Quantium Virtual Internship - Retail Strategy and Analytics

Sample solution for Data preparation and customer analytics task

This file outlines a sample response to the data insights and strategy task using R.

Load required libraries and datasets

```
#### Example code to install packages
#install.packages("data.table")

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

#### Point the filePath to where you have downloaded the datasets to and
#### assign the data files to data.tables
filePath <- "C:/Users/victorb/Quantium/Sarah D'oliveiro - InsideSherpa
 ↪  (internal)/"
transactionData <- fread(paste0(filePath,"QVI_transaction_data.csv"))
customerData <- fread(paste0(filePath,"QVI_purchase_behaviour.csv"))
```

Exploratory data analysis

The first step in any analysis is to first understand the data. Let's take a look at each of the datasets provided.

Examining transaction data

We can use `str()` to look at the format of each column and see a sample of the data. As we have read in the dataset as a `data.table` , we can also run `transactionData` in the console to see a sample of the data or use `head(transactionData)` to look at the first 10 rows.

Let's check if columns we would expect to be numeric are in numeric form and date columns are in date format.

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

```
## Classes 'data.table' and 'data.frame': 264836 obs. of 8 variables:
## $ DATE : int 43390 43599 43605 43329 43330 43604 43601 43601 43332 43330 ...
## $ STORE_NBR : int 1 1 1 2 2 4 4 4 5 7 ...
## $ LYLTY_CARD_NBR: int 1000 1307 1343 2373 2426 4074 4149 4196 5026 7150 ...
## $ TXN_ID : int 1 348 383 974 1038 2982 3333 3539 4525 6900 ...
## $ PROD_NBR : int 5 66 61 69 108 57 16 24 42 52 ...
## $ PROD_NAME : chr "Natural Chip Compny SeaSalt175g" "CCs Nacho Cheese 175g"
"Smiths Crinkle Cut Chips Chicken 170g" "Smiths Chip Thinly S/Cream&Onion 175g"
...
## $ PROD_QTY : int 2 3 2 5 3 1 1 1 1 2 ...
## $ TOT_SALES : num 6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
## - attr(*, ".internal.selfref")=<externalptr>
```

We can see that the date column is in an integer format. Let's change this to a date format.

```
#### Convert DATE column to a date format
#### A quick Google search 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.

```
#### Examine PROD_NAME
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
```

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

```
#### Examine the words in PROD_NAME to see if there are any incorrect entries
#### such as products that are not chips
productWords <- data.table(unlist(strsplit(unique(transactionData[,
 ↪ PROD_NAME]), " ")))
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
#### Removing digits
productWords <- productWords[grepl("\\d", words) == FALSE, ]
#### Removing special characters
productWords <- productWords[grepl("[:alpha:]", words), ]
#### Let's look at the most common words by counting the number of times a word
↪ appears and
#### sorting them by this frequency in order of highest to lowest frequency
productWords[, .N, words][order(N, decreasing = TRUE)]
```

```
##              words  N
##    1:        Chips 21
##    2:       Smiths 16
##    3:      Crinkle 14
##    4:       Kettle 13
##    5:       Cheese 12
##   ---
## 127: Chikn&Garlic  1
## 128:        Aioli  1
## 129:         Slow  1
## 130:        Belly  1
## 131:    Bolognese  1
```

```r
#### Note that sorting by negative N gives us the same result
#productWords[, .N, words][order(-N)]
#### This shortcut is used in later steps
```

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

Next, we can use `summary()` to check summary statistics such as mean, min and max values for each feature to see if there are any obvious outliers in the data and if there are any nulls in any of the columns (`NA's : number of nulls` will appear in the output if there are any nulls).

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

```
##       DATE              STORE_NBR     LYLTY_CARD_NBR        TXN_ID
##  Min.   :2018-07-01   Min.   :  1.0   Min.   :   1000   Min.   :      1
##  1st Qu.:2018-09-30   1st Qu.: 70.0   1st Qu.:  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
```

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

```r
#### Filter the dataset to find the outlier
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
```

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

```r
#### Let's see if the customer has had other transactions
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
```

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 <- transactionData[LYLTY_CARD_NBR != 226000, ]

#### Re-examine transaction data
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.:  70021   1st Qu.:  67601
##  Median :2018-12-30   Median :130.0   Median : 130357   Median : 135137
##  Mean   :2018-12-30   Mean   :135.1   Mean   : 135549   Mean   : 135158
##  3rd Qu.:2019-03-31   3rd Qu.:203.0   3rd Qu.: 203094   3rd Qu.: 202700
##  Max.   :2019-06-30   Max.   :272.0   Max.   :2373711   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
```

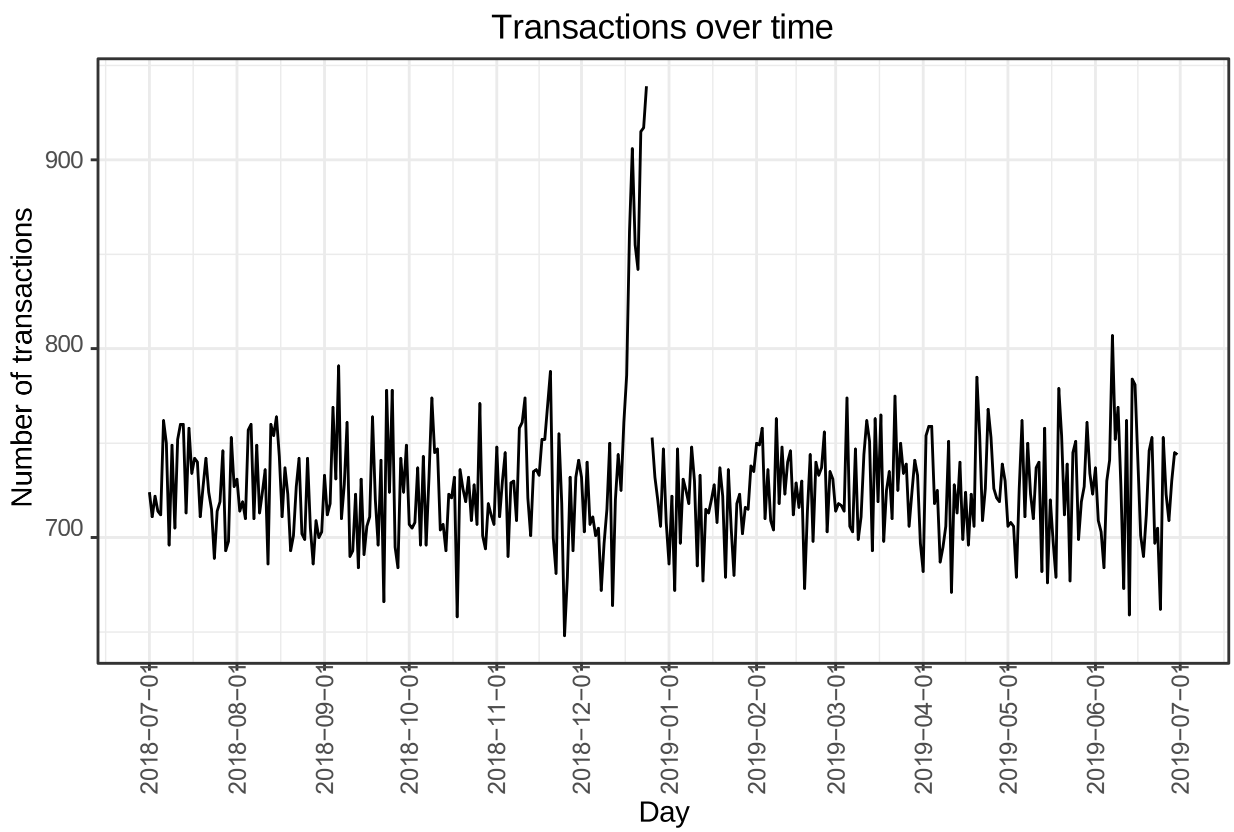
Now, let's look at the number of transaction lines over time to see if there are any obvious data issues such as missing data.

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

```
##            DATE   N
##   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
```

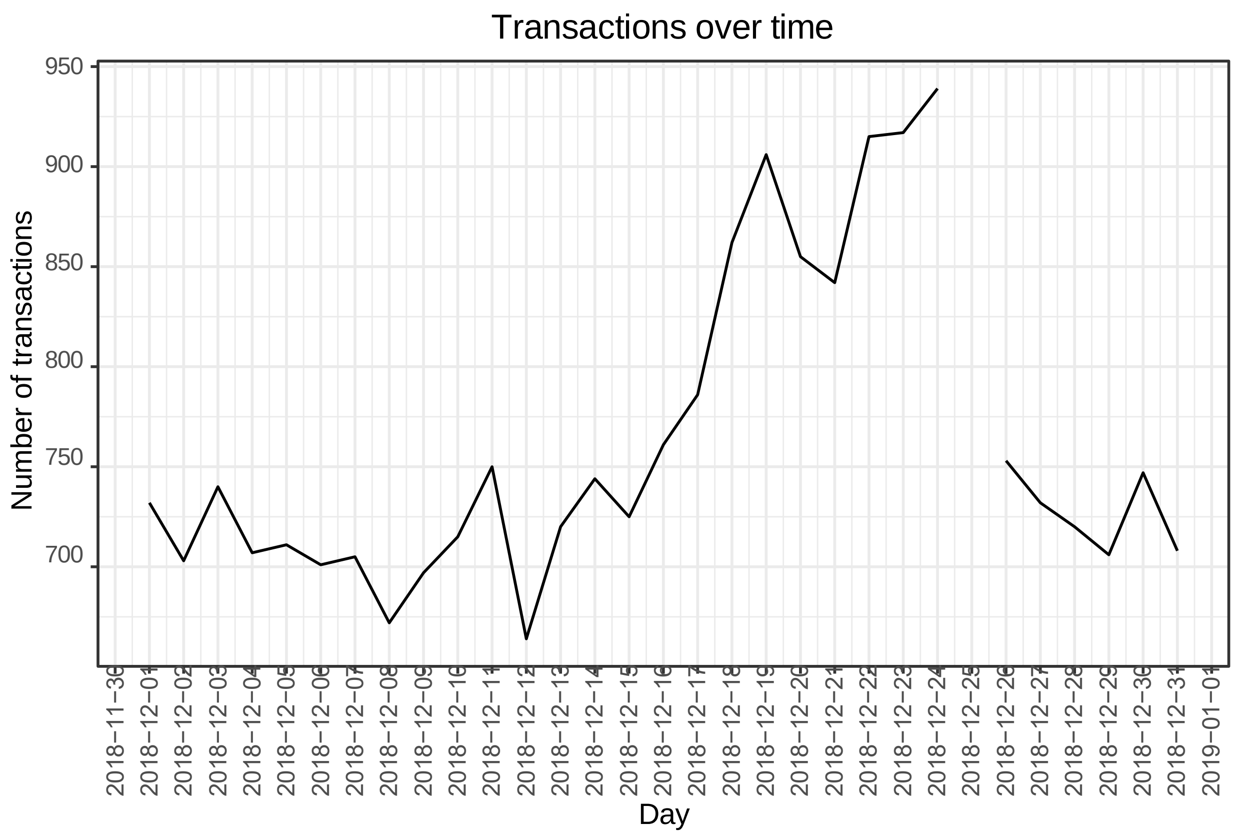
There's only 364 rows, meaning only 364 dates which indicates a missing date. Let's create a sequence of dates from 1 Jul 2018 to 30 Jun 2019 and use this to create a chart of number of transactions over time to find the missing date.

```
#### Create a sequence of dates and join this the count of transactions by date
allDates <- data.table(seq(as.Date("2018/07/01"), as.Date("2019/06/30"), by =
↪   "day"))
setnames(allDates, "DATE")
transactions_by_day <- merge(allDates, transactionData[, .N, by = DATE], all.x
↪   = TRUE)

#### 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. Let's zoom in on this.

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



We can see that the increase in sales occurs in the lead-up to Christmas and that there are zero sales on Christmas day itself. This is due to shops being closed on Christmas day.

Now that we are satisfied that the data no longer has outliers, we can move on to creating other features such as brand of chips or pack size from PROD_NAME. We will start with pack size.

```
#### Pack size
#### We can work this out by taking the digits that are in PROD_NAME
transactionData[, PACK_SIZE := parse_number(PROD_NAME)]

#### Always check your output
#### Let's check if the pack sizes look sensible
transactionData[, .N, PACK_SIZE][order(PACK_SIZE)]
```

```
##      PACK_SIZE     N
##   1:        70  1507
##   2:        90  3008
##   3:       110 22387
##   4:       125  1454
##   5:       134 25102
##   6:       135  3257
##   7:       150 43131
##   8:       160  2970
##   9:       165 15297
##  10:       170 19983
##  11:       175 66390
```

```
## 12:       180   1468
## 13:       190   2995
## 14:       200   4473
## 15:       210   6272
## 16:       220   1564
## 17:       250   3169
## 18:       270   6285
## 19:       300  15166
## 20:       330  12540
## 21:       380   6416
##     PACK_SIZE      N
```

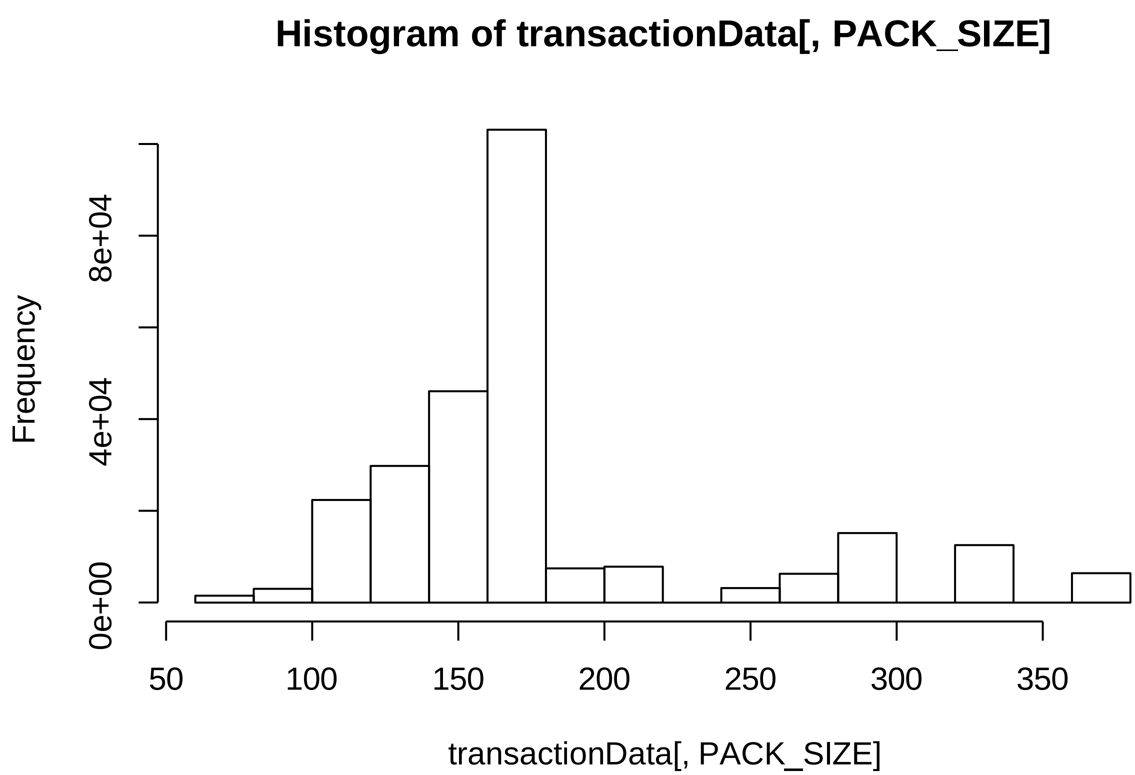
The largest size is 380g and the smallest size is 70g - seems sensible!

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

```
##                DATE  STORE_NBR  LYLTY_CARD_NBR TXN_ID  PROD_NBR
##      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
##     ---
## 264830:  2019-03-09        272          272319 270088        89
## 264831:  2018-08-13        272          272358 270154        74
## 264832:  2018-11-06        272          272379 270187        51
## 264833:  2018-12-27        272          272379 270188        42
## 264834:  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
##     ---
## 264830:  Kettle Sweet Chilli And Sour Cream 175g        2       10.8
## 264831:            Tostitos Splash Of  Lime 175g        1        4.4
## 264832:                 Doritos Mexicana     170g        2        8.8
## 264833:  Doritos Corn Chip Mexican Jalapeno  150g        2        7.8
## 264834:            Tostitos Splash Of  Lime  175g        2        8.8
##         PACK_SIZE
##      1:       175
##      2:       175
##      3:       170
##      4:       175
##      5:       150
##     ---
## 264830:       175
## 264831:       175
## 264832:       170
```

```
## 264833:        150
## 264834:        175
```

```r
#### Let's plot a histogram of PACK_SIZE since we know that it is a categorical
↪  variable and not a continuous variable even though it is numeric.
hist(transactionData[, PACK_SIZE])
```



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

```r
#### Brands
transactionData[, BRAND := toupper(substr(PROD_NAME, 1, regexpr(pattern = ' ',
↪  PROD_NAME) - 1))]

#### Checking brands
transactionData[, .N, by = BRAND][order(-N)]
```

```
##           BRAND     N
##   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
##  11:    TWISTIES  9454
##  12:         OLD  9324
##  13:    TYRRELLS  6442
##  14:       GRAIN  6272
##  15:     NATURAL  6050
##  16:         RED  5885
##  17:    CHEEZELS  4603
##  18:         CCS  4551
##  19:  WOOLWORTHS  4437
##  20:      DORITO  3183
##  21:      INFZNS  3144
##  22:       SMITH  2963
##  23:     CHEETOS  2927
##  24:       SNBTS  1576
##  25:      BURGER  1564
##  26:     GRNWVES  1468
##  27:    SUNBITES  1432
##  28:         NCC  1419
##  29:      FRENCH  1418
##            BRAND     N
```

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

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

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

```
##            BRAND     N
##   1:      BURGER  1564
##   2:         CCS  4551
##   3:     CHEETOS  2927
##   4:    CHEEZELS  4603
##   5:        COBS  9693
##   6:     DORITOS 28145
##   7:      FRENCH  1418
##   8:     GRNWVES  7740
##   9:   INFUZIONS 14201
##  10:      KETTLE 41288
##  11:     NATURAL  7469
##  12:         OLD  9324
```

```
## 13:    PRINGLES 25102
## 14:         RRD 17779
## 15:      SMITHS 31823
## 16:    SUNBITES  3008
## 17:       THINS 14075
## 18:    TOSTITOS  9471
## 19:    TWISTIES  9454
## 20:    TYRRELLS  6442
## 21: WOOLWORTHS 14757
##           BRAND     N
```

Examining customer data

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

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

```
summary(customerData)
```

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

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

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

```
##                 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
```

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

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

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

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

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.

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

```
data[is.null(LIFESTAGE), .N]
```

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

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

```
## [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, paste0(filePath,"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:

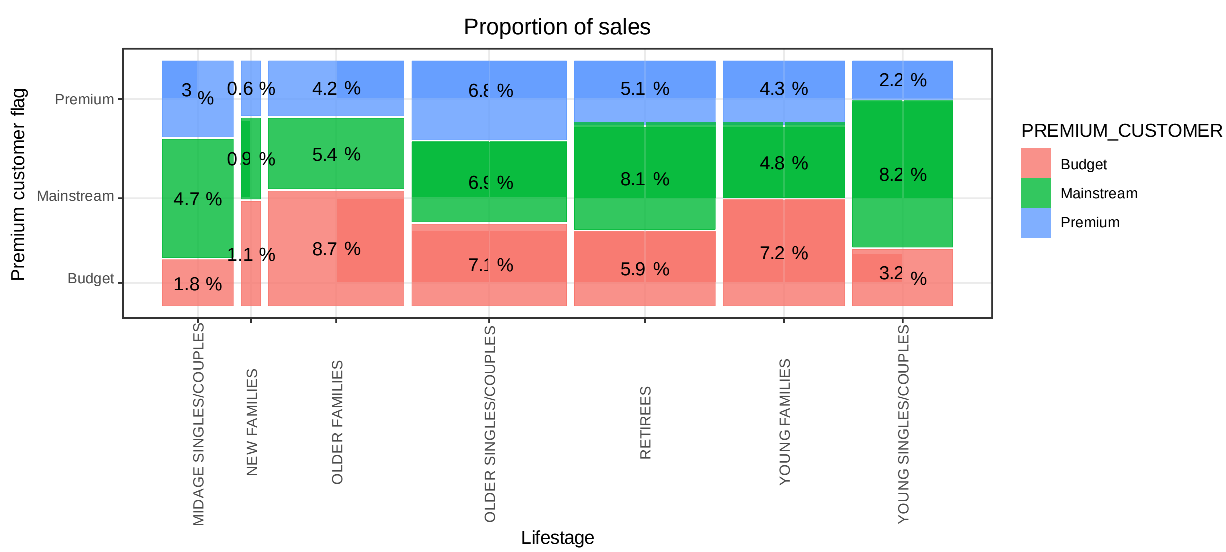
- Who spends the most on chips (total sales), describing customers by lifestage and how premium their general purchasing behaviour is
- How many customers are in each segment
- How many chips are bought per customer by segment
- What's the average chip price by customer segment

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

sales <- data[, .(SALES = sum(TOT_SALES)), .(LIFESTAGE, PREMIUM_CUSTOMER)]

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

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

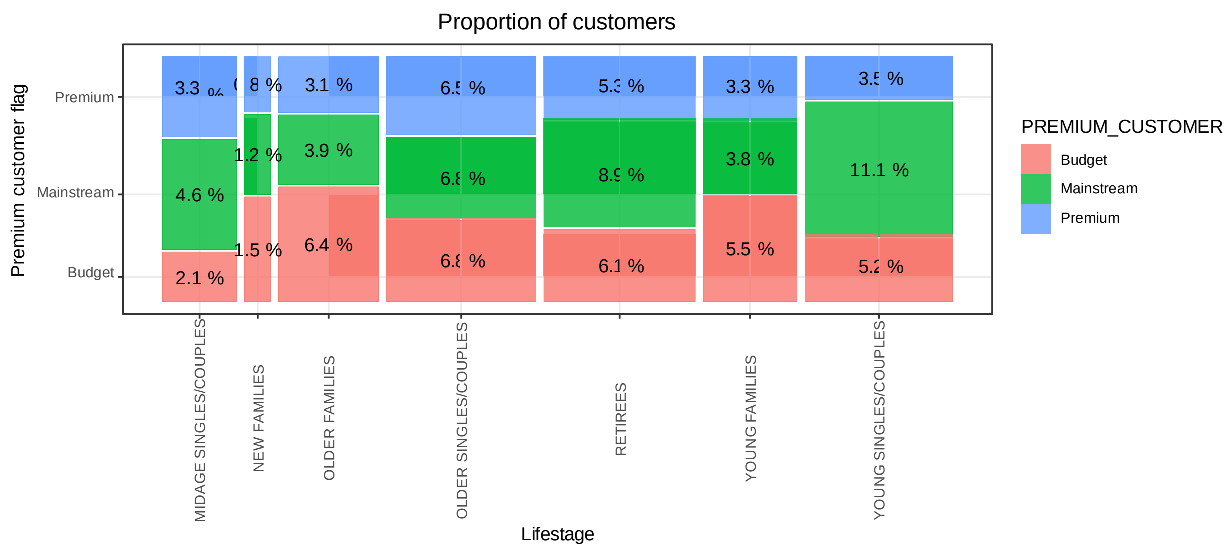


Sales are coming mainly from Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees

Let's see if the higher sales are due to there being more customers who buy chips.

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

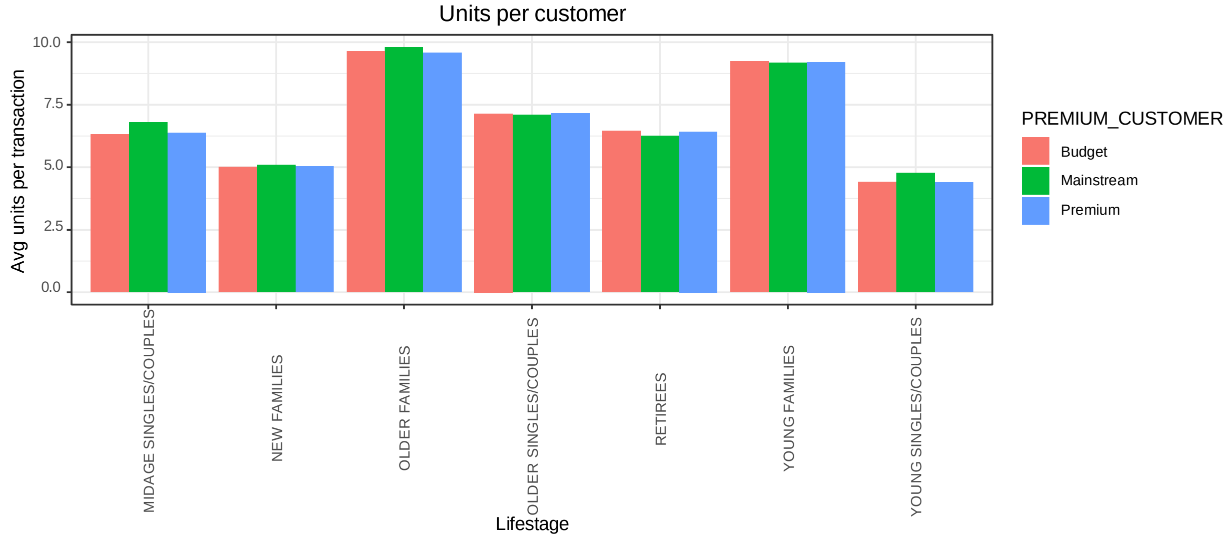
```r
#### Create plot
p <- ggplot(data = customers) +
    geom_mosaic(aes(weight = CUSTOMERS, x = product(PREMIUM_CUSTOMER,
  ↪   LIFESTAGE), fill = PREMIUM_CUSTOMER)) +
    labs(x = "Lifestage", y = "Premium customer flag", title = "Proportion of
  ↪   customers") +
    theme(axis.text.x = element_text(angle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

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



There are more Mainstream - young singles/couples and Mainstream - retirees who buy chips. This contributes to there being more sales to these customer segments but this is not a major driver for the Budget - Older families segment.

Higher sales may also be driven by more units of chips being bought per customer. Let's have a look at this next.

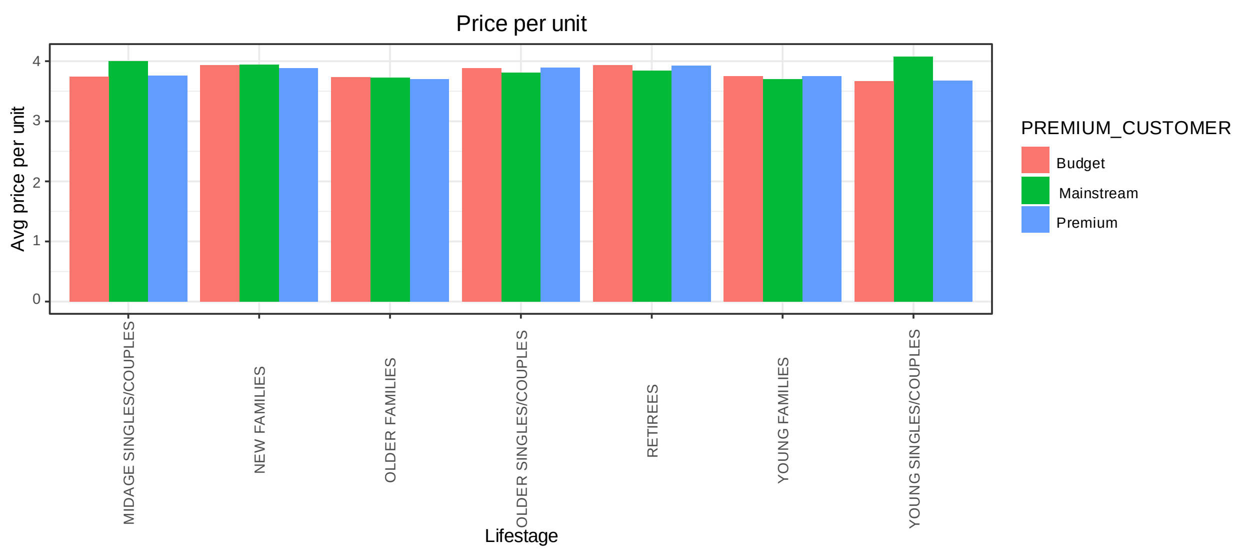
```r
#### Average number of units per customer by LIFESTAGE and PREMIUM_CUSTOMER
avg_units <- data[, .(AVG = sum(PROD_QTY)/uniqueN(LYLTY_CARD_NBR)),
  ↪   .(LIFESTAGE, PREMIUM_CUSTOMER)][order(-AVG)]

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



Older families and young families in general buy more chips per customer

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

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

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



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.

```
#### Perform an independent t-test between mainstream vs premium and budget
 ↪  midage and
#### young singles and couples
pricePerUnit <- data[, price := TOT_SALES/PROD_QTY]
t.test(data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES")
 ↪  & PREMIUM_CUSTOMER == "Mainstream", price]
      , data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES")
 ↪  & PREMIUM_CUSTOMER != "Mainstream", price]
      , alternative = "greater")
```

```
##
## Welch Two Sample t-test
##
## data: data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE
SINGLES/COUPLES") & and data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE
SINGLES/COUPLES") & PREMIUM_CUSTOMER == "Mainstream", price] and
PREMIUM_CUSTOMER != "Mainstream", price]
## t = 40.61, df = 58792, p-value < 2.2e-16
## 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 < 2.2e-16, i.e. the unit price for mainstream, young and mid-age singles and couples are significantly higher than that of budget or premium, young and midage singles and couples.

Deep dive into specific customer segments for insights

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

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

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

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

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

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

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

```
##          BRAND  targetSegment       other  affinityToBrand
##  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
##          BRAND  targetSegment       other  affinityToBrand
```

We can see that :

- Mainstream young singles/couples are 23% more likely to purchase Tyrrells chips compared to the rest of the population
- Mainstream young singles/couples are 56% less likely to purchase Burger Rings compared to the rest of the population

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

```
#### Preferred pack size compared to the rest of the population
quantity_segment1_by_pack <- segment1[, .(targetSegment =
 ↪  sum(PROD_QTY)/quantity_segment1), by = PACK_SIZE]
quantity_other_by_pack <- other[, .(other = sum(PROD_QTY)/quantity_other), by =
 ↪  PACK_SIZE]

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

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

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

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

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

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

Conclusion

Let's recap what we've found!

Sales have mainly been due to Budget - older families, Mainstream - young singles/couples, and Mainstream - retirees shoppers. We found that the high spend in chips for mainstream young singles/couples and retirees is due to there being more of them than other buyers. 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. 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.

Quantium can help the Category Manager with recommendations of where these segments are and further help them with measuring the impact of the changed placement. We'll work on measuring the impact of trials in the next task and putting all these together in the third task.