Saugat_Task1

Saugat Pyakuryal

2025-06-23

```
# Load required libraries
library(data.table)
library(ggplot2)
library(readxl)
library(stringr)
```

```
# Load datasets
transaction_data <- as.data.table(read_excel("QVI_transaction_data.xlsx"))
customer_data <- fread("QVI_purchase_behaviour.csv")

# Check structure
str(transaction_data)
```

```
## Classes 'data.table' and 'data.frame':   264836 obs. of  8 variables:
##  $ DATE          : num  43390 43599 43605 43329 43330 ...
##  $ STORE_NBR     : num  1 1 1 2 2 4 4 4 5 7 ...
##  $ LYLTY_CARD_NBR: num  1000 1307 1343 2373 2426 ...
##  $ TXN_ID        : num  1 348 383 974 1038 ...
##  $ PROD_NBR      : num  5 66 61 69 108 57 16 24 42 52 ...
##  $ PROD_NAME     : chr  "Natural Chip        Compny SeaSalt175g" "CCs Nacho Cheese    175g" "Smiths C
##  $ PROD_QTY      : num  2 3 2 5 3 1 1 1 1 2 ...
##  $ TOT_SALES     : num  6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
head(transaction_data)
```

```
##     DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##    <num>     <num>          <num>  <num>    <num>
## 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
##                                       <char>    <num>     <num>
## 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
```

```r
# Check summary of customer data
str(customer_data)
```

```
## Classes 'data.table' and 'data.frame':   72637 obs. of  3 variables:
##  $ LYLTY_CARD_NBR  : int  1000 1002 1003 1004 1005 1007 1009 1010 1011 1012 ...
##  $ LIFESTAGE       : chr  "YOUNG SINGLES/COUPLES" "YOUNG SINGLES/COUPLES" "YOUNG FAMILIES" "OLDER SI
##  $ PREMIUM_CUSTOMER: chr  "Premium" "Mainstream" "Budget" "Mainstream" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```r
head(customer_data)
```

```
##    LYLTY_CARD_NBR              LIFESTAGE PREMIUM_CUSTOMER
##             <int>                 <char>           <char>
## 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
```

```r
# Convert DATE to proper Date format
transaction_data[, DATE := as.Date(DATE, origin = "1899-12-30")]

# Remove salsa products from PROD_NAME
transaction_data <- transaction_data[!grepl("salsa", tolower(PROD_NAME))]

# View outliers
transaction_data[PROD_QTY > 100]
```

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

```r
# Save loyalty ID of outlier customer
outlier_ids <- transaction_data[PROD_QTY > 100, unique(LYLTY_CARD_NBR)]
print(outlier_ids)
```

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

```r
# Remove outliers
transaction_data <- transaction_data[!LYLTY_CARD_NBR %in% outlier_ids]
```

```r
# Extract pack size from product name using regex
transaction_data[, PACK_SIZE := as.numeric(str_extract(PROD_NAME, "\\d+"))]
```

```r
# Extract brand
transaction_data[, BRAND := tstrsplit(PROD_NAME, " ")[[1]]]

# Clean up known brand aliasing
transaction_data[BRAND == "RED", BRAND := "RRD"]
transaction_data[BRAND == "SNB", BRAND := "SUNBITES"]
transaction_data[BRAND == "WW", BRAND := "WOOLWORTHS"]
transaction_data[BRAND == "INFZ", BRAND := "INFUZIONS"]

# Extract PACK_SIZE from product name
transaction_data[, PACK_SIZE := as.numeric(str_extract(PROD_NAME, "\\d+"))]

# Show how many transactions occurred for each pack size
transaction_data[, .N, by = PACK_SIZE][order(PACK_SIZE)]
```

```
##     PACK_SIZE     N
##         <num> <int>
##  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
##     PACK_SIZE     N
```

```r
# Extract brand name as the first word in PROD_NAME
transaction_data[, BRAND := tstrsplit(PROD_NAME, " ")[[1]]]

# inspect unique brand names
unique(transaction_data$BRAND)
```

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

```r
# Clean common brand aliases for consistency
transaction_data[BRAND == "RED", BRAND := "RRD"]
transaction_data[BRAND == "SNB", BRAND := "SUNBITES"]
transaction_data[BRAND == "WW", BRAND := "WOOLWORTHS"]
transaction_data[BRAND == "INFZ", BRAND := "INFUZIONS"]

# Merge customer attributes into the transaction data
merged_data <- merge(transaction_data, customer_data, by = "LYLTY_CARD_NBR", all.x = TRUE)
```

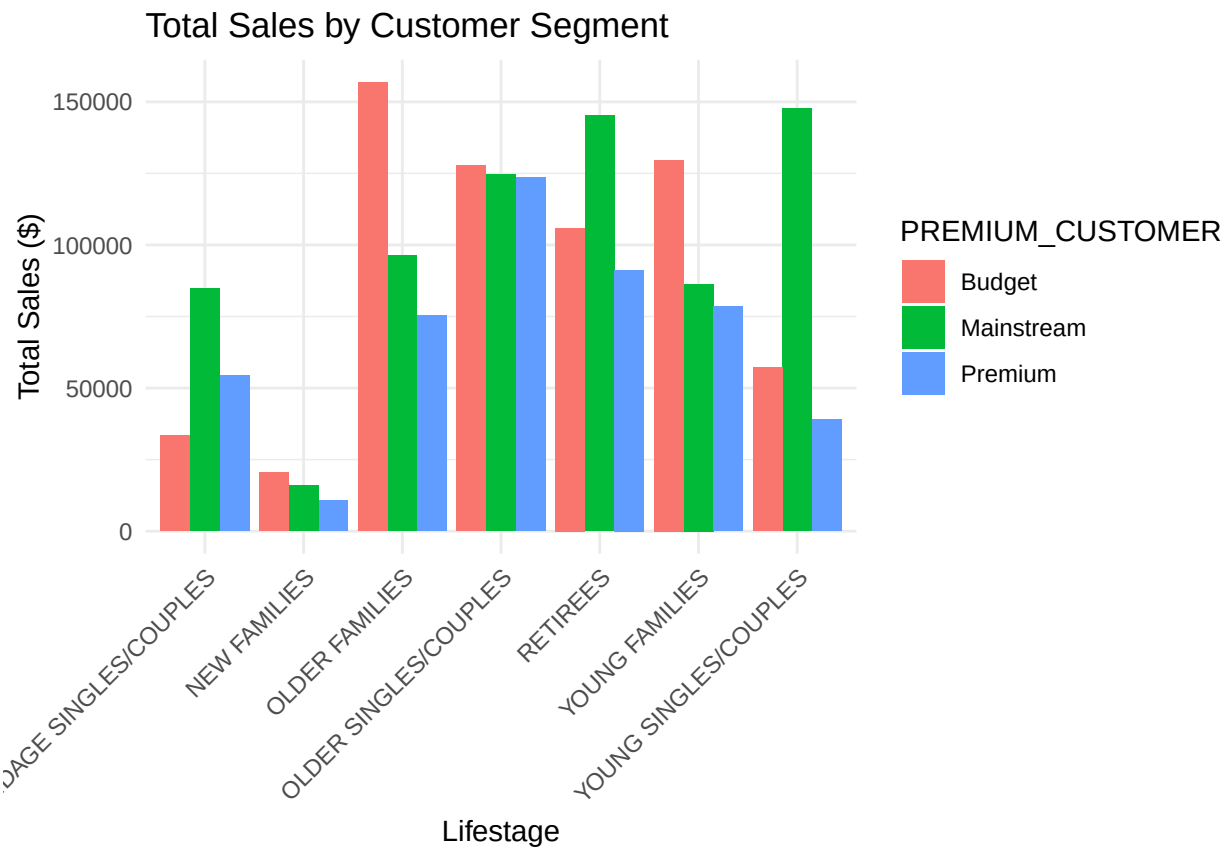
```r
# Check for missing customer info
sum(is.na(merged_data$LIFESTAGE))
```

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

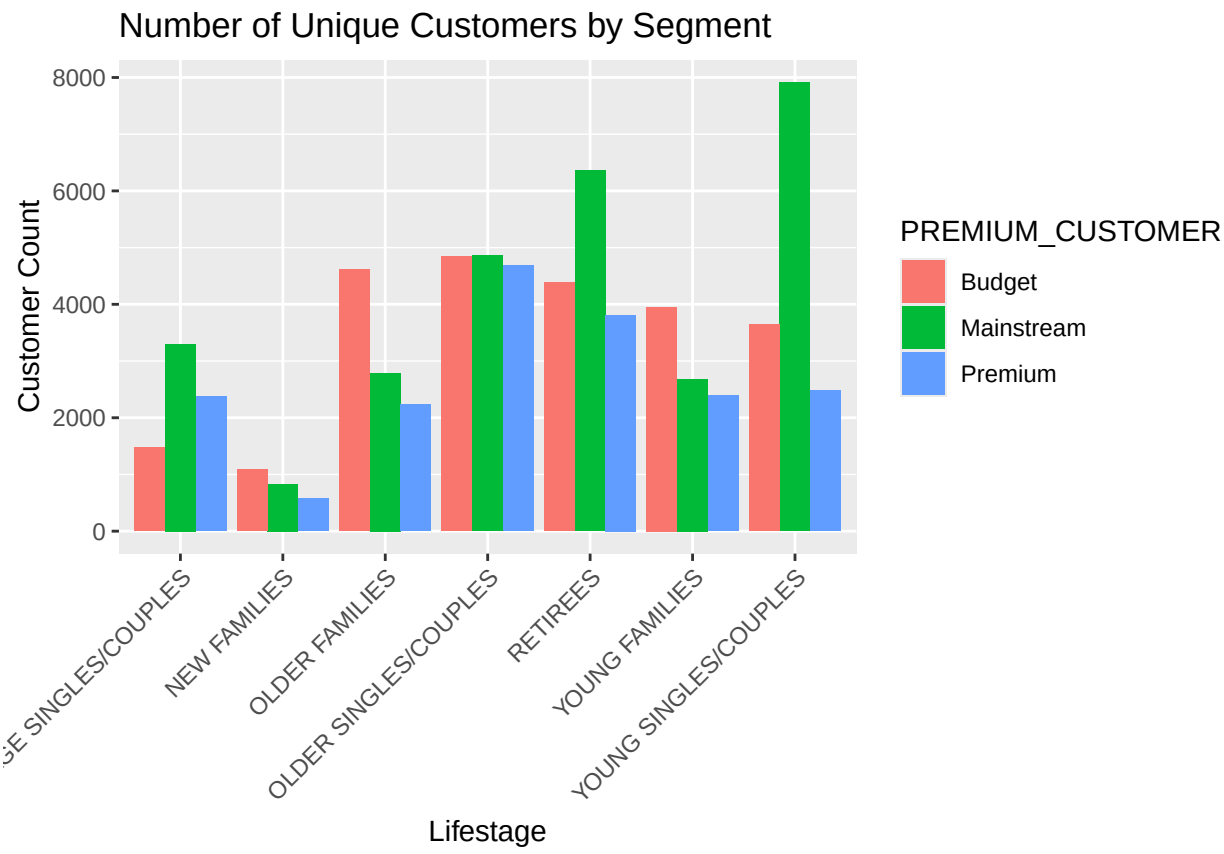
```r
sum(is.na(merged_data$PREMIUM_CUSTOMER))
```

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

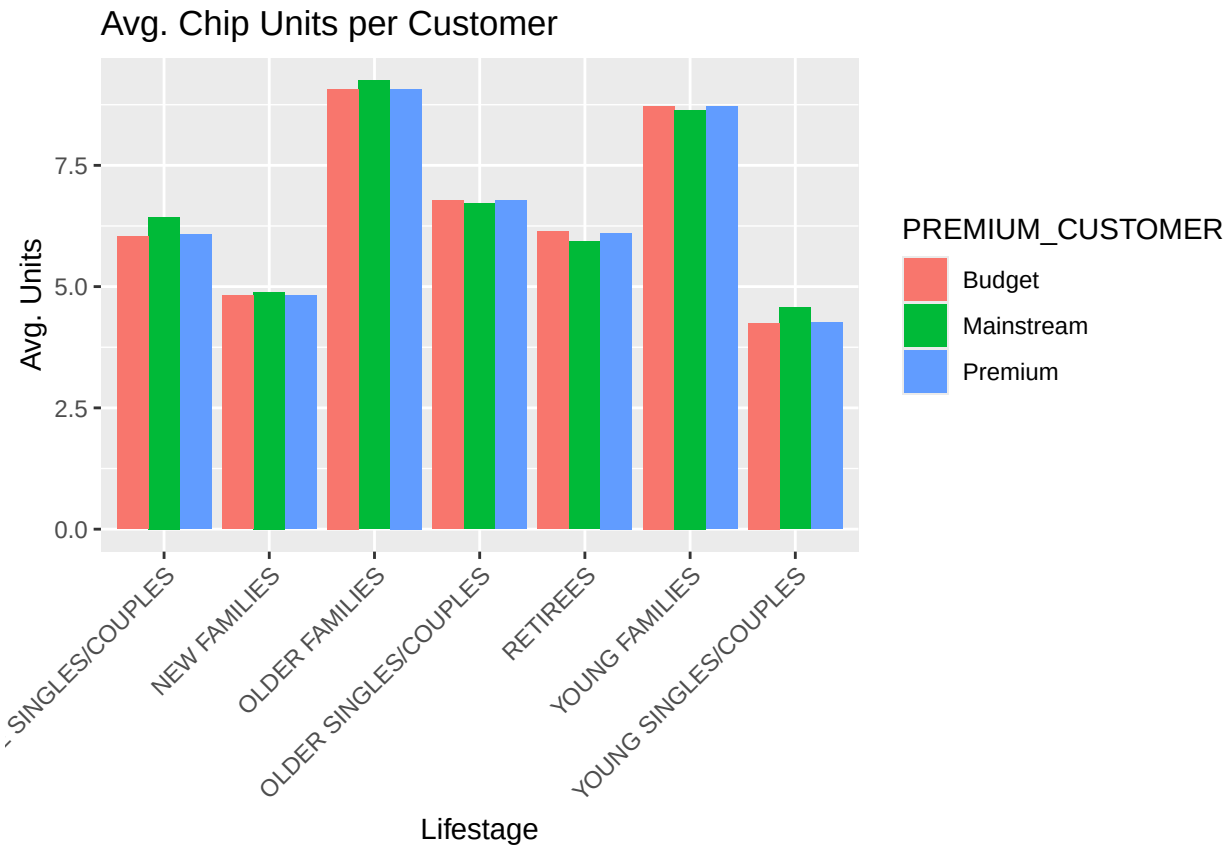
```r
# Group by lifestage and premium segment, sum total sales
sales_by_segment <- merged_data[, .(Total_Sales = sum(TOT_SALES)), by = .(LIFESTAGE, PREMIUM_CUSTOMER)]

# Visualize
ggplot(sales_by_segment, aes(x = LIFESTAGE, y = Total_Sales, fill = PREMIUM_CUSTOMER)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme_minimal(base_size = 11) +
  labs(title = "Total Sales by Customer Segment",
       x = "Lifestage",
       y = "Total Sales ($)") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



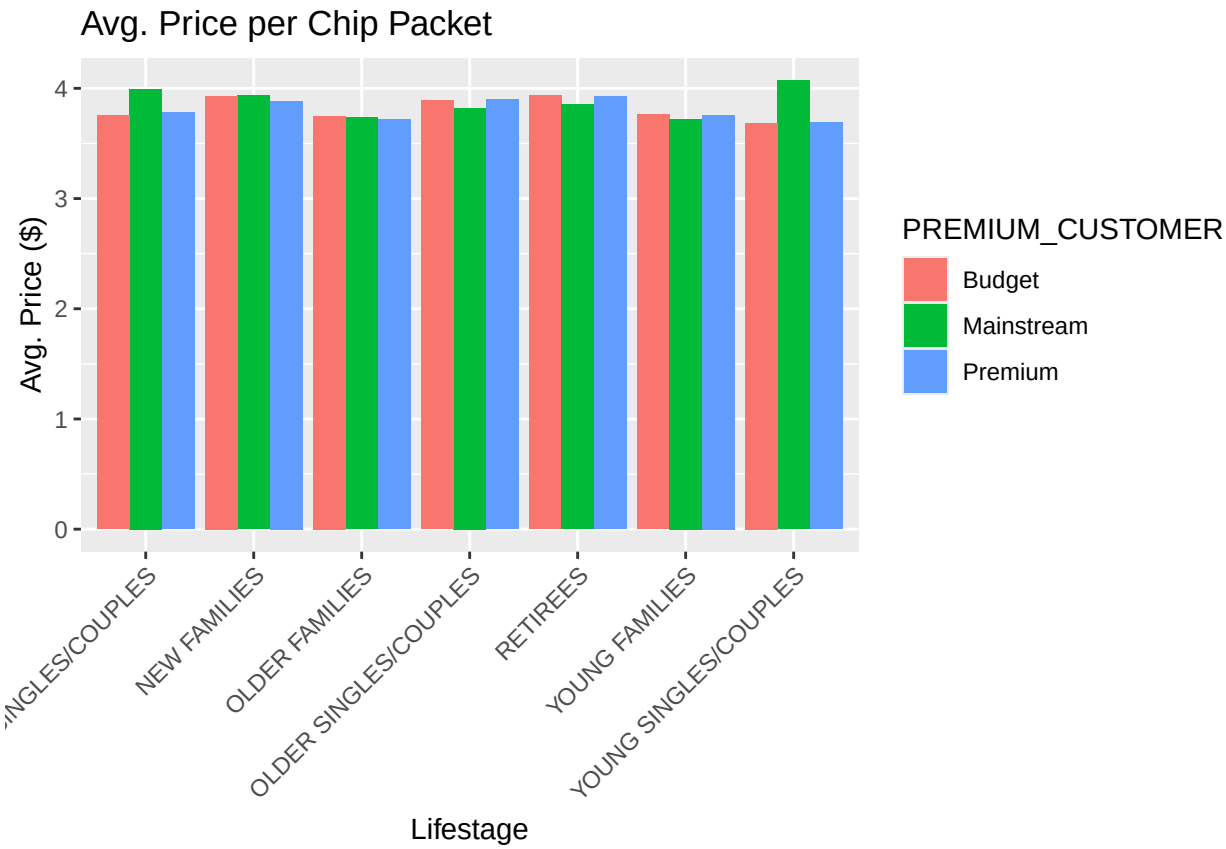

```r
# Count unique customers per segment
cust_count_by_segment <- unique(merged_data[, .(LYLTY_CARD_NBR, LIFESTAGE, PREMIUM_CUSTOMER)])
cust_count_by_segment <- cust_count_by_segment[, .N, by = .(LIFESTAGE, PREMIUM_CUSTOMER)]

# Visualize
ggplot(cust_count_by_segment, aes(x = LIFESTAGE, y = N, fill = PREMIUM_CUSTOMER)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Number of Unique Customers by Segment",
       x = "Lifestage",
       y = "Customer Count") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Sum quantity and get unique customer count per segment
units_by_segment <- merged_data[, .(Total_Units = sum(PROD_QTY)), by = .(LIFESTAGE, PREMIUM_CUSTOMER)]
units_by_segment[, N_Customers := cust_count_by_segment$N]
units_by_segment[, Avg_Units_Per_Customer := Total_Units / N_Customers]

# Visualize
ggplot(units_by_segment, aes(x = LIFESTAGE, y = Avg_Units_Per_Customer, fill = PREMIUM_CUSTOMER)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Avg. Chip Units per Customer",
       y = "Avg. Units", x = "Lifestage") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Total sales / total quantity = avg unit price
price_by_segment <- merged_data[, .(Avg_Price = sum(TOT_SALES) / sum(PROD_QTY)), by = .(LIFESTAGE, PREM

# Visualize
ggplot(price_by_segment, aes(x = LIFESTAGE, y = Avg_Price, fill = PREMIUM_CUSTOMER)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Avg. Price per Chip Packet",
       y = "Avg. Price ($)", x = "Lifestage") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
## Insights
# Mainstream & Budget Older Families, Mainstream Retirees, and Mainstream Young Singles/Couples are dri
# Premium customers buy fewer chips overall, less engaged with the category.
# Big sales from Mainstream Retirees and Young Singles/Couples are due to their large numbers.
# Budget Older Families punch above their weight - fewer in number but buy a lot of chips.
# Families (Older & Young) buy more chip units per person.
# Price per pack is steady, but Mainstream Young Singles/Couples pay slightly more, possibly open to pr
```

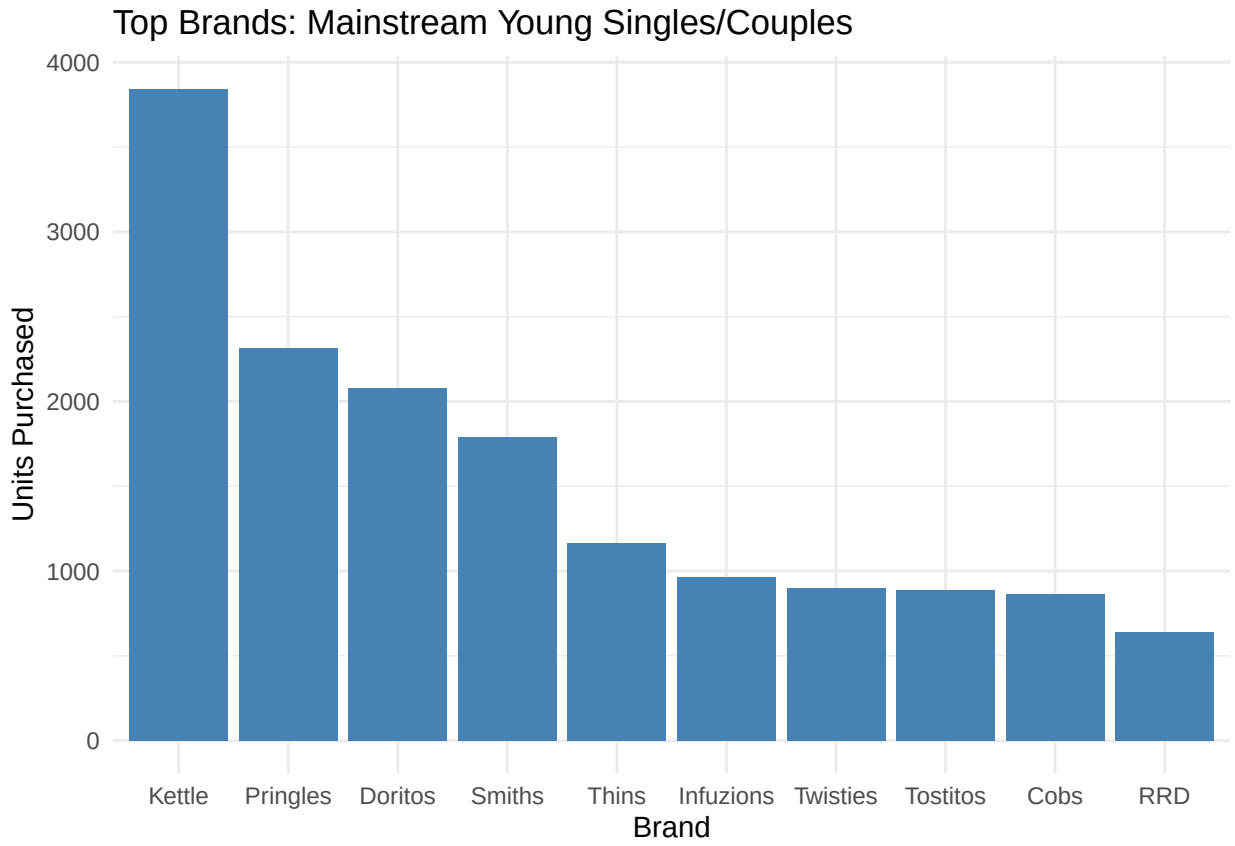
```
# Create a unit price column
merged_data[, UNIT_PRICE := TOT_SALES / PROD_QTY]

# Filter to Young Singles/Couples
ysc <- merged_data[LIFESTAGE == "YOUNG SINGLES/COUPLES"]

# Run t-test: Mainstream vs Others
t_test_result <- t.test(UNIT_PRICE ~ PREMIUM_CUSTOMER,
                        data = ysc[PREMIUM_CUSTOMER %in% c("Mainstream", "Budget")])
print(t_test_result)
```

```
##
##  Welch Two Sample t-test
##
## data:  UNIT_PRICE by PREMIUM_CUSTOMER
## t = -29.522, df = 15099, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group Budget and group Mainstream is not eq
```

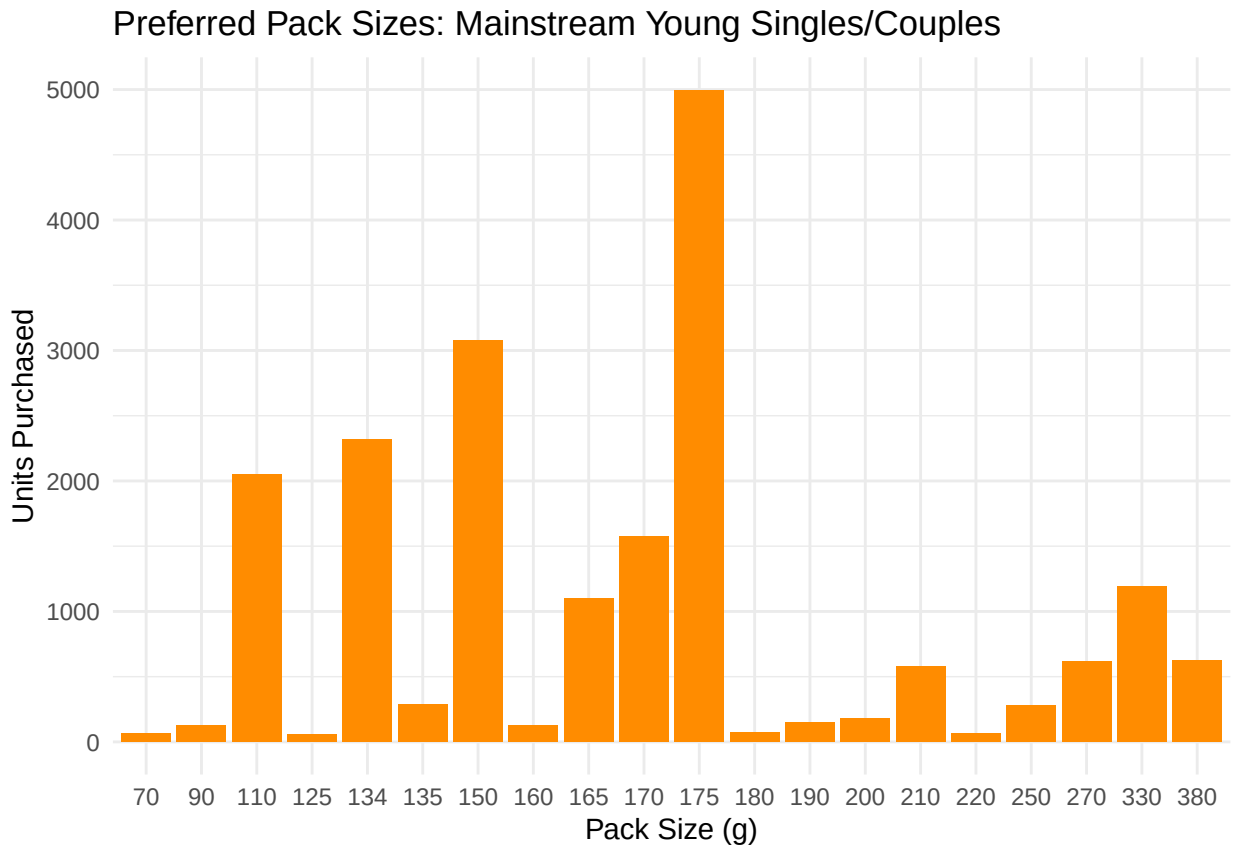
```
## 95 percent confidence interval:
##  -0.4353828 -0.3811682
## sample estimates:
##     mean in group Budget mean in group Mainstream
##                 3.657366                 4.065642
```

```
# Mainstream Young Singles/Couples pay an average of $4.07 per chip pack
# Budget Young Singles/Couples pay around $3.66 per pack
# p-value < 2.2 → statistically significant difference
# Mainstream customers in this lifestage are willing to pay more, a solid premium positioning opportuni
```

```
# Filter to target group
target_segment <- merged_data[LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream"]

# Count top brands
top_brands <- target_segment[, .N, by = BRAND][order(-N)]

# Visualize
ggplot(top_brands[1:10], aes(x = reorder(BRAND, -N), y = N)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Top Brands: Mainstream Young Singles/Couples",
       x = "Brand", y = "Units Purchased") +
  theme_minimal()
```



```r
# Count preferred pack sizes
top_packs <- target_segment[, .N, by = PACK_SIZE][order(-N)]

# Visualize
ggplot(top_packs, aes(x = factor(PACK_SIZE), y = N)) +
  geom_bar(stat = "identity", fill = "darkorange") +
  labs(title = "Preferred Pack Sizes: Mainstream Young Singles/Couples",
       x = "Pack Size (g)", y = "Units Purchased") +
  theme_minimal()
```




```r
# Segment Deep Dive: Mainstream Young Singles/Couples

# This segment buys the most chips among young customers and pays more per pack (confirmed by t-test).
# Top brands: Kettle, Pringles, Doritos - suggests preference for premium/well-known options.
# Most popular pack size: 175g by far, followed by 150g and 135g (mid-sized packs are key).

# Strategic Rec:
# Focus marketing on Mainstream Young Singles/Couples.
# Run promos on 150g-175g packs of premium brands like Kettle/Pringles.
# Try limited-edition flavors or slight price bumps, they'll likely accept it.
```