Data Exploration and Analysis - Task 1

Siddharth Gada

```
## ## Chunk: Loading Libraries
```

```r
library(data.table)
library(ggplot2)
library(ggmosaic)
library(readr)
library(readxl)
library(stringr)
library(dplyr)
library(lubridate)
library(scales)
options(scipen = 999)  # Turn off scientific notation
```

```
## ## Chunk: Loading Datasets
```

```r
transactionData <- read_excel(paste0(
  "C:/Users/gadas/OneDrive/Desktop/",
  "Classes Outside UNT/Forage Project/",
  "Quantium Data Analytics/QVI_transaction_data.xlsx"))

customerData <- read.csv(paste0(
  "C:/Users/gadas/OneDrive/Desktop/",
  "Classes Outside UNT/Forage Project/",
  "Quantium Data Analytics/QVI_purchase_behaviour.csv"))

head(transactionData)
```

```
## # A tibble: 6 x 8
##    DATE STORE_NBR LYLTY_CAR~1 TXN_ID PROD_~2 PROD_~3 PROD_~4
##   <dbl>     <dbl>       <dbl>  <dbl>   <dbl> <chr>     <dbl>
## 1 43390         1        1000      1       5 Natura~       2
## 2 43599         1        1307    348      66 CCs Na~       3
## 3 43605         1        1343    383      61 Smiths~       2
## 4 43329         2        2373    974      69 Smiths~       5
## 5 43330         2        2426   1038     108 Kettle~       3
## 6 43604         4        4074   2982      57 Old El~       1
## # ... with 1 more variable: TOT_SALES <dbl>, and
## #   abbreviated variable names 1: LYLTY_CARD_NBR,
## #   2: PROD_NBR, 3: PROD_NAME, 4: PROD_QTY
```

```r
head(customerData)
```

```
##   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
```

```
## ## Data Exploration Start
```

```
## ## Examining Transaction Data
```

```r
# Examine date variable from transaction data
summary(transactionData$DATE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   43282   43373   43464   43464   43555   43646
```

```r
# Need to convert date variable from numeric to date format
transactionData$DATE_Converted <- as.Date(transactionData$DATE,
                                          origin = "1899-12-30")

# Remove DATE column which is not converted
transactionData$DATE <- NULL

# Examine PROD_NAME
summary(transactionData$PROD_NAME)
```

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

```r
# Split all product names into words
words <- unlist(strsplit(transactionData$PROD_NAME, "\\s+"))

# Keep only words with letters a-z or A-Z (remove digits/special chars)
clean_words <- words[!grepl("[^A-Za-z]", words)]

# Create frequency table
word_freq <- data.table(word = clean_words)[, .N, by = word]
setorder(word_freq, -N)

# Remove salsa products because we are only interested in keeping the
# data related to chips sales
transactionData <- subset(transactionData, !grepl("salsa", tolower(PROD_NAME)))

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

```
##    STORE_NBR     LYLTY_CARD_NBR        TXN_ID
##  Min.   :  1.0   Min.   :   1000   Min.   :      1
##  1st Qu.: 70.0   1st Qu.:  70015   1st Qu.:  67569
```

```
##  Median :130.0   Median : 130367   Median : 135183
##  Mean   :135.1   Mean   : 135531   Mean   : 135131
##  3rd Qu.:203.0   3rd Qu.: 203084   3rd Qu.: 202654
##  Max.   :272.0   Max.   :2373711   Max.   :2415841
##     PROD_NBR        PROD_NAME            PROD_QTY
##  Min.   :  1.00   Length:246742      Min.   :  1.000
##  1st Qu.: 26.00   Class :character   1st Qu.:  2.000
##  Median : 53.00   Mode  :character   Median :  2.000
##  Mean   : 56.35                      Mean   :  1.908
##  3rd Qu.: 87.00                      3rd Qu.:  2.000
##  Max.   :114.00                      Max.   :200.000
##    TOT_SALES       DATE_Converted
##  Min.   :  1.700   Min.   :2018-07-01
##  1st Qu.:  5.800   1st Qu.:2018-09-30
##  Median :  7.400   Median :2018-12-30
##  Mean   :  7.321   Mean   :2018-12-30
##  3rd Qu.:  8.800   3rd Qu.:2019-03-31
##  Max.   :650.000   Max.   :2019-06-30
```

```
data.frame(
  Column = names(transactionData),
  Total_Obs = nrow(transactionData),
  Non_Missing = colSums(!is.na(transactionData)),
  Missing = colSums(is.na(transactionData))
)
```

```
##                          Column Total_Obs Non_Missing Missing
## STORE_NBR             STORE_NBR    246742      246742       0
## LYLTY_CARD_NBR LYLTY_CARD_NBR    246742      246742       0
## TXN_ID                   TXN_ID    246742      246742       0
## PROD_NBR               PROD_NBR    246742      246742       0
## PROD_NAME             PROD_NAME    246742      246742       0
## PROD_QTY               PROD_QTY    246742      246742       0
## TOT_SALES             TOT_SALES    246742      246742       0
## DATE_Converted DATE_Converted    246742      246742       0
```

```
# There are no nulls in the columns but product quantity appears to have an outlier
# Filter the dataset to find the outlier in product quantity
# where 200 chip packets were bought in one transaction
setDT(transactionData)
transactionData[PROD_QTY==200]
```

```
##    STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 1:       226         226000 226201        4
## 2:       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
##    DATE_Converted
## 1:     2018-08-19
## 2:     2019-05-20
```

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

# Checking if the customer has had other transactions
transactionData[LYLTY_CARD_NBR==226000]
```

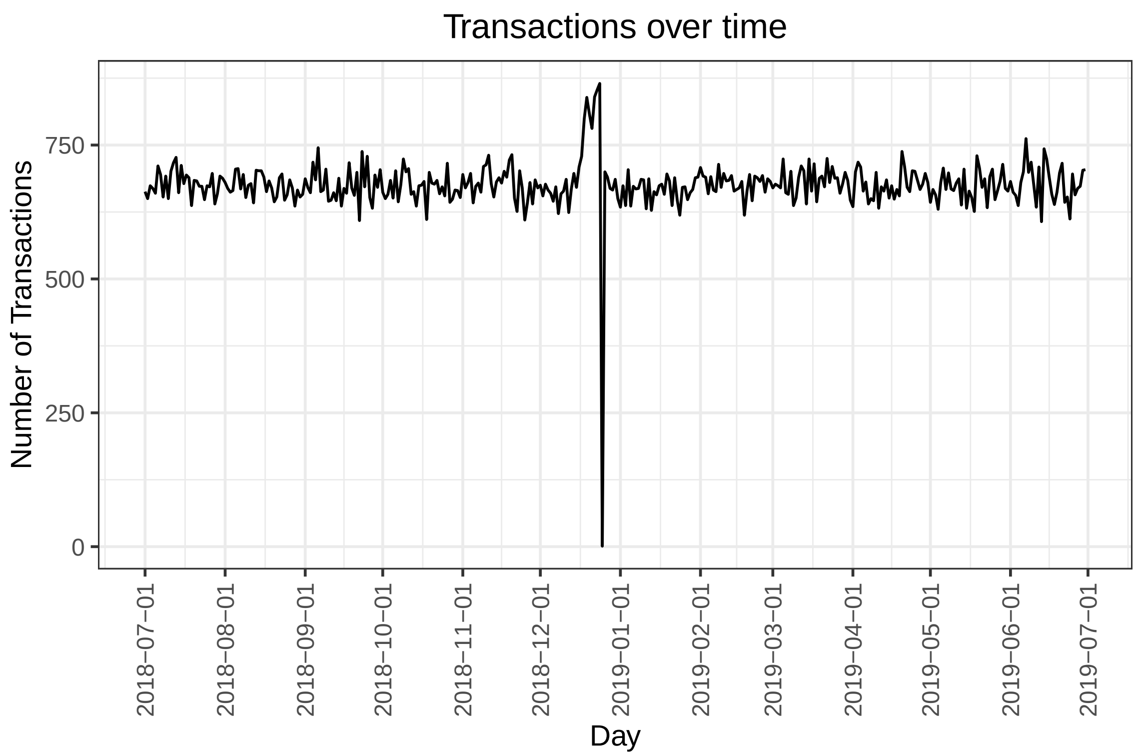
```
##    STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
## 1:       226         226000 226201        4
## 2:       226         226000 226210        4
##                           PROD_NAME PROD_QTY TOT_SALES
## 1: Dorito Corn Chp     Supreme 380g      200       650
## 2: Dorito Corn Chp     Supreme 380g 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

```r
# 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.
transactionData <- transactionData[LYLTY_CARD_NBR != 226000]

# Checking if all transactions from the card number have been removed
transactionData[LYLTY_CARD_NBR==226000]
```

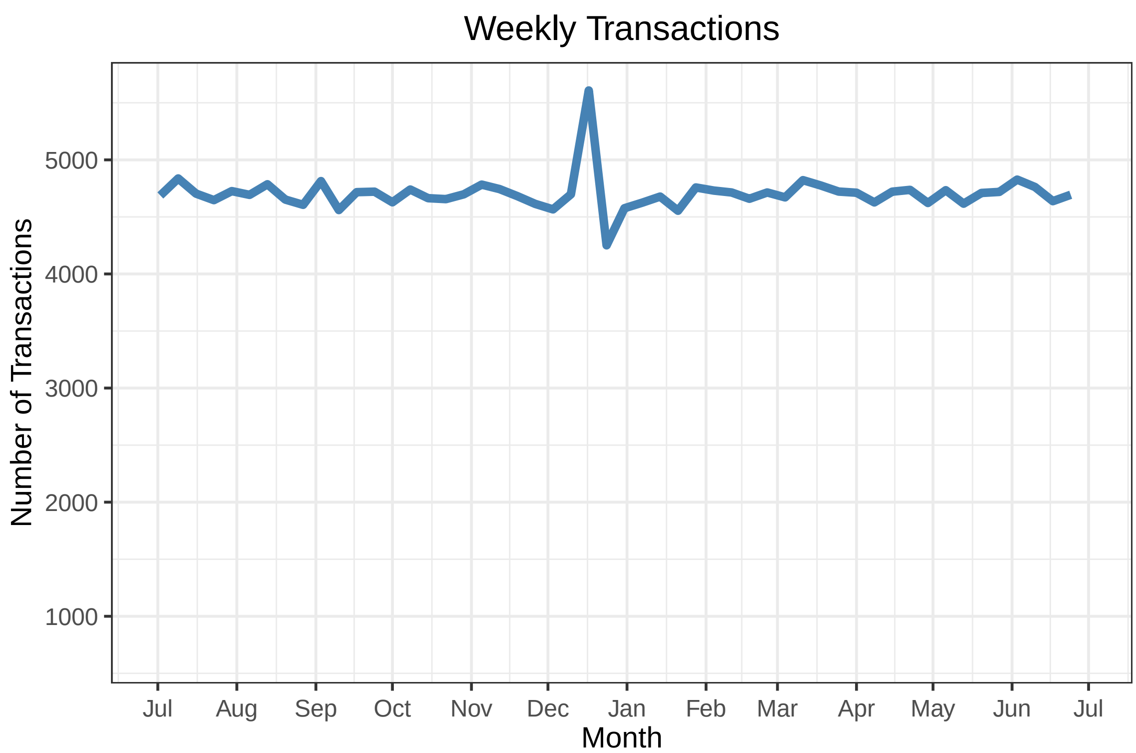
```
## Empty data.table (0 rows and 8 cols): STORE_NBR,LYLTY_CARD_NBR,TXN_ID,PROD_NBR,PROD_NAME,PROD_QTY...
```

```r
# Count the number of transactions by date
transactions_by_date <- transactionData %>%
  group_by(DATE_Converted) %>%
  summarise(Transaction_Count = n())

# There's only 364 rows, meaning only 364 dates which indicates a missing date.
# Create a full 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.
ALLDATES <- data.table(DATE_Converted = seq(as.Date("2018-07-01"),
                                            as.Date("2019-06-30"), by = "day"))

# Join all_dates with transactions_by_date (left join)
fullTransactionData <- merge(ALLDATES, transactionData, by = "DATE_Converted",
                             all.x = TRUE)

# Count the number of transactions by date to see if the missing date was added
transactions_by_date_1 <- fullTransactionData %>%
  group_by(DATE_Converted) %>%
  summarise(Transaction_Count = n())

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

# Plot transactions over time
```

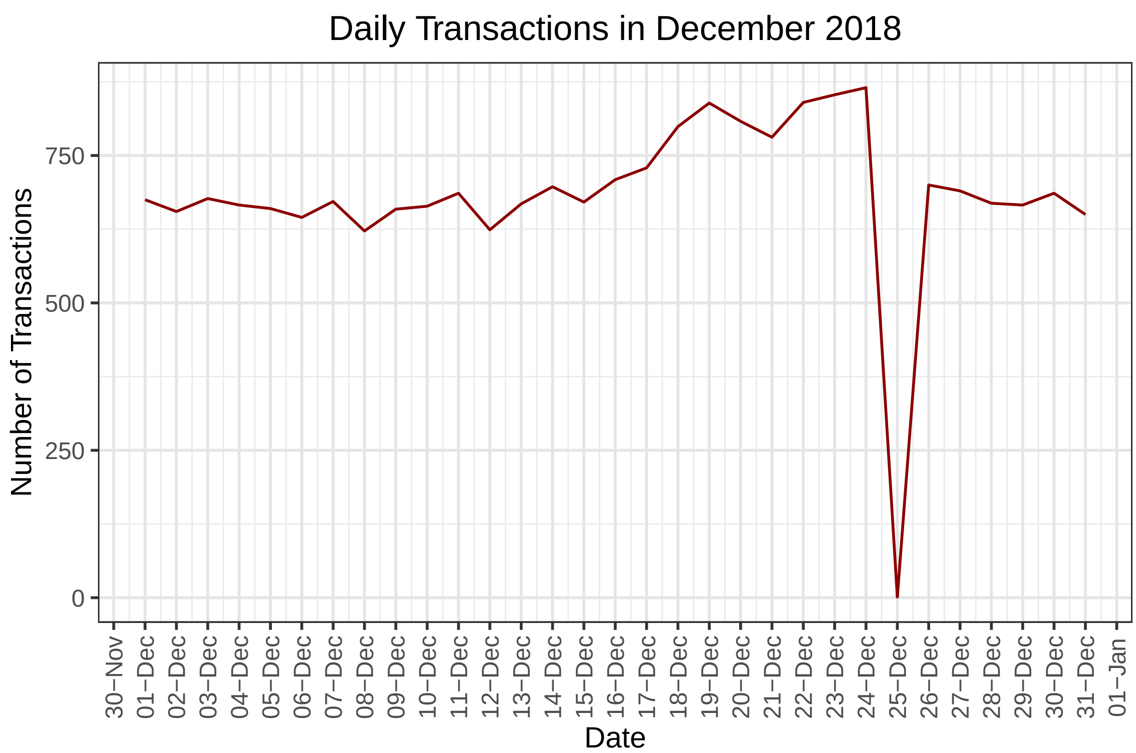
```r
ggplot(transactions_by_date_1, aes(x = DATE_Converted, y = Transaction_Count))+
  geom_line() +
  labs(x = "Day", y = "Number of Transactions", title = "Transactions over time")+
  scale_x_date(breaks = "1 month")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




```r
# We can see that there is an increase in purchases in December and a break
# in late December. Let's zoom in on this.

# Weekly transaction plot for Presentation
fullTransactionData$WeekStart <- floor_date(fullTransactionData$DATE_Converted,
                             unit = "week", week_start = 1)

weekly_transactions <- fullTransactionData[, .(
  weekly_transactions = uniqueN(TXN_ID)  # Count of unique transactions
), by = WeekStart][order(WeekStart)]

ggplot(weekly_transactions, aes(x = WeekStart, y = weekly_transactions))+
  geom_line(color = "steelblue", size = 1.5) +
  labs(x = "Month", y = "Number of Transactions",
       title = "Weekly Transactions") +
  scale_x_date(date_breaks = "1 month", date_labels = "%b",
               limits = as.Date(c("2018-07-01", "2019-06-30")))+
  theme(axis.text.x = element_text(vjust = 0.5))
```




```
fullTransactionData$WeekStart <- NULL
weekly_transactions <- NULL

# Filter to December and look at individual days
# Filter for December 2018
setDT(transactions_by_date_1)
december_data <- transactions_by_date_1[
  transactions_by_date_1$DATE_Converted >= as.Date("2018-12-01")
  & transactions_by_date_1$DATE_Converted <= as.Date("2018-12-31")
]

# Plot daily transactions for December
ggplot(december_data, aes(x = DATE_Converted, y = Transaction_Count))+
  geom_line(color = "darkred")+
  labs(
    x = "Date",
    y = "Number of Transactions",
    title = "Daily Transactions in December 2018"
  ) +
  scale_x_date(
    date_breaks = "1 day",
    date_labels = "%d-%b"
  ) +
  theme(
    axis.text.x = element_text(angle = 90, vjust = 0.5),
    panel.background = element_blank(),
    panel.grid.major = element_line(color = "grey90")
  )
```

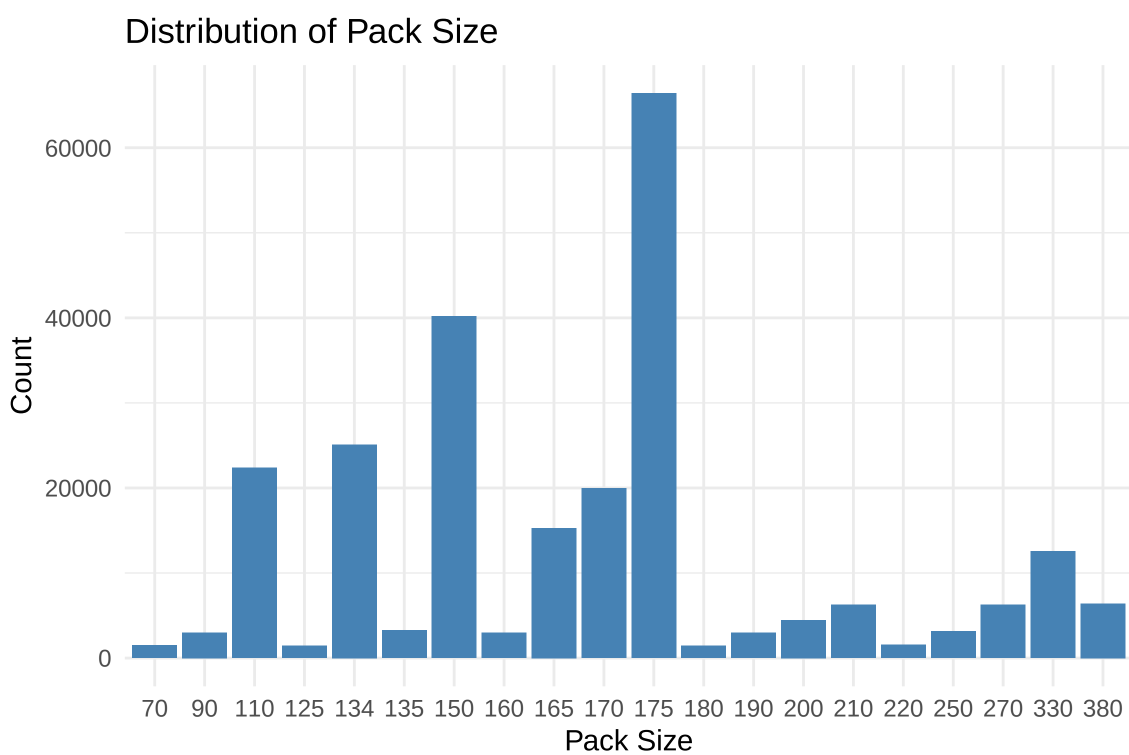



```r
# We can see that the increase in sales occurs in the lead-up to Christmas and
# that there are zero sales on Christmas day itself which was the missing date.
# This is due to shops being closed on Christmas day.

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

# Checking 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 40203
##  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:        330 12540
## 20:        380  6416
```

```r
## The largest size is 380g and the smallest size is 70g - seems sensible

# Histogram of PACK_SIZE
# Treat PACK_SIZE as a factor (categorical)
ggplot(transactionData, aes(x = factor(PACK_SIZE)))+
  geom_bar(fill = "steelblue")+
  labs(
    x = "Pack Size",
    y = "Count",
    title = "Distribution of Pack Size"
  )+
  theme_minimal()
```




```r
# Creating a Brand variable
# We can work this out by taking the first word that is in PROD_NAME
transactionData[, Brand := ifelse(is.na(PROD_NAME), NA,
                                  tstrsplit(PROD_NAME, " ")[[1]])]
# Finding out unique Brand names
unique(transactionData$Brand)
```

```
##  [1] "Natural"    "CCs"        "Smiths"     "Kettle"
##  [5] "Grain"      "Doritos"    "Twisties"   "WW"
```

```
##  [9] "Thins"      "Burger"     "NCC"        "Cheezels"
## [13] "Infzns"     "Red"        "Pringles"   "Dorito"
## [17] "Infuzions"  "Smith"      "GrnWves"    "Tyrrells"
## [21] "Cobs"       "French"     "RRD"        "Tostitos"
## [25] "Cheetos"    "Woolworths" "Snbts"      "Sunbites"
```

```r
# Clean Brand names
transactionData[Brand == "Red", Brand := "RRD"]
transactionData[Brand == "Smith", Brand := "Smiths"]
transactionData[Brand == "Infzns", Brand := "Infuzions"]
transactionData[Brand == "Snbts", Brand := "Sunbites"]
transactionData[Brand == "WW", Brand := "Woolworths"]
transactionData[Brand == "NCC", Brand := "Natural"]
transactionData[Brand == "Dorito", Brand := "Doritos"]
transactionData[Brand == "Grain", Brand := "GrnWves"]

# Checking if any more discrepancies in names
unique(transactionData$Brand)
```

```
##  [1] "Natural"    "CCs"        "Smiths"     "Kettle"
##  [5] "GrnWves"    "Doritos"    "Twisties"   "Woolworths"
##  [9] "Thins"      "Burger"     "Cheezels"   "Infuzions"
## [13] "RRD"        "Pringles"   "Tyrrells"   "Cobs"
## [17] "French"     "Tostitos"   "Cheetos"    "Sunbites"
```

```r
# Clean each PROD_NAME by removing words that contain non-letter characters
transactionData[, PROD_NAME := sapply(strsplit(PROD_NAME, "\\s+"),
                                      function(words) {
  clean_words <- words[!grepl("[^A-Za-z]", words)]
  paste(clean_words, collapse = " ")
})]
```

```
## ## Examining Customer Data
```

```r
# Making sure there are no missing values
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
```

```r
length(customerData$LYLTY_CARD_NBR)
```

```
## [1] 72637
```

```r
length(unique(transactionData$LYLTY_CARD_NBR))
```

```
## [1] 71287
```

```r
# Performing a left join because we only want matches from 'transactionData'
# since 'customerData' has customer 226000 which we excluded from our dataset
# because he buys for commercial use
# This is the reason we did not want to delete the customers that bought SALSA
# so that we get exact matches from the two datasets.

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

length(unique(data$LYLTY_CARD_NBR))
```

```
## [1] 71287
```

```r
# Check if some customers were not matched on by checking for nulls.
sum(is.na(data$LYLTY_CARD_NBR))
```

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

```r
sum(is.na(data$LIFESTAGE))
```

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

```r
sum(is.na(data$PREMIUM_CUSTOMER))
```

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

```r
# Code to save dataset as a csv
write.csv(data,
          file = paste0(
  "C:/Users/gadas/OneDrive/Desktop/",
  "Classes Outside UNT/Forage Project/",
  "Quantium Data Analytics/QVI_data.csv"))
```

```
## ## Data analysis on customer segments Start
```

```
## ## Chunk: Total sales by Loyalty Card Number
```

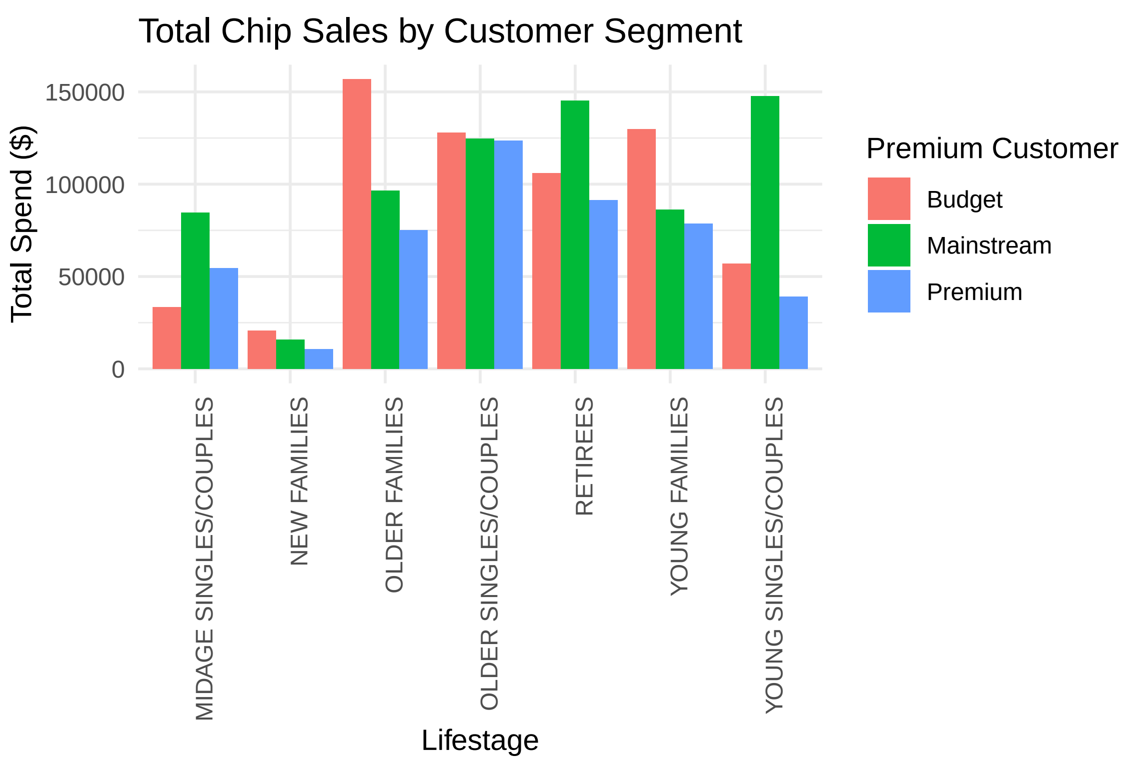
```r
data[, .(Total_Spend = sum(TOT_SALES)), by = .(LYLTY_CARD_NBR)][order(-Total_Spend)]
```

```
##        LYLTY_CARD_NBR Total_Spend
##     1:         230078       138.6
##     2:          58361       124.8
##     3:          63197       122.6
##     4:         162039       121.6
##     5:         179228       120.8
```

```
##    ---
## 71283:         268247         1.7
## 71284:         268313         1.7
## 71285:         268315         1.7
## 71286:         268390         1.7
## 71287:         268476         1.7
```

```
## ## Chunk: Total sales by Lifestage and Premium Customer
```

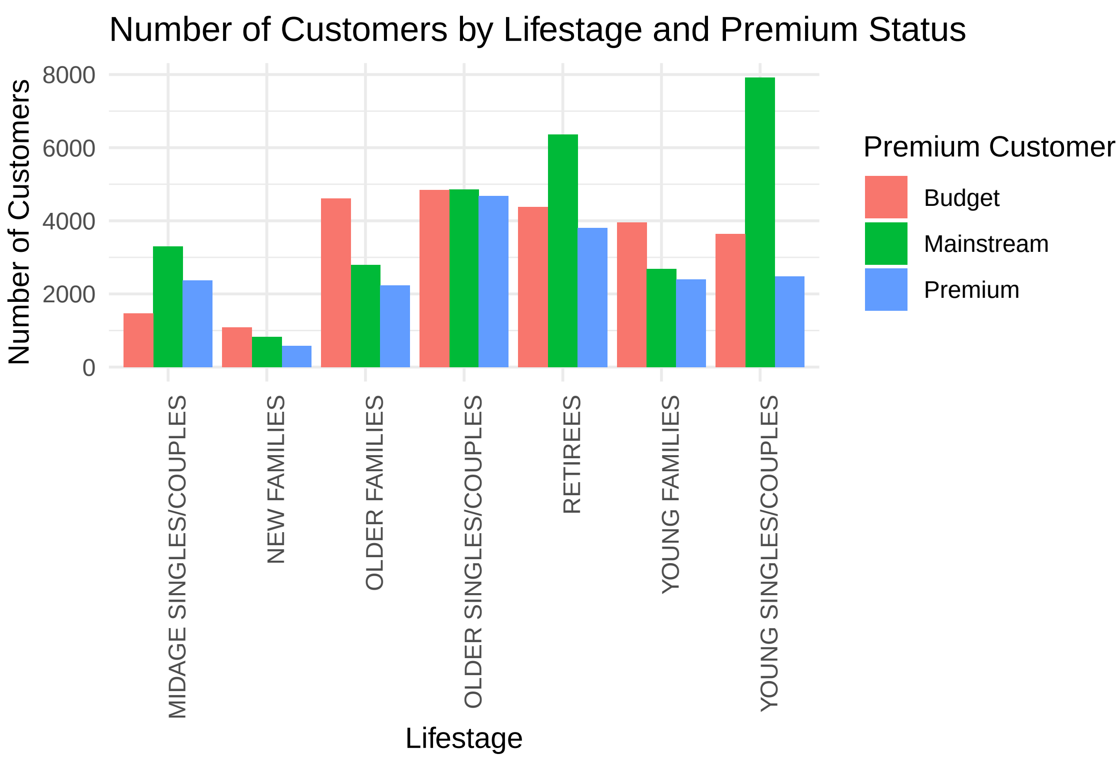
```
# LIFESTAGE: Customer attribute that identifies whether a customer has a family
# or not and what point in life they are at
# PREMIUM_CUSTOMER: Customer segmentation used to differentiate shoppers by the
# price point of products they buy and the types of products they buy.
sales_by_segment <- data[, .(Total_Spend = sum(TOT_SALES)),
                         by = .(PREMIUM_CUSTOMER, LIFESTAGE)][order(-Total_Spend)]
# Plot the sales split
ggplot(sales_by_segment, aes(x = LIFESTAGE, y = Total_Spend, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Total Chip Sales by Customer Segment",
       x = "Lifestage", y = "Total Spend ($)",
       fill = "Premium Customer")+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




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

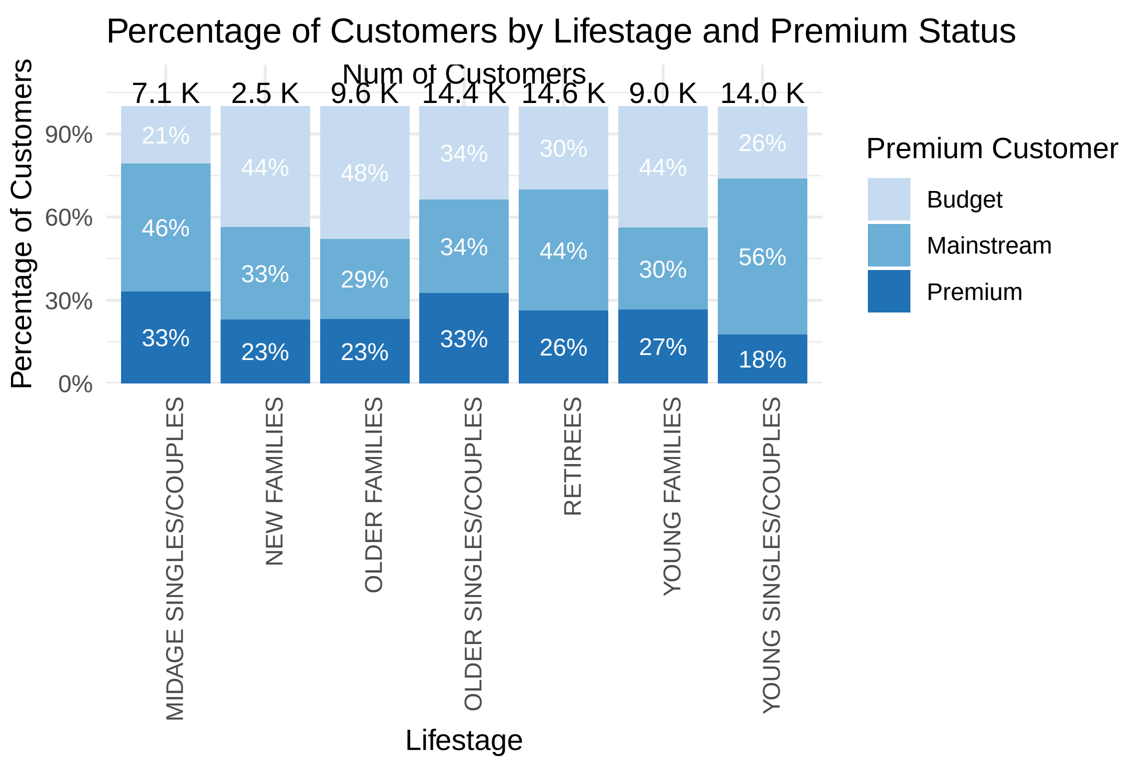
```
## ## Chunk: How many customers are in each segment
```

```r
# How many customers are in each segment
customers_by_segment <- data[, uniqueN(LYLTY_CARD_NBR),
                             by = .(LIFESTAGE, PREMIUM_CUSTOMER)]
setnames(customers_by_segment, "V1", "N")
# Plot number of customers
ggplot(customers_by_segment, aes(x = LIFESTAGE, y = N, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Number of Customers by Lifestage and Premium Status",
       x = "Lifestage", y = "Number of Customers",
       fill = "Premium Customer")+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```r
# 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.

# Percentage of Customers by Lifestage and Premium Status plot for Presentation
customers_by_segment_pct <- customers_by_segment %>%
  group_by(LIFESTAGE) %>%
  mutate(pct = N / sum(N),
         pct = pct / sum(pct),
         total_N = sum(N)) %>%
  ungroup()

ggplot(customers_by_segment_pct, aes(x = LIFESTAGE, y = pct, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "stack") +
  geom_text(aes(label = scales::percent(pct, accuracy = 1)),
```

```r
           position = position_stack(vjust = 0.5),
           size = 3, color = "white") +
  geom_text(data = customers_by_segment_pct %>%
              distinct(LIFESTAGE, total_N),
            aes(x = LIFESTAGE, y = 1.05,
                label = paste(scales::number(total_N / 1000, accuracy = 0.1), "K")),
            inherit.aes = FALSE) +
  annotate("text", x = length(unique(customers_by_segment_pct$LIFESTAGE)) / 2 + 0.5,
           y = 1.12, label = "Num of Customers") +
  labs(title = "Percentage of Customers by Lifestage and Premium Status",
       x = "Lifestage", y = "Percentage of Customers",
       fill = "Premium Customer")+
  scale_y_continuous(labels = percent_format(), limits = c(0, 1.15), expand = c(0,0)) +
  scale_fill_manual(values = c("Budget" = "#c6dbef",     # light blue
                               "Mainstream" = "#6baed6", # medium blue
                               "Premium" = "#2171b5")) + # dark blue
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

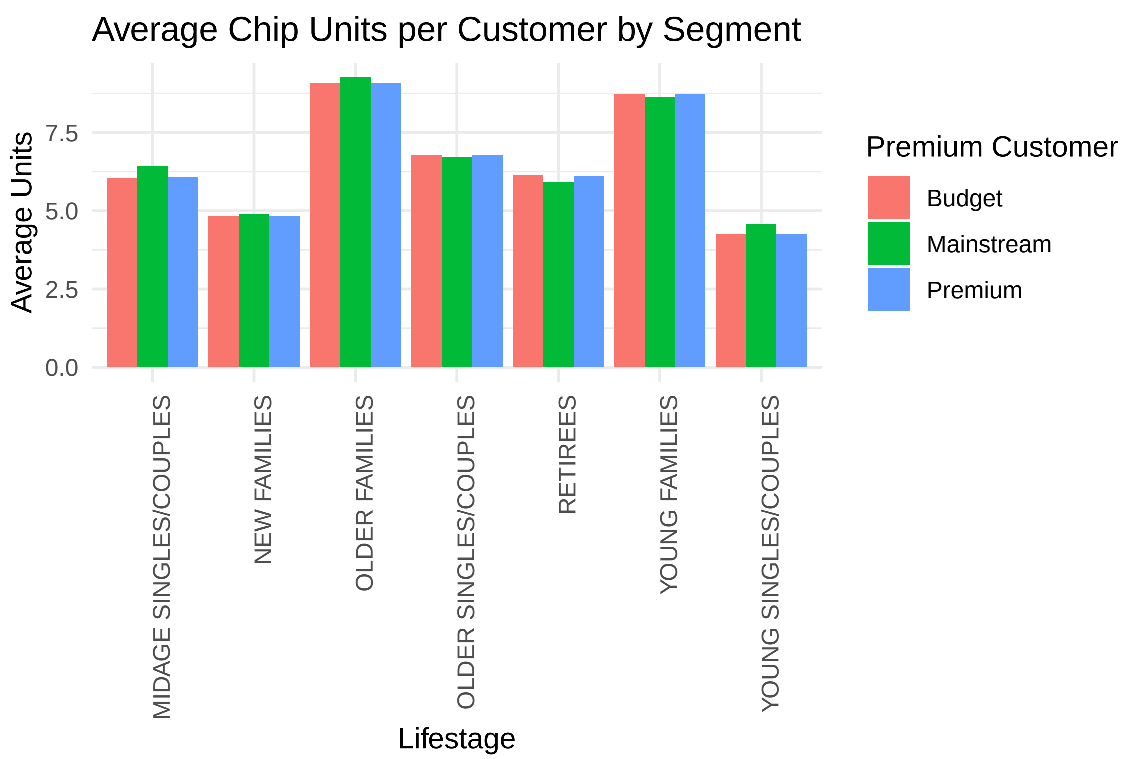



```r
customers_by_segment_pct <- NULL
```

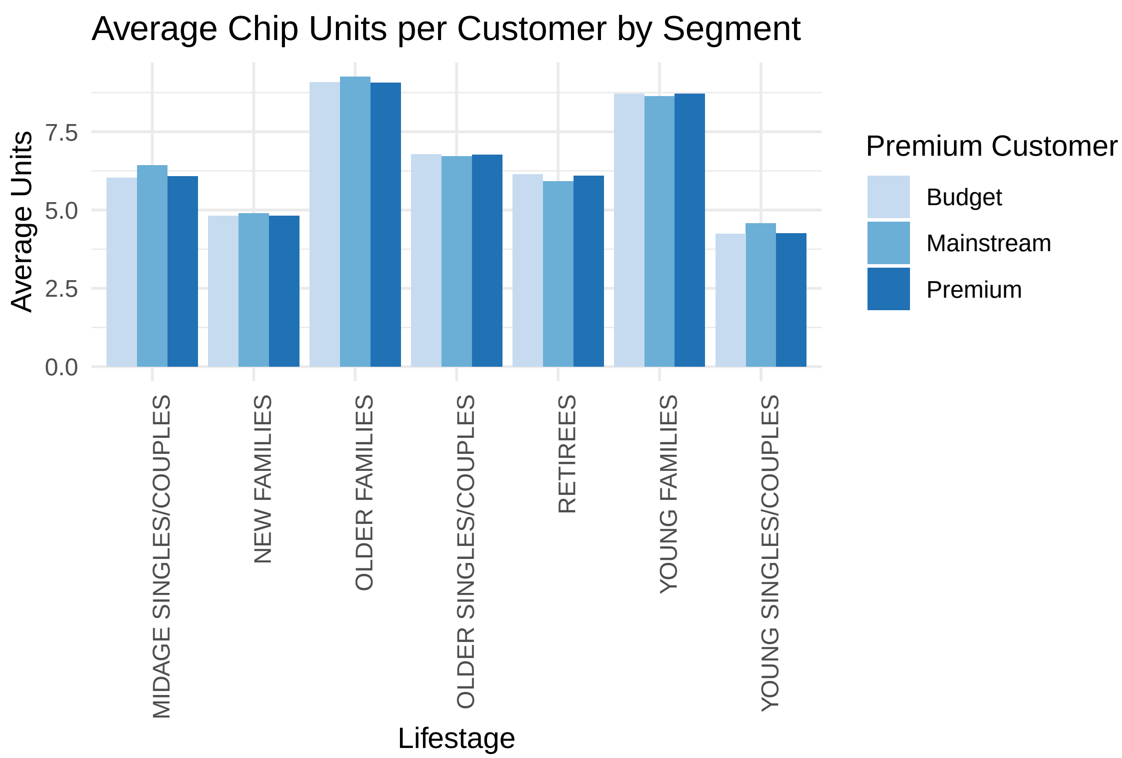
```
## ## Chunk: How many chips are bought per customer by segment
```

```r
avg_units_per_customer <- data[, .(Num_Chips_bought = sum(PROD_QTY),
         Unique_Customers = uniqueN(LYLTY_CARD_NBR)),
         by = .(PREMIUM_CUSTOMER, LIFESTAGE)][order(-Num_Chips_bought)]
# Compute average units per customer
```

```r
avg_units_per_customer[, Avg_Units := Num_Chips_bought / Unique_Customers]
# Plot
ggplot(avg_units_per_customer, aes(x = LIFESTAGE, y = Avg_Units, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Average Chip Units per Customer by Segment",
       x = "Lifestage", y = "Average Units",
       fill = "Premium Customer")+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

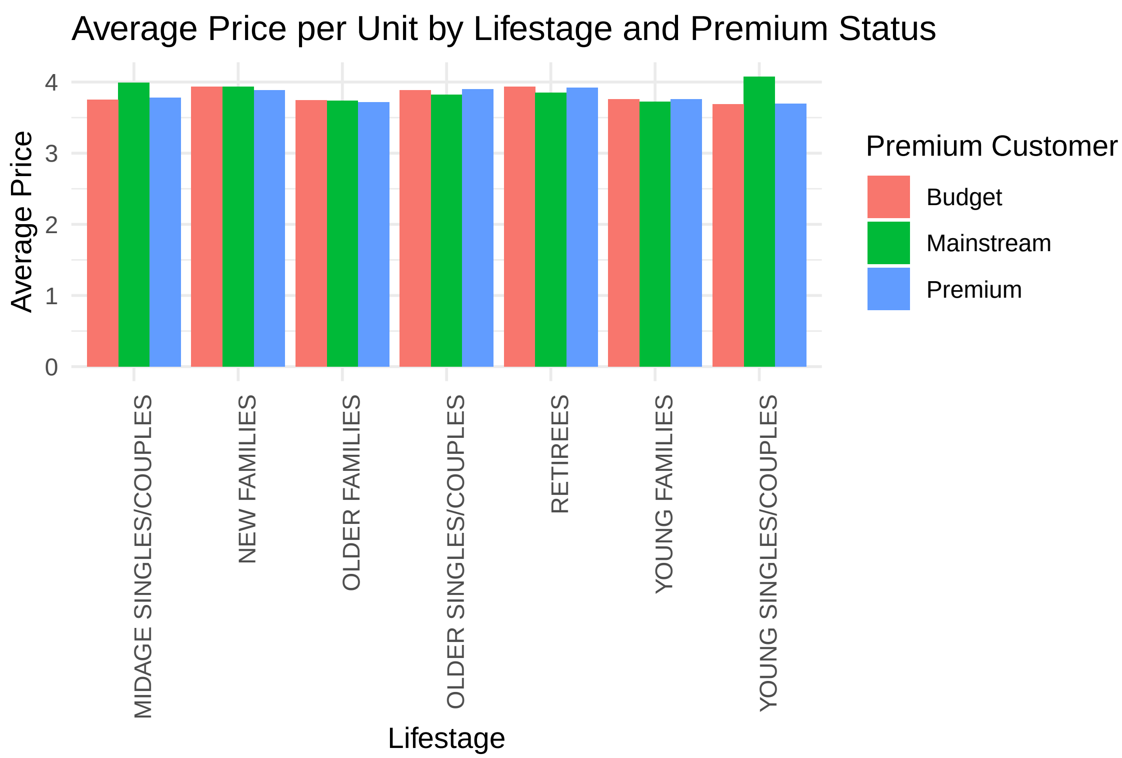



```r
# Older families and young families in general buy more chips per customer

# Plot for Presentation
ggplot(avg_units_per_customer, aes(x = LIFESTAGE, y = Avg_Units, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Average Chip Units per Customer by Segment",
       x = "Lifestage", y = "Average Units",
       fill = "Premium Customer")+
  scale_fill_manual(values = c("Budget" = "#c6dbef",     # light blue
                               "Mainstream" = "#6baed6", # medium blue
                               "Premium" = "#2171b5")) + # dark blue
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```
## ## Chunk: Calculate average price per unit by segment
```

```
avg_price_data <- data[, .(  Total_Sales = sum(TOT_SALES),
                             Total_Units = sum(PROD_QTY)),
                        by = .(LIFESTAGE, PREMIUM_CUSTOMER)]
# Compute average price per unit
avg_price_data[, Avg_Price := Total_Sales / Total_Units]
# Plot the results
ggplot(avg_price_data, aes(x = LIFESTAGE, y = Avg_Price, fill = PREMIUM_CUSTOMER))+
  geom_bar(stat = "identity", position = "dodge")+
  labs(title = "Average Price per Unit by Lifestage and Premium Status",
       x = "Lifestage", y = "Average Price",
       fill = "Premium Customer")+
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```




```r
# 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.
```

```
## ## Chunk: T-Tests
```

```r
# Perform an independent t-test between mainstream vs premium and budget
# midage and young singles and couples
# T-test for Budget vs. Mainstream midage and young singles and couples
t.test(TOT_SALES / PROD_QTY ~ PREMIUM_CUSTOMER,
       data = data,
       subset = LIFESTAGE %in% c("MIDAGE SINGLES/COUPLES", "YOUNG SINGLES/COUPLES")
       & PREMIUM_CUSTOMER %in% c("Budget", "Mainstream"))
```

```
##
##  Welch Two Sample t-test
##
## data:  TOT_SALES/PROD_QTY by PREMIUM_CUSTOMER
## t = -31.671, df = 23526, p-value <
## 0.00000000000000022
## alternative hypothesis: true difference in means between group Budget and group Mainstream is not equ
## 95 percent confidence interval:
##  -0.3738041 -0.3302324
## sample estimates:
##     mean in group Budget mean in group Mainstream
```

```
##                 3.687768                 4.039786
```

```r
# T-test for Budget vs. Premium midage and young singles and couples
t.test(TOT_SALES / PROD_QTY ~ PREMIUM_CUSTOMER,
       data = data,
       subset = LIFESTAGE %in% c("MIDAGE SINGLES/COUPLES", "YOUNG SINGLES/COUPLES")
       & PREMIUM_CUSTOMER %in% c("Budget", "Premium"))
```

```
##
##  Welch Two Sample t-test
##
## data:  TOT_SALES/PROD_QTY by PREMIUM_CUSTOMER
## t = -2.7694, df = 26724, p-value = 0.005619
## alternative hypothesis: true difference in means between group Budget and group Premium is not equal
## 95 percent confidence interval:
##  -0.06347549 -0.01086297
## sample estimates:
##  mean in group Budget mean in group Premium
##              3.687768              3.724937
```

```r
# The t-test results in a p-value of < 0.00000000000000022, 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.
```

```
## ## Chunk: # Find out if the Mainstream, Young singles/couples
##     customer segment tend to buy a particular brand of chips
```

```r
brand_segment <- data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER, Brand) %>%
  summarise(SegmentBrandQty = sum(PROD_QTY), .groups = "drop")
# Total quantity bought by each segment
segment_total <- data %>%
  group_by(LIFESTAGE, PREMIUM_CUSTOMER) %>%
  summarise(SegmentTotalQty = sum(PROD_QTY), .groups = "drop")
# Total quantity bought per brand
brand_total <- data %>%
  group_by(Brand) %>%
  summarise(BrandTotalQty = sum(PROD_QTY), .groups = "drop")
# Total quantity overall
total_qty <- sum(data$PROD_QTY)
# Merge data
affinity_data <- brand_segment %>%
  left_join(segment_total, by = c("LIFESTAGE", "PREMIUM_CUSTOMER")) %>%
  left_join(brand_total, by = "Brand") %>%
  mutate(
    SegmentBrandShare = SegmentBrandQty / SegmentTotalQty,
    BrandShareOverall = BrandTotalQty / total_qty,
    AffinityScore = SegmentBrandShare / BrandShareOverall
  )
# Focus on Mainstream Young Singles/Couples
affinity_data %>%
  filter(LIFESTAGE == "YOUNG SINGLES/COUPLES", PREMIUM_CUSTOMER == "Mainstream") %>%
```

```
  arrange(desc(AffinityScore)) %>%
  print (n=21)
```

```
## # A tibble: 20 x 9
##    LIFESTAGE   PREMI~1 Brand Segme~2 Segme~3 Brand~4 Segme~5
##    <chr>       <chr>   <chr>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 YOUNG SING~ Mainst~ Tyrr~    1143   36225   12298 0.0316
##  2 YOUNG SING~ Mainst~ Twis~    1673   36225   18118 0.0462
##  3 YOUNG SING~ Mainst~ Dori~    4447   36225   48331 0.123
##  4 YOUNG SING~ Mainst~ Kett~    7172   36225   79051 0.198
##  5 YOUNG SING~ Mainst~ Tost~    1645   36225   18134 0.0454
##  6 YOUNG SING~ Mainst~ Prin~    4326   36225   48019 0.119
##  7 YOUNG SING~ Mainst~ Cobs     1617   36225   18571 0.0446
##  8 YOUNG SING~ Mainst~ Infu~    2343   36225   27119 0.0647
##  9 YOUNG SING~ Mainst~ Thins    2187   36225   26929 0.0604
## 10 YOUNG SING~ Mainst~ GrnW~    1185   36225   14726 0.0327
## 11 YOUNG SING~ Mainst~ Chee~     651   36225    8747 0.0180
## 12 YOUNG SING~ Mainst~ Smit~    3491   36225   57582 0.0964
## 13 YOUNG SING~ Mainst~ Fren~     143   36225    2643 0.00395
## 14 YOUNG SING~ Mainst~ Chee~     291   36225    5530 0.00803
## 15 YOUNG SING~ Mainst~ RRD      1587   36225   30891 0.0438
## 16 YOUNG SING~ Mainst~ Natu~     710   36225   14106 0.0196
## 17 YOUNG SING~ Mainst~ CCs       405   36225    8609 0.0112
## 18 YOUNG SING~ Mainst~ Sunb~     230   36225    5692 0.00635
## 19 YOUNG SING~ Mainst~ Wool~     873   36225   22333 0.0241
## 20 YOUNG SING~ Mainst~ Burg~     106   36225    2970 0.00293
## # ... with 2 more variables: BrandShareOverall <dbl>,
## #   AffinityScore <dbl>, and abbreviated variable names
## #   1: PREMIUM_CUSTOMER, 2: SegmentBrandQty,
## #   3: SegmentTotalQty, 4: BrandTotalQty,
## #   5: SegmentBrandShare
```

```
# Mainstream, Young singles/couples customer segment prefer to buy chips from
# brands like Tyrrells, Twisties, Kettle, Tostitos and Old.
# Mainstream young singles/couples are 21% more likely to buy Tyrrells chips than
# the other customers.
# Mainstream young singles/couples are 54% less likely to purchase Burger Rings
# compared to the overall customer base.
```

```
## ## Chunk: Find out if our target segment tends to buy larger packs of chips
```

```
# Calculate average PACK_SIZE for Mainstream Young Singles/Couples
# Define your target and all other segment
target_segment <- data %>%
  filter(LIFESTAGE == "YOUNG SINGLES/COUPLES", PREMIUM_CUSTOMER == "Mainstream")
rest_segment <- data %>%
  filter(!(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream"))
# Calculate proportion of each pack size in target segment
target_pack_share <- target_segment %>%
  group_by(PACK_SIZE) %>%
  summarise(TargetQty = sum(PROD_QTY)) %>%
  mutate(TotalTarget = sum(TargetQty),
```

```r
         TargetProp = TargetQty / TotalTarget)
# Calculate proportion of each pack size in other segments
rest_pack_share <- rest_segment %>%
  group_by(PACK_SIZE) %>%
  summarise(RestQty = sum(PROD_QTY)) %>%
  mutate(TotalRest = sum(RestQty),
         RestProp = RestQty / TotalRest)
# Merge both and calculate affinity score
pack_affinity <- left_join(target_pack_share, rest_pack_share, by = "PACK_SIZE") %>%
  mutate(Affinity_Score = TargetProp / RestProp) %>%
  arrange(desc(Affinity_Score))
# View results
print(pack_affinity)
```

```
## # A tibble: 20 x 8
##    PACK_SIZE Targe~1 Total~2 Targe~3 RestQty Total~4 RestP~5
##        <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1       270    1153   36225 0.0318    10896  434174 0.0251
##  2       380    1165   36225 0.0322    11108  434174 0.0256
##  3       330    2220   36225 0.0613    21779  434174 0.0502
##  4       134    4326   36225 0.119     43693  434174 0.101
##  5       110    3850   36225 0.106     38985  434174 0.0898
##  6       210    1055   36225 0.0291    10907  434174 0.0251
##  7       135     535   36225 0.0148     5677  434174 0.0131
##  8       250     520   36225 0.0144     5549  434174 0.0128
##  9       170    2926   36225 0.0808    35162  434174 0.0810
## 10       150    5709   36225 0.158     70953  434174 0.163
## 11       175    9237   36225 0.255    117230  434174 0.270
## 12       165    2016   36225 0.0557    27035  434174 0.0623
## 13       190     271   36225 0.00748    5402  434174 0.0124
## 14       180     130   36225 0.00359    2634  434174 0.00607
## 15       160     232   36225 0.00640    5372  434174 0.0124
## 16        90     230   36225 0.00635    5462  434174 0.0126
## 17       125     109   36225 0.00301    2621  434174 0.00604
## 18       200     325   36225 0.00897    8100  434174 0.0187
## 19        70     110   36225 0.00304    2745  434174 0.00632
## 20       220     106   36225 0.00293    2864  434174 0.00660
## # ... with 1 more variable: Affinity_Score <dbl>, and
## #   abbreviated variable names 1: TargetQty,
## #   2: TotalTarget, 3: TargetProp, 4: TotalRest,
## #   5: RestProp
```

```r
# 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.

# Check which products are the closest to selling 270g chip packets for targetted marketting
data[PACK_SIZE == 270, unique(PROD_NAME)]
```

```
## [1] "Twisties Cheese" "Twisties"
```

```
## ## Conclusion
```

```
# 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.
```