Retail Strategy and Analytics

Gavan Corke

```
library(readxl)
library(data.table)
library(ggplot2)
library(ggmosaic)
library(readr)
library(stringr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(scales)
```

```
##
## Attaching package: 'scales'

## The following object is masked from 'package:readr':
##
##     col_factor
```

Data Cleaning

Examine transaction data – look for inconsistencies, missing data across the data set, outliers, correctly identified category items, numeric data across all tables.

Examine customer data – check for similar issues in the customer data.

```
qvi_purchase <- read.csv("QVI_purchase_behaviour.csv")
head(qvi_purchase)
```

```
##   LYLTY_CARD_NBR              LIFESTAGE PREMIUM_CUSTOMER
## 1           1000  YOUNG SINGLES/COUPLES          Premium
## 2           1002  YOUNG SINGLES/COUPLES       Mainstream
## 3           1003         YOUNG FAMILIES           Budget
## 4           1004  OLDER SINGLES/COUPLES       Mainstream
## 5           1005 MIDAGE SINGLES/COUPLES       Mainstream
## 6           1007  YOUNG SINGLES/COUPLES           Budget
```

```
qvi_trans <- read_excel("QVI_transaction_data.xlsx")
head(qvi_trans)
```

```
## # A tibble: 6 x 8
## DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_~1 TOT_S~2
## <dbl> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <dbl>
## 1 43390 1 1000 1 5 Natural Chip ~ 2 6
## 2 43599 1 1307 348 66 CCs Nacho Chee~ 3 6.3
## 3 43605 1 1343 383 61 Smiths Crinkle~ 2 2.9
## 4 43329 2 2373 974 69 Smiths Chip Th~ 5 15
## 5 43330 2 2426 1038 108 Kettle Tortill~ 3 13.8
## 6 43604 4 4074 2982 57 Old El Paso Sa~ 1 5.1
## # ... with abbreviated variable names 1: PROD_QTY, 2: TOT_SALES
```

```
# Column Names

col_qvi <- colnames(qvi_purchase)
trans_col <- colnames(qvi_trans)
```

```
#==========================#
# For transaction data set
#==========================#

#1)

# check for data types

trans_col # names of columns
```

```
## [1] "DATE"           "STORE_NBR"      "LYLTY_CARD_NBR" "TXN_ID"
## [5] "PROD_NBR"       "PROD_NAME"      "PROD_QTY"       "TOT_SALES"
```

```
str(qvi_trans)
```

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

```
dt_table<-data.frame(Names=names(qvi_trans), Type=sapply(qvi_trans,class))
labels(dt_table)
```

```
## [[1]]
## [1] "DATE"            "STORE_NBR"       "LYLTY_CARD_NBR" "TXN_ID"
## [5] "PROD_NBR"        "PROD_NAME"       "PROD_QTY"        "TOT_SALES"
##
## [[2]]
## [1] "Names" "Type"
```

```
# change data DATE column into date type.

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

str(qvi_trans)
```

```
## tibble [264,836 x 8] (S3: tbl_df/tbl/data.frame)
## $ DATE : Date[1:264836], format: "2018-10-17" "2019-05-14" ...
## $ STORE_NBR : num [1:264836] 1 1 1 2 2 4 4 4 5 7 ...
## $ LYLTY_CARD_NBR: num [1:264836] 1000 1307 1343 2373 2426 ...
## $ TXN_ID : num [1:264836] 1 348 383 974 1038 ...
## $ PROD_NBR : num [1:264836] 5 66 61 69 108 57 16 24 42 52 ...
## $ PROD_NAME : chr [1:264836] "Natural Chip Compny SeaSalt175g" "CCs Nacho
Cheese 175g" "Smiths Crinkle Cut Chips Chicken 170g" "Smiths Chip Thinly
S/Cream&Onion 175g" ...
## $ PROD_QTY : num [1:264836] 2 3 2 5 3 1 1 1 1 2 ...
## $ TOT_SALES : num [1:264836] 6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
```

```
#2)

#Check for missing data across the data set

sum(is.null(qvi_trans))
```

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

```
sum(is.na(qvi_trans))
```

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

```
# no null or na data.

#3)
```

```
#Check product names are correctly entered.


unique(qvi_trans$PROD_NAME)
```

```
##   [1] "Natural Chip        Compny SeaSalt175g"
##   [2] "CCs Nacho Cheese    175g"
##   [3] "Smiths Crinkle Cut  Chips Chicken 170g"
##   [4] "Smiths Chip Thinly  S/Cream&Onion 175g"
##   [5] "Kettle Tortilla ChpsHny&Jlpno Chili 150g"
##   [6] "Old El Paso Salsa   Dip Tomato Mild 300g"
##   [7] "Smiths Crinkle Chips Salt & Vinegar 330g"
##   [8] "Grain Waves         Sweet Chilli 210g"
##   [9] "Doritos Corn Chip Mexican Jalapeno 150g"
##  [10] "Grain Waves Sour    Cream&Chives 210G"
##  [11] "Kettle Sensations   Siracha Lime 150g"
##  [12] "Twisties Cheese     270g"
##  [13] "WW Crinkle Cut      Chicken 175g"
##  [14] "Thins Chips Light&  Tangy 175g"
##  [15] "CCs Original 175g"
##  [16] "Burger Rings 220g"
##  [17] "NCC Sour Cream &    Garden Chives 175g"
##  [18] "Doritos Corn Chip Southern Chicken 150g"
##  [19] "Cheezels Cheese Box 125g"
##  [20] "Smiths Crinkle      Original 330g"
##  [21] "Infzns Crn Crnchers Tangy Gcamole 110g"
##  [22] "Kettle Sea Salt     And Vinegar 175g"
##  [23] "Smiths Chip Thinly  Cut Original 175g"
##  [24] "Kettle Original 175g"
##  [25] "Red Rock Deli Thai  Chilli&Lime 150g"
##  [26] "Pringles Sthrn FriedChicken 134g"
##  [27] "Pringles Sweet&Spcy BBQ 134g"
##  [28] "Red Rock Deli SR    Salsa & Mzzrlla 150g"
##  [29] "Thins Chips         Originl saltd 175g"
##  [30] "Red Rock Deli Sp    Salt & Truffle 150G"
##  [31] "Smiths Thinly       Swt Chli&S/Cream175G"
##  [32] "Kettle Chilli 175g"
##  [33] "Doritos Mexicana    170g"
##  [34] "Smiths Crinkle Cut  French OnionDip 150g"
##  [35] "Natural ChipCo      Hony Soy Chckn175g"
##  [36] "Dorito Corn Chp     Supreme 380g"
##  [37] "Twisties Chicken270g"
##  [38] "Smiths Thinly Cut   Roast Chicken 175g"
##  [39] "Smiths Crinkle Cut  Tomato Salsa 150g"
##  [40] "Kettle Mozzarella   Basil & Pesto 175g"
##  [41] "Infuzions Thai SweetChili PotatoMix 110g"
##  [42] "Kettle Sensations   Camembert & Fig 150g"
##  [43] "Smith Crinkle Cut   Mac N Cheese 150g"
##  [44] "Kettle Honey Soy    Chicken 175g"
##  [45] "Thins Chips Seasonedchicken 175g"
##  [46] "Smiths Crinkle Cut  Salt & Vinegar 170g"
##  [47] "Infuzions BBQ Rib   Prawn Crackers 110g"
##  [48] "GrnWves Plus Btroot & Chilli Jam 180g"
##  [49] "Tyrrells Crisps     Lightly Salted 165g"
##  [50] "Kettle Sweet Chilli And Sour Cream 175g"
##  [51] "Doritos Salsa       Medium 300g"
##  [52] "Kettle 135g Swt Pot Sea Salt"
##  [53] "Pringles SourCream  Onion 134g"
##  [54] "Doritos Corn Chips  Original 170g"
##  [55] "Twisties Cheese     Burger 250g"
##  [56] "Old El Paso Salsa   Dip Chnky Tom Ht300g"
##  [57] "Cobs Popd Swt/Chlli &Sr/Cream Chips 110g"
##  [58] "Woolworths Mild     Salsa 300g"
##  [59] "Natural Chip Co     Tmato Hrb&Spce 175g"
##  [60] "Smiths Crinkle Cut  Chips Original 170g"
##  [61] "Cobs Popd Sea Salt  Chips 110g"
##  [62] "Smiths Crinkle Cut  Chips Chs&Onion170g"
##  [63] "French Fries Potato Chips 175g"
##  [64] "Old El Paso Salsa   Dip Tomato Med 300g"
##  [65] "Doritos Corn Chips  Cheese Supreme 170g"
##  [66] "Pringles Original   Crisps 134g"
##  [67] "RRD Chilli&         Coconut 150g"
##  [68] "WW Original Corn    Chips 200g"
##  [69] "Thins Potato Chips  Hot & Spicy 175g"
##  [70] "Cobs Popd Sour Crm  &Chives Chips 110g"
##  [71] "Smiths Crnkle Chip  Orgnl Big Bag 380g"
##  [72] "Doritos Corn Chips  Nacho Cheese 170g"
##  [73] "Kettle Sensations   BBQ&Maple 150g"
##  [74] "WW D/Style Chip     Sea Salt 200g"
##  [75] "Pringles Chicken    Salt Crips 134g"
##  [76] "WW Original Stacked Chips 160g"
##  [77] "Smiths Chip Thinly  CutSalt/Vinegr175g"
##  [78] "Cheezels Cheese 330g"
##  [79] "Tostitos Lightly    Salted 175g"
##  [80] "Thins Chips Salt &  Vinegar 175g"
##  [81] "Smiths Crinkle Cut  Chips Barbecue 170g"
##  [82] "Cheetos Puffs 165g"
##  [83] "RRD Sweet Chilli &  Sour Cream 165g"
##  [84] "WW Crinkle Cut      Original 175g"
##  [85] "Tostitos Splash Of  Lime 175g"
##  [86] "Woolworths Medium   Salsa 300g"
##  [87] "Kettle Tortilla ChpsBtroot&Ricotta 150g"
##  [88] "CCs Tasty Cheese    175g"
##  [89] "Woolworths Cheese   Rings 190g"
##  [90] "Tostitos Smoked     Chipotle 175g"
##  [91] "Pringles Barbeque   134g"
##  [92] "WW Supreme Cheese   Corn Chips 200g"
##  [93] "Pringles Mystery    Flavour 134g"
##  [94] "Tyrrells Crisps     Ched & Chives 165g"
##  [95] "Snbts Whlgrn Crisps Cheddr&Mstrd 90g"
##  [96] "Cheetos Chs & Bacon Balls 190g"
##  [97] "Pringles Slt Vingar 134g"
##  [98] "Infuzions SourCream&Herbs Veg Strws 110g"
##  [99] "Kettle Tortilla ChpsFeta&Garlic 150g"
## [100] "Infuzions Mango     Chutny Papadums 70g"
## [101] "RRD Steak &         Chimuchurri 150g"
## [102] "RRD Honey Soy       Chicken 165g"
## [103] "Sunbites Whlegrn    Crisps Frch/Onin 90g"
## [104] "RRD Salt & Vinegar  165g"
## [105] "Doritos Cheese      Supreme 330g"
## [106] "Smiths Crinkle Cut  Snag&Sauce 150g"
## [107] "WW Sour Cream &OnionStacked Chips 160g"
## [108] "RRD Lime & Pepper   165g"
## [109] "Natural ChipCo Sea  Salt & Vinegr 175g"
## [110] "Red Rock Deli Chikn&Garlic Aioli 150g"
## [111] "RRD SR Slow Rst     Pork Belly 150g"
## [112] "RRD Pc Sea Salt     165g"
## [113] "Smith Crinkle Cut   Bolognese 150g"
## [114] "Doritos Salsa Mild  300g"
```

```
summary(qvi_trans$PROD_NAME)
```

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

```
#Make sure we are looking at chip products. Remove any incorrect
#products and clean the PROD NAME e.g. characters etc.\

product_unique <- unique(qvi_trans[ , "PROD_NAME"])
split_words_by <- strsplit(as.character(product_unique), " ")

productWords <- data.table(unlist(split_words_by))


setnames(productWords, 'words')

# Remove special characters from words

x <- "a1~!@#$%^&*(){}_+:\"<>?,./;'[]-="


productWords_clean <- str_replace_all(productWords, "[^[:alnum:]]", " ")
```

```
## Warning in stri_replace_all_regex(string, pattern,
## fix_replacement(replacement), : argument is not an atomic vector; coercing
```

```
productWords_num2 <- trimws(gsub("\\w*[0-9]+\\w*\\s*", "", productWords_clean))


#productWords_final <- trimws(gsub("[^\\s]*[0-9][^\\s]*", "", productWords_clean, perl=T))

#productWords_final <- strsplit(as.character(productWords_final), " ")

# use subset to remove empty values ""

productWords_final <- subset(productWords_num2[1], productWords_num2[1]!=" ")
```

Now we can make the frequency table of the words and remove the SALSA product which is not a chip product.

```r
# Frequency table
productword_freq <- strsplit(as.character(productWords_final), " ")
productword_freqtab <- table(productword_freq)

# Remove Salsa

qvi_superclean <- qvi_trans[!grepl("salsa", qvi_trans$PROD_NAME, ignore.case = TRUE), ]
```

OUTLIERS

In order to make sure our analysis of the products can tell us something about the general population, it is important that we investigate and deal with potential outliers as this could affect our generalization and analysis.

From a quick glance it can be seen that there is an outlier with the PROD_QTY variable. The quantity purchased from one customer totals 200 which is much higher than the average. This also affects the total sales for this transaction at being over 600. We can deal with this outlier singularly or develop a more systemic method to identify potential outliers. For example, one can generate boxplots and define the outliers to be any datapoint + or -1.5x IQR.

Another method is to determine the relevant z-scores for each datapoint for the particular variable in question and determine if they are greater than 3 or less than -3. This will tell us that the data is 3 standard deviations below or above the mean. Both methods are outlined in this analysis however given the nature and scope of the project the more intuitive method (by glancing the 200 PROD_QTY datapoint) will be the method used here.

Boxplot Method

```r
summary(qvi_superclean)

# interested in product quantity and total sales for outliers.

qty_and_sales <- summary(qvi_superclean[c("PROD_QTY","TOT_SALES")])


# Look at boxplots for detecting outliers.
# ======================================#

boxplot(qvi_superclean$PROD_QTY, horizontal = TRUE)
boxplot(qvi_superclean$TOT_SALES, horizontal = TRUE)

boxplot.stats(qvi_superclean$PROD_QTY)$out

hist(qvi_superclean$PROD_QTY)
hist(qvi_superclean$TOT_SALES, breaks = 150)

#Thanks to the which() function it is possible to extract the row number corresponding to these outlier
#e.g.:

out <- boxplot.stats(dat$hwy)$out
out_ind <- which(dat$hwy %in% c(out))
out_ind
```

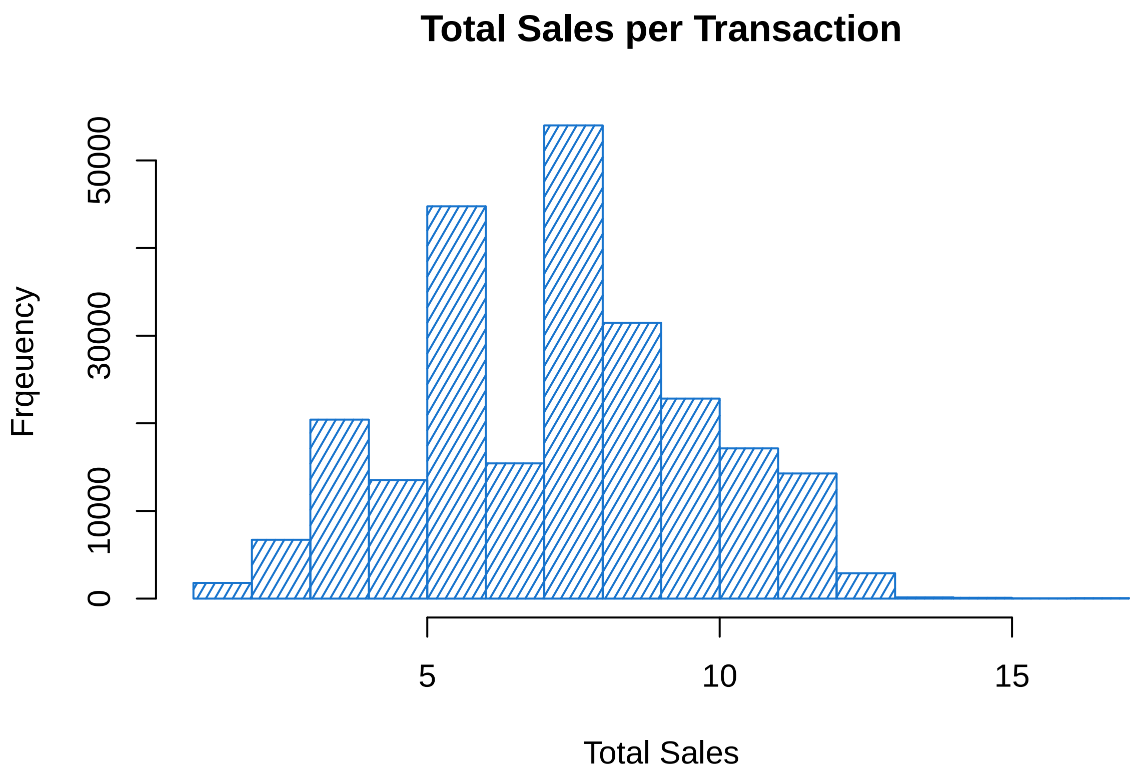
```
#outliers index
# boxplot()$out shows the datapoints which are >1.5x IQR

outliers_pq <- boxplot.stats(qvi_superclean$PROD_QTY)$out
outliers_idx_pq <- which(qvi_superclean$PROD_QTY %in% c(outliers_pq))


length(qvi_superclean$PROD_QTY)

no_outliers_trans <- qvi_trans[-outliers_idx_pq, ]

#outliers sales

qvi_pq <- qvi_trans$TOT_SALES

outliers_sales <- boxplot.stats(qvi_pq)$out
outliers_idx_sal <- which(qvi_pq %in% c(outliers_idx_pq))


no_outliers_qvi <- no_outliers_trans[-outliers_idx_sal, ]

clean_qvi <- no_outliers_qvi

#IQR takes too many datapoints away
```

However for this dataset it seems to take away too many data points and thus will not be used further in this analysis.

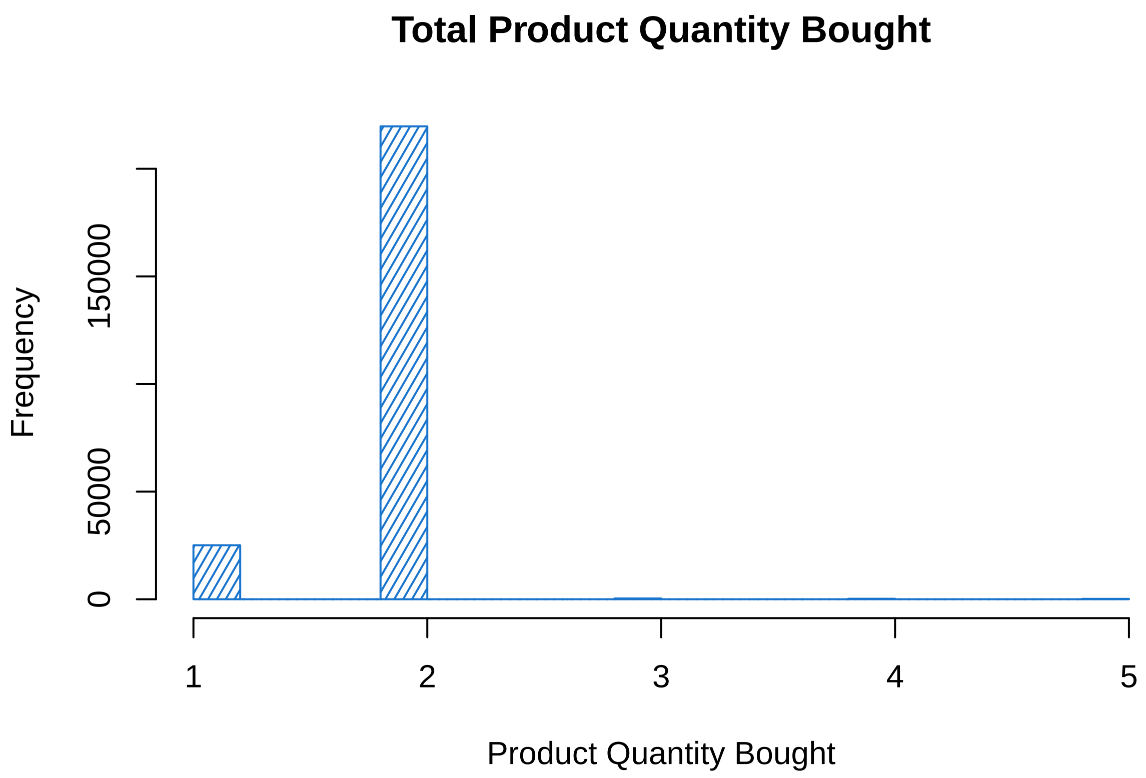
Z score method.

```
#===================#

# Z score approach

# ==================#
# use z scores to get rid of outliers for PROD_QTY

z_score = (qvi_superclean$PROD_QTY - mean(qvi_superclean$PROD_QTY))/
  sd(qvi_superclean$PROD_QTY)
outliers_idx_pq <- which(!(-3 < z_score & z_score < 3))

#use z scores to get rid of outliers for TOT_SALES

z_score2 <- (qvi_superclean$TOT_SALES - mean(qvi_superclean$TOT_SALES))/
  sd(qvi_superclean$TOT_SALES)
outliers_idx_sales <- which(!(-3 < z_score2 & z_score2 < 3))


clean_qvi_trans <- qvi_superclean[-outliers_idx_sales, ]
clean_qvi_trans <- clean_qvi_trans[-outliers_idx_pq, ]
clean_qvi_trans <- clean_qvi_trans
```

```
hist(clean_qvi_trans$TOT_SALES, xlab = "Total Sales", ylab = "Frqeuency",
     main = "Total Sales per Transaction", col="dodgerblue3", density=25,
     angle=60)
```



```
hist(clean_qvi_trans$PROD_QTY, xlab = "Product Quantity Bought",
     ylab = "Frequency", main = "Total Product Quantity Bought", col="dodgerblue3",
     density=25,
     angle=60)
```



Looking at the singular outlier.

```
# CHECK the 200 OUTLIER

qvi_superclean[qvi_superclean$PROD_QTY == 200, ]
```

```
## # A tibble: 2 x 8
## DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_~1 TOT_S~2
## <date> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <dbl>
## 1 2018-08-19 226 226000 226201 4 Dorito Co~ 200 650
## 2 2019-05-20 226 226000 226210 4 Dorito Co~ 200 650
## # ... with abbreviated variable names 1: PROD_QTY, 2: TOT_SALES
```

```
unique(qvi_superclean[qvi_superclean$PROD_QTY == 200, ]$LYLTY_CARD_NBR)
```

```
## [1] 226000
```

```
# 226000 Loyalty Card Number
# other transactions he made

qvi_superclean[qvi_superclean$LYLTY_CARD_NBR == 226000, ]
```

```
## # A tibble: 2 x 8
## DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR PROD_NAME PROD_~1 TOT_S~2
## <date> <dbl> <dbl> <dbl> <dbl> <chr> <dbl> <dbl>
## 1 2018-08-19 226 226000 226201 4 Dorito Co~ 200 650
## 2 2019-05-20 226 226000 226210 4 Dorito Co~ 200 650
## # ... with abbreviated variable names 1: PROD_QTY, 2: TOT_SALES
```

```
# Remove from further analysis

qvi_superclean2 <- qvi_superclean[qvi_superclean$LYLTY_CARD_NBR != 226000, ]

# Number of transactions by date

length(unique(qvi_superclean2$DATE))
```

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

```
table(qvi_superclean2$DATE)
```

```
#create a sequence of dates and join with the counts of transactions per date

seq_1 <- seq(from = as.Date("2018-7-1"), to = as.Date("2019-06-30"),
    by = "days")

date_trans <- as.data.frame(seq_1)

date_wit_trans <- as.data.frame(t(table(qvi_superclean2$DATE)))[ , c(2, 3)]

date_wit_trans$Var2 <- as.Date(date_wit_trans$Var2)

freq_table <- merge(date_trans, date_wit_trans, by.x = "seq_1", by.y = "Var2", all.x = TRUE)


which(is.na(freq_table$Freq))# 178
```

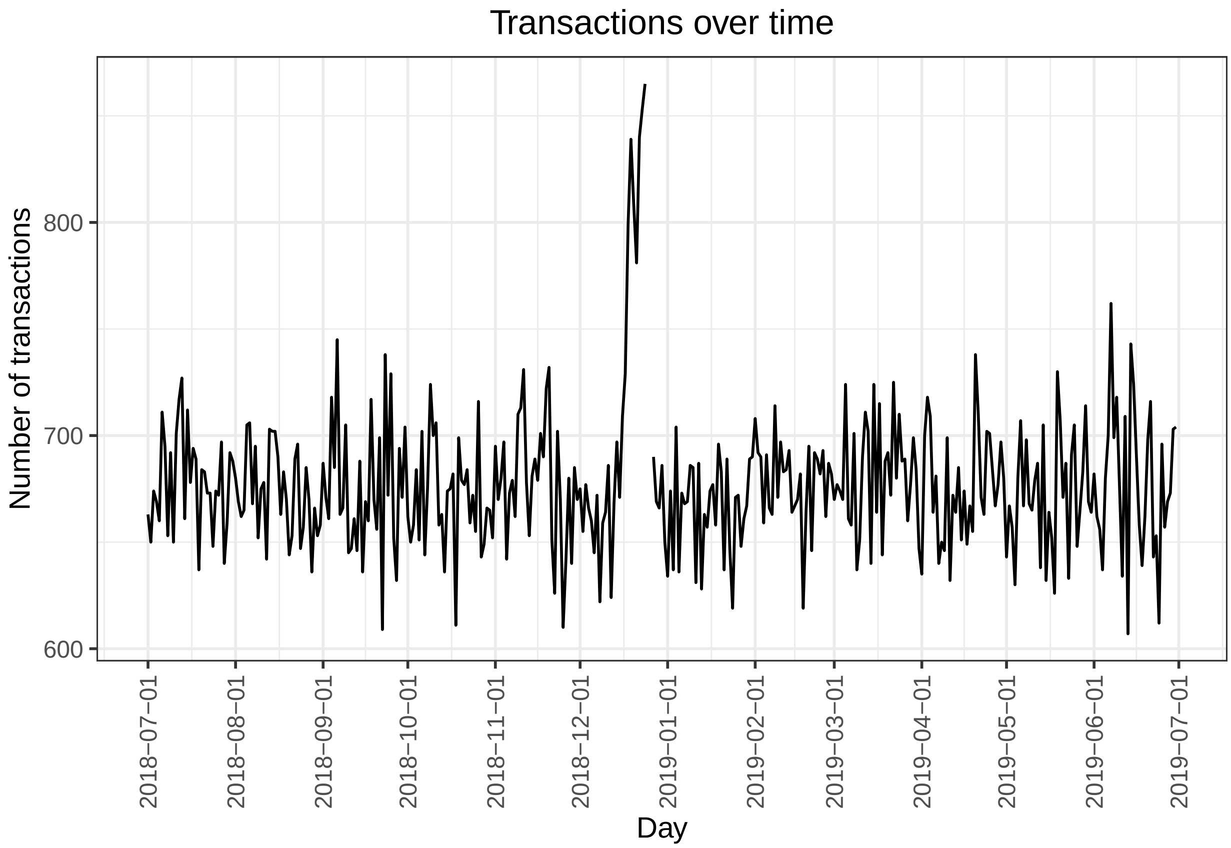
```
## [1] 178
```

```
freq_table2 <- freq_table[-178, ]
freq_table2[178, ] <- freq_table[178, ]


freq_table2$Freq <- as.numeric(freq_table2$Freq)
freq_table2$seq_1 <- as.Date(freq_table2$seq_1)
```

Generating relevant graphs.

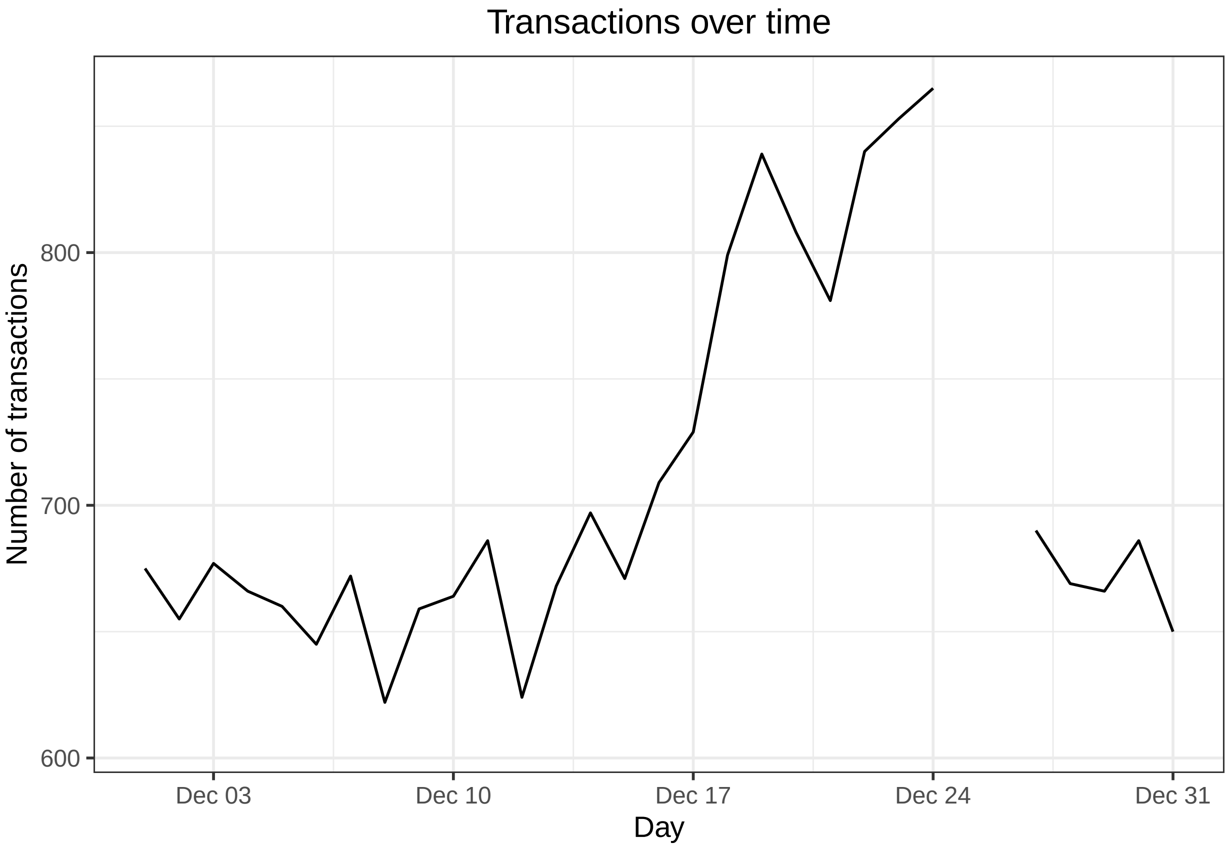
```
#Graph counts
#### Setting plot themes to format graphs
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))
```

```
#### Plot transactions over time
ggplot(freq_table2, aes(x = seq_1, y = Freq)) +
  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))
```



```
##Zoom into december

ggplot(freq_table2, aes(x = seq_1, y = Freq)) +
  geom_line() +
  labs(x = "Day", y = "Number of transactions",
       title = "Transactions over time") +
  scale_x_date(limits = c(as.Date("2018-12-01"), as.Date("2018-12-31")))
```

```
## Warning: Removed 334 rows containing missing values (‘geom_line()‘).
```



From the graphs above we can see there is an uptick of sales all the way until the 25th of december. Here we can see that no transactions occur. This is the 25th of December, which is christmas day - a public holiday.

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

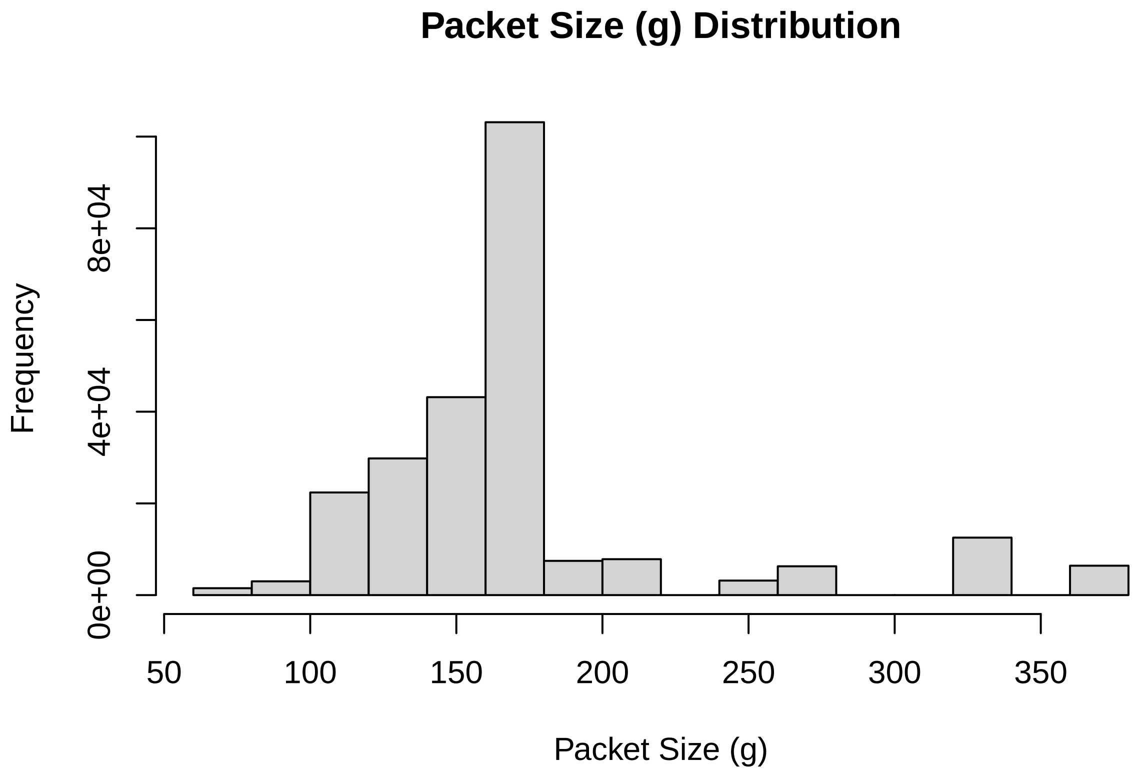
```
#### Pack size
#### We can work this out by taking the digits that are in PROD_NAME

qvi_superclean2$PACK_SIZE <- parse_number(qvi_superclean2$PROD_NAME)

#See the sizes in order

qvi_superclean2[order(qvi_superclean2$PACK_SIZE), "PACK_SIZE"]
```

```
## # A tibble: 246,740 x 1
##    PACK_SIZE
##        <dbl>
##  1        70
##  2        70
##  3        70
##  4        70
##  5         70
##  6         70
##  7         70
##  8         70
##  9         70
## 10         70
## # ... with 246,730 more rows
```

```
summary(qvi_superclean2[order(qvi_superclean2$PACK_SIZE), "PACK_SIZE"])
```

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

```
#### 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(qvi_superclean2$PACK_SIZE, xlab = "Packet Size (g)", ylab = "Frequency",
     main = "Packet Size (g) Distribution")
```



```r
# FIRST WORD OF EACH Product.
first_word <- strsplit(qvi_superclean2$PROD_NAME, " ")

brand <- list()

#for( i in 1:length(first_word)){
# brand <- c(brand, first_word[[i]][1])
# }

#OR

first_words <- sapply(first_word, function(x) strsplit(x, " ")[[1]][1])

qvi_superclean2$brands <- first_words

# make sure the same products are referred to by the same string

qvi_superclean3 <- qvi_superclean2


qvi_superclean3$brands <- gsub("RRD", "Red", qvi_superclean3$brands )
qvi_superclean3$brands <-gsub("GrnWves", "Grain", qvi_superclean3$brands )
qvi_superclean3$brands <-gsub("Snbts", "Sunbites", qvi_superclean3$brands )
qvi_superclean3$brands <-gsub("Infzns", "Infuzions", qvi_superclean3$brands )
qvi_superclean3$brands <-gsub("Smiths", "Smith", qvi_superclean3$brands )
qvi_superclean3$brands <-gsub("WW", "Woolworths", qvi_superclean3$brands )

unique(qvi_superclean3$brands)
```

```
##  [1] "Natural"    "CCs"        "Smith"      "Kettle"     "Grain"
##  [6] "Doritos"    "Twisties"   "Woolworths" "Thins"      "Burger"
## [11] "NCC"        "Cheezels"   "Infuzions"  "Red"        "Pringles"
## [16] "Dorito"     "Tyrrells"   "Cobs"       "French"     "Tostitos"
## [21] "Cheetos"    "Sunbites"
```

From the summary of the pack sizes we find that the minimum size is 70g, the median size is 170g and the maximum size (the largest chip packet size) is 380g. In Australian supermarkets you can find 70g packet sizes and the 380g size is usually called a 'party size' as it is to be shared among a few people. Thus the sizes shown in the summary seem fine. There is no need to worry about any potential outliers or high leverage points with regards to packet size.

The Purchase Dataset

```r
#==========================#
# For purchase data set
#==========================#

head(qvi_purchase)
```

```
##   LYLTY_CARD_NBR              LIFESTAGE PREMIUM_CUSTOMER
## 1           1000   YOUNG SINGLES/COUPLES          Premium
## 2           1002   YOUNG SINGLES/COUPLES       Mainstream
## 3           1003          YOUNG FAMILIES           Budget
## 4           1004   OLDER SINGLES/COUPLES       Mainstream
## 5           1005 MIDAGE SINGLES/COUPLES       Mainstream
## 6           1007   YOUNG SINGLES/COUPLES           Budget
```

```
#1)

# check for data types

trans_col # names of columns
```

```
## [1] "DATE"           "STORE_NBR"      "LYLTY_CARD_NBR" "TXN_ID"
## [5] "PROD_NBR"       "PROD_NAME"      "PROD_QTY"       "TOT_SALES"
```

```
str(qvi_purchase)
```

```
## '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" ...
```

```
dp_table<-data.frame(Names=names(qvi_purchase), Type=sapply(qvi_purchase,class))
labels(dp_table)
```

```
## [[1]]
## [1] "LYLTY_CARD_NBR"   "LIFESTAGE"        "PREMIUM_CUSTOMER"
##
## [[2]]
## [1] "Names" "Type"
```

```
#2)
# check for null or na values

sum(is.na(qvi_purchase))
```

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

```
sum(is.null(qvi_purchase))
```

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

```
#3)

#Check product names are correctly entered.


unique(qvi_purchase$LIFESTAGE)
```

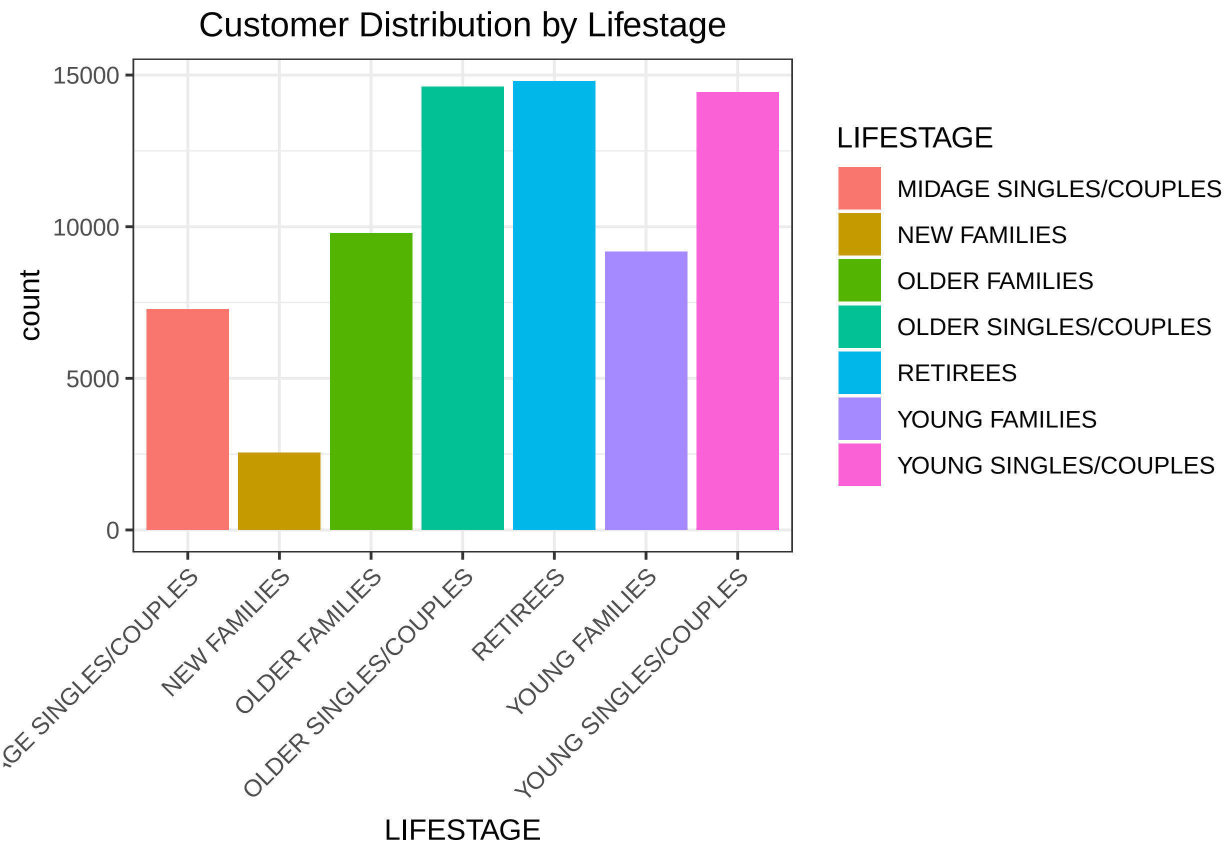
```
## [1] "YOUNG SINGLES/COUPLES" "YOUNG FAMILIES" "OLDER SINGLES/COUPLES"
## [4] "MIDAGE SINGLES/COUPLES" "NEW FAMILIES" "OLDER FAMILIES"
## [7] "RETIREES"
```

```
unique(qvi_purchase$PREMIUM_CUSTOMER)
```

```
## [1] "Premium"    "Mainstream" "Budget"
```

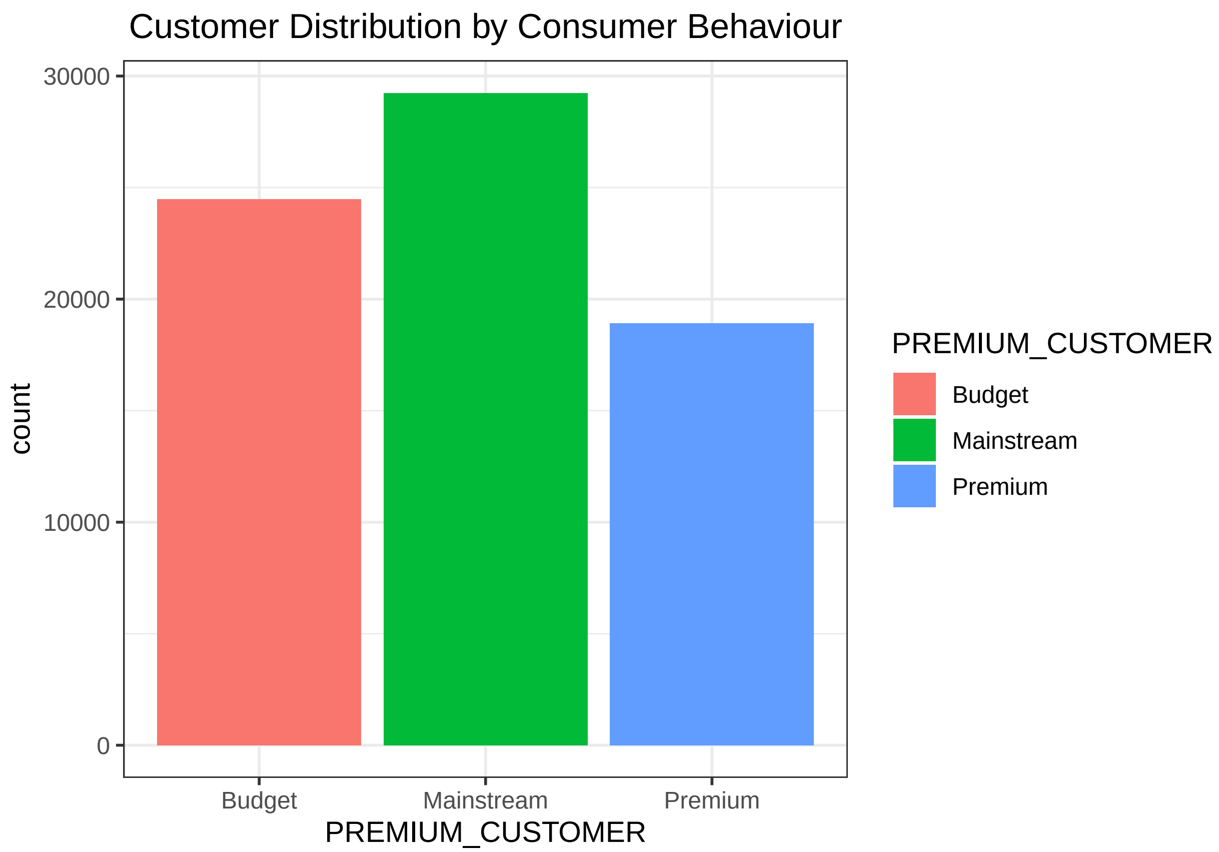
```
ggplot(qvi_purchase, aes(x = LIFESTAGE, fill = LIFESTAGE)) +
     geom_histogram(stat = "count")+
  theme(axis.text.x = element_text(angle=45, hjust=1))+
  ggtitle("Customer Distribution by Lifestage")
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## ‘binwidth‘, ‘bins‘, and ‘pad‘
```



```
ggplot(qvi_purchase, aes(x=PREMIUM_CUSTOMER, fill = PREMIUM_CUSTOMER))+
         geom_histogram(stat = "count")+
  ggtitle("Customer Distribution by Consumer Behaviour")
```

```
## Warning in geom_histogram(stat = "count"): Ignoring unknown parameters:
## ‘binwidth‘, ‘bins‘, and ‘pad‘
```



```
# merge dataframes

qvi_data <- merge(qvi_superclean3, qvi_purchase, by = 'LYLTY_CARD_NBR')
dim(qvi_data)
```

```
## [1] 246740     12
```

```
# I think these two (above) and below are the same left joins.

qvi_data2 <- merge(qvi_superclean3, qvi_purchase, all.x = TRUE)

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

is.null(qvi_data2$LYLTY_CARD_NBR) # No Nulls.
```

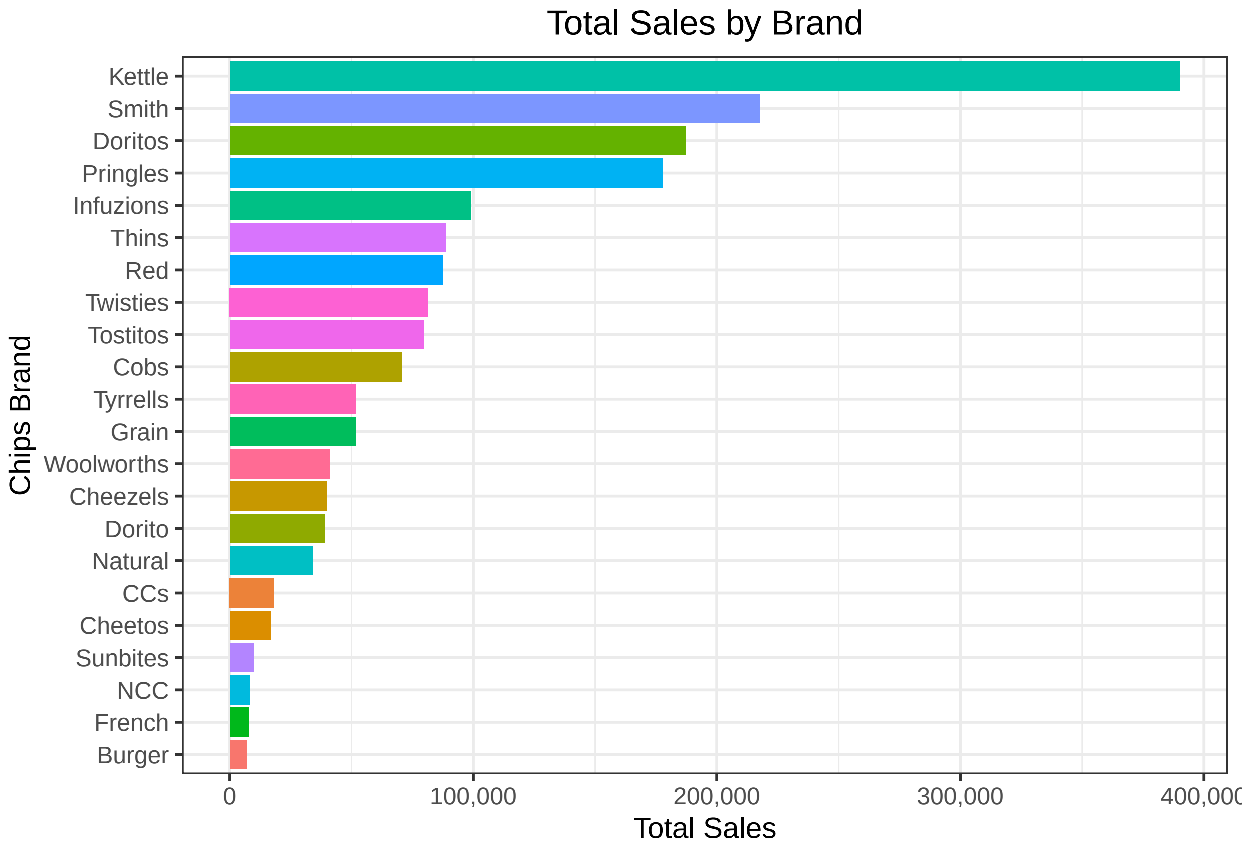
```
## [1] FALSE
```

```
fwrite(qvi_data2, "qvi_data2.csv", sep = ",", row.names = FALSE)
```

DATA ANALYSIS

In this section we can perform data analysis on the merged data set to gleam any particular insights. This can be done via filtering the data via particular sub categories and looking at variables of interest to ideally discover some underlying trends that can be help in improving product sales in the future.

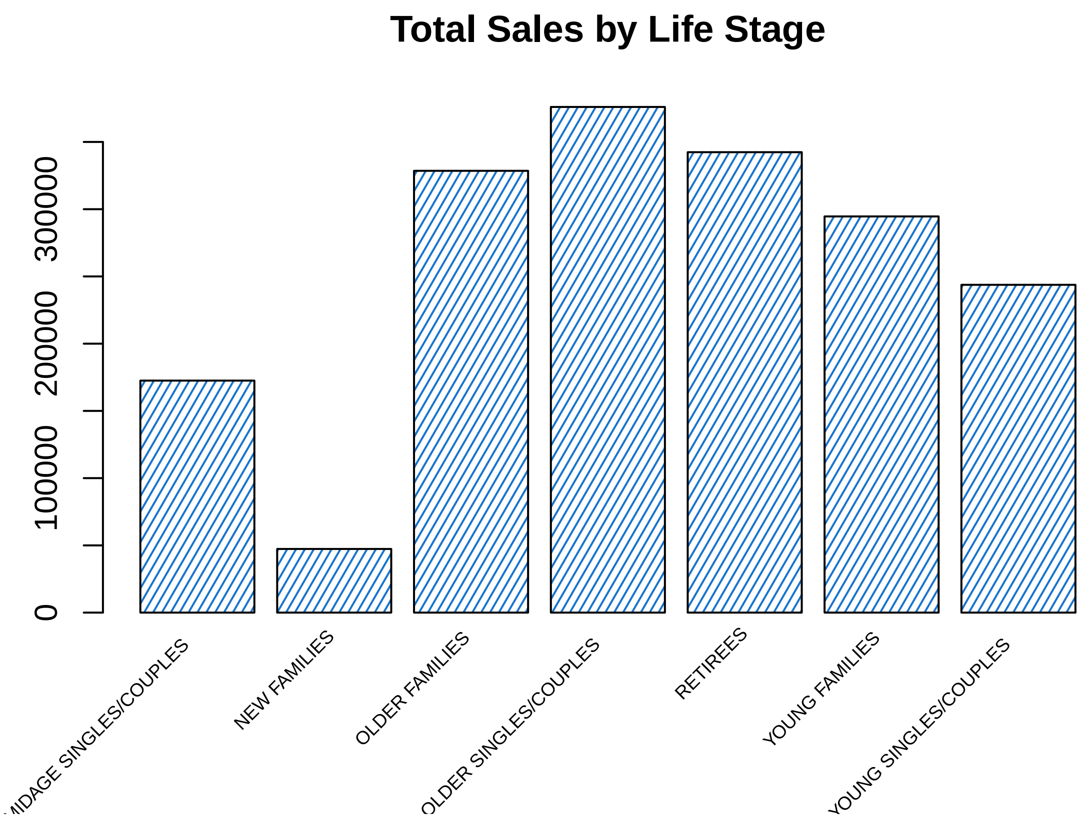
```
# ToTAL SALES BY BRAND

agg_brand_sales <- aggregate(qvi_data2$TOT_SALES, by = list(qvi_data2$brands),
                             sum)

ggplot(agg_brand_sales, aes(x = reorder(Group.1, x), y = x, fill = Group.1)) +
  geom_bar(position = 'dodge', stat = 'identity', show.legend = FALSE) +
  labs(x = "Chips Brand", y= "Total Sales", title = "Total Sales by Brand")+
  scale_y_continuous(labels = comma)+
  coord_flip()
```



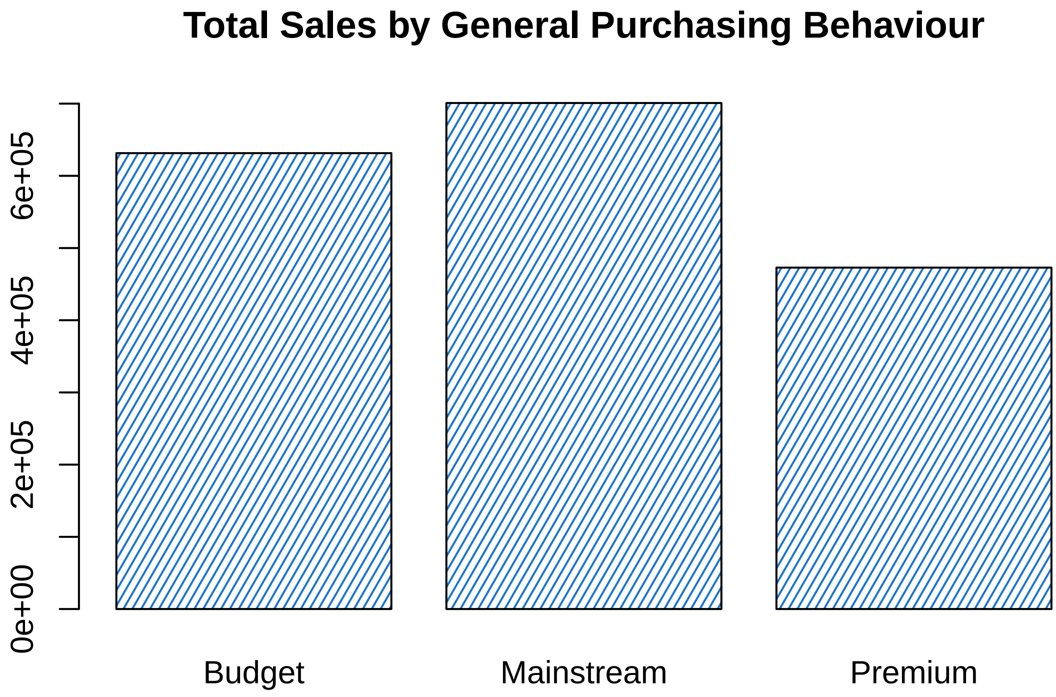
Here we can see that Kettle's chips are by far the brand with the most sales. Coming in second and third are Smith's and Dorito's respectively.

One metric we can look at is who spends the most on chips by life stage and general purchasing behaviour.

```
sales_by_life <- aggregate(qvi_data2$TOT_SALES,
                           by = list(qvi_data2$LIFESTAGE), sum)

sales_by_customer <-aggregate(qvi_data$TOT_SALES,
                              by = list(qvi_data2$PREMIUM_CUSTOMER), sum)



rotate_x <- function(data, column_to_plot, labels_vec, rot_angle, title) {
    plt <- barplot(data[[column_to_plot]], col='dodgerblue3',
                   xaxt="n", density = 25,
                   angle = 60, main = title)
    text(plt, par("usr")[3], labels = labels_vec, srt = rot_angle,
         adj = c(1.1,1.1), xpd = TRUE, cex=0.6)
}

rotate_x(sales_by_life, 'x', sales_by_life$Group.1, 45, "Total Sales by Life Stage")
```



```
barplot(sales_by_customer$x, beside = TRUE, names.arg=sales_by_customer$Group.1,
        main = "Total Sales by General Purchasing Behaviour",
        col ='dodgerblue3', angle = 60, density = 25)
```

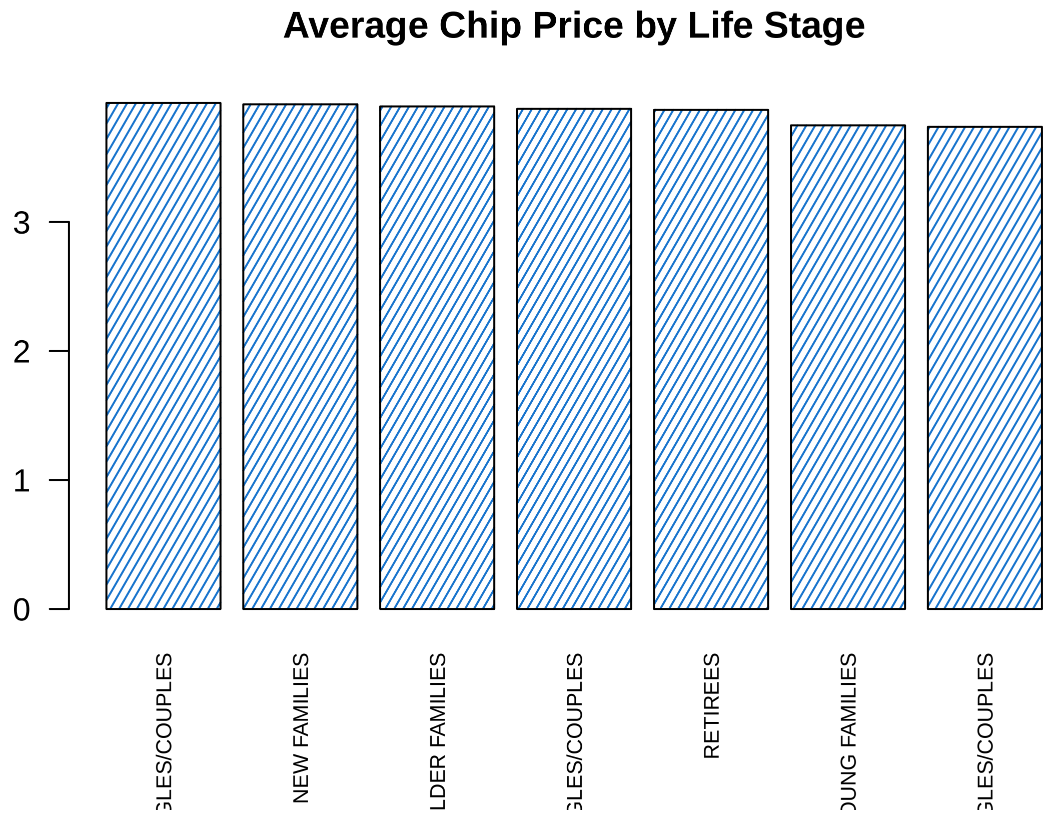


In the first graph above we can see that order singles and couples contribute to the largest amount of sales compared to other groups filtered just by life stage. In the second graph we can see that the mainstream group contribute to the highest amount of total sales compared to all other groups.

```
# Find total sales by LIFESTAGE and CUSTOMER BEHAVIOUR, then divide each by PROD_QTY for each category

total_product_qty_life <- aggregate(qvi_data2$PROD_QTY,
                                    by = list(qvi_data2$LIFESTAGE), sum)

avgprice_qty_lifestage <- sales_by_life[,2]/total_product_qty_life[,2]

avgprice_qty_lifestage
```

```
## [1] 3.877288 3.922780 3.737598 3.869112 3.896238 3.749544 3.912626
```

```
barplot(sort(avgprice_qty_lifestage, decreasing = TRUE), beside = TRUE,
        names.arg=total_product_qty_life$Group.1,
        main = "Average Chip Price by Life Stage", las = 2, col = 'dodgerblue3',
        angle = 60, density = 25, cex.names = 0.7)
```



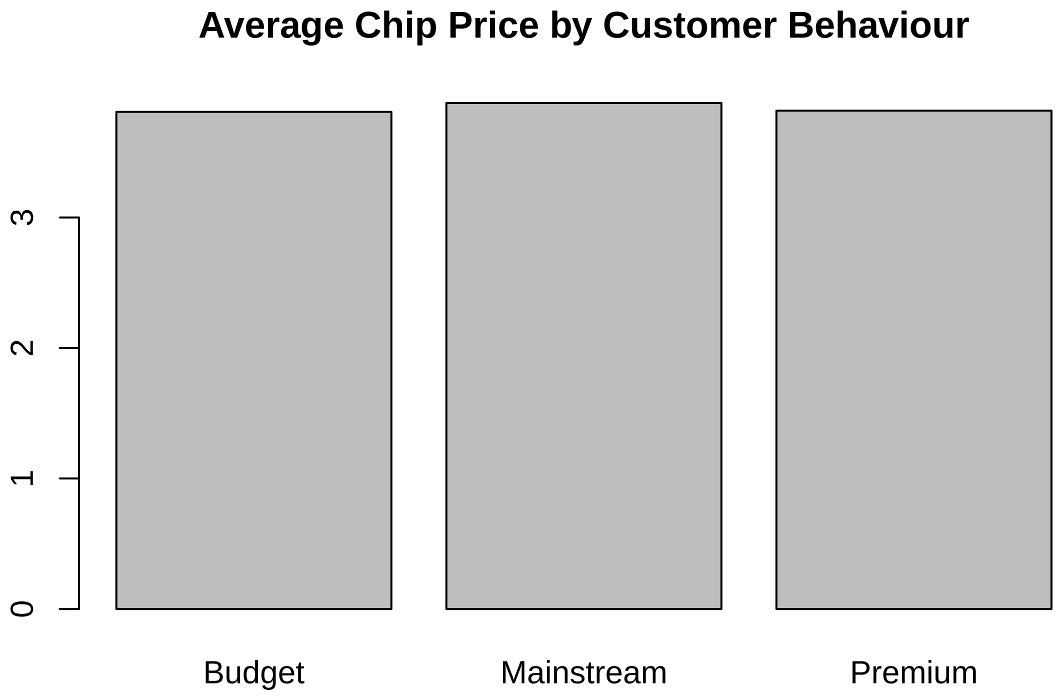
```
summary(avgprice_qty_lifestage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.738   3.809   3.877   3.852   3.904   3.923
```

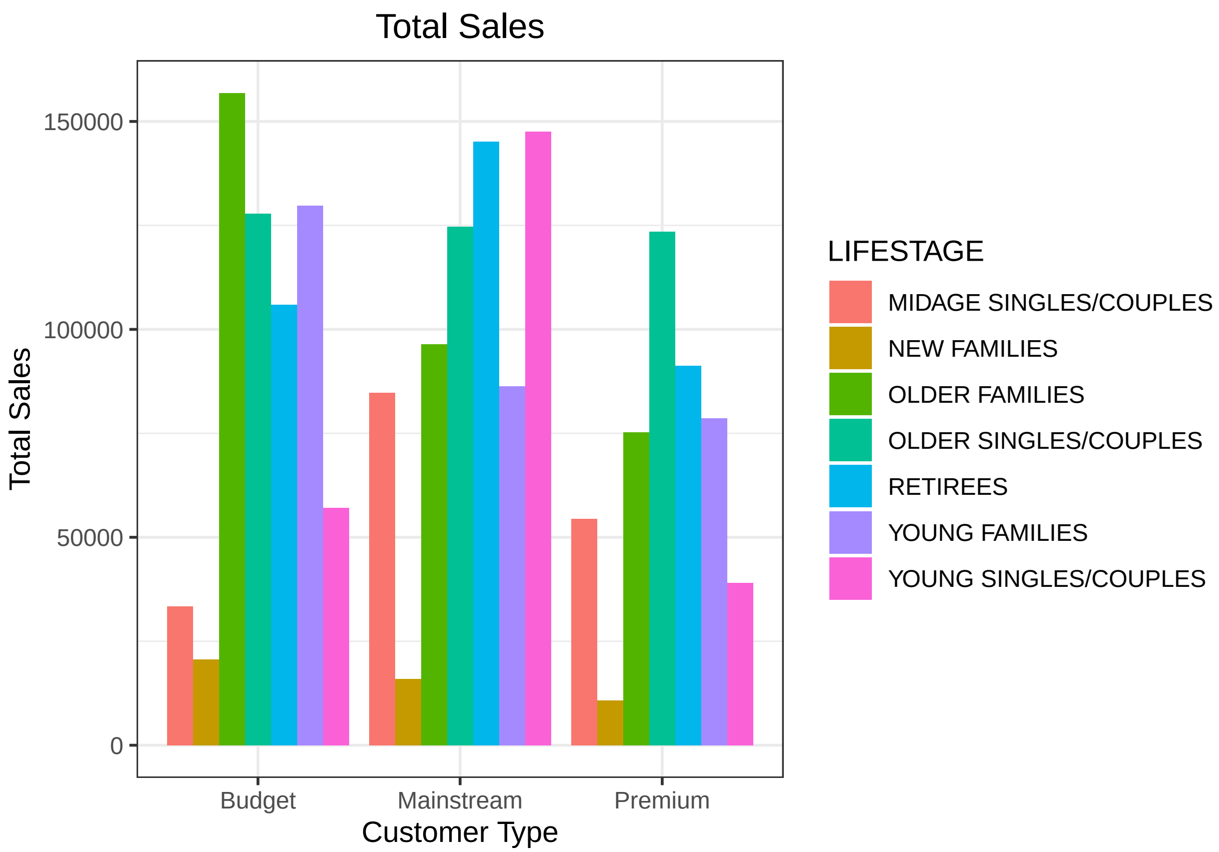
```
total_product_qty_customer_behaviour <- aggregate(qvi_data2$PROD_QTY, by = list(qvi_data2$PREMIUM_CUSTOM

avgprice_qty_customer_behaviour <- sales_by_customer[,2]/total_product_qty_customer_behaviour[,2]

avgprice_qty_customer_behaviour
```

```
## [1] 3.808841 3.876897 3.818527
```

```
barplot(avgprice_qty_customer_behaviour, beside = TRUE,
        names.arg = total_product_qty_customer_behaviour$Group.1,
        main = "Average Chip Price by Customer Behaviour")
```

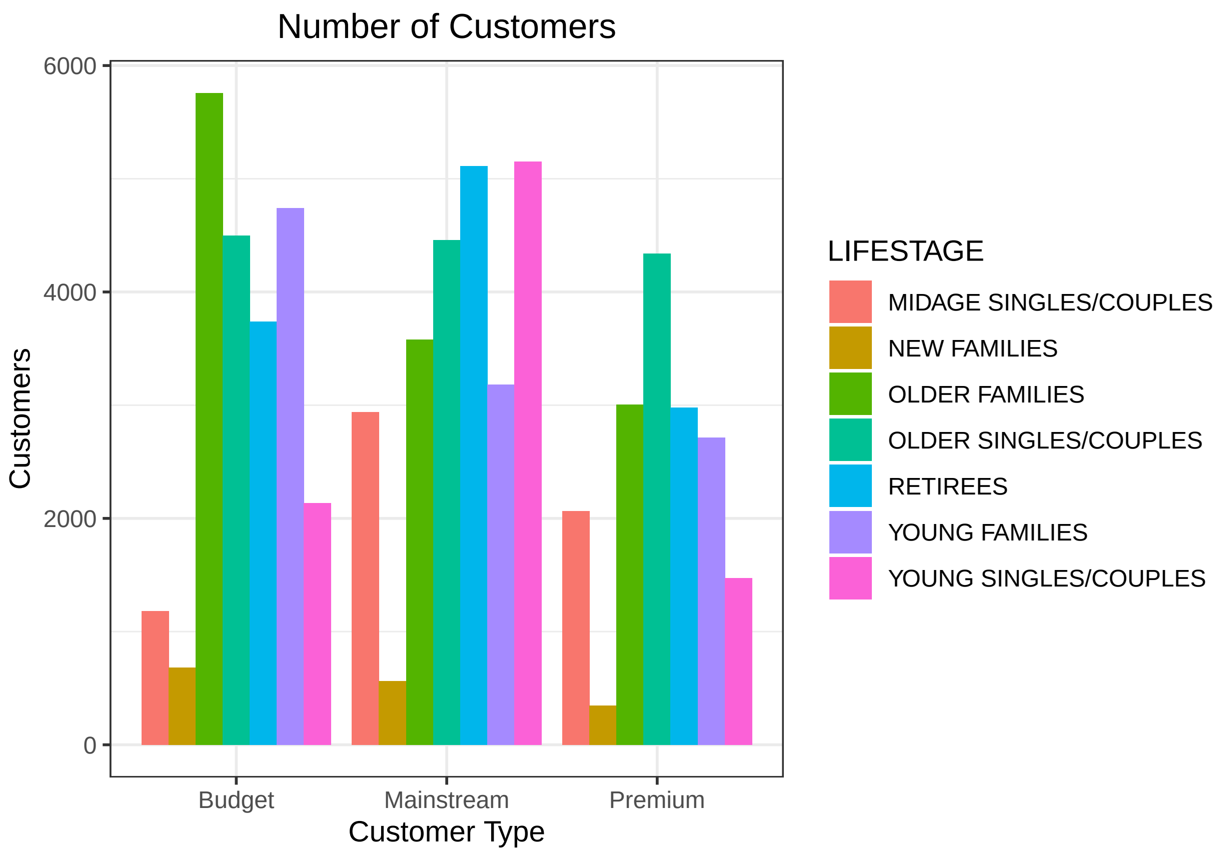


```r
#### Total sales by LIFESTAGE and PREMIUM_CUSTOMER


tot_sales_premium_life <- aggregate(TOT_SALES~LIFESTAGE+PREMIUM_CUSTOMER,
                                    qvi_data2, sum)

ggplot(tot_sales_premium_life, aes(x = PREMIUM_CUSTOMER,
                                   y = TOT_SALES,fill = LIFESTAGE)) +
  geom_bar(position = 'dodge', stat = 'identity')+
  labs (x = "Customer Type", y = "Total Sales", title = "Total Sales")
```

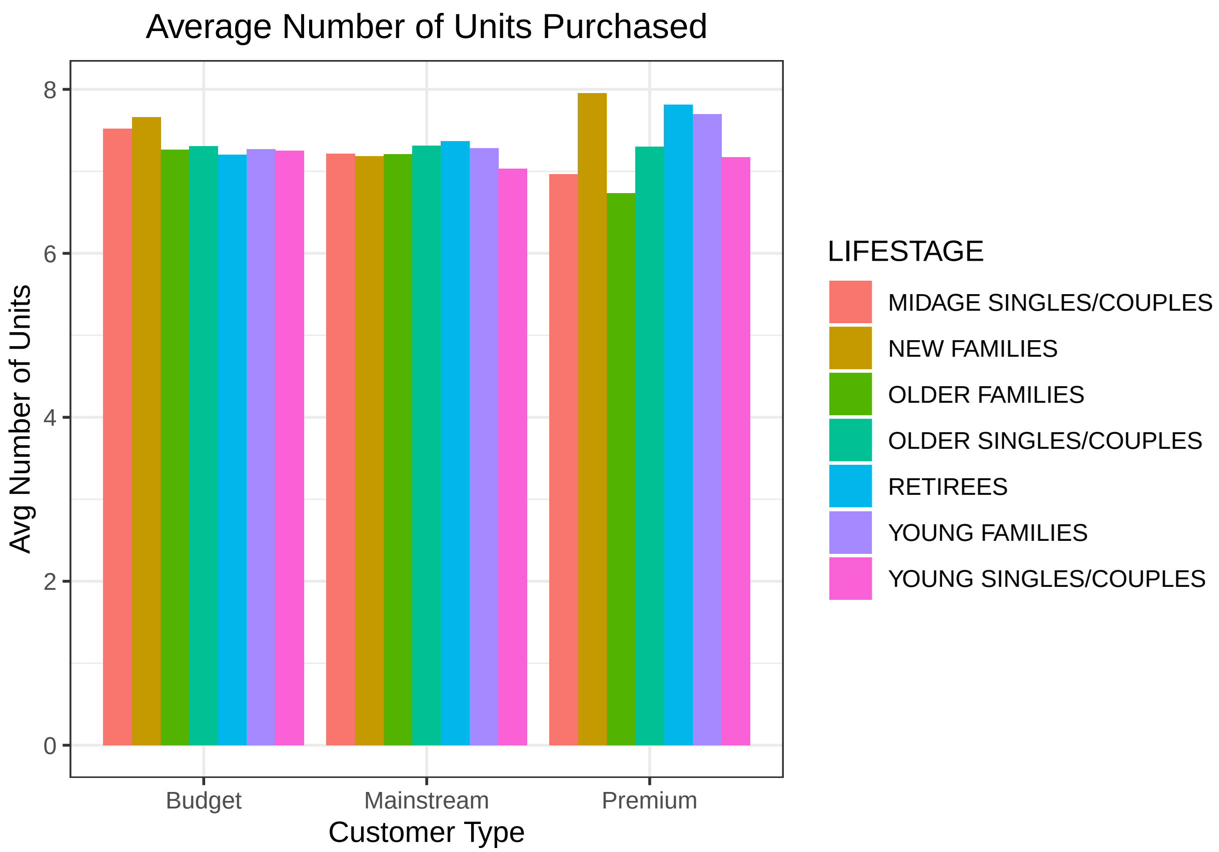


In this graph we can see that older families contribute the most to total budget sales, whilst for the mainstream, young couples and singles contribute the highest percentage of sales. Lastly, in the premium category, older singles and couples are the highest contributers to total sales.

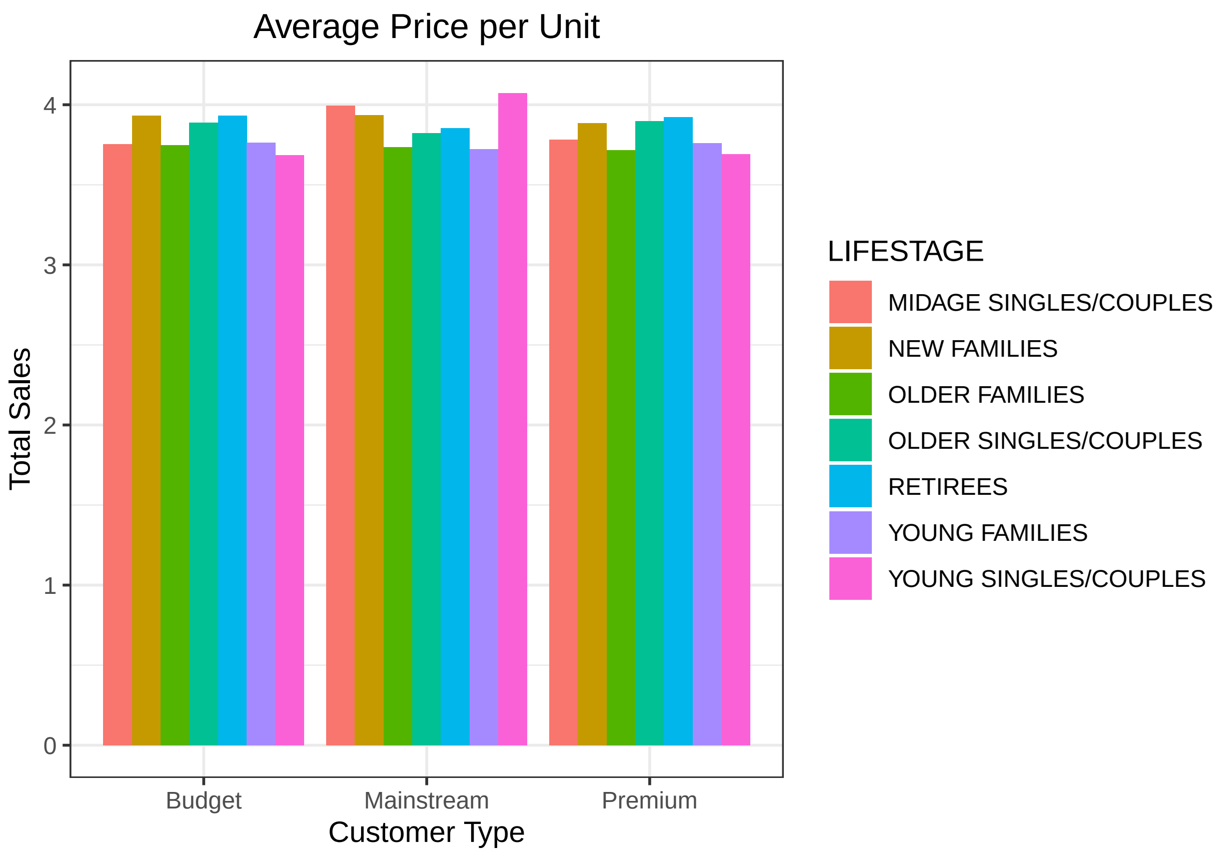
```
#Number of Customers by LIFESTAGE and PREMIUM_CUSTOMER


filtered_lylty_card <- qvi_data2[unique(qvi_data2$LYLTY_CARD_NBR), ]

number_customers_premium_life <- aggregate(LYLTY_CARD_NBR~LIFESTAGE+PREMIUM_CUSTOMER,
                                        filtered_lylty_card, length)

ggplot(number_customers_premium_life, aes(x =PREMIUM_CUSTOMER,
                                          y = LYLTY_CARD_NBR,fill = LIFESTAGE))+
  geom_bar(position = 'dodge', stat = 'identity')+
  labs(x = "Customer Type", y = "Customers", title = "Number of Customers")
```



```
# Average number of units per customer by LIFESTAGE and PREMIUM_CUSTOMER

units_per_category <- aggregate(PROD_QTY~LIFESTAGE+PREMIUM_CUSTOMER,
                                qvi_data2, sum)

units_per_customer_category <- units_per_category

units_per_customer_category[, 4] <- units_per_category[,3]/number_customers_premium_life[,3]

ggplot(units_per_customer_category, aes(x =PREMIUM_CUSTOMER,
                                        y = V4,fill = LIFESTAGE)) +
  geom_bar(position = 'dodge', stat = 'identity')+
  labs(x = "Customer Type", y = "Avg Number of Units",
       title = "Average Number of Units Purchased")
```



```r
#Average price per unit in LIFESTAGE and PREMIUM_CUSTOMER

average_price_per_unit <- units_per_category

average_price_per_unit[, 4] <- tot_sales_premium_life[ , 3]

average_price_per_unit$total_sales <- average_price_per_unit[ ,4]

average_price_per_unit <- average_price_per_unit[, -4]

average_price_per_unit_LIFE_PREMIUM <- average_price_per_unit

average_price_per_unit_LIFE_PREMIUM[ , 5] <- average_price_per_unit[,4]/
  average_price_per_unit[3]

names(average_price_per_unit_LIFE_PREMIUM)[5] <- c("Average_Price")

ggplot(average_price_per_unit, aes(x =PREMIUM_CUSTOMER,
                                   y =total_sales/PROD_QTY,fill = LIFESTAGE)) +
  geom_bar(position = 'dodge', stat = 'identity')+
  labs(x = "Customer Type", y = "Total Sales", title = "Average Price per Unit")
```



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.

T-Test

In this section, we will perform a T-test between mainstream vs premium and budget mid age and young singles and couples. Therefore in order to do this we must state our null and alternative hypothesis:

H_{0} = The null hypothesis states that there is NO significant difference between the two groups.

H_{1} = The alternate hypothesis states that there is a significant difference between the two groups (mainstream mid age and young singles and couples vs premium and budget mid age and young singles and couples).

In order to reject the null hypothesis we will use the standard p-value of being less than or equal to 0.05.

```
#T test is between two groups mainstream midage and young singles/couples vs
# premimum and mainstream midage and young singles/couples


# null hypothesis, there is not a statistically significant difference between
# the two groups.
```

```r
df_main <- qvi_data2 %>% filter(PREMIUM_CUSTOMER == "Mainstream",
                                LIFESTAGE %in% c("MIDAGE SINGLES/COUPLES",
                                                 "YOUNG SINGLES/COUPLES"))

df_other <- qvi_data2 %>% filter(PREMIUM_CUSTOMER %in% c("Budget", "Premium"),
                                 LIFESTAGE %in% c("MIDAGE SINGLES/COUPLES",
                                                  "YOUNG SINGLES/COUPLES"))
main_avgsales <- df_main$TOT_SALES/df_main$PROD_QTY

other_avgsales <- df_other$TOT_SALES/df_other$PROD_QTY

tTest <- t.test(x = main_avgsales, y = other_avgsales,
                alternative = "two.sided", var.equal = TRUE)

tTest
```

```
##
##  Two Sample t-test
##
## data:  main_avgsales and other_avgsales
## t = 37.832, df = 57365, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.3160272 0.3505620
## sample estimates:
## mean of x mean of y
##  4.039786  3.706491
```

RESULTS

The t-test results in a p-value of <2.2e-16, i.e. the unit price for mainstream, young and mid-age singles and couples ARE significantly higher than that of budget or premium, young and mid age singles and couples. Therefore we can reject the null hypothesis that there is no significant difference between the two groups.

This result will enable us to develop strategies and recommendations which could improve sales as we now know that there is a quantitative difference suggested by the data among the groups of concern. Thus we can optimize and focus our strategies to each respective group in a more tailored and meaningful way.