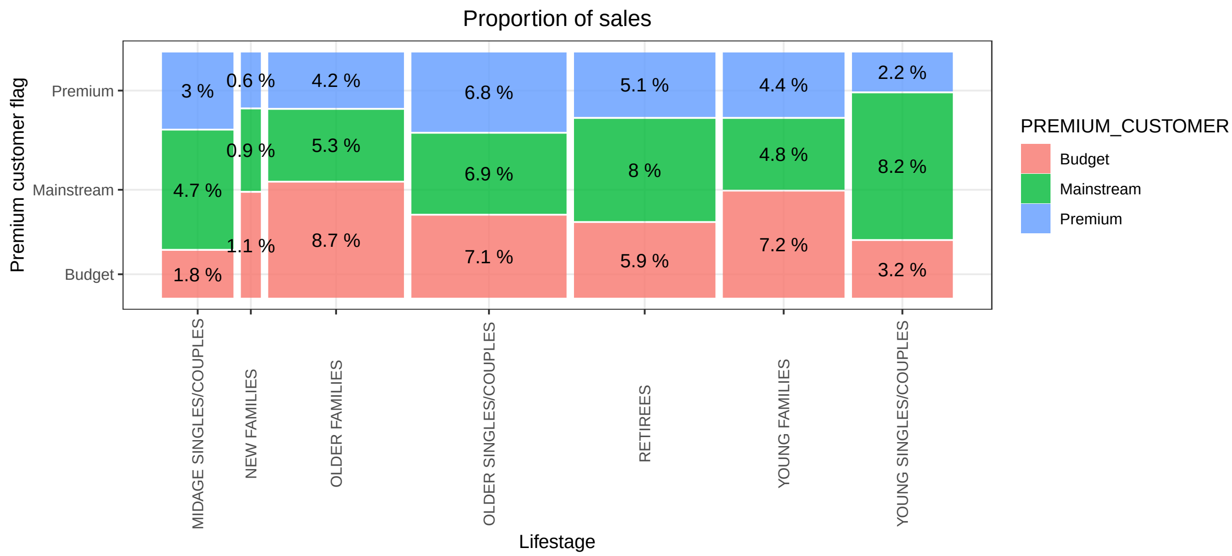
```
#### Total sales and items sold by LIFESTAGE and PREMIUM_CUSTOMER
sumsBySegment <- data[,list(SALES=sum(TOT_SALES), packets=sum(PROD_QTY)), by=c('LIFESTAGE', 'PREMIUM_CUS
# sales <- data[, .(SALES = sum(TOT_SALES)), .(LIFESTAGE, PREMIUM_CUSTOMER)] # Data.table way to do the


# Grouped bar plot
# ggplot(sumsBySegment, aes(fill=PREMIUM_CUSTOMER, y=LIFESTAGE, x=total_sales)) +
#     geom_bar(position="dodge", stat="identity")+
#         ggtitle("Total sales by LIFESTAGE and PREMIUM_CUSTOMER")

p <- ggplot(data = sumsBySegment) +
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,'%'))))
```




Sales are coming mainly from Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees

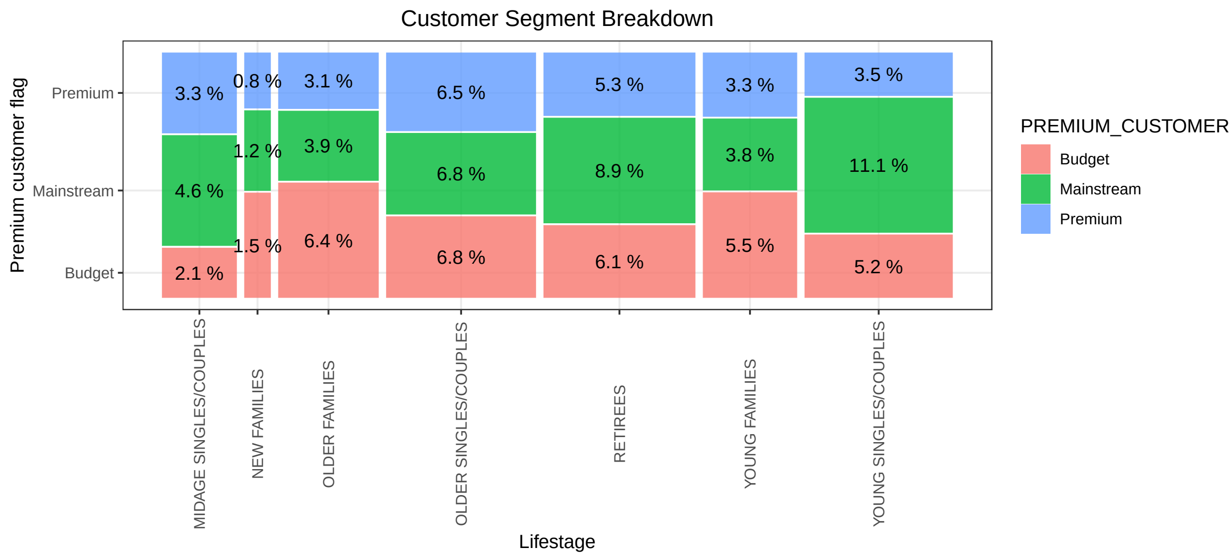
### Total Customers and Packets Per Customer by Customer Segment

Let's calculate number of customers by Lifestage and Premium to see if the higher sales in those customer segments are due to a higher population

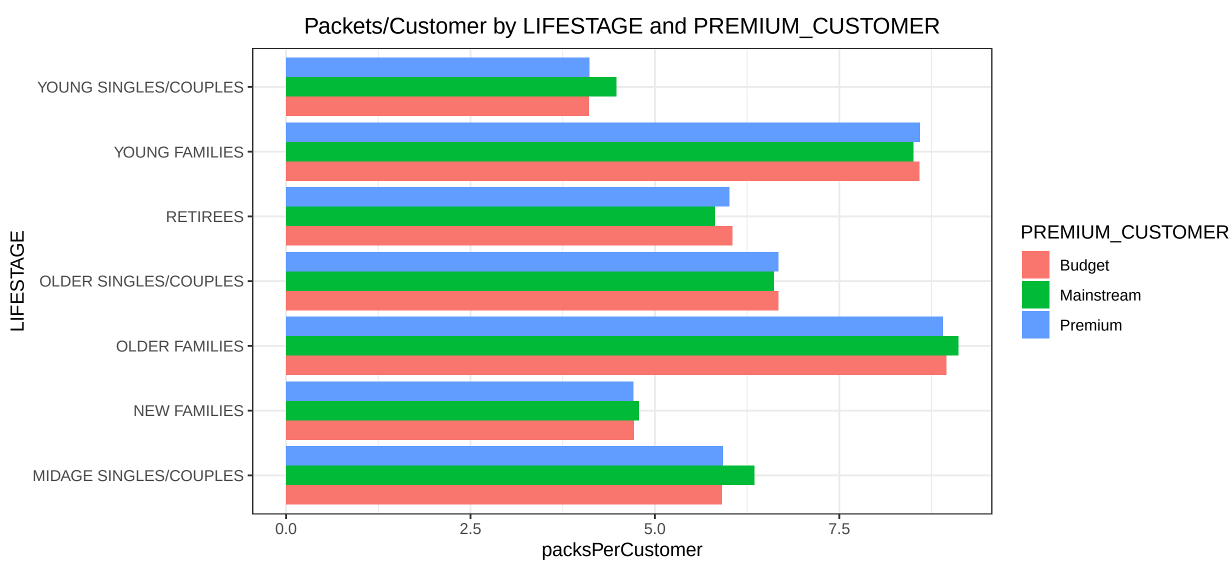
```
#### Total customers by LIFESTAGE and PREMIUM_CUSTOMER
customersBySegment <- customerData[,.N,by=c('LIFESTAGE', 'PREMIUM_CUSTOMER')]

p <- ggplot(data = customersBySegment) +
geom_mosaic(aes(weight = N, x = product(PREMIUM_CUSTOMER, LIFESTAGE),
fill = PREMIUM_CUSTOMER)) +
labs(x = "Lifestage", y = "Premium customer flag", title = "Customer Segment Breakdown") +
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,'%'))))
```




```r
# Packets per Customer by LIFESTAGE and PREMIUM_CUSTOMER
packetsPerCustomerBySegment <- customersBySegment[, packsPerCustomer := sumsBySegment$packets / N ]
ggplot(packetsPerCustomerBySegment, aes(fill=PREMIUM_CUSTOMER, y=LIFESTAGE, x=packsPerCustomer)) +
    geom_bar(position="dodge", stat="identity")+
        ggtitle("Packets/Customer by LIFESTAGE and PREMIUM_CUSTOMER")
```




Mainstream young singles/couples dominate the customer base, followed by retirees. By plotting the chip packets per customer we can see that families buy the most as they are likely buying for multiple people.

There appears to be a trend in the age of the customer segment. The older a single, couple or family is, the more packets they buy

The main takeaway is that older and young families buy the most chips per customer

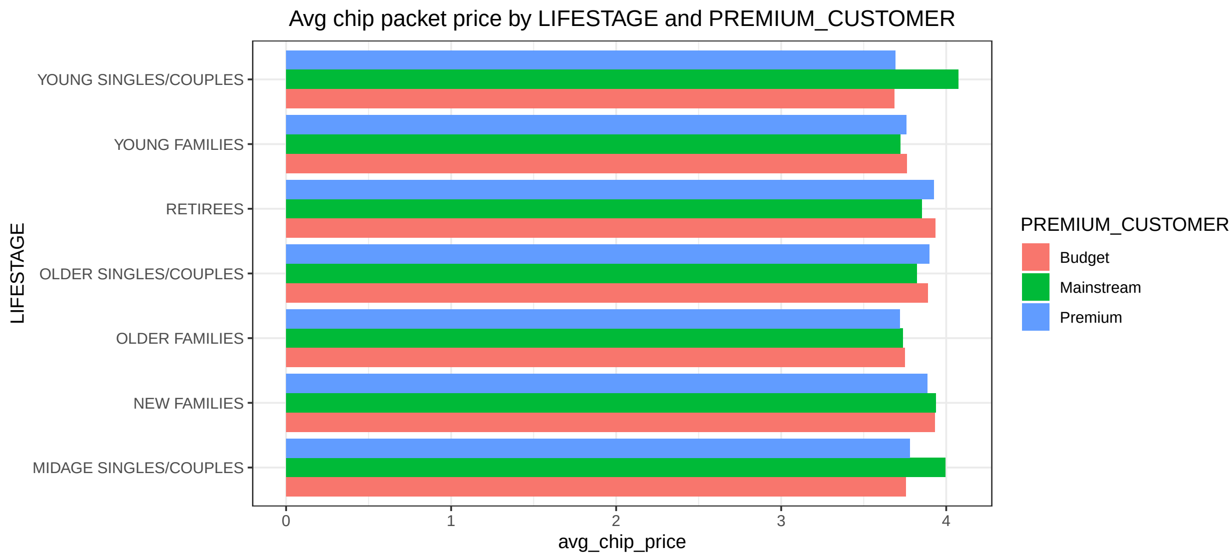
### Average Chip Prices By Customer Segment

Calculate avg chip price per customer segment

```
#### Total customers by LIFESTAGE and PREMIUM_CUSTOMER
sumsBySegment[, avg_chip_price := SALES / packets]
print(sumsBySegment)
```

```
##                  LIFESTAGE PREMIUM_CUSTOMER     SALES packets avg_chip_price
##  1:  YOUNG SINGLES/COUPLES          Premium  39052.30   10575       3.692889
##  2:  YOUNG SINGLES/COUPLES       Mainstream 147582.20   36225       4.074043
##  3:         YOUNG FAMILIES           Budget 129717.95   34482       3.761903
##  4:  OLDER SINGLES/COUPLES       Mainstream 124648.50   32607       3.822753
##  5: MIDAGE SINGLES/COUPLES       Mainstream  84734.25   21213       3.994449
##  6:  YOUNG SINGLES/COUPLES           Budget  57122.10   15500       3.685297
##  7:           NEW FAMILIES          Premium  10760.80    2769       3.886168
##  8:         OLDER FAMILIES       Mainstream  96413.55   25804       3.736380
##  9:               RETIREES           Budget 105916.30   26932       3.932731
## 10:  OLDER SINGLES/COUPLES          Premium 123537.55   31695       3.897698
## 11:         OLDER FAMILIES           Budget 156863.75   41853       3.747969
## 12: MIDAGE SINGLES/COUPLES          Premium  54443.85   14400       3.780823
## 13:         OLDER FAMILIES          Premium  75242.60   20239       3.717703
## 14:               RETIREES       Mainstream 145168.95   37677       3.852986
## 15:               RETIREES          Premium  91296.65   23266       3.924037
## 16:         YOUNG FAMILIES       Mainstream  86338.25   23194       3.722439
## 17: MIDAGE SINGLES/COUPLES           Budget  33345.70    8883       3.753878
## 18:           NEW FAMILIES       Mainstream  15979.70    4060       3.935887
## 19:  OLDER SINGLES/COUPLES           Budget 127833.60   32883       3.887529
## 20:         YOUNG FAMILIES          Premium  78571.70   20901       3.759232
## 21:           NEW FAMILIES           Budget  20607.45    5241       3.931969
##                  LIFESTAGE PREMIUM_CUSTOMER     SALES packets avg_chip_price
```

```
ggplot(sumsBySegment, aes(fill=PREMIUM_CUSTOMER, y=LIFESTAGE, x=avg_chip_price)) +
    geom_bar(position="dodge", stat="identity") +
        ggtitle("Avg chip packet price by LIFESTAGE and PREMIUM_CUSTOMER")
```




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

### T-test to Verify Statistical Significance

Do a t test on avg chip packet price between Mainstream vs Premium & Budget wrt Young and Midage Single/Couples to see if there is a statistically significant difference

```
# Calculate avg chip prices
data <- data[, avgChipPacketPrice := TOT_SALES / PROD_QTY]

mainstream <- data[(LIFESTAGE == 'YOUNG SINGLES/COUPLES' | LIFESTAGE == 'MIDAGE SINGLES/COUPLES') & PREM
premiumBudget <- data[(LIFESTAGE == 'YOUNG SINGLES/COUPLES' | LIFESTAGE == 'MIDAGE SINGLES/COUPLES') &

t.test(mainstream,premiumBudget, alternative = "greater")
```

```
##
##  Welch Two Sample t-test
##
## data:  mainstream and premiumBudget
## 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 yields a p-value of less than 2.2e-16 concluding 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.

**Investigate Target Segments**

Mainstream - young singles/couples and budget older families are two of the top contributors to sales and are thus apt target segments. Let's look at their most preferred brand. We may want to target them to retain or increase sales.

```
# mainstream - young singles/couples
myscBrands <- data[LIFESTAGE == 'YOUNG SINGLES/COUPLES'  & PREMIUM_CUSTOMER == 'Mainstream'][, .N, BRAN
print(myscBrands)
```

```
##          BRAND    N
##  1:     Kettle 3844
##  2:    Doritos 2379
##  3:   Pringles 2315
##  4:     Smiths 1921
##  5:  Infuzions 1250
##  6:      Thins 1166
##  7:   Twisties  900
##  8:   Tostitos  890
##  9:        RRD  875
## 10:       Cobs  864
## 11:    GrnWves  646
## 12:   Tyrrells  619
## 13: Woolworths  479
## 14:   Cheezels  346
## 15:    Natural  321
## 16:        CCs  222
## 17:    Cheetos  166
## 18:   Sunbites  128
## 19:     French   78
## 20:        NCC   73
## 21:     Burger   62
##          BRAND    N
```

```
# budget - older families
bofBrands <- data[LIFESTAGE == 'OLDER FAMILIES'  & PREMIUM_CUSTOMER == 'Budget'][, .N, BRAND][order(N, 
print(bofBrands)
```

```
##          BRAND    N
##  1:     Kettle 3320
##  2:     Smiths 2948
##  3:    Doritos 2032
##  4:   Pringles 1996
##  5:        RRD 1708
##  6: Woolworths 1213
##  7:  Infuzions 1185
##  8:      Thins 1171
##  9:   Twisties  810
## 10:       Cobs  760
## 11:   Tostitos  705
## 12:    GrnWves  671
## 13:    Natural  576
## 14:   Tyrrells  489
```

```
## 15:        CCs  451
## 16:   Cheezels  427
## 17:   Sunbites  305
## 18:    Cheetos  281
## 19:        NCC  165
## 20:     Burger  159
## 21:     French  142
##          BRAND    N
```

Both segments share the same top 4 brands in slightly different order. However, both share Kettle as number 1 brand. If the client wanted to target these segments, Kettle would cover both. EDIT: Upon seeing the solution, it's clear that the above analysis is flawed as it does not take into account the affinity of these target segments for certain brands with respect to all OTHER segments. Below is an affinity analysis which does just this.

```r
#### 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)]
# print(quantity_segment1)

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

print(quantity_segment1_by_brand)
```

**Affinity Analysis**

```
##          BRAND targetSegment
##  1:        RRD   0.043809524
##  2:    Doritos   0.122760524
##  3:     Kettle   0.197984817
##  4:  Infuzions   0.064679089
##  5:     Smiths   0.096369910
##  6:    GrnWves   0.032712215
##  7:   Tyrrells   0.031552795
##  8:   Twisties   0.046183575
##  9:       Cobs   0.044637681
## 10:   Pringles   0.119420290
## 11:    Natural   0.015955832
## 12:   Cheezels   0.017971014
## 13:     Burger   0.002926156
## 14: Woolworths   0.024099379
## 15:   Sunbites   0.006349206
## 16:      Thins   0.060372671
## 17:   Tostitos   0.045410628
## 18:     French   0.003947550
## 19:        CCs   0.011180124
## 20:    Cheetos   0.008033126
## 21:        NCC   0.003643892
```

```
##          BRAND targetSegment
```

```
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 := tar
brand_proportions[order(-affinityToBrand)]
```

```
##          BRAND targetSegment       other affinityToBrand
##  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   0.015955832 0.024980768       0.6387246
## 17:        NCC   0.003643892 0.005873221       0.6204248
## 18:        CCs   0.011180124 0.018895650       0.5916771
## 19:   Sunbites   0.006349206 0.012580210       0.5046980
## 20: Woolworths   0.024099379 0.049427188       0.4875733
## 21:     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

**Investigate Packet Size of Target Segments** Let's also look at packsize relative to these target segments

```
# mainstream - young singles/couples
myscBrands <- data[LIFESTAGE == 'YOUNG SINGLES/COUPLES'  & PREMIUM_CUSTOMER == 'Mainstream'][, .N, PACK_
print(myscBrands)
```

```
##     PACK_SIZE    N
##  1:       175 4997
##  2:       150 3080
##  3:       134 2315
##  4:       110 2051
##  5:       170 1575
##  6:       330 1195
##  7:       165 1102
##  8:       380  626
##  9:       270  620
## 10:       210  576
```

```
## 11:       135  290
## 12:       250  280
## 13:       200  179
## 14:       190  148
## 15:        90  128
## 16:       160  128
## 17:       180   70
## 18:        70   63
## 19:       220   62
## 20:       125   59
```

```
# budget - older families
bofBrands <- data[LIFESTAGE == 'OLDER FAMILIES'  & PREMIUM_CUSTOMER == 'Budget'][, .N, PACK_SIZE][order
print(bofBrands)
```

```
##     PACK_SIZE    N
##  1:       175 5808
##  2:       150 3588
##  3:       134 1996
##  4:       110 1803
##  5:       170 1786
##  6:       165 1358
##  7:       330 1092
##  8:       270  532
##  9:       380  510
## 10:       210  505
## 11:       200  448
## 12:       190  312
## 13:       160  306
## 14:        90  305
## 15:       250  278
## 16:       135  268
## 17:       180  166
## 18:       220  159
## 19:       125  152
## 20:        70  142
```

They both share the same top 5 pack sizes, with 175g being principly preferred. EDIT: Like with brand affinity above, we will do similarly for packet sizes

```
#### Preferred pack size compared to the rest of the population
quantity_segment1_by_pack <- segment1[, .(targetSegment = sum(PROD_QTY)/quantity_segment1), by = PACK_S
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 := targetS
pack_proportions[order(-affinityToPack)]
```

```
##     PACK_SIZE targetSegment       other affinityToPack
##  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
```

We can see that our target segment is 27% more likely to purchase a pack size of 270g compared to the rest of the population. Let's look at the relationship between pack size and brand

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

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

Only Twisties sell 270g, this suggests the pack size affinity may actually reflect a higher likelihood of buying twisties

## Recommendation

Initial findings for Julia in regards to chip sales with respect to customer segments are as follows.

Sales have mainly been due to Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees shoppers.

It was determined that mainstream young singles/couples and retirees contributed more to sales due to being highly represented in customer base.

Mainstream, midage and young singles and couples are also more likely to pay more per packet of chips. This is indicative of impulse buying behaviour given that they are likely buying for themselves unlike other segments.

We've also found that Mainstream young singles and couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population. The Category Manager may want to increase the category's performance by off-locating some Tyrrells and smaller packs of chips in discretionary space near segments where young singles and couples frequent more often to increase visibilty and impulse behaviour.