Customer Segment Analysis

Mopelade Ademuyiwa

2023-04-16

Quantium Virtaul Internship

This is an exploratory analysis on the dataset. The aim is to clean the data and find interesting pattern and trends to work with and analysis to provide more information and customers that buy Chips.

```
library(dplyr)
```

Installation of relevant pakaages

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

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

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

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.4
## v ggplot2   3.4.1     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.1     v tidyr     1.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

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

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following object is masked from 'package:purrr':
##
##     transpose
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(ggmosaic)
library(readr)
library(stringr)
```

```
# This is the code for Import the `Transaction Data`

Transaction_data <- read.csv("QVI_transaction_data.csv")

head(Transaction_data)
```

Firstly,Import the Transaction Data for Cleaning and preperation.

```
##    DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 1 43390         1           1000      1        5
## 2 43599         1           1307    348       66
## 3 43605         1           1343    383       61
## 4 43329         2           2373    974       69
## 5 43330         2           2426   1038      108
## 6 43604         4           4074   2982       57
##                                 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
## 6 Old El Paso Salsa   Dip Tomato Mild 300g        1       5.1
```

```
tail(Transaction_data)
```

```
##         DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 264831 43416       272         272319 270087       44
## 264832 43533       272         272319 270088       89
## 264833 43325       272         272358 270154       74
## 264834 43410       272         272379 270187       51
```

```
## 264835 43461       272         272379 270188       42
## 264836 43365       272         272380 270189       74
##                                       PROD_NAME PROD_QTY TOT_SALES
## 264831          Thins Chips Light&  Tangy 175g        2       6.6
## 264832 Kettle Sweet Chilli And Sour Cream 175g        2      10.8
## 264833           Tostitos Splash Of  Lime 175g        1       4.4
## 264834                Doritos Mexicana    170g        2       8.8
## 264835 Doritos Corn Chip Mexican Jalapeno 150g        2       7.8
## 264836           Tostitos Splash Of  Lime 175g        2       8.8
```

Get acquainted with the Transaction dataset.

```
View(Transaction_data)
summary(Transaction_data)
```

```
##       DATE         STORE_NBR     LYLTY_CARD_NBR        TXN_ID
##  Min.   :43282   Min.   :  1.0   Min.   :   1000   Min.   :      1
##  1st Qu.:43373   1st Qu.: 70.0   1st Qu.:  70021   1st Qu.:  67602
##  Median :43464   Median :130.0   Median : 130358   Median : 135138
##  Mean   :43464   Mean   :135.1   Mean   : 135550   Mean   : 135158
##  3rd Qu.:43555   3rd Qu.:203.0   3rd Qu.: 203094   3rd Qu.: 202701
##  Max.   :43646   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
```

```
str(Transaction_data)
```

```
## '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 C
##  $ 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 ...
```

The **DATE** column is in Integral format, and change it to date format for better analysis.

```
# Convert the date column to a date format

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

head(Transaction_data)
```

```
##         DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 1 2018-10-17         1           1000      1        5
## 2 2019-05-14         1           1307    348       66
## 3 2019-05-20         1           1343    383       61
## 4 2018-08-17         2           2373    974       69
## 5 2018-08-18         2           2426   1038      108
## 6 2019-05-19         4           4074   2982       57
##                                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
## 6 Old El Paso Salsa   Dip Tomato Mild 300g        1       5.1
```

```
tail(Transaction_data)
```

```
##              DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 264831 2018-11-12       272         272319 270087       44
## 264832 2019-03-09       272         272319 270088       89
## 264833 2018-08-13       272         272358 270154       74
## 264834 2018-11-06       272         272379 270187       51
## 264835 2018-12-27       272         272379 270188       42
## 264836 2018-09-22       272         272380 270189       74
##                                      PROD_NAME PROD_QTY TOT_SALES
## 264831          Thins Chips Light&  Tangy 175g        2       6.6
## 264832 Kettle Sweet Chilli And Sour Cream 175g        2      10.8
## 264833           Tostitos Splash Of  Lime 175g        1       4.4
## 264834                Doritos Mexicana    170g        2       8.8
## 264835 Doritos Corn Chip Mexican Jalapeno 150g        2       7.8
## 264836           Tostitos Splash Of  Lime 175g        2       8.8
```

Analyse the product column, to identify the outliners, the Top and less 5 product.

```
names(Transaction_data)
```

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

```
Product_frequency <- Transaction_data %>%
  count(PROD_NAME) %>%
  arrange(desc(n))

head(Product_frequency)
```

```
##                                PROD_NAME    n
## 1   Kettle Mozzarella   Basil & Pesto 175g 3304
## 2 Kettle Tortilla ChpsHny&Jlpno Chili 150g 3296
## 3 Cobs Popd Swt/Chlli &Sr/Cream Chips 110g 3269
## 4   Tyrrells Crisps     Ched & Chives 165g 3268
## 5           Cobs Popd Sea Salt  Chips 110g 3265
## 6             Kettle 135g Swt Pot Sea Salt 3257
```

```
Total_sales <- Transaction_data %>%
  group_by(PROD_NAME) %>%
  summarise(Total_Sales = sum(TOT_SALES))
# The is the code for thr Top 5 products By sales

Top_5_product <- Total_sales %>%
  arrange(desc(Total_Sales)) %>%
  head(5)

Top_5_product
```

```
## # A tibble: 5 x 2
##   PROD_NAME                                Total_Sales
##   <chr>                                          <dbl>
## 1 Dorito Corn Chp     Supreme 380g              40352
## 2 Smiths Crnkle Chip  Orgnl Big Bag 380g        36368.
## 3 Smiths Crinkle Chips Salt & Vinegar 330g      34804.
## 4 Kettle Mozzarella   Basil & Pesto 175g        34457.
## 5 Smiths Crinkle      Original 330g             34303.
```

```
# This the code for the Least performing products by sales

Least_5_products <- Total_sales %>%
  arrange(Total_Sales) %>%
  head(5)

  Least_5_products
```

```
## # A tibble: 5 x 2
##   PROD_NAME                                Total_Sales
##   <chr>                                          <dbl>
## 1 Woolworths Medium   Salsa 300g                 4050
## 2 Woolworths Mild     Salsa 300g                 4234.
## 3 WW Crinkle Cut      Original 175g              4532.
## 4 Sunbites Whlegrn    Crisps Frch/Onin 90g       4600.
## 5 WW Crinkle Cut      Chicken 175g               4702.
```

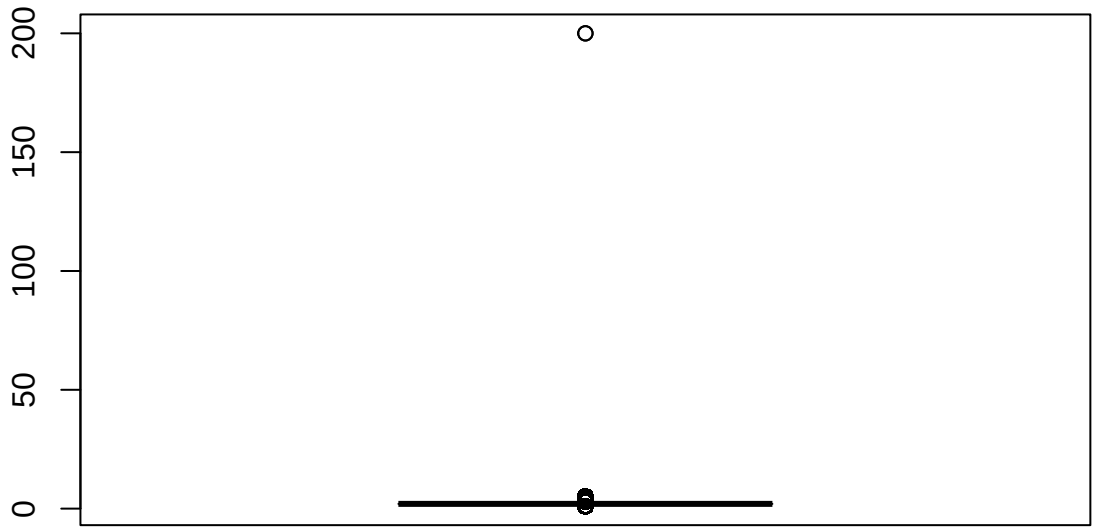
```
  Isnull_product <- sum(is.null(Transaction_data$PROD_NAME))
```

Next, Analyse the **Product Quality** to find outliners

```
summary(Transaction_data$PROD_QTY)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.000   1.907   2.000 200.000
```

```
boxplot(Transaction_data$PROD_QTY)
```




```
# 200, which is the highest product Quality, and also and an outliner. I will further analysis to find

quality200 <- Transaction_data %>%
  filter(PROD_QTY == 200)

quality200
```

```
##         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 only two occasions were 200 packets were bought in a transaction and These transactions wer
# I will filter out these transactions

Transaction_data <- filter(Transaction_data, PROD_QTY != 200)

summary(Transaction_data$PROD_QTY)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.000   1.906   2.000   5.000
```

The Transaction Over time Analysis, this is to out if there are missing date(days the company didnt make any sale).

```
# Extract both the month and year from the Date for further analysis

Transaction_data <- Transaction_data %>%
  mutate(Year = as.POSIXlt(DATE)$year + 1900)

# Extract the month from data
Transaction_data <- mutate(Transaction_data, Month_num = as.POSIXlt(DATE)$mon + 1)

head(Transaction_data)
```

```
##         DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 1 2018-10-17         1           1000      1        5
## 2 2019-05-14         1           1307    348       66
## 3 2019-05-20         1           1343    383       61
## 4 2018-08-17         2           2373    974       69
## 5 2018-08-18         2           2426   1038      108
## 6 2019-05-19         4           4074   2982       57
##                                PROD_NAME PROD_QTY TOT_SALES Year Month_num
## 1   Natural Chip        Compny SeaSalt175g        2       6.0 2018        10
## 2                 CCs Nacho Cheese    175g        3       6.3 2019         5
## 3   Smiths Crinkle Cut  Chips Chicken 170g        2       2.9 2019         5
## 4   Smiths Chip Thinly  S/Cream&Onion 175g        5      15.0 2018         8
## 5 Kettle Tortilla ChpsHny&Jlpno Chili 150g        3      13.8 2018         8
## 6 Old El Paso Salsa   Dip Tomato Mild 300g        1       5.1 2019         5
```

```
Transa_dates <- Transaction_data %>%
  count(DATE)

head(Transa_dates)
```

```
##         DATE   n
## 1 2018-07-01 724
## 2 2018-07-02 711
## 3 2018-07-03 722
## 4 2018-07-04 714
## 5 2018-07-05 712
## 6 2018-07-06 762
```

```
tail(Transa_dates)
```

```
##           DATE   n
## 359 2019-06-25 753
## 360 2019-06-26 723
## 361 2019-06-27 709
## 362 2019-06-28 730
## 363 2019-06-29 745
## 364 2019-06-30 744
```

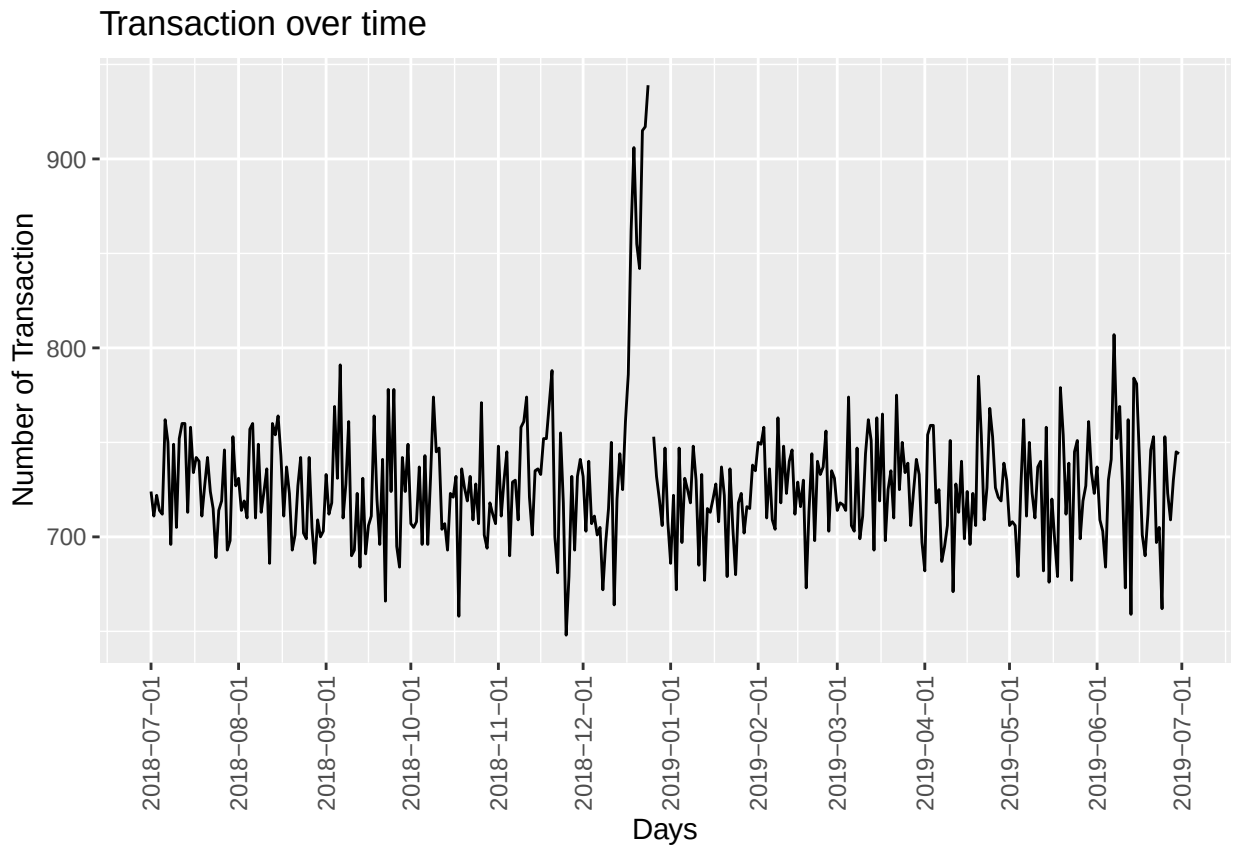
```r
# There are 364 rows, which means the company did not make any sales in one day, to find out. I will cr

dates_seQ <- tibble(DATE = seq(as.Date("2018-07-01"), as.Date("2019-06-30"), by = "day"))

# And creating the sequence data table, merge the sequence table and the actual data table.

Transa_by_day <- left_join(dates_seQ, Transa_dates, by = "DATE")

head(Transa_by_day)
```

```
## # A tibble: 6 x 2
##   DATE           n
##   <date>     <int>
## 1 2018-07-01   724
## 2 2018-07-02   711
## 3 2018-07-03   722
## 4 2018-07-04   714
## 5 2018-07-05   712
## 6 2018-07-06   762
```

```r
tail(Transa_by_day)
```

```
## # A tibble: 6 x 2
##   DATE           n
##   <date>     <int>
## 1 2019-06-25   753
## 2 2019-06-26   723
## 3 2019-06-27   709
## 4 2019-06-28   730
## 5 2019-06-29   745
## 6 2019-06-30   744
```

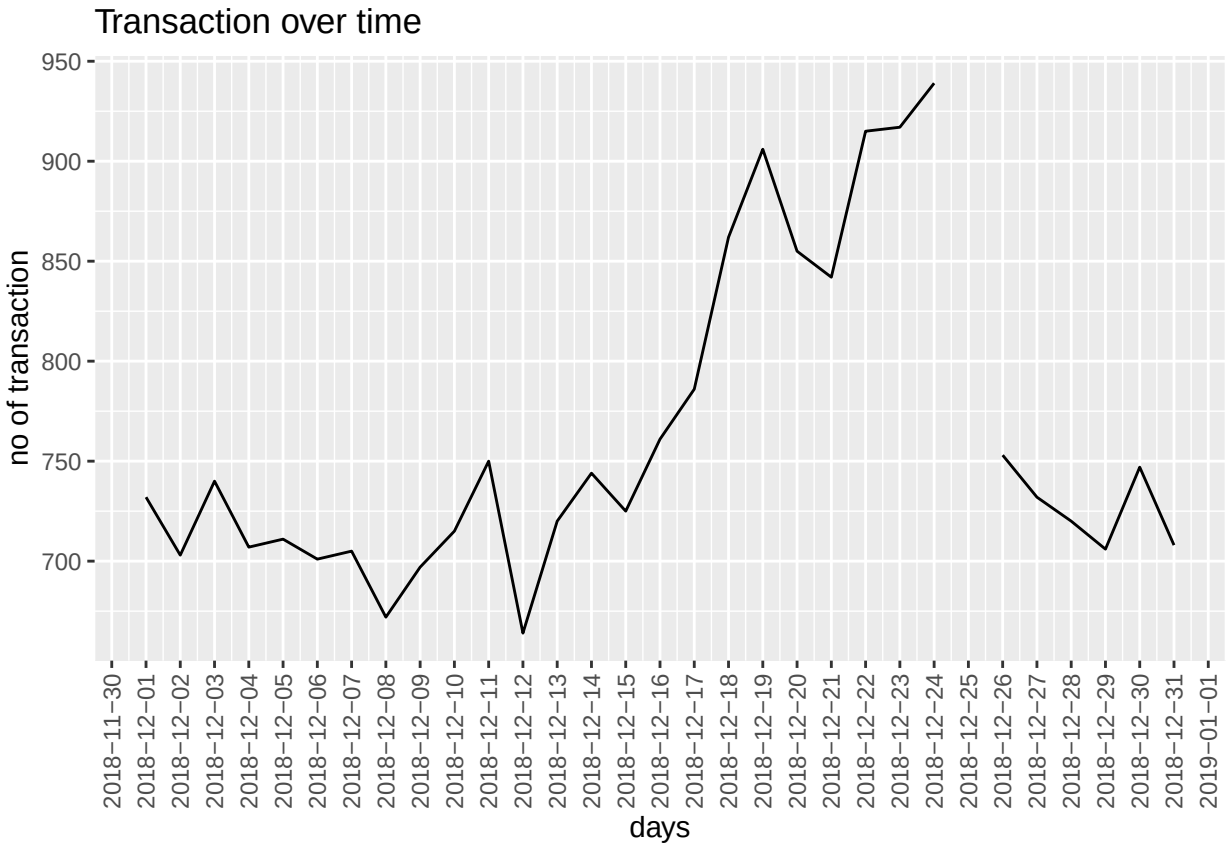
```r
ggplot(Transa_by_day, aes(DATE,n))+
  geom_line()+
  labs(x = "Days",  y = "Number of Transaction", title = "Transaction over time")+
  scale_x_date(breaks = "1 month")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




```
# Notably, there is a missing day in December. Next, visualize the transaction that took place in Decem

Filtered_dec <- Transa_by_day %>%
  filter(month(DATE) == 12)
head(Filtered_dec)
```

```
## # A tibble: 6 x 2
##   DATE           n
##   <date>     <int>
## 1 2018-12-01   732
## 2 2018-12-02   703
## 3 2018-12-03   740
## 4 2018-12-04   707
## 5 2018-12-05   711
## 6 2018-12-06   701
```

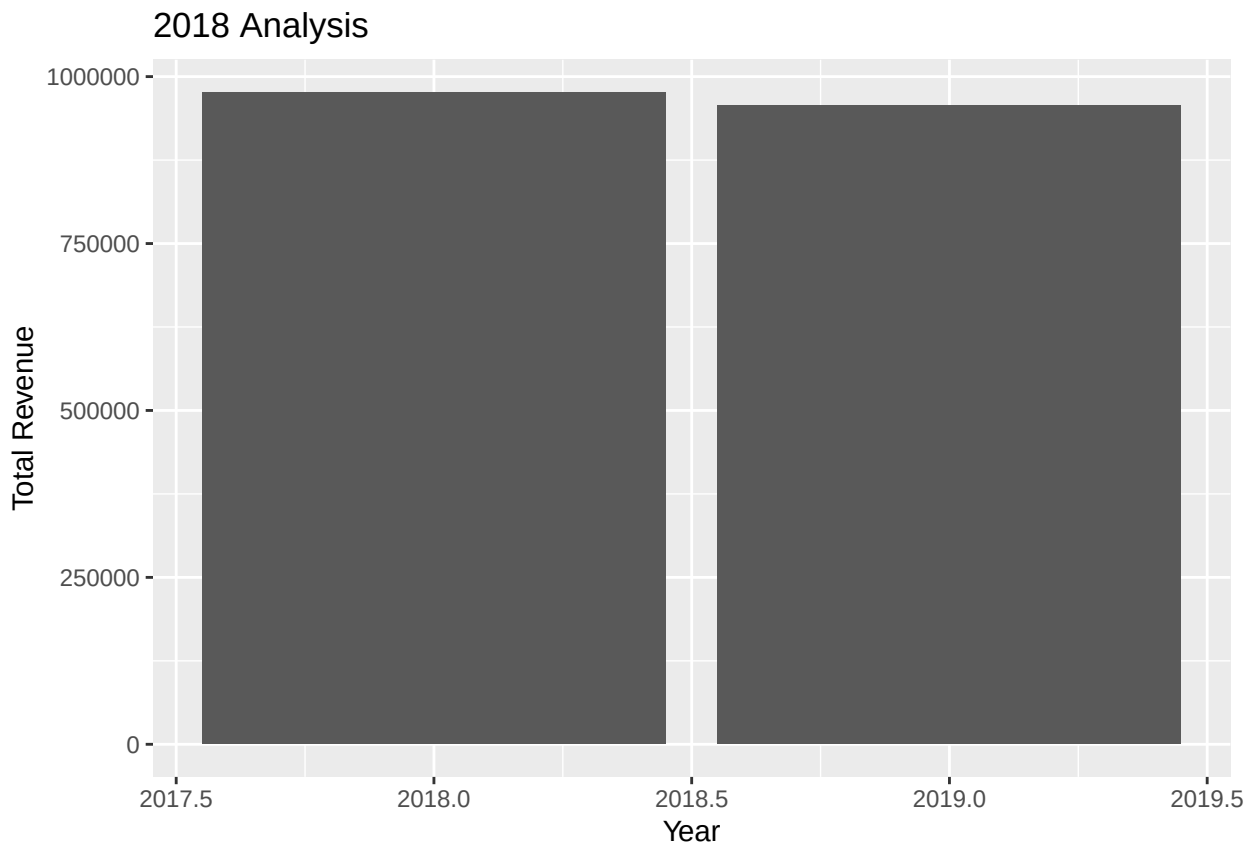
```
ggplot(Filtered_dec, aes(DATE,n))+
  geom_line()+
  labs(x = "days", y = "no of transaction", title = "Transaction over time")+
  scale_x_date(breaks = "1 day")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




```
# Notably, the missing day is 25th of December, which is understandable as that day is Xmas day, store
```

Further Analysis on DATE

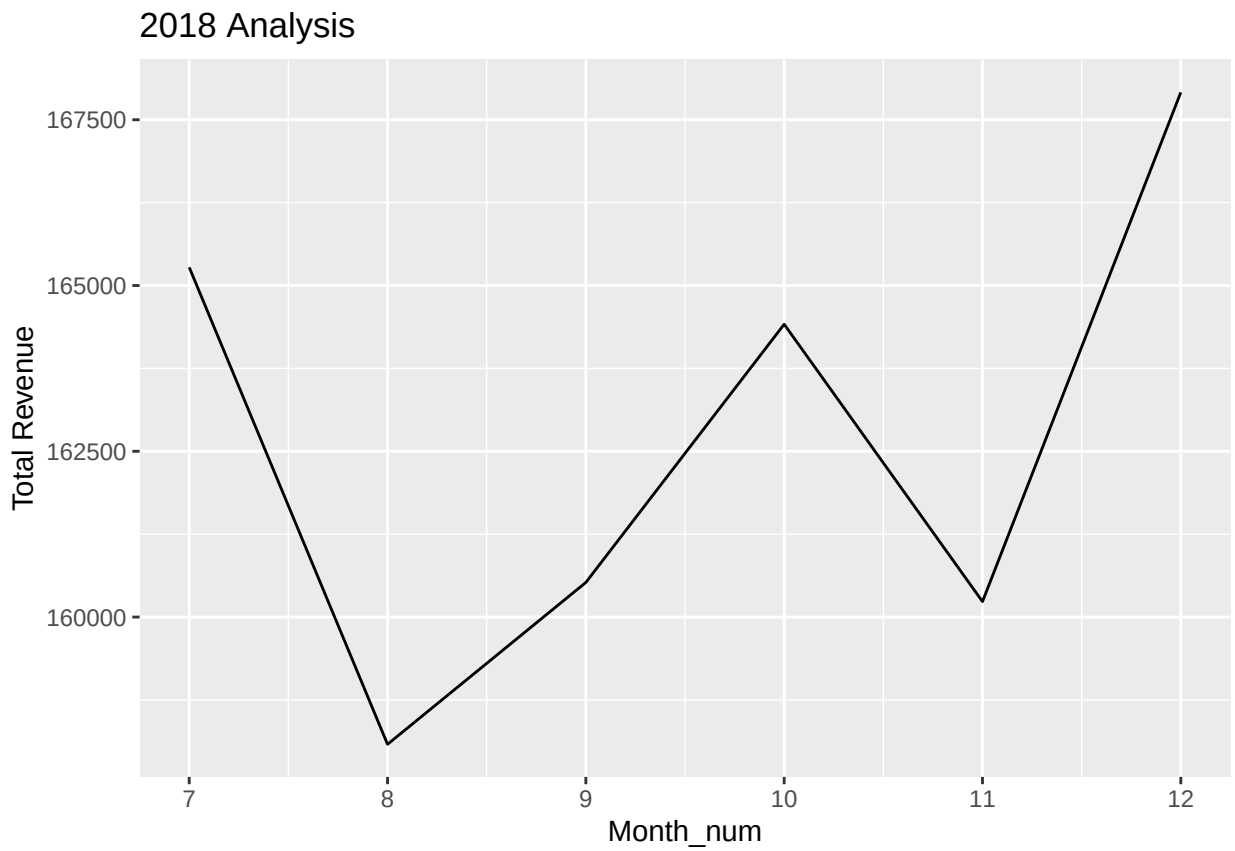
```
# The year the company generated more revenue

sales_by_year <- Transaction_data %>%
  group_by(Year) %>%
  summarise(Total_revenue = sum(TOT_SALES)) %>%
  arrange(desc(Total_revenue))

ggplot(sales_by_year, aes(Year, Total_revenue))+
  geom_col()+
  labs(x = "Year",y = "Total Revenue", title = "2018 Analysis")
```




```
sales_by_year
```

```
## # A tibble: 2 x 2
##    Year Total_revenue
##   <dbl>         <dbl>
## 1  2018       976441.
## 2  2019       956674.
```

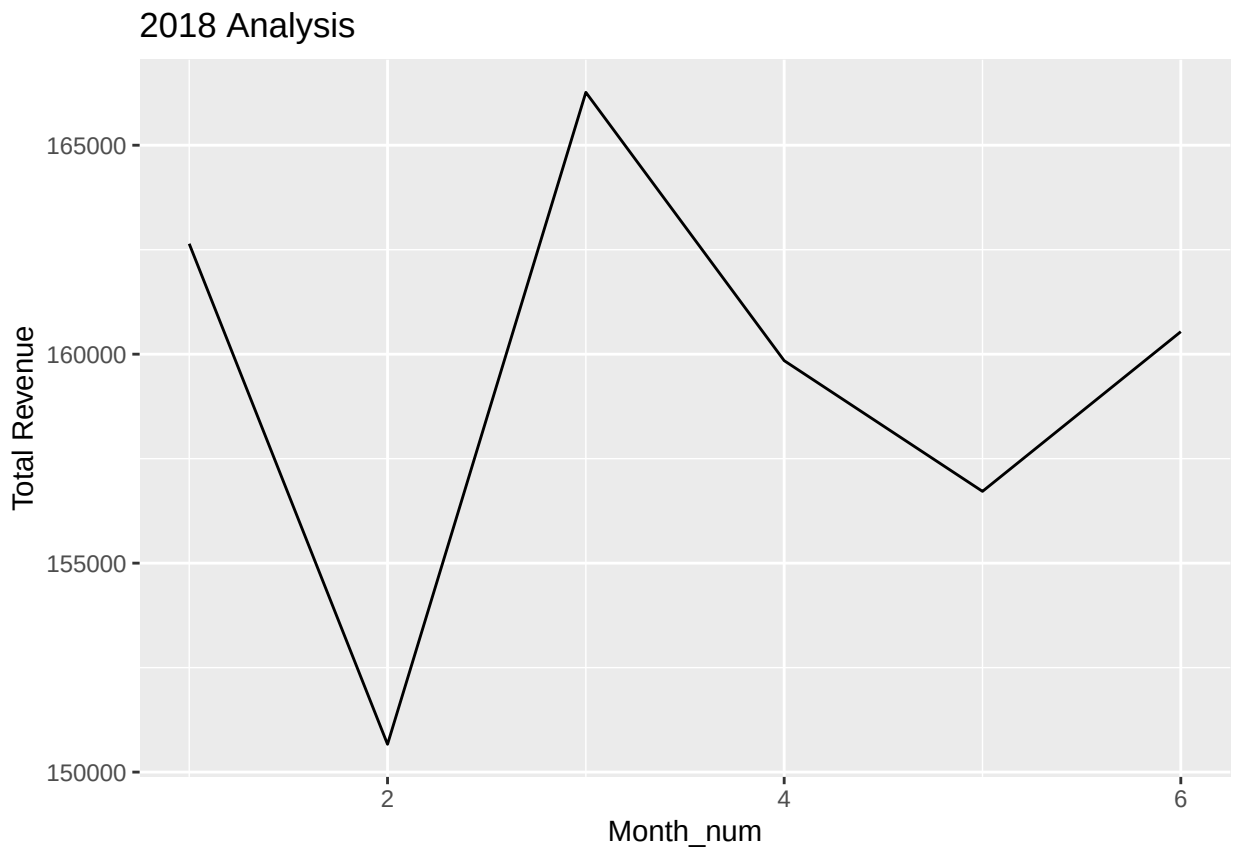
```
# The company generated the most revenue In 2019.

# What of the by Month on both year

sales_2018 <- Transaction_data %>%
  filter(Year == "2018") %>%
  group_by(Month_num) %>%
  summarise(Total_revenue = sum(TOT_SALES)) %>%
  arrange(desc(Total_revenue))

ggplot(sales_2018, aes(Month_num, Total_revenue))+
  geom_line()+
  labs(x = "Month_num",y = "Total Revenue", title = "2018 Analysis")
```




```
sales_2018
```

```
## # A tibble: 6 x 2
##   Month_num Total_revenue
##       <dbl>         <dbl>
## 1        12       167913.
## 2         7       165275.
## 3        10       164416.
## 4         9       160522
## 5        11       160234.
## 6         8       158081.
```

```
sales_2019 <- Transaction_data %>%
  filter(Year == "2019") %>%
  group_by(Month_num) %>%
  summarise(Total_revenue = sum(TOT_SALES)) %>%
  arrange(desc(Total_revenue))

ggplot(sales_2019, aes(Month_num, Total_revenue))+
  geom_line()+
  labs(x = "Month_num",y = "Total Revenue", title = "2018 Analysis")
```

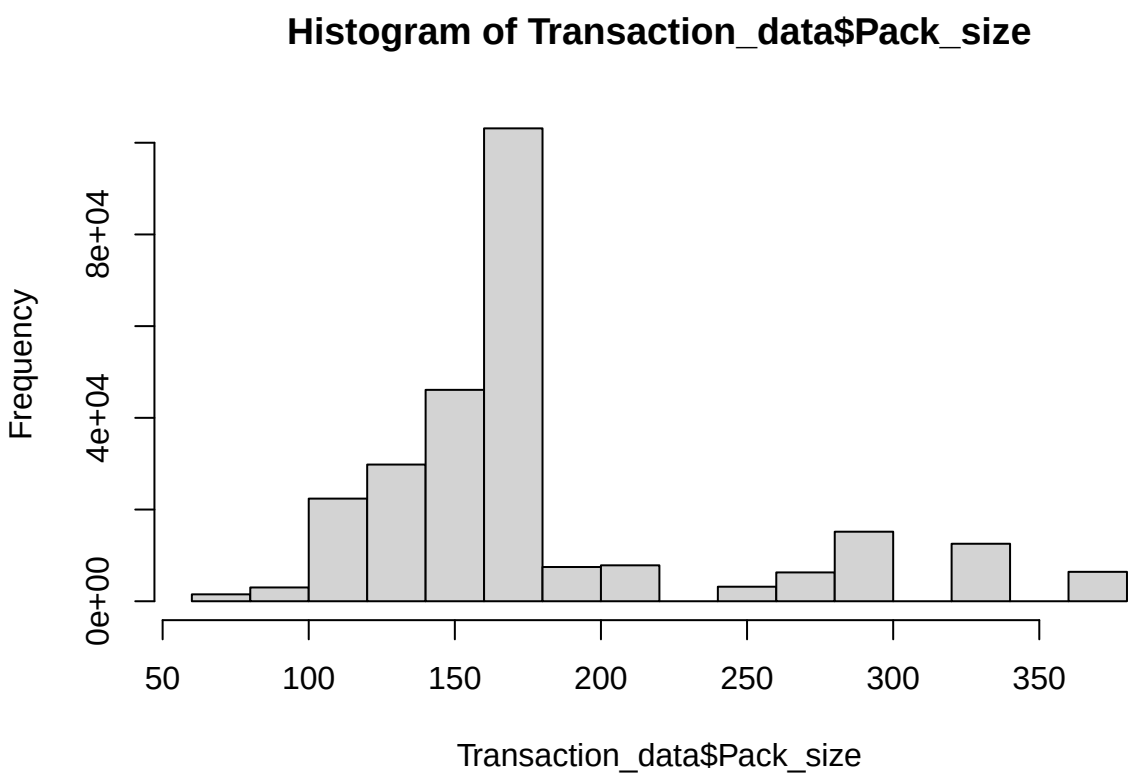



```
sales_2019
```

```
## # A tibble: 6 x 2
##   Month_num Total_revenue
##       <dbl>         <dbl>
## 1         3       166265.
## 2         1       162642.
## 3         6       160539.
## 4         4       159845.
## 5         5       156718.
## 6         2       150665
```

Moving on, analyse the pack size, to find out the pack sizes that are mostly bought.

```
# This is the code to extract the Packet size from the Product Name.

Transaction_data <- mutate(Transaction_data, Pack_size := parse_number(PROD_NAME))

# Then  find out the most bought Packet size and the least

packet_size <- Transaction_data %>%
  count(Pack_size) %>%
  arrange(desc(n))


hist(Transaction_data$Pack_size)
```




```
head(packet_size)
```

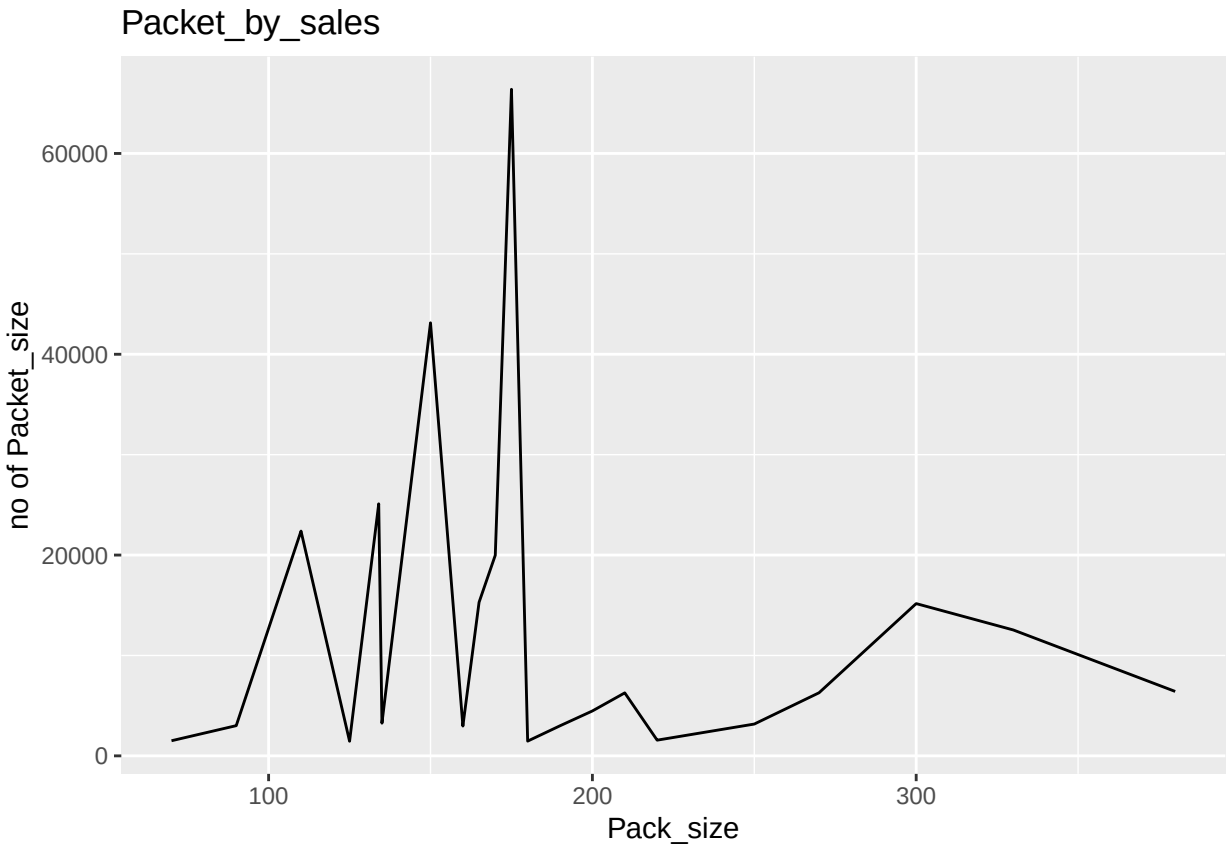
```
##   Pack_size     n
## 1       175 66390
## 2       150 43131
## 3       134 25102
## 4       110 22387
## 5       170 19983
## 6       165 15297
```

```
tail(packet_size)
```

```
##    Pack_size    n
## 16       190 2995
## 17       160 2970
## 18       220 1564
## 19        70 1507
## 20       180 1468
## 21       125 1454
```

```
packet_sales <- Transaction_data %>%
  group_by(Pack_size) %>%
  summarise(PAck_sales = sum(TOT_SALES)) %>%
  arrange(desc(PAck_sales))
```

```
ggplot(packet_size, aes(Pack_size,n))+
  geom_line()+
  labs(x = "Pack_size", y = "no of Packet_size", title = "Packet_by_sales")
```




```
head(packet_sales)
```

```
## # A tibble: 6 x 2
##   Pack_size PAck_sales
##       <dbl>      <dbl>
## 1       175    485437.
## 2       150    304288.
## 3       134    177656.
## 4       110    162765.
## 5       170    146673
## 6       330    136794.
```

```
tail(packet_sales)
```

```
## # A tibble: 6 x 2
##   Pack_size PAck_sales
##       <dbl>      <dbl>
## 1       160     10648.
## 2        90      9676.
```

```
## 3       180      8568.
## 4        70      6852
## 5       220      6831
## 6       125      5733
```

Brand Analysis, identify the best selling Brand.

```
# Firstly, I will extract brands from the product name.
# This is the code to extract words from a column.

library(stringr)

Transaction_data <- Transaction_data %>%
  mutate(Brand_name = str_extract(PROD_NAME, "^\\w+"))

# I will summarise the brand sales

Brand_sales <- Transaction_data %>%
  group_by(Brand_name) %>%
  summarise(Total_sales = sum(TOT_SALES)) %>%
  arrange(desc(Total_sales))

head(Brand_sales)
```

```
## # A tibble: 6 x 2
##   Brand_name Total_sales
##   <chr>            <dbl>
## 1 Kettle         390240.
## 2 Smiths         210077.
## 3 Doritos        201539.
## 4 Pringles       177656.
## 5 Old             90785.
## 6 Thins           88853.
```

```
tail(Brand_sales)
```

```
## # A tibble: 6 x 2
##   Brand_name Total_sales
##   <chr>            <dbl>
## 1 GrnWves          8568.
## 2 NCC              8046
## 3 French           7929
## 4 Burger           6831
## 5 Snbts            5076.
## 6 Sunbites         4600.
```

```
# Some of the Brand names are spelt incorrectly,  change to make corrections.

Transaction_data <- Transaction_data %>%
  mutate(Brand_name = case_when(
           Brand_name == "Dorito" ~ "Doritos",
```

```
            Brand_name == "GmWves" ~ "Grain",
            Brand_name == "Smith" ~ "Smiths",
            Brand_name == "Snbts" ~ "Sunbites",
            Brand_name == "WW" ~ "Woolworths",
            Brand_name == "Red" ~ "RRD",
            Brand_name == "NCC" ~ "Natural",
            Brand_name == "Infzns" ~ "Infuzions",
            TRUE ~ Brand_name
  ))


#  Check if they Brand Names are corrected

Brand_sales <- Transaction_data %>%
  group_by(Brand_name) %>%
  summarise(Total_sales = sum(TOT_SALES)) %>%
  arrange(desc(Total_sales))

head(Brand_sales)
```

```
## # A tibble: 6 x 2
##   Brand_name Total_sales
##   <chr>            <dbl>
## 1 Kettle         390240.
## 2 Doritos        240591.
## 3 Smiths         224660.
## 4 Pringles       177656.
## 5 Infuzions       99048.
## 6 RRD             95046
```

```
tail(Brand_sales)
```

```
## # A tibble: 6 x 2
##   Brand_name Total_sales
##   <chr>            <dbl>
## 1 CCs              18079.
## 2 Cheetos          16884.
## 3 Sunbites          9676.
## 4 GrnWves           8568.
## 5 French            7929
## 6 Burger            6831
```

Purchase Behavior exploratory Analysis

```
Purchase_behaviour <- read.csv("QVI_purchase_behaviour.csv")

str(Purchase_behaviour)
```

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

```
# There are three Column in purchase_behaviour table.
# Analyse Life Stage and Premium_customers,
#to check if there are missing data, wrong spelling, duplicate etc
```

Analyse Life Stage and Premium_customers, to check if there are missing data, wrong spelling, duplicate Etc.

```
# Examining the values of the Life stage column.

Life_stage <- Purchase_behaviour %>%
  count(LIFESTAGE) %>%
  arrange(desc(n))

Life_stage
```

```
##                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
```

```
# Examining the values of the Premium_Customers column.
Premium_customers <- Purchase_behaviour %>%
  count(PREMIUM_CUSTOMER) %>%
  arrange(desc(n))

Premium_customers
```

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

Merge Purchase behaviour data table and Transaction_ data table for the major analysis

```
# This is the code To merge the two data tables,

Customer_segmentation <- full_join(Transaction_data, Purchase_behaviour,
                                   by =  "LYLTY_CARD_NBR")

# Find out if there are Null in the Customer segmentation data table.

isnul <- sum(is.null(Customer_segmentation$PREMIUM_CUSTOMER))

isnul <- sum(is.null(Customer_segmentation$LIFESTAGE))
```

End of Data Exploration Analysis

DATA ANALYSIS ON CUSTOMER ANALYSIS

```
# Calculate the total sales by Life stage and Premium Customer
# Set the theme for the plot with this code
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))

# This calculate the total revenue by Life stage and Premium Customer

Total_Revenue <- Customer_segmentation %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(Total_Revenue = sum(TOT_SALES)) %>%
  mutate(percentage = 100 * Total_Revenue / sum(Total_Revenue))
```

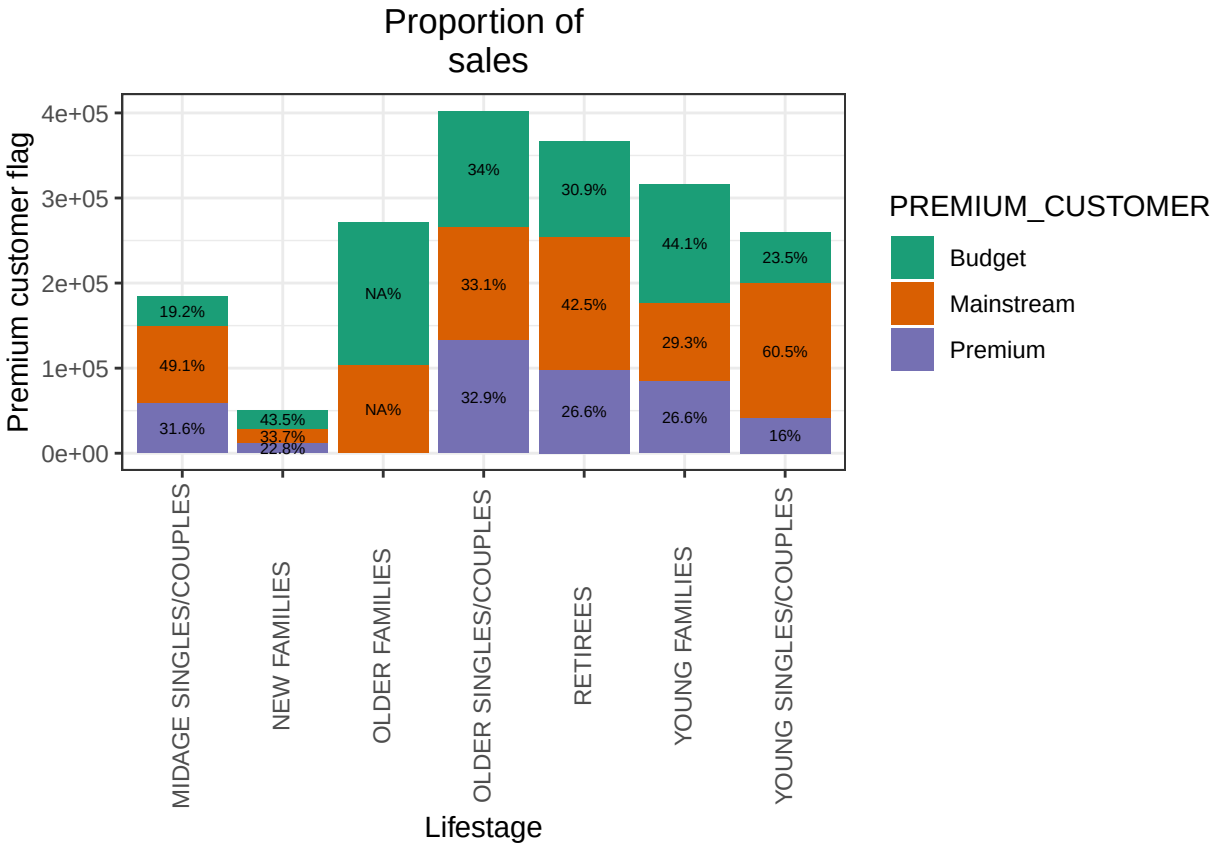
```
## `summarise()` has grouped output by 'LIFESTAGE'. You can override using the
## `.groups` argument.
```

```
head(Total_Revenue)
```

```
## # A tibble: 6 x 4
## # Groups:   LIFESTAGE [2]
##   LIFESTAGE              PREMIUM_CUSTOMER Total_Revenue percentage
##   <chr>                  <chr>                   <dbl>      <dbl>
## 1 MIDAGE SINGLES/COUPLES Budget                 35515.       19.2
## 2 MIDAGE SINGLES/COUPLES Mainstream             90804.       49.1
## 3 MIDAGE SINGLES/COUPLES Premium                58433.       31.6
## 4 NEW FAMILIES           Budget                 21928.       43.5
## 5 NEW FAMILIES           Mainstream             17014.       33.7
## 6 NEW FAMILIES           Premium                11491.       22.8
```

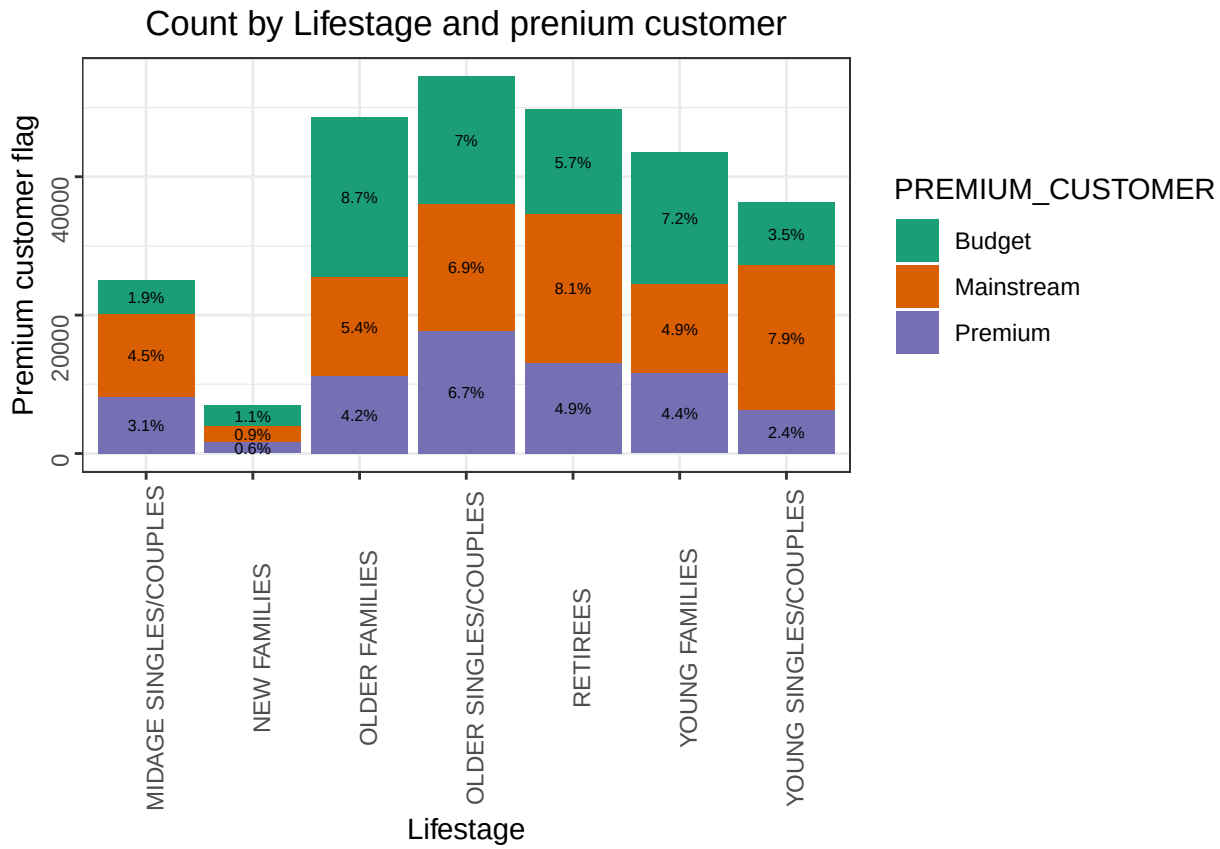
```
# create a plot

ggplot(Total_Revenue, aes(x = LIFESTAGE, y = Total_Revenue, fill = PREMIUM_CUSTOMER))+
geom_col()+
  geom_text(aes(label = paste0(round(percentage, 1), "%")), position = position_stack(vjust = 0.5 ), si
  labs(x = "Lifestage", y = "Premium customer flag", title = "Proportion of
 sales") +
  scale_fill_brewer(palette = "Dark2")+
theme(axis.text.x = element_text(angle = 90, vjust = 0.4))
```

```
## Warning: Removed 1 rows containing missing values (`position_stack()`).
## Removed 1 rows containing missing values (`position_stack()`).
```



The company generate most revenue from Young singles/couples = Mainstream , Retirees = Mainstream and Older Families = Budget. This might be attributed to the number of customers who buys chip

```
# The count by Life stage and Premium_customer

customer_number <- Customer_segmentation %>%
  count(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  mutate(total_count = sum(n),
         percentage = n/total_count*100)

ggplot(customer_number,aes(x = LIFESTAGE, y = n, fill = PREMIUM_CUSTOMER))+
  geom_col()+
  geom_text(aes(label = paste0(round(percentage, 1),"%")), position = position_stack(vjust = 0.5), size
  labs(x = "Lifestage", y = "Premium customer flag", title = "Count by Lifestage and prenium customer")
  scale_fill_brewer(palette = "Dark2")+
  theme(axis.text = element_text(angle = 90, vjust = 0.1))
```



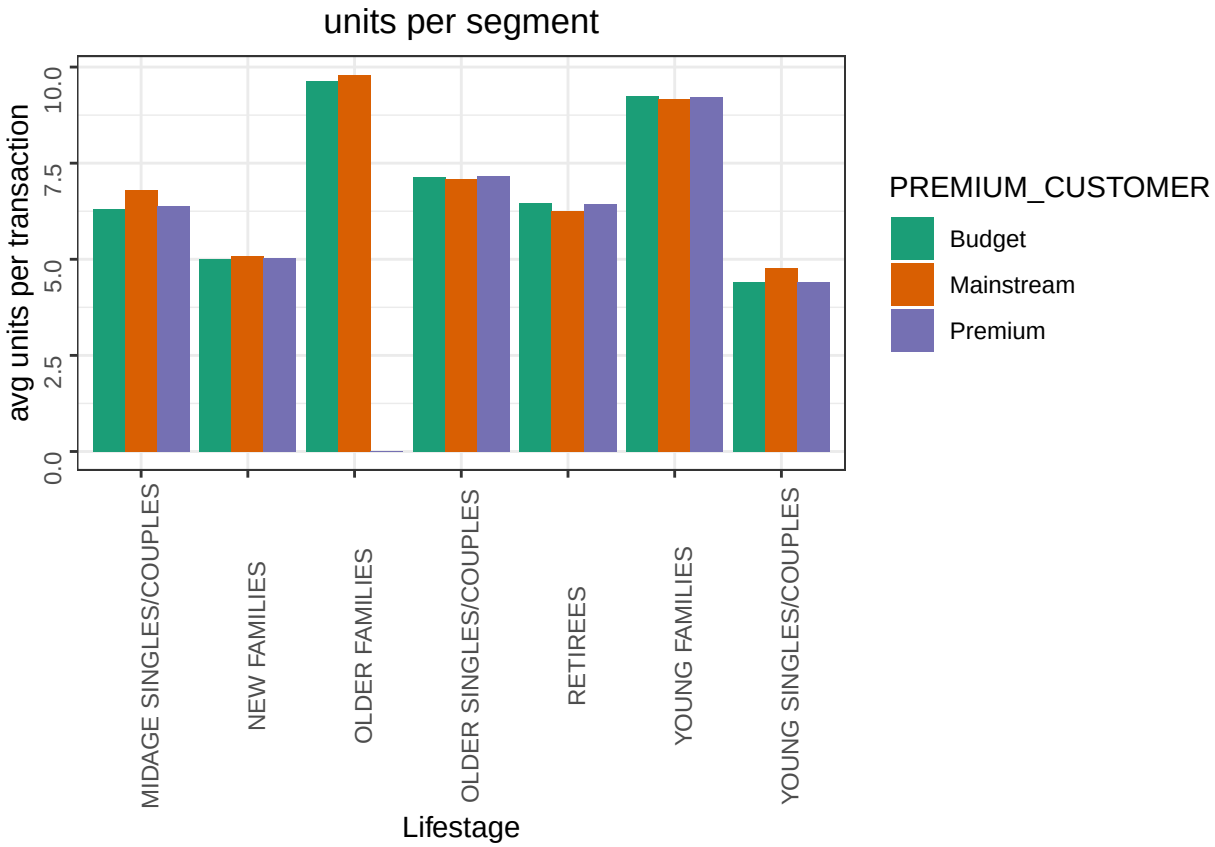
Notably, Number of customer is a driver factor for sales, as the Segment of customers that generated the most revenue , have the highest number of people. Young singles/couples = Mainstream , Retirees = Mainstream and Older Families = Budget, have more people who buy chips.

Analyse the units of chips bought by each segment, as this is also a driving factor for sales .

```
# The Average by Life stage and Premium Customers.

avg <- Customer_segmentation %>%
group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarize(AVG = sum(PROD_QTY) / n_distinct(LYLTY_CARD_NBR)) %>%
  arrange(desc(AVG))
```

```
## `summarise()` has grouped output by 'LIFESTAGE'. You can override using the
## `.groups` argument.
```

```
# Create a plot

ggplot(avg, aes( weight = AVG, x = LIFESTAGE, fill = PREMIUM_CUSTOMER))+
  geom_bar(position = position_dodge())+
  labs(x = "Lifestage", y = "avg units per transaction", title = "units per segment") +
  scale_fill_brewer(palette = "Dark2")+
  theme(axis.text = element_text(angle = 90, vjust = 0.5))
```



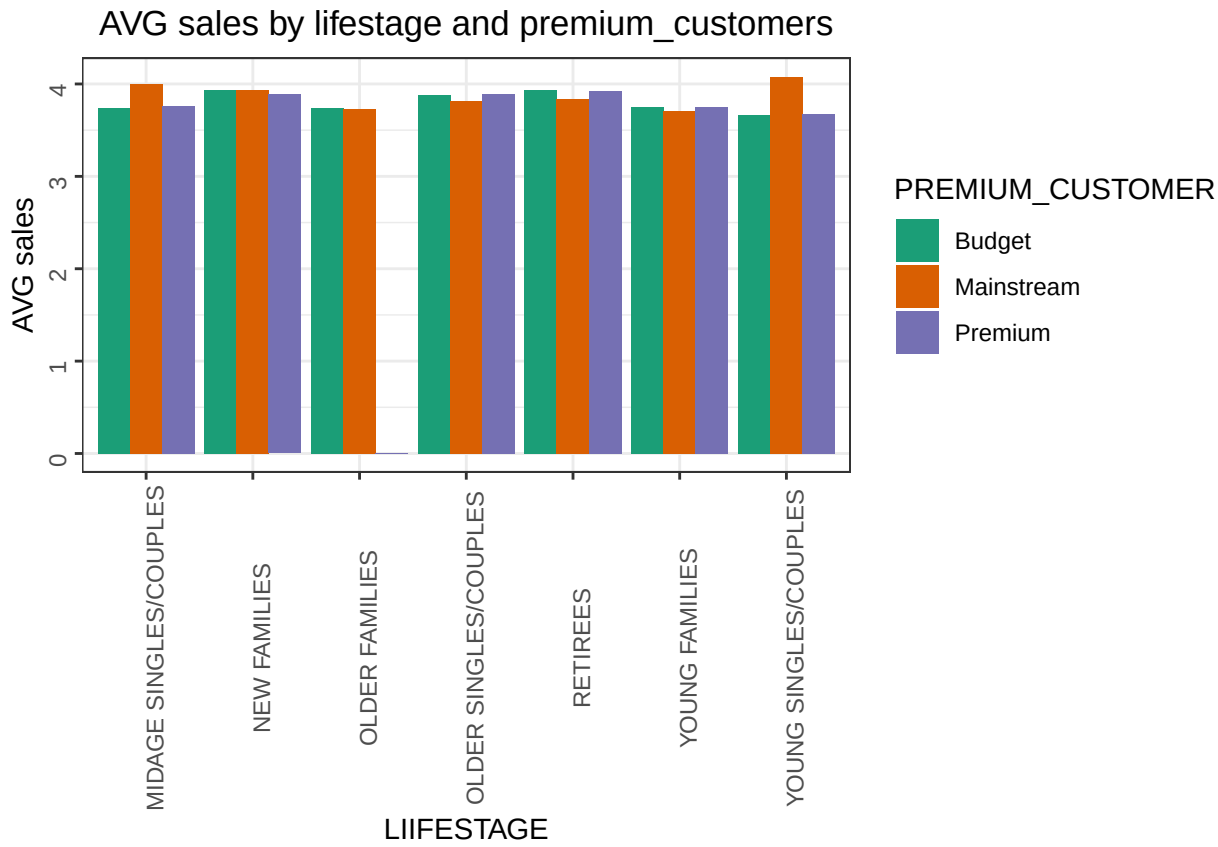
Notably, older families followed by young families buy more chips than other groups in the LIFESTAGE and also Older families/ mainstream seems to buy more chips compared to its premium and budget counterpart. while young families / budget also buy more chips compared to its premium and mainstream counterpart.

Analyse avg sales by LIFE_STAGE and PREMIUM_CUSTOMER, to identity the segment that spent more money in buying chips.

```
# Here is the code for the avg sales

Avg_sales <- Customer_segmentation %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(Avg_sa = sum(TOT_SALES)/sum(PROD_QTY)) %>%
  arrange(desc(Avg_sa))
```

```
## `summarise()` has grouped output by 'LIFESTAGE'. You can override using the
## `.groups` argument.
```

```
# Create a plot

ggplot(Avg_sales, aes(weight = Avg_sa, x = LIFESTAGE, fill = PREMIUM_CUSTOMER))+
  geom_bar(position = position_dodge())+
  labs(x = "LIIFESTAGE",y = "AVG sales", title = "AVG sales by lifestage and premium_customers")+
  scale_fill_brewer(palette = "Dark2")+
  theme(axis.text = element_text(angle = 90, vjust = 0.5))
```



young singles/couples and Mid-age singles/couples Mainstream spent more money on chips compared to other groups in The LIFESTAGE and their Premium and budget counterparts. Premium people often have the tendency of spending more money on healthy snacks and budget shoppers mostly don't have as money money as their mainstream counterparts. This explains the reason there are more mainstream young singles and couples and mid-age singles and couples shoppers.

```
library(stats)

Customer_segmentation <- mutate(Customer_segmentation, Avg_price = TOT_SALES / PROD_QTY)

group1 <- Customer_segmentation %>%
  filter(LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") & PREMIUM_CUSTOMER == "Mai

group2 <- Customer_segmentation %>%
  filter(LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") & PREMIUM_CUSTOMER != "Mai

t.test(group1$Avg_price, group2$Avg_price, alternative = "greater")
```

```
##
##  Welch Two Sample t-test
##
## data:  group1$Avg_price and group2$Avg_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
```

Based on the T.test,the p_value is significantly small, it is less than 0.5. The avg price of mainstream /Young singles/couples and Mid-age singles and couples is greater than that of avg price Young singles and couples and Mid-age singles and couples Premium and Budget counterparts.

To futher my analysis I would focus on the mainstream young singles and couples as the major target customer contributing more revenue to the company. And i would identify the brand they are most likely to buy.

```
# To analysis this , group  target  customer and other customers in two data frames.

main <- Customer_segmentation %>%
  filter(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream")
str(main)
```

```
## 'data.frame':    20854 obs. of  15 variables:
##  $ DATE            : Date, format: "2018-08-16" "2018-08-17" ...
##  $ STORE_NBR       : int  1 1 1 3 3 3 3 3 3 3 ...
##  $ LYLTY_CARD_NBR  : int  1020 1163 1291 3031 3118 3120 3182 3316 3329 3354 ...
##  $ TXN_ID          : int  26 188 333 1227 1574 1580 1864 2449 2491 2594 ...
##  $ PROD_NBR        : int  19 46 27 14 62 33 65 60 16 14 ...
##  $ PROD_NAME       : chr  "Smiths Crinkle Cut  Snag&Sauce 150g" "Kettle Original 175g" "WW Supreme Cl
##  $ PROD_QTY        : int  1 1 1 1 1 1 5 1 1 1 ...
##  $ TOT_SALES       : num  2.6 5.4 1.9 5.9 3.7 3.8 10.2 4.6 5.7 5.9 ...
##  $ Year            : num  2018 2018 2018 2019 2019 ...
##  $ Month_num       : num  8 8 8 5 5 8 5 8 8 5 ...
##  $ Pack_size       : num  150 175 200 380 134 110 300 150 330 380 ...
##  $ Brand_name      : chr  "Smiths" "Kettle" "Woolworths" "Smiths" ...
##  $ LIFESTAGE       : chr  "YOUNG SINGLES/COUPLES" "YOUNG SINGLES/COUPLES" "YOUNG SINGLES/COUPLES" "YU
##  $ PREMIUM_CUSTOMER: chr  "Mainstream" "Mainstream" "Mainstream" "Mainstream" ...
##  $ Avg_price       : num  2.6 5.4 1.9 5.9 3.7 3.8 2.04 4.6 5.7 5.9 ...
```

```
others <- Customer_segmentation %>%
  filter(!LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream")

# Calculate the Product quality By each customers grouped.

Quality_main <- main %>%
  summarise(Product_quality = sum(PROD_QTY))

other_quality <- others %>%
  summarise(Product_quality = sum(PROD_QTY))

# calculate the Brand each segments buy the most

Quality_main_brand <- main %>%
  group_by(Brand_name) %>%
  summarise(Target_brand =  sum(PROD_QTY) / Quality_main) %>%
```

```
  arrange(desc(Target_brand))

other_quality_brand <-others %>%
  group_by(Brand_name) %>%
  summarise(other_target = sum(PROD_QTY) / other_quality)

# This is the code to find out the chips the target customers most likey buy and less likely buy compar

brand_proportions <- full_join(Quality_main_brand, other_quality_brand, by = "Brand_name") %>%
  mutate(affinity_to_brand = Target_brand / other_target) %>%
  arrange(desc(affinity_to_brand))

brand_proportions <- brand_proportions %>%
  mutate(Target_brand = pull(Target_brand, Product_quality),
         other_target = pull(other_target, Product_quality),
         affinity_to_brand = pull(affinity_to_brand, Product_quality))

head(brand_proportions)
```

```
## # A tibble: 6 x 4
##   Brand_name Target_brand other_target affinity_to_brand
##   <chr>             <dbl>        <dbl>             <dbl>
## 1 Tyrrells         0.0296       0.0242              1.22
## 2 Twisties         0.0433       0.0356              1.22
## 3 Tostitos         0.0426       0.0354              1.20
## 4 Pringles         0.112        0.0936              1.20
## 5 Kettle           0.186        0.156               1.19
## 6 Old              0.0416       0.0353              1.18
```

```
tail(brand_proportions)
```

```
## # A tibble: 6 x 4
##   Brand_name Target_brand other_target affinity_to_brand
##   <chr>             <dbl>        <dbl>             <dbl>
## 1 Cheetos         0.00753      0.0116              0.651
## 2 GrnWves         0.00337      0.00545             0.618
## 3 CCs             0.0105       0.0173              0.604
## 4 Sunbites        0.00595      0.0112              0.531
## 5 Woolworths      0.0282       0.0562              0.501
## 6 Burger          0.00274      0.00601             0.456
```

Tyrrells, the affinity score is 1.2235936. This means that the target segment is 1.22 times more likely to buy Tyrrells compared to the other segment. Similarly, for Burger, the affinity score is 0.4563263, which means that the target segment is 54% less likely to buy Burger compared to the other segment.

Calculate the Pack size The target customer will most likely and less likely buy compared to other customer.

```
# This is the code to calculate the Pack size target customer  buy the most, the purpose of this, is to

quality_pack_size <- main %>%
```

```
  group_by(Pack_size) %>%
  summarise(Target_packSize = sum(PROD_QTY) / Quality_main) %>%
arrange(desc(Target_packSize))

# This is for the other segments.

other_packSIze <- others %>%
  group_by(Pack_size) %>%
  summarise(Other_packsizes = sum(PROD_QTY) / other_quality) %>%
  arrange(desc(Other_packsizes))

# This code is to identify how likely Target customer will most likely buy a particular pack size compa

Affinity_to_packsize <- full_join(quality_pack_size, other_packSIze, by = "Pack_size") %>%
  mutate(Affinity_score = Target_packSize / Other_packsizes) %>%
  arrange(desc(Affinity_score))

Affinity_to_packsize <- Affinity_to_packsize %>%
  mutate(Target_packSize = pull(Target_packSize, Product_quality),
         Other_packsizes = pull(Other_packsizes, Product_quality),
         Affinity_score = pull(Affinity_score, Product_quality))

write.csv(Affinity_to_packsize, "Affinity_to_packsize.csv")

head(Affinity_to_packsize)
```

```
## # A tibble: 6 x 4
##   Pack_size Target_packSize Other_packsizes Affinity_score
##       <dbl>           <dbl>           <dbl>          <dbl>
## 1       270          0.0298          0.0238           1.26
## 2       380          0.0302          0.0240           1.26
## 3       330          0.0575          0.0464           1.24
## 4       134          0.112           0.0936           1.20
## 5       110          0.0997          0.0850           1.17
## 6       210          0.0273          0.0240           1.14
```

```
tail(Affinity_to_packsize)
```

```
## # A tibble: 6 x 4
##   Pack_size Target_packSize Other_packsizes Affinity_score
##       <dbl>           <dbl>           <dbl>          <dbl>
## 1       160         0.00601         0.0109           0.552
## 2        90         0.00595         0.0112           0.531
## 3       125         0.00282         0.00538          0.525
## 4        70         0.00285         0.00565          0.504
## 5       200         0.00841         0.0168           0.501
## 6       220         0.00274         0.00601          0.456
```

Target segment are 25% most likely to buy 270g pack size and 55% less likely to buy 220g pack size compared to other segement .

```
 packs270 <- main %>%
  filter(Pack_size == 270) %>%
           distinct(PROD_NAME)

packs270
```

```
##                   PROD_NAME
## 1 Twisties Cheese     270g
## 2     Twisties Chicken270g
```

This is the only brand offering 270g ,might be a driving factor for sales for these brands.

#conclusions

The mainstream Young singles and couples segment generates the most revenue for the company, they are 22% most likely to buy Tyrells and 27% most likely to buy 270g pack size, which means twisties brand will be most likely bought by this segment because they prefer 270g pack size.