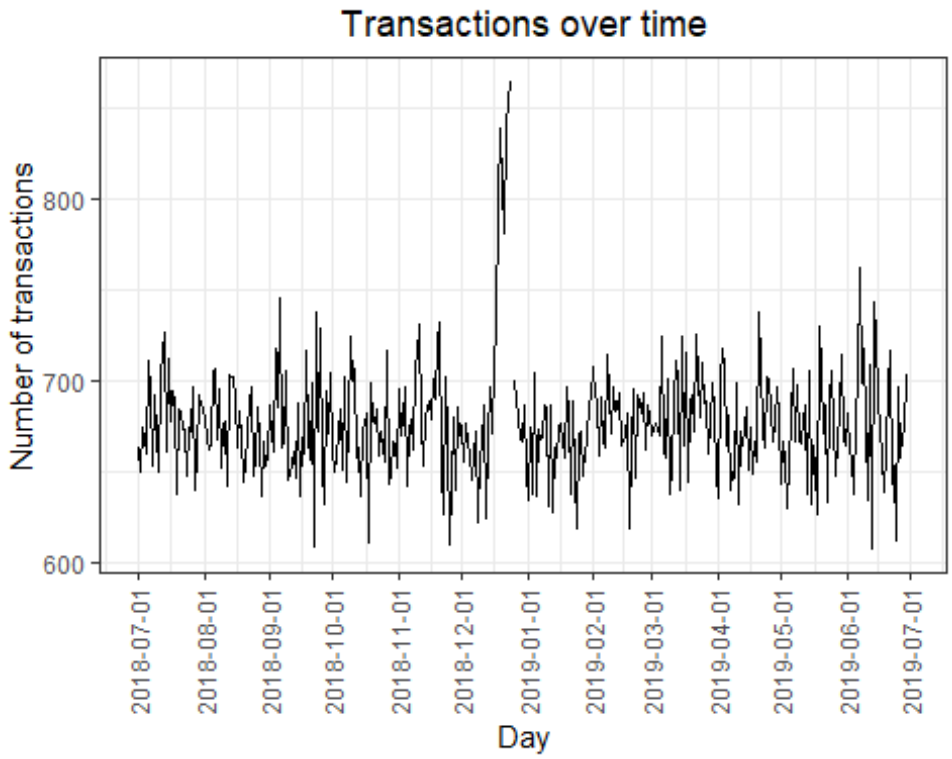
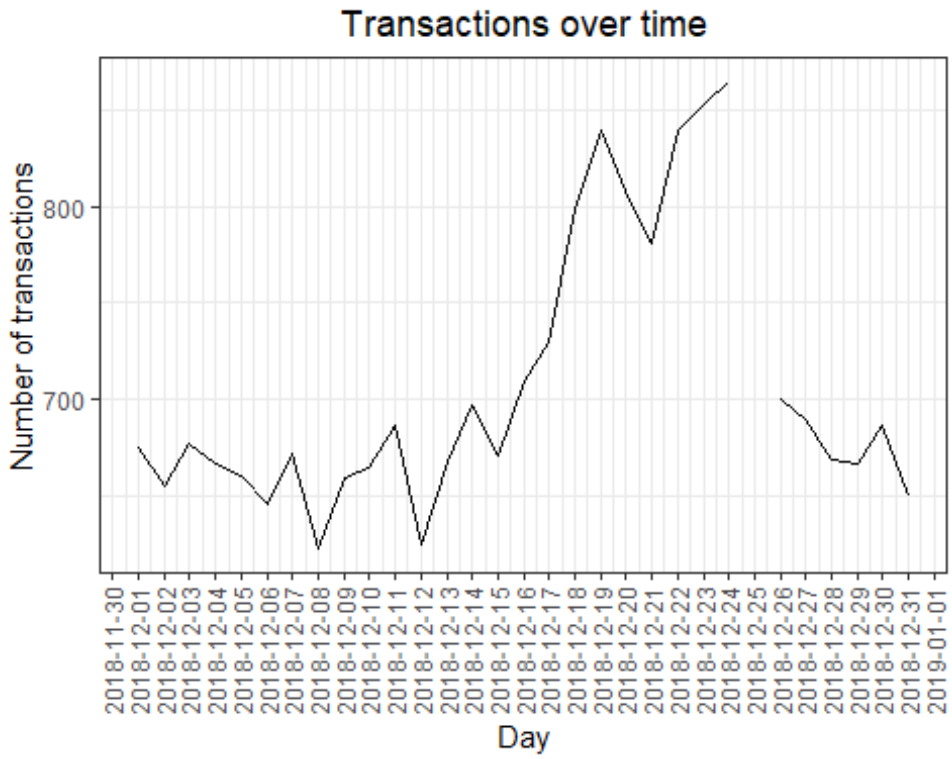
Task 1

```r
library(data.table)

library(ggplot2)
library(ggmosaic)

library(readr)

library(plyr)
library(dplyr)

filePath = "C:/Users/jerem/Desktop/Quantium/"
transactionData = fread(paste0(filePath,"QVI_transaction_data.csv"))
customerData = fread(paste0(filePath,"QVI_purchase_behaviour.csv"))
###Exploratory data analysis

str(transactionData)

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

# checking the type of product
unique(transactionData$PROD_NAME)

productWords <- data.table(unlist(strsplit(unique(transactionData[,
PROD_NAME]), "
")))
setnames(productWords, 'words')
## removing digits and special characters

productWords = apply(productWords, 2,function(productWords)
gsub("[[:punct:]]", "",gsub("[[:digit:]]",
"",gsub("\\g$","",tolower(productWords)))))

##SORT PRODUCT WORDS BY OCCURENCE (NOT DONE YET)


##Remove salsa products
transactionData[, SALSA := grepl("salsa", tolower(PROD_NAME))]
transactionData = transactionData[SALSA == FALSE, ][, SALSA := NULL]


##Summary stats
summary(transactionData$PROD_NBR)

summary(transactionData$PROD_QTY)

summary(transactionData$TOT_SALES)
```

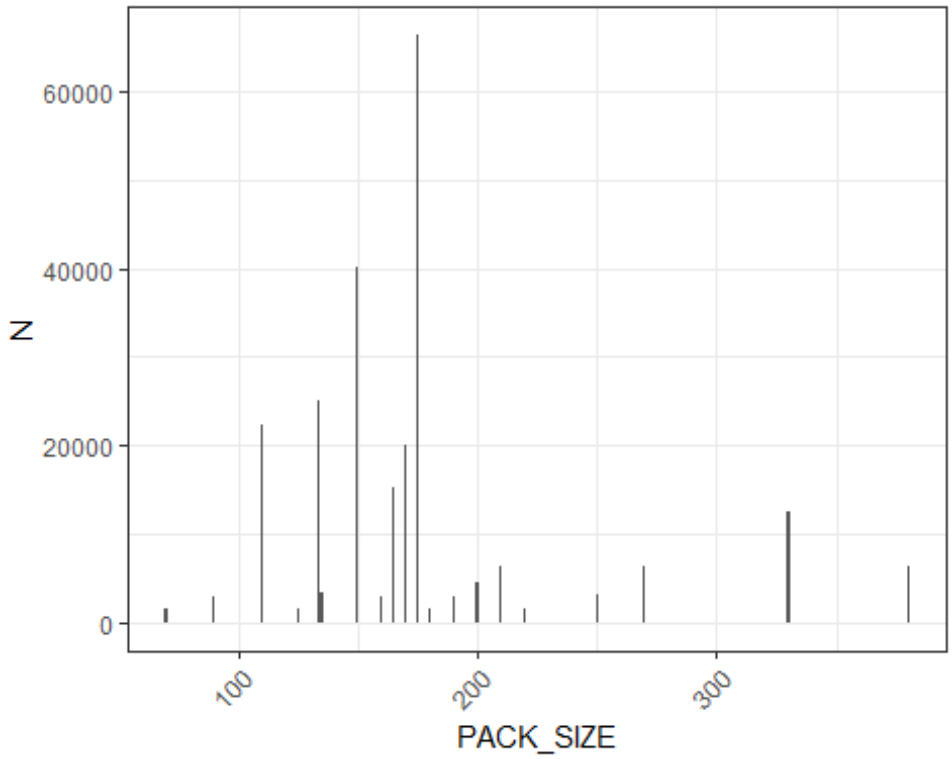
```r
#finding and eliminating outlier
outlier = subset(transactionData, transactionData$PROD_QTY == 200)
outlier_ID = outlier$LYLTY_CARD_NBR[1]
outlier_transaction = subset(transactionData, transactionData$LYLTY_CARD_NBR
== outlier_ID)

transactionData = transactionData[LYLTY_CARD_NBR != outlier_ID]

#Re-examining transaction Data
unique_Dates = as.data.frame(unique(transactionData$DATE))
setnames(unique_Dates, "Dates")

# filling in missing day
dates <- as.data.frame(seq(as.Date('2018-07-01'), as.Date('2019-06-30'), by =
'days'))
setnames(dates, "DATE")
transactions_by_dates = transactionData %>% group_by(DATE) %>% count(DATE)
Dates_with_missing = merge(transactions_by_dates, dates, by = "DATE", all.y =
T)




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


#### Plot transactions over time
ggplot(Dates_with_missing, 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))
```




```r
#plotting december
december =  Dates_with_missing %>% filter(DATE < as.Date("2019-01-01"))

december =  december %>% filter(DATE > as.Date("2018-11-30") )


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



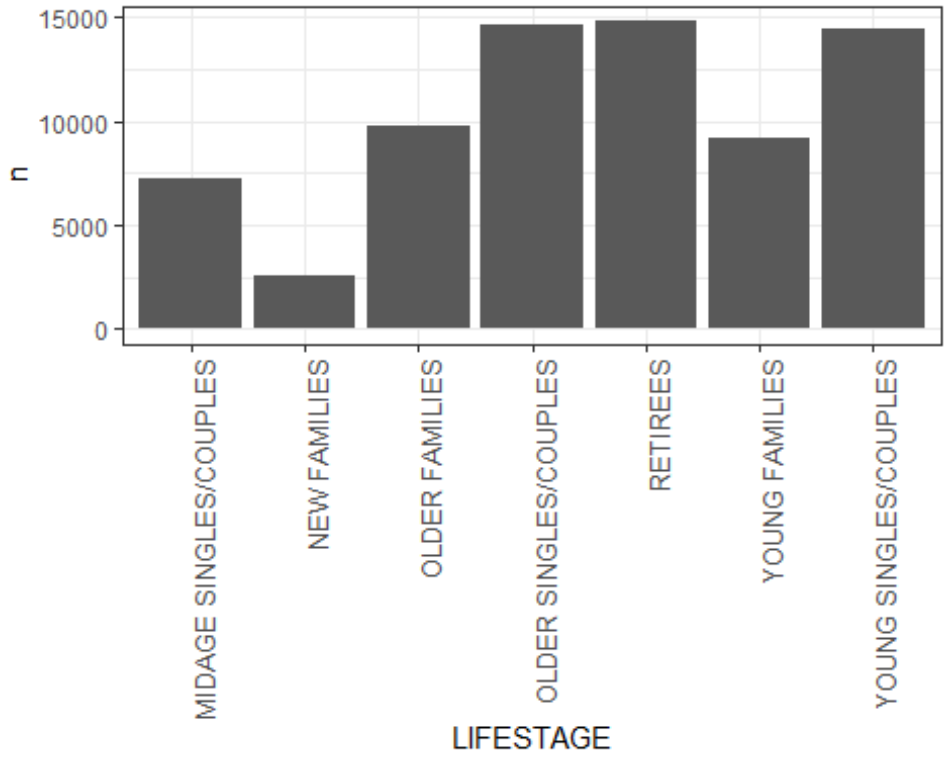

```
# We can see that the increase in sales occurs in the lead-up to Christmas
and that
# there are zero sales on Christmas day itself. This is due to shops being
closed on
# Christmas day.
# Now that we are satisfied that the data no longer has outliers, we can move
on to
# creating other features such as brand of chips or pack size from PROD_NAME.
We will
# start with pack size.

# pack size
transactionData[, PACK_SIZE := parse_number(PROD_NAME)]
Pack_size = transactionData[, .N, PACK_SIZE][order(PACK_SIZE)]
#histogram

plot_pack_size  = ggplot(data= Pack_size, aes(x=PACK_SIZE, y= N))+
geom_bar(stat="identity")+
theme(axis.text.x = element_text(size = 10, angle = 45, hjust = 1))
plot_pack_size
```



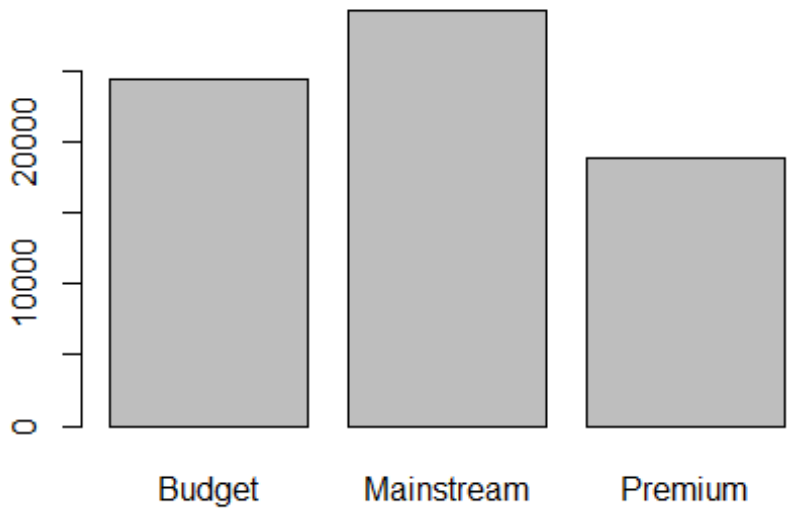

```
# brand_name; combining the same brands.

transactionData[, BRAND_NAME := gsub("([A-Za-z]+).*", "\\1",PROD_NAME)]
transactionData[BRAND_NAME == "Red", BRAND_NAME := "RRD"]
unique(transactionData$BRAND_NAME)

transactionData[BRAND_NAME == "Dorito", BRAND_NAME := "Doritos"]
transactionData[BRAND_NAME == "Snbts", BRAND_NAME := "Sunbites"]
transactionData[BRAND_NAME == "Grain", BRAND_NAME := "GrnWves"]
transactionData[BRAND_NAME == "WW", BRAND_NAME := "Woolworths"]
transactionData[BRAND_NAME == "NCC", BRAND_NAME := "Natural"]
transactionData[BRAND_NAME == "Infzns", BRAND_NAME := "Infuzions"]
transactionData[BRAND_NAME == "Infzns", BRAND_NAME := "Infuzions"]
transactionData[BRAND_NAME == "SMITH", BRAND_NAME := "SMITHS"]
# Examining customer Data

str(customerData)

Customer_by_LifeStage = customerData %>% group_by(LIFESTAGE) %>%
count(LIFESTAGE)
```

```
plot_lifestage  = ggplot(data= Customer_by_LifeStage, aes(x=LIFESTAGE, y=
n))+
geom_bar(stat="identity")+
```

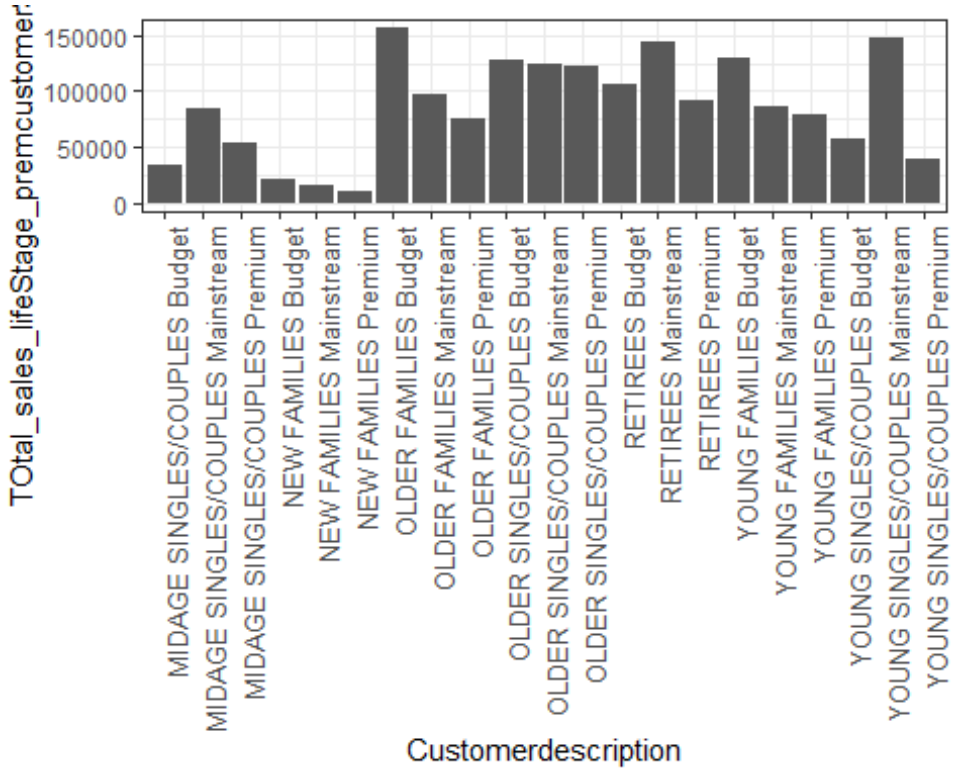
```
theme(axis.text.x = element_text(size = 10, angle = 90, hjust = 1))
plot_lifestage
```



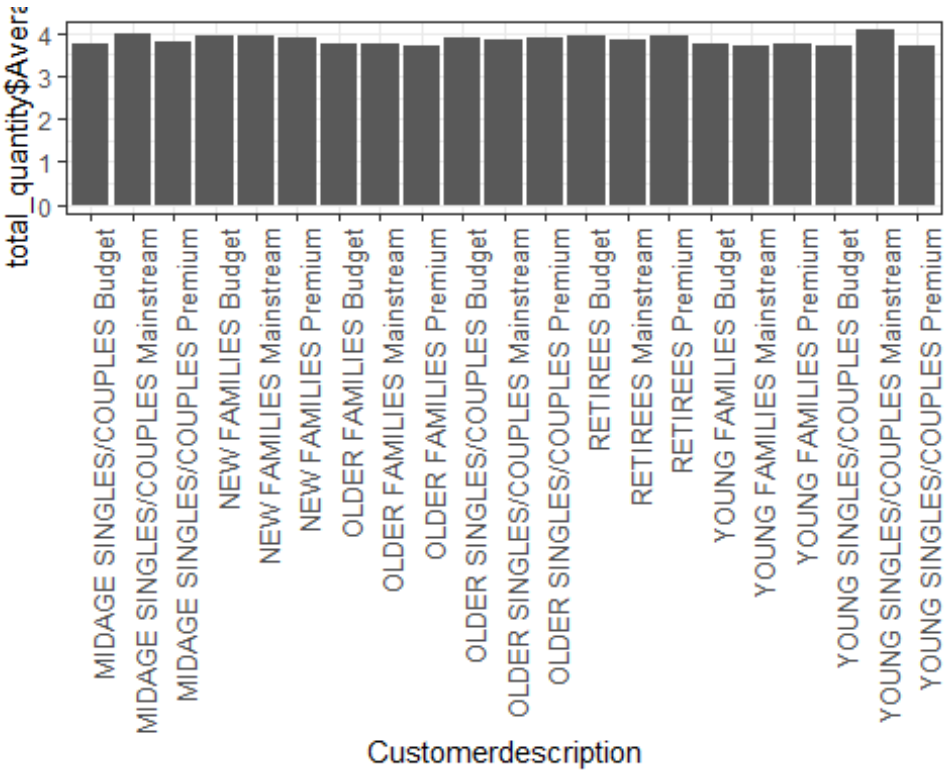

```
Customer_by_PREMstatus = customerData %>% group_by(PREMIUM_CUSTOMER) %>%
count(PREMIUM_CUSTOMER)
barplot(height  = Customer_by_PREMstatus$n, names.arg =
Customer_by_PREMstatus$PREMIUM_CUSTOMER)
```




```
is.null(data)

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

write.csv(data,"QVI_data.csv")
#PLotting data by premium customer and by life stage

TOtal_sales_lifeStage_premcustomer = aggregate(data$TOT_SALES~data$LIFESTAGE
+ data$PREMIUM_CUSTOMER, data, sum)
TOtal_sales_lifeStage_premcustomer$Customerdescription =
paste(TOtal_sales_lifeStage_premcustomer$`data$LIFESTAGE`,
TOtal_sales_lifeStage_premcustomer$`data$PREMIUM_CUSTOMER`)
plot  = ggplot(data=  TOtal_sales_lifeStage_premcustomer,
aes(x=Customerdescription, y=
TOtal_sales_lifeStage_premcustomer$`data$TOT_SALES`))+
geom_bar(stat="identity")+
theme(axis.text.x = element_text(size = 10, angle = 90, hjust = 1))
plot
```

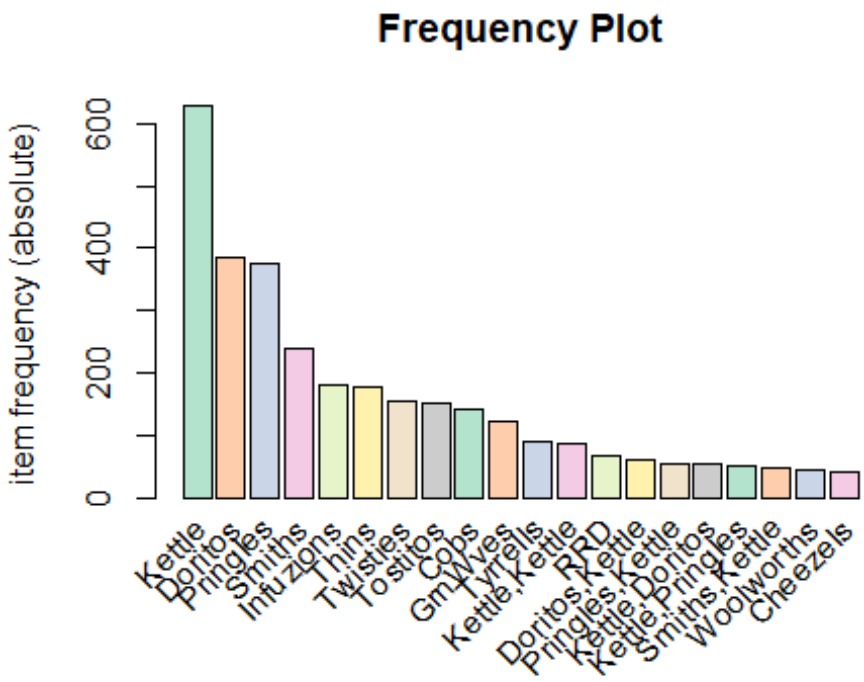



```
total_quantity = aggregate(data$PROD_QTY~data$LIFESTAGE +
data$PREMIUM_CUSTOMER, data, sum)
total_quantity = as.data.frame(aggregate(data$PROD_QTY~data$LIFESTAGE +
data$PREMIUM_CUSTOMER, data, sum))
total_quantity$Customerdescription = paste(total_quantity$`data$LIFESTAGE`,
total_quantity$`data$PREMIUM_CUSTOMER`)
total_quantity = subset(total_quantity, select = c("Customerdescription",
"data$PROD_QTY"))
total_quantity = merge(total_quantity, TOtal_sales_lifeStage_premcustomer, by
= "Customerdescription")
total_quantity$Average =
total_quantity$`data$TOT_SALES`/total_quantity$`data$PROD_QTY`


plot_average_sales  = ggplot(data= total_quantity, aes(x=Customerdescription,
y= total_quantity$Average))+
geom_bar(stat="identity")+
theme(axis.text.x = element_text(size = 10, angle = 90, hjust = 1))
plot_average_sales
```

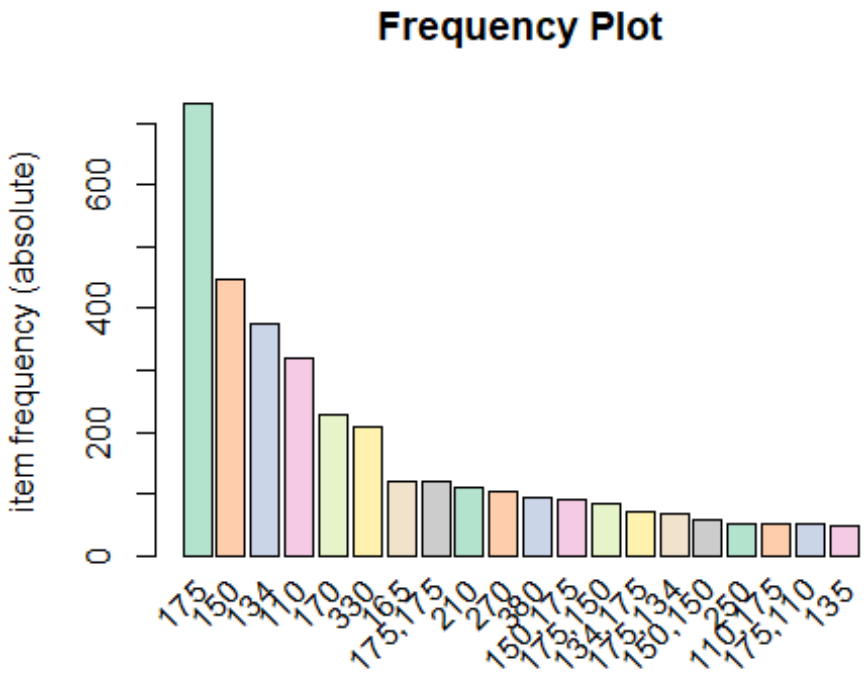



```
#### t test

t.test(data[PREMIUM_CUSTOMER == "Mainstream" &
              LIFESTAGE %in% c("YOUNG SINGLES/COUPLES",
                               "MIDAGE SINGLES/COUPLES"),
data$TOT_SALES/data$PROD_QTY],data[ PREMIUM_CUSTOMER != "Mainstream" &
LIFESTAGE %in% c("YOUNG SINGLES/COUPLES", "MIDAGE SINGLES/COUPLES"),
data$TOT_SALES/data$PROD_QTY], alternative = "greater")

## Deep dive into specific customer segments for insights
library(arules)

library(RColorBrewer)
main_young = data[PREMIUM_CUSTOMER == "Mainstream" & LIFESTAGE == "YOUNG
SINGLES/COUPLES"]

main_young_PROD_NAME = ddply(main_young,c("LYLTY_CARD_NBR"),
                       function(dataframe)paste(dataframe$PROD_NAME,
                                                collapse = ","))

main_young_BRAND = ddply(main_young,c("LYLTY_CARD_NBR"),
                       function(dataframe)paste(dataframe$BRAND_NAME,
                                                collapse = ","))
write.csv(main_young_BRAND,"main_young_brand.csv")
main_young_transaction <-
read.transactions("C:/Users/jerem/Documents/R/Quantium/Quantium/main_young_br
and.csv", format = 'basket', sep=',')
```

```
itemFrequencyPlot(main_young_transaction,topN = 20,type =
"absolute",col=brewer.pal(8,'Pastel2'), main="Frequency Plot", angle = 90)
```




```
main_young_SIZE =  ddply(main_young,c("LYLTY_CARD_NBR"),
                         function(dataframe)paste(dataframe$PACK_SIZE,
                                                  collapse = ","))

write.csv(main_young_SIZE,"main_young.csv")
main_young_transaction <-
read.transactions("C:/Users/jerem/Documents/R/Quantium/Quantium/main_young.cs
v", format = 'basket', sep=',')
itemFrequencyPlot(main_young_transaction,topN = 20,type =
"absolute",col=brewer.pal(8,'Pastel2'), main="Frequency Plot")
```


Frequency Plot
item frequency (absolute)
0
200
400
600
175
150
134
110
170
330
165
175,175
210
270
380
150,175
175,150
134,175
175,134
150,150
250
110,175
175,110
135