```
install.packages(c("data.table", "ggplot2", "ggmosaic", "readr"))
```

Show hidden output

Double-click (or enter) to edit

```
library(data.table)
library(ggplot2)
library(ggmosaic)
library(readr)


filePath <- "/content/"

# Load datasets
transactionData <- fread(paste0(filePath,"QVI_transaction_data -csv_file.csv"))
customerData <- fread(paste0(filePath,"QVI_purchase_behaviour.csv"))


# ================================================================================
# EXPLORATORY DATA ANALYSIS
# ================================================================================

# Examine transaction data structure
str(transactionData)
head(transactionData)
summary(transactionData)
```

Show hidden output

```
# Convert DATE column to proper date format
transactionData$DATE <- as.Date(transactionData$DATE, origin = "1899-12-30")


# Examine PROD_NAME column
summary(transactionData$PROD_NAME)
length(unique(transactionData$PROD_NAME))
```

Show hidden output

```
# Examine individual words in product names
productWords <- data.table(unlist(strsplit(unique(transactionData[, PROD_NAME]), " ")))
setnames(productWords, 'words')


# Remove digits and special characters, then count word frequency
productWords <- productWords[grepl("^[A-Za-z]+$", words)]
productWords_summary <- productWords[, .N, by = words][order(-N)]
head(productWords_summary, 20)
```

Show hidden output

```
# Remove salsa products (already provided in template)
transactionData[, SALSA := grepl("salsa", tolower(PROD_NAME))]
transactionData <- transactionData[SALSA == FALSE, ][, SALSA := NULL]


# Check for outliers and nulls
summary(transactionData)
```

Show hidden output

```
# Investigate the outlier (200 packets)
outlier_transactions <- transactionData[PROD_QTY == 200]
print(outlier_transactions)
```

Show hidden output

```
# Check other transactions by the same customer
outlier_customer <- unique(outlier_transactions$LYLTY_CARD_NBR)
customer_transactions <- transactionData[LYLTY_CARD_NBR %in% outlier_customer]
print(customer_transactions)
```

Show hidden output

```
# Remove the outlier customer
transactionData <- transactionData[LYLTY_CARD_NBR != outlier_customer]
```

```
# Re-examine the data
summary(transactionData)
```

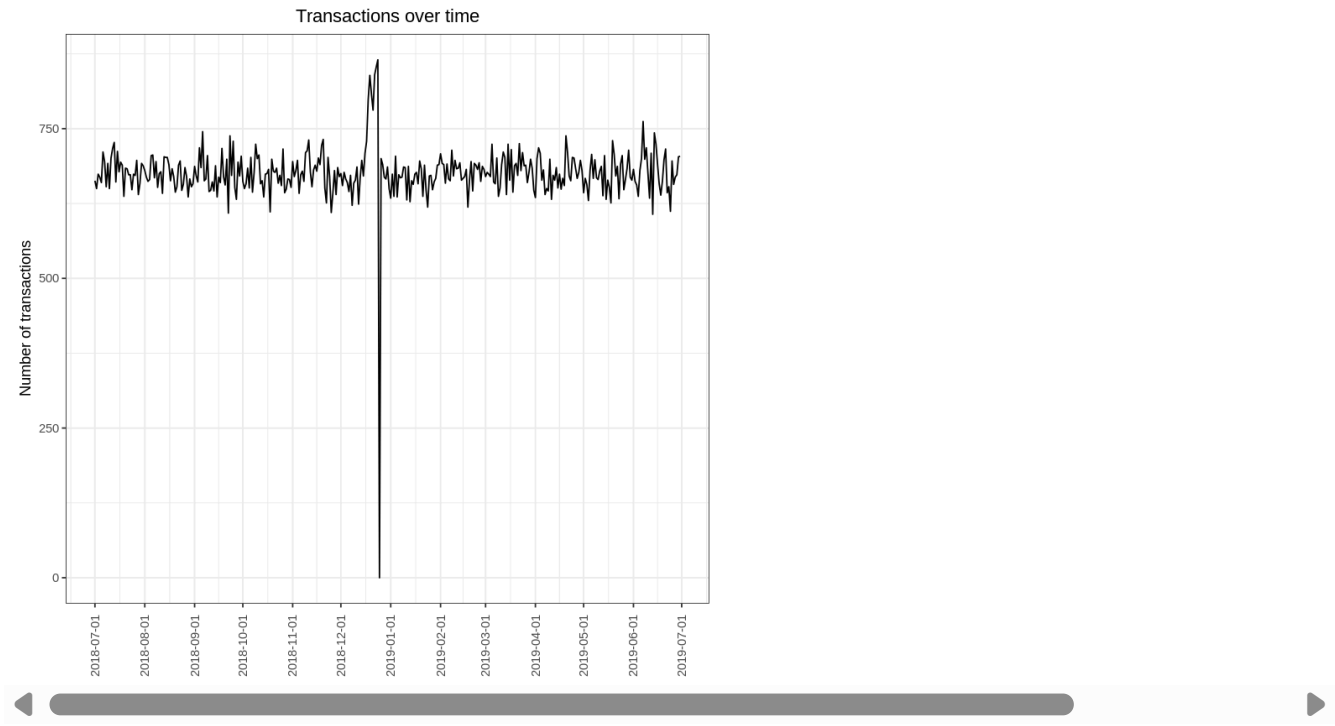
Show hidden output

```
# Count transactions by date
transactions_by_day <- transactionData[, .N, by = DATE][order(DATE)]
print(nrow(transactions_by_day))  # Should be 364, missing 1 day
```

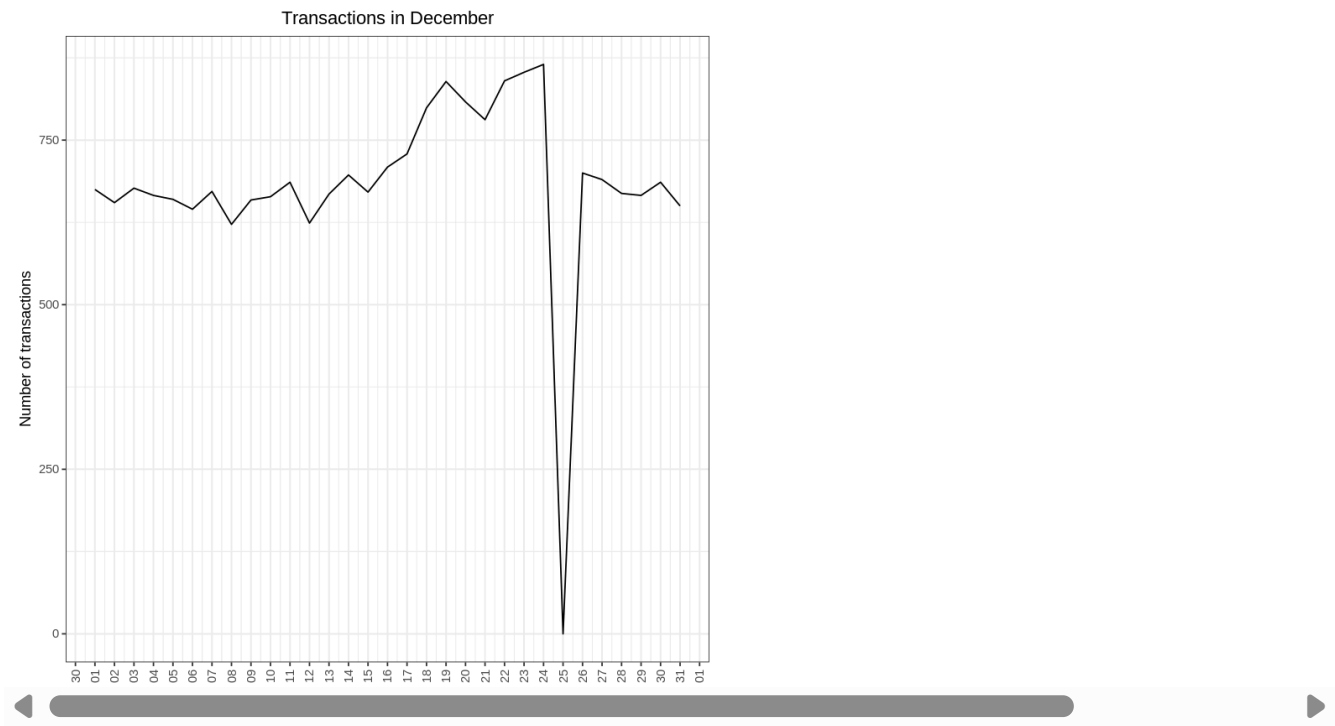
Show hidden output

```
# Create complete sequence of dates and join
all_dates <- data.table(DATE = seq(as.Date("2018-07-01"), as.Date("2019-06-30"), by = "day"))
transactions_by_day <- merge(all_dates, transactions_by_day, all.x = TRUE)
transactions_by_day[is.na(N), N := 0]
```

```
# Plot transactions over time
theme_set(theme_bw())
theme_update(plot.title = element_text(hjust = 0.5))

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



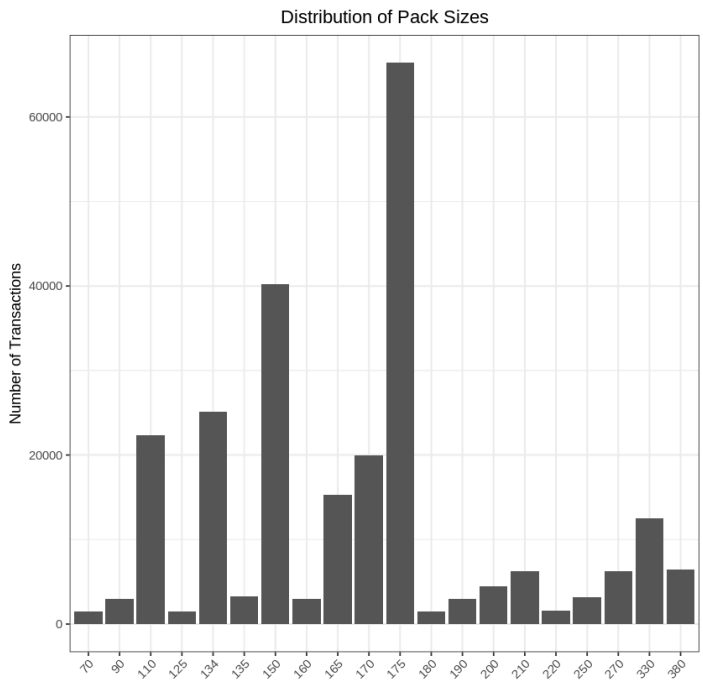

```
# Zoom in on December
december_data <- transactions_by_day[DATE >= as.Date("2018-12-01") & DATE <= as.Date("2018-12-31")]

ggplot(december_data, aes(x = DATE, y = N)) +
  geom_line() +
  labs(x = "Day", y = "Number of transactions", title = "Transactions in December") +
  scale_x_date(date_labels = "%d", date_breaks = "1 day") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```




```
# Create pack size feature
transactionData[, PACK_SIZE := parse_number(PROD_NAME)]
transactionData[, .N, PACK_SIZE][order(PACK_SIZE)]
```

Show hidden output

```
# Plot histogram of pack sizes
ggplot(transactionData, aes(x = factor(PACK_SIZE))) +
  geom_bar() +
  labs(x = "Pack Size (g)", y = "Number of Transactions", title = "Distribution of Pack Sizes") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Create brand names
transactionData[, BRAND := toupper(substr(PROD_NAME, 1, regexpr(" ", PROD_NAME) - 1))]
```

```
# Check brand names
brand_summary <- transactionData[, .N, by = BRAND][order(-N)]
print(brand_summary)
```

Show hidden output

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

```
# Check cleaned brands
brand_summary_clean <- transactionData[, .N, by = BRAND][order(-N)]
print(brand_summary_clean)
```

Show hidden output

```
# ================================================================================
# EXAMINE CUSTOMER DATA
# ================================================================================
```

```
# Basic summaries of customer data
str(customerData)
summary(customerData)
```

Show hidden output

```
# Distribution of LIFESTAGE
lifestage_dist <- customerData[, .N, by = LIFESTAGE][order(-N)]
print(lifestage_dist)
```

Show hidden output

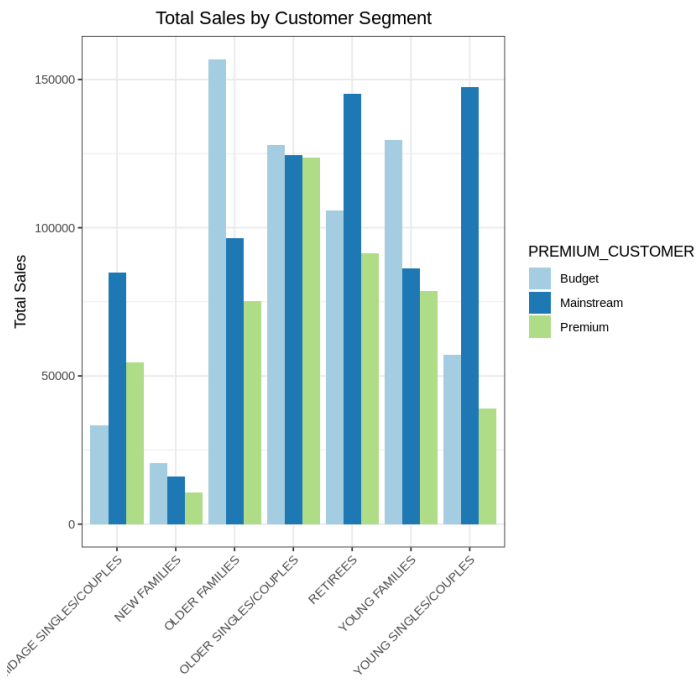
```
# Distribution of PREMIUM_CUSTOMER
premium_dist <- customerData[, .N, by = PREMIUM_CUSTOMER][order(-N)]
print(premium_dist)
```

Show hidden output

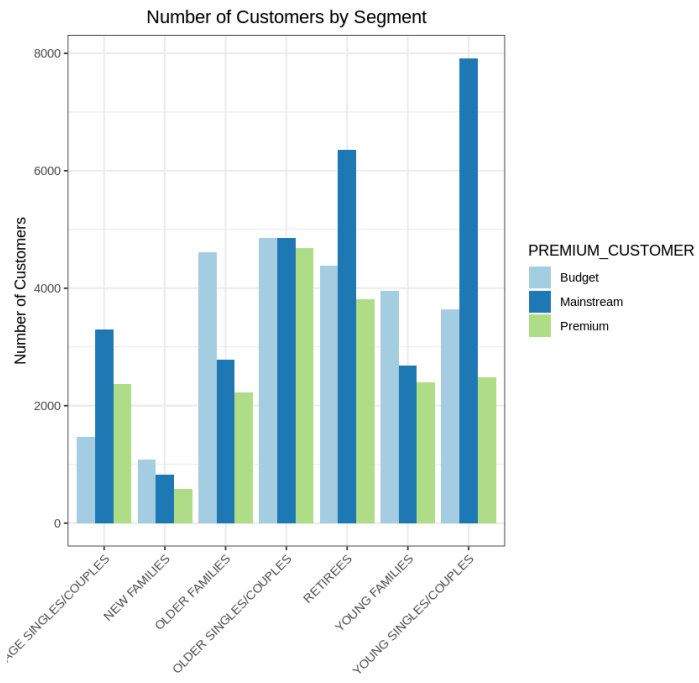
```
# Merge transaction and customer data
data <- merge(transactionData, customerData, all.x = TRUE)


# Check for missing customer details
missing_customers <- sum(is.na(data$LIFESTAGE))
print(paste("Missing customers:", missing_customers))
```

Show hidden output

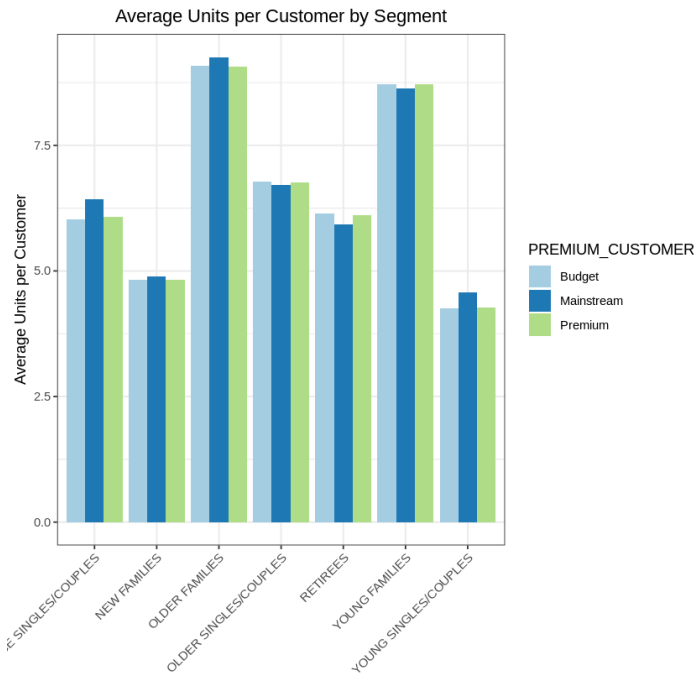
```
# Save the merged dataset
fwrite(data, paste0(filePath,"QVI_data.csv"))


# ================================================================================
# DATA ANALYSIS ON CUSTOMER SEGMENTS
# ================================================================================

# Total sales by LIFESTAGE and PREMIUM_CUSTOMER
sales_summary <- data[, .(TOTAL_SALES = sum(TOT_SALES)), by = .(LIFESTAGE, PREMIUM_CUSTOMER)]

ggplot(sales_summary, aes(x = LIFESTAGE, y = TOTAL_SALES, fill = PREMIUM_CUSTOMER)) +
  geom_col(position = "dodge") +
  labs(x = "Lifestage", y = "Total Sales", title = "Total Sales by Customer Segment") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(type = "qual", palette = 3)
```



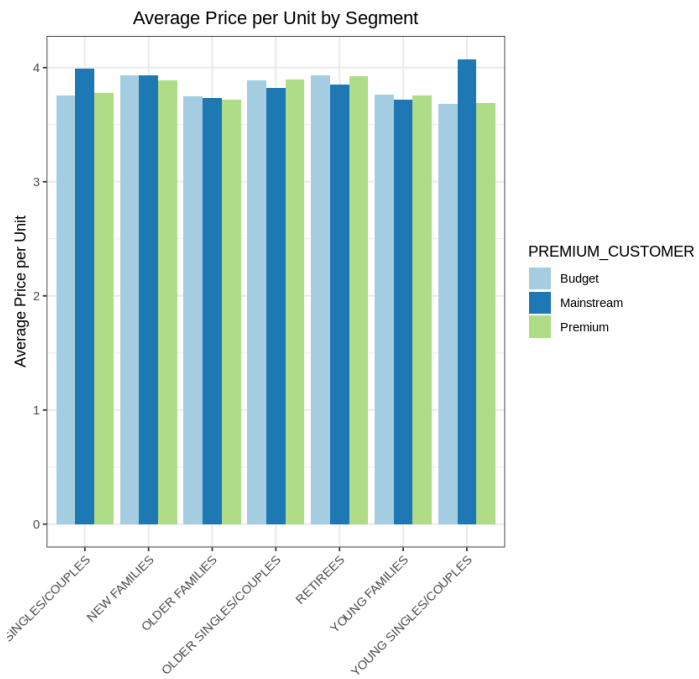

```
# Number of customers by segment
customer_summary <- data[, .(CUSTOMERS = uniqueN(LYLTY_CARD_NBR)), by = .(LIFESTAGE, PREMIUM_CUSTOMER)]

ggplot(customer_summary, aes(x = LIFESTAGE, y = CUSTOMERS, fill = PREMIUM_CUSTOMER)) +
  geom_col(position = "dodge") +
  labs(x = "Lifestage", y = "Number of Customers", title = "Number of Customers by Segment") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(type = "qual", palette = 3)
```




```
# Average units per customer by segment
units_summary <- data[, .(AVG_UNITS = sum(PROD_QTY)/uniqueN(LYLTY_CARD_NBR)), by = .(LIFESTAGE, PREMIUM_CUS

ggplot(units_summary, aes(x = LIFESTAGE, y = AVG_UNITS, fill = PREMIUM_CUSTOMER)) +
  geom_col(position = "dodge") +
  labs(x = "Lifestage", y = "Average Units per Customer", title = "Average Units per Customer by Segment")
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(type = "qual", palette = 3)
```




```
# Average price per unit by segment
price_summary <- data[, .(AVG_PRICE = sum(TOT_SALES)/sum(PROD_QTY)), by = .(LIFESTAGE, PREMIUM_CUSTOMER)]

ggplot(price_summary, aes(x = LIFESTAGE, y = AVG_PRICE, fill = PREMIUM_CUSTOMER)) +
  geom_col(position = "dodge") +
  labs(x = "Lifestage", y = "Average Price per Unit", title = "Average Price per Unit by Segment") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(type = "qual", palette = 3)
```




```
# Statistical test - comparing mainstream vs others for young/midage singles/couples
mainstream_data <- data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") &
                        PREMIUM_CUSTOMER == "Mainstream", TOT_SALES/PROD_QTY]
others_data <- data[LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES") &
                    PREMIUM_CUSTOMER != "Mainstream", TOT_SALES/PROD_QTY]
```

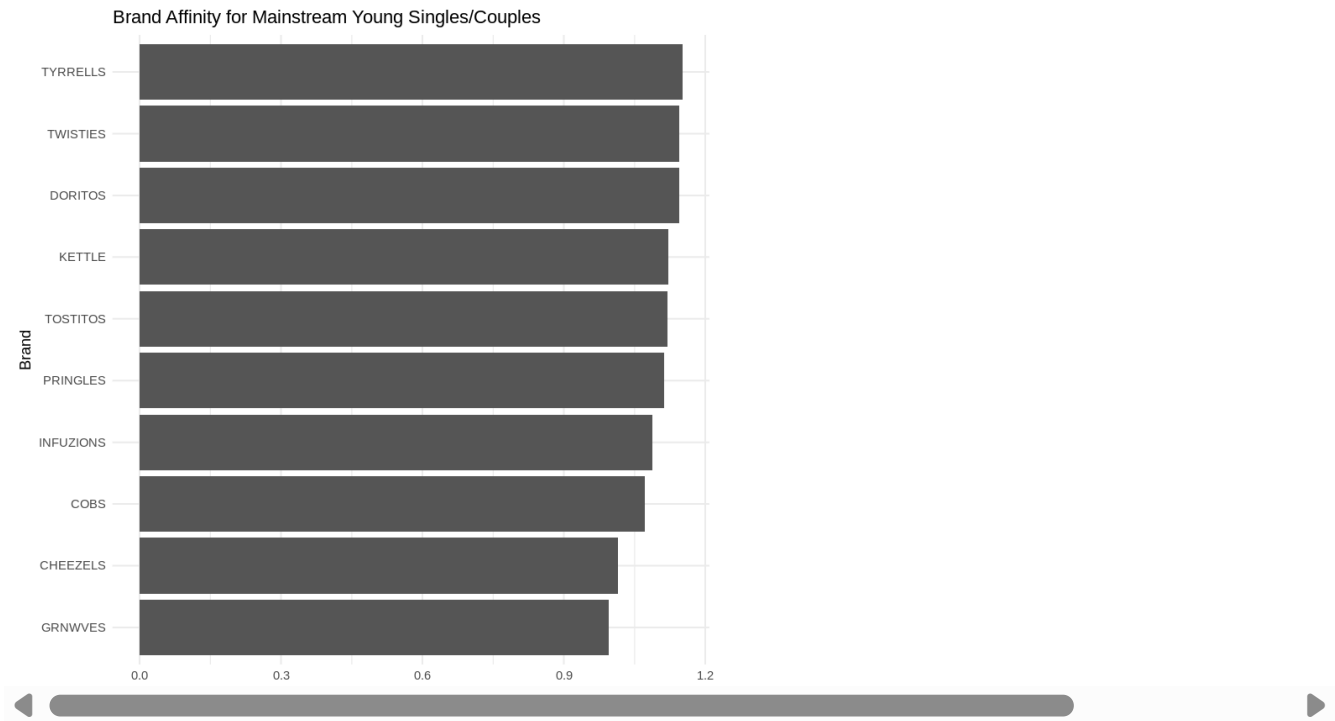
```
t_test_result <- t.test(mainstream_data, others_data, alternative = "greater")
print(t_test_result)
```

Show hidden output

```
# ============================================================================
# DEEP DIVE INTO MAINSTREAM YOUNG SINGLES/COUPLES
# ============================================================================

# Brand preference analysis
target_segment <- data[LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream"]
other_segments <- data[!(LIFESTAGE == "YOUNG SINGLES/COUPLES" & PREMIUM_CUSTOMER == "Mainstream")]


# Calculate brand affinity
target_brand_prop <- target_segment[, .(TARGET_SALES = sum(TOT_SALES)), by = BRAND]
target_brand_prop[, TARGET_PROP := TARGET_SALES/sum(TARGET_SALES)]

other_brand_prop <- other_segments[, .(OTHER_SALES = sum(TOT_SALES)), by = BRAND]
other_brand_prop[, OTHER_PROP := OTHER_SALES/sum(OTHER_SALES)]

brand_affinity <- merge(target_brand_prop, other_brand_prop, by = "BRAND")
brand_affinity[, AFFINITY := TARGET_PROP/OTHER_PROP]
brand_affinity <- brand_affinity[order(-AFFINITY)]

print("Brand Affinity Analysis (Top 10):")
print(head(brand_affinity[, .(BRAND, AFFINITY)], 10))
```

Show hidden output

```
# Visualize brand preference
top_brands <- head(brand_affinity, 10)
ggplot(top_brands, aes(x = reorder(BRAND, AFFINITY), y = AFFINITY)) +
  geom_col() +
  coord_flip() +
  labs(x = "Brand", y = "Affinity Index",
       title = "Brand Affinity for Mainstream Young Singles/Couples") +
  theme_minimal()
```

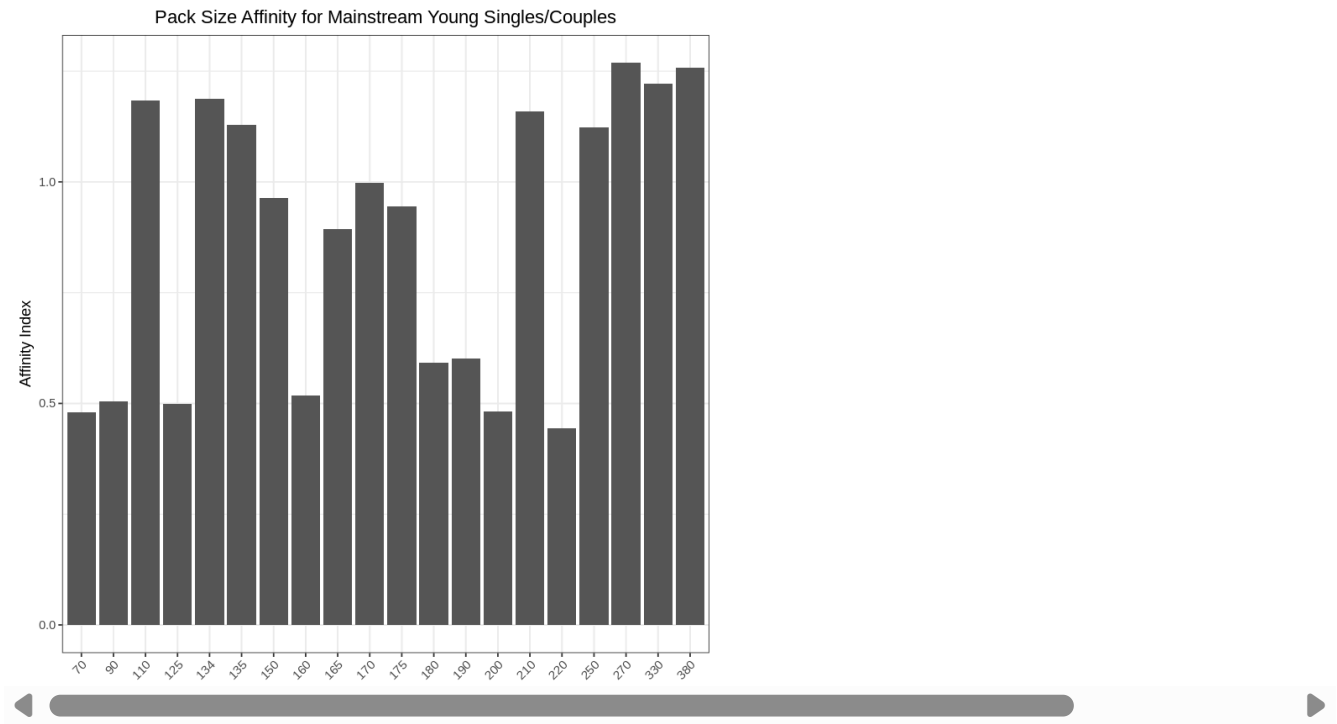



```
# Pack size preference analysis
target_pack_prop <- target_segment[, .(TARGET_QTY = sum(PROD_QTY)), by = PACK_SIZE]
```

```
target_pack_prop[, TARGET_PROP := TARGET_QTY/sum(TARGET_QTY)]

other_pack_prop <- other_segments[, .(OTHER_QTY = sum(PROD_QTY)), by = PACK_SIZE]
other_pack_prop[, OTHER_PROP := OTHER_QTY/sum(OTHER_QTY)]

pack_affinity <- merge(target_pack_prop, other_pack_prop, by = "PACK_SIZE")
pack_affinity[, AFFINITY := TARGET_PROP/OTHER_PROP]
pack_affinity <- pack_affinity[order(-AFFINITY)]

print("Pack Size Affinity Analysis:")
print(pack_affinity[, .(PACK_SIZE, AFFINITY)])
```

Show hidden output

```
# Visualize pack size preference
ggplot(pack_affinity, aes(x = factor(PACK_SIZE), y = AFFINITY)) +
  geom_col() +
  labs(x = "Pack Size (g)", y = "Affinity Index",
       title = "Pack Size Affinity for Mainstream Young Singles/Couples") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# ================================================================================
# SUMMARY OF INSIGHTS
# ================================================================================

cat("\n=== KEY INSIGHTS ===\n")
cat("1. Sales Analysis:\n")
cat("   - Budget Older Families, Mainstream Young Singles/Couples, and Mainstream Retirees contribute most t
cat("   - Mainstream Young Singles/Couples have higher numbers but Budget Older Families buy more per custom

cat("2. Price Analysis:\n")
```