Appendix I - Rmd File

GBA424: Analytics Design - Assignment 2

Pin Li, Jiawen Liang, Ruiling Shen, Chenxi Tao, Khanh Tran

2/2/2020

Set Up Environment and Load Datasets

```r
# Set up environments
library(tidyverse)
library(formattable)
library(lubridate)
dir = "E:/Studying/Simon/Classes/GBA424 - Analytics Design/Assignments/Assignment 2"
setwd(dir)

#################################
#          SQL CODE             #
#################################

# SQL command to extract `storeItemSales.csv` and `itemAttributes.csv`
# select * from storeItemSales
# select * from itemAttributes

# Load datasets
# For question 1, 2
itemsAttributes = read.csv('itemsAttributes.csv')
storeItemSales = read.csv('storeItemSales.csv')
# For question 3, 4
survResponses = read.csv('survResponses.csv')
survQuestions = read.csv('survQuestions.csv')
```

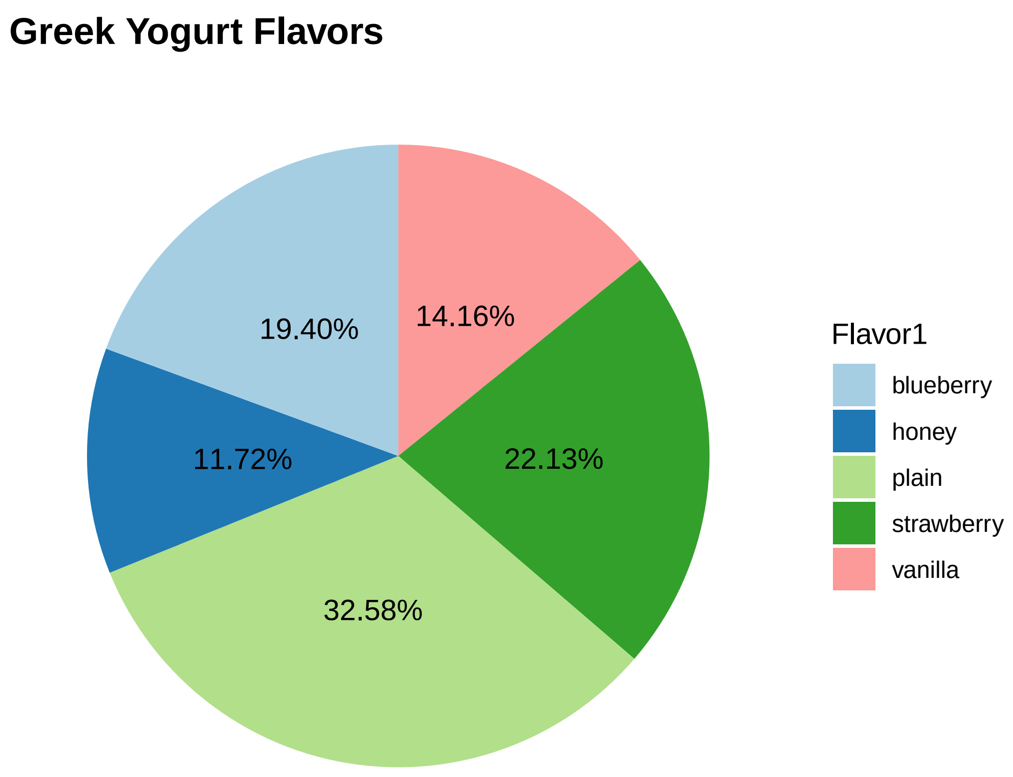
Question 1: Percentage of sales of existing flavors in the Greek yogurt category

We chose sales data from all the stores because it is the most representative data for the whole population.

```r
# Merge sales data table with item attributes data table
salesData = merge(itemsAttributes, storeItemSales, right_on='Item.Num')
```

For Greek yogurt, we only include six flavors mentioned in the case as currently existing flavors.

```r
#################################
#      OUTPUT FOR SLIDE 2       #
#################################

existingFlavor=c('plain','strawberry','blueberry','honey','vanilla','peach')

# calculate the percentage of sales
greekYogurt=
  salesData %>%
  filter(Class=='GREEK', Flavor1 %in% existingFlavor) %>%   # Where
  group_by(Flavor1) %>%   # Group by
  summarize(total_sales=sum(Sales)) %>% # Aggregation
  mutate(Percentage=percent(total_sales/sum(total_sales))) %>% # Calculate percentage
  arrange(desc(Percentage)) # Order by

# PLOT PIE CHART
library(ggplot2)

# Create a bar plot
bp<- ggplot(greekYogurt, aes(x="", y=Percentage, fill=Flavor1))+
  geom_bar(width = 1, stat = "identity")

# Create a pie chart with title
pie <- bp + coord_polar("y", start=0)+ggtitle('Greek Yogurt Flavors')

# Define the theme with blank grids
blank_theme <- theme_minimal()+
  theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  panel.border = element_blank(),
  panel.grid=element_blank(),
  axis.ticks = element_blank(),
  plot.title=element_text(size=14, face="bold")
  )

# Add labels to the pie chart
pie <- pie + scale_fill_brewer(palette="Paired")+
  theme_minimal()+
  blank_theme+
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        panel.grid  = element_blank())+
  geom_text(aes(label=Percentage),position = position_stack(vjust = 0.5))
pie
```

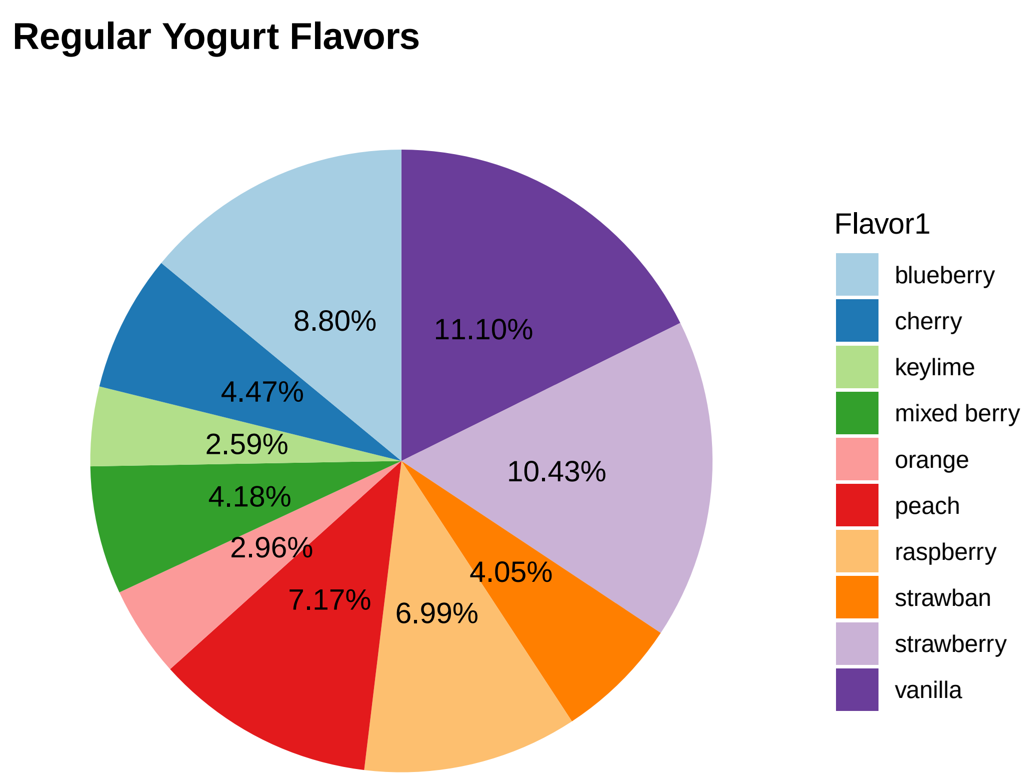


Question 2: Percentage of sales of existing flavors outside the Greek yogurt category

For regular yogurt, we included all the flavors. We showed the percentage data of the top 10 popular flavors and use "Others" to represent all the other flavors.

```
################################
#      OUTPUT FOR SLIDE 3      #
################################

# Calculate the percentage of sales
regular =
  salesData %>%
  filter(Class=='REGULAR') %>%
  group_by(Flavor1) %>%
  summarize(total_sales=sum(Sales)) %>%
  mutate(Percentage=percent(total_sales/sum(total_sales))) %>%
  arrange(desc(Percentage))

# Change the category of flavors outside the top10 popular flavors as "Others"
levels(regular$Flavor1) <- c(levels(regular$Flavor1), "Others")
regular[11:82,1] = "Others"

# Calculate the percentage of sales
Regular_Yogurt =
    regular %>%
        group_by(Flavor1) %>%# group by %>% #aggregation
        summarize(Percentage=sum(Percentage)) %>%
        arrange(desc(Percentage))

# PLOT PIE CHART

# create a bar plot
bp2<- ggplot(Regular_Yogurt, aes(x="", y=Percentage, fill=Flavor1))+
  geom_bar(width = 1, stat = "identity")

# create a pie chart with title
pie2 <- bp2 + coord_polar("y", start=0)+ggtitle('Regular Yogurt Flavors')

# add labels to the pie chart
pie2 <- pie2 + scale_fill_brewer(palette="Paired")+
  theme_minimal()+
  blank_theme+
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        panel.grid  = element_blank())+
  geom_text(aes(label=Percentage),position = position_stack(vjust = 0.5))
pie2
```

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

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



Question 3: Describe survey respondents' preferences for Greek yogurt flavors

Before conducting analysis, first we needed to do some data cleaning and processing.

```
# Change column names of `survResponses` in `Question 12` to flavors' name

# Extract flavor names from `survQuestions`. Before running this line,
# we identify an error in survQuestions.csv's Q3_8 column and fix it with Excel.
flavorNames = sapply(survQuestions[1, ], function (x) {substring(x, 105, nchar(as.character(x), type="c

# Change column names of `survResponses`
names(survResponses)[c(15:37)] <- flavorNames[c(15:37)]
```

We removed unreliable survey samples which: - are incomplete (V10 == 0), - skipped Question 12 entirely, and - take longer than 30 minutes to complete.

```
# Remove incomplete answers
sum(survResponses$V10 == 0) # 129 incomplete answers
```

```
## [1] 129
```

```
survResponses <- survResponses[survResponses$V10 != 0, ]

# Remove samples which skipped Q12 entirely
```

```r
emptyQ12 <- apply(survResponses[, c(15:37)], 1, function(x) {all(is.na(x))})
sum(emptyQ12) # 19 people skipped Q12 entirely
```

```
## [1] 19
```

```r
survResponses <- survResponses[!emptyQ12, ]

# Calculate time taken to answer survey
survResponses$V8 <- ymd_hms(survResponses$V8)
survResponses$V9 <- ymd_hms(survResponses$V9)
survResponses$timeElapsed <- as.numeric(difftime(survResponses$V9, survResponses$V8,
                                                 units="mins"))
# Remove answers took more than 30 minutes to complete
longAnswer <- survResponses$timeElapsed > 30
sum(longAnswer) # 25 unreliable answers
```

```
## [1] 25
```

```r
survResponses <- survResponses[!longAnswer, ]

# After remove unreliable responses, there are 579 samples left.
dim(survResponses)
```

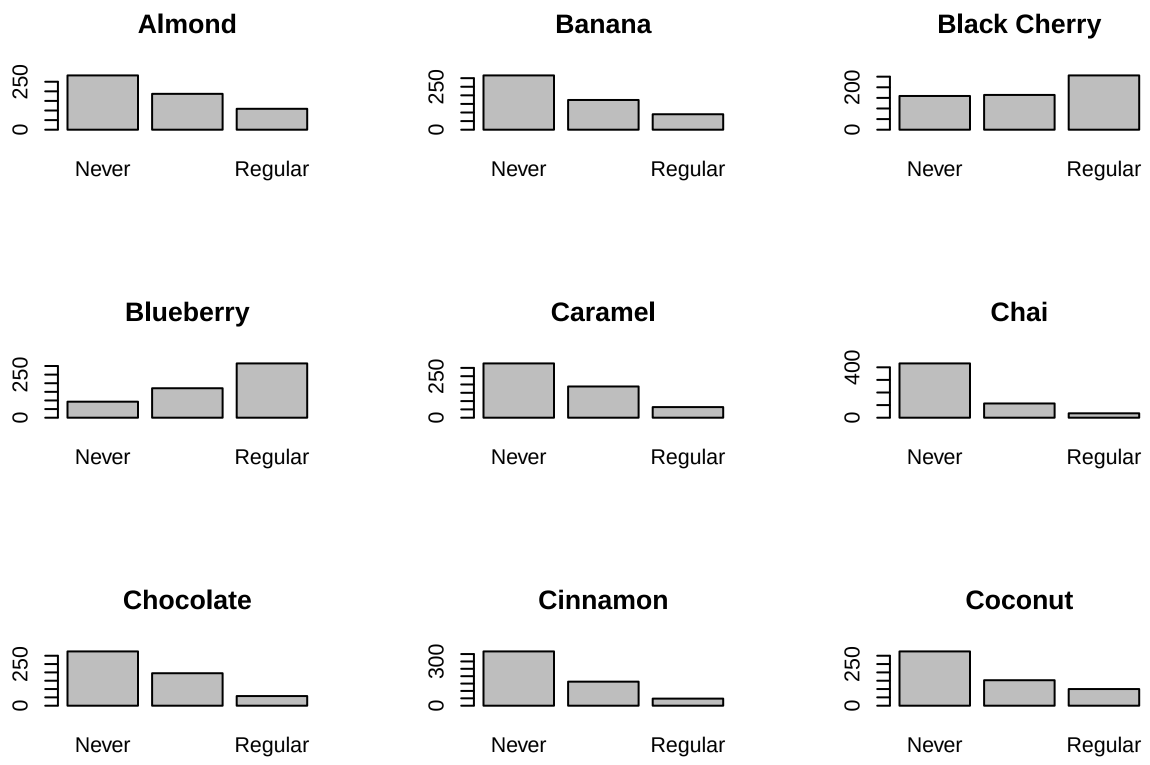
```
## [1] 579  38
```

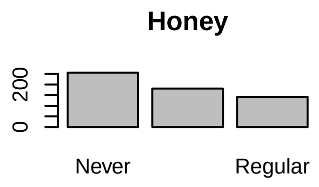
After removing unreliable samples, we assumed that NA values in Question 12 columns mean `Never` and replaced these NA values with 2.

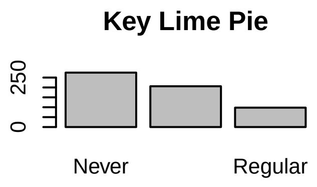
```r
# Replace missing values with value `2`
flavors <- survResponses[, c(15: 37)]
flavors[is.na(flavors)] <- 2
```

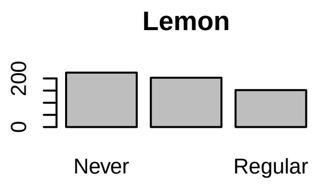
After data cleaning, we exported our data and use Tableau to create visualization for the our 4th slides. The code below will create similar bar charts.

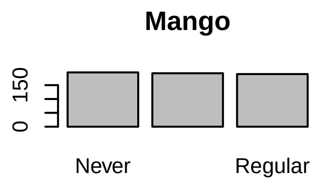
```r
################################
#      OUTPUT FOR SLIDE 4      #
################################

# Export data for visualization in Tableau
flavorData <- ifelse(flavors==0, "Regular", ifelse(flavors==2, "Never", "Occasionally"))
write.csv(flavorData,'flavorData')

# Create bar chart to understand preference
par(mfrow=c(3, 3));
for (i in 1:length(colnames(flavorData))) {
    x = flavorData[,i]
    plot(factor(x),main=colnames(flavorData)[i])
}
```

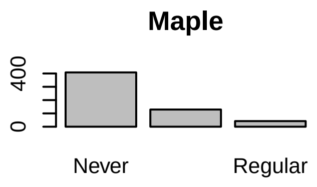

Almond
0
250
Never
Regular
Banana
0
250
Never
Regular
Black Cherry
0
200
Never
Regular
Blueberry
0
250
Never
Regular
Caramel
0
250
Never
Regular
Chai
0
400
Never
Regular
Chocolate
0
250
Never
Regular
Cinnamon
0
300
Never
Regular
Coconut
0
250
Never
Regular

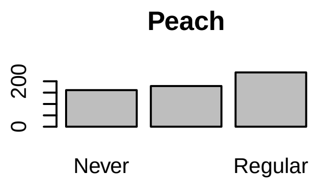

Honey
0
200
Never
Regular

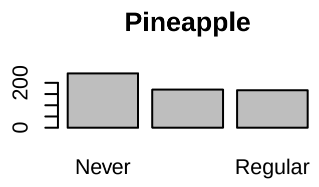

Key Lime Pie
0
250
Never
Regular

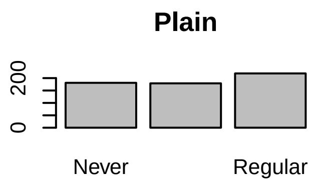

Lemon
0
200
Never
Regular

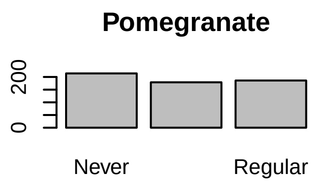

Mango
0
150
Never
Regular

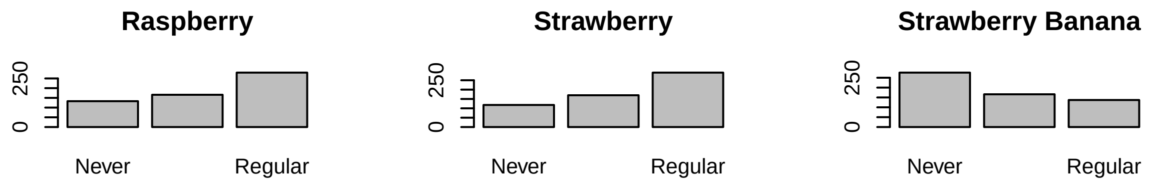

Maple
0
400
Never
Regular

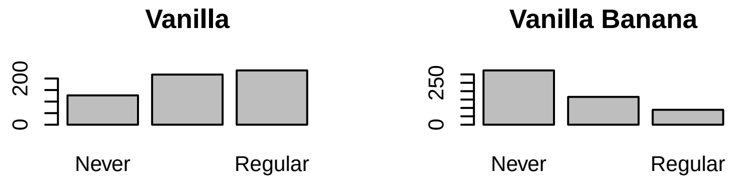

Peach
0
200
Never
Regular


Pineapple
0
200
Never
Regular


Plain
0
200
Never
Regular


Pomegranate
0
200
Never
Regular





Question 4: Predict the best set of next flavors to add to achieve the highest reach using survey data

Before conducting TURF analysis, we will encode the data as following: - Regularly: 4 - Occasionally: 1 - Never: 0

Originally, "Regularly" is encoded as 0, "Occasionally" as 1, and "Never" as 2.

```
# Encode data
flavorPurchase <- ifelse(flavors==0, 4, ifelse(flavors==2, 0, 1))
```

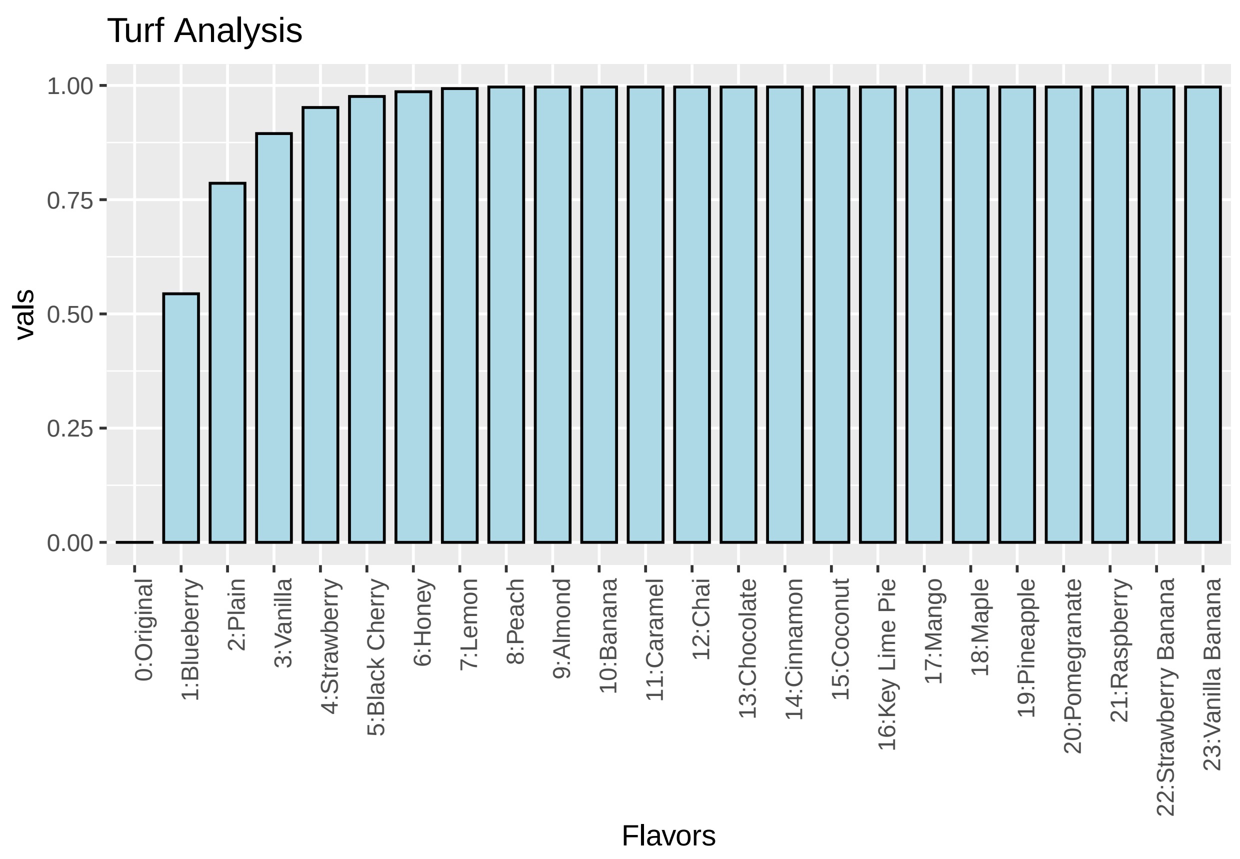
The code below will be used to do TURF Analysis. The codes are taken from `WegmansSurveyCase.Rmd` file provided by Prof. Lovett. We created a new function `measFreq` to measure total frequency.

```
#measReach: measures reach given set of options and data
measReach = function(data){
  if(is.null(dim(data))){
    ret = sum(data>1,na.rm=TRUE)/length(data)
  } else if(ncol(data)==1){
    ret = sum(data>1,na.rm=TRUE)/length(data)
  }
  else {
    ret = sum(apply(data>1,1,any),na.rm=TRUE)/nrow(data)
  }
}
```

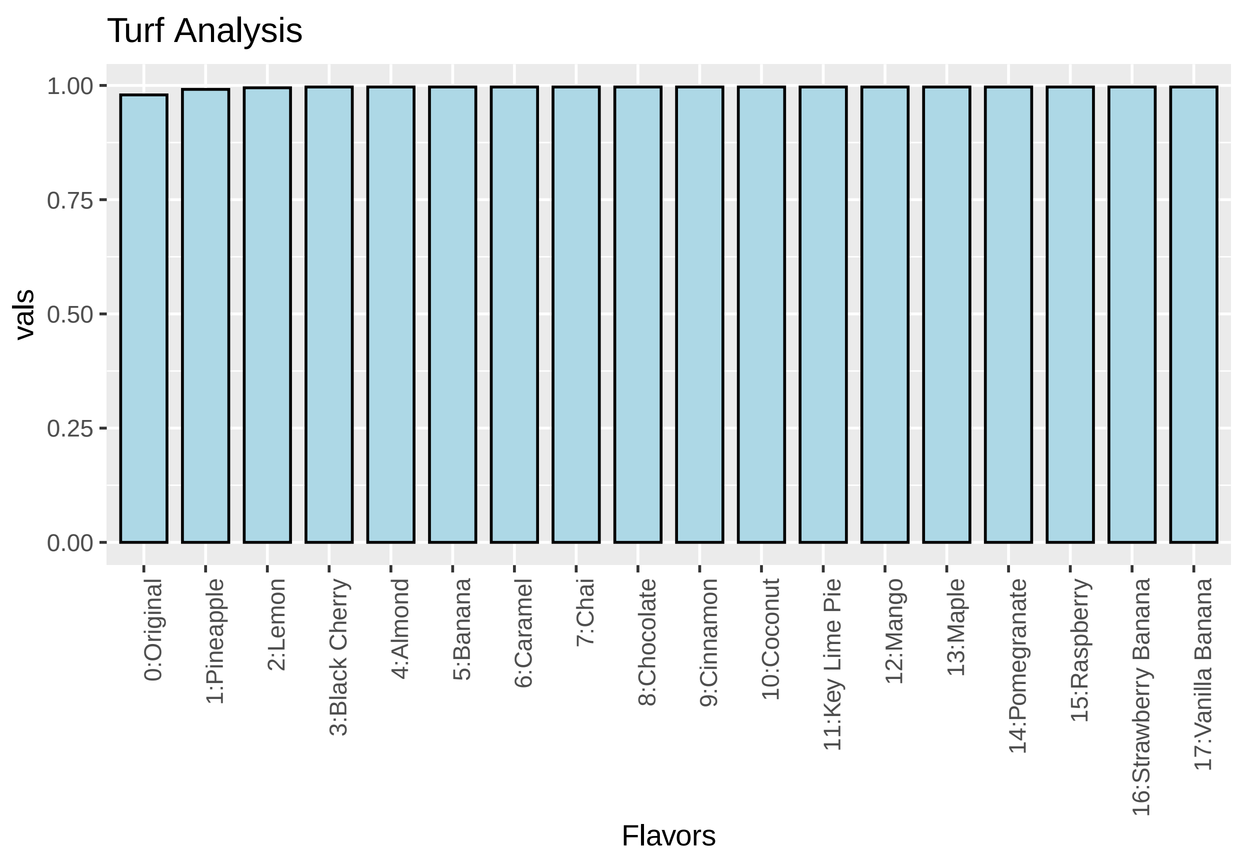
```
#measFreq: measures frequency given set of options and data
measFreq = function(data){
    if(is.null(dim(data))){
        ret = sum(data,na.rm=TRUE)
    } else if(ncol(data)==1){
        ret = sum(data,na.rm=TRUE)
    }
    else {
        ret = sum(apply(data,2,sum),na.rm=TRUE)
    }
    ret
}

#evalNext: evaluates the next set, nextSet using measure given existing set in data
evalNext = function(nextSet,set,data,measure=measReach){
  vals = numeric(length(nextSet))
  for(k in 1:length(nextSet)){
    if(length(set)==0){
      vals[k] = measure(data[,nextSet[k]])
    } else {
      vals[k] = measure(data[,c(set,nextSet[k])])
    }
  }
  vals
}

#evalFull: creates optimal full evaluation starting from origSet and
#considering remaining options fullSet
evalFull = function(fullSet,data,origSet=numeric(0),measure=measReach){
  #evaluates full set of cases picking optimal at each stage, returns a TURF object
  curSet = origSet;
  remSet = fullSet[!(fullSet%in%origSet)];
  K = length(remSet)
  optVals = numeric(K);
  ordSet = numeric(K);
  for(i in 1:K){
    tmpVals = evalNext(remSet,curSet,data,measure);
    k = which.max(tmpVals)
    optVals[i] = tmpVals[k]
    ordSet[i] = remSet[k]
    curSet = c(curSet,ordSet[i]);
    remSet = remSet[-k];
  }
  #creaets a "TURF object" containing ordSet, optVals, origSet, origVal, measure, and pnames
  turf = list(ordSet=ordSet,optVals=optVals,origSet=origSet,
              origVal=measure(data[,origSet]),measure=measure,
              pnames=colnames(data))
  class(turf)="TURF"
  turf
}

#creates ggplot barplot for a turf object
plot.TURF=function(turf,...){
  if(class(turf)!="TURF"){
    cat("Object not a turf.") #concatenate and print
  } else {
    df = with(turf,data.frame(vals = c(origVal,optVals),
                              titles=paste(0:length(ordSet),
                                           c("Original",pnames[ordSet]),sep=":")))
    dodge = position_dodge(width=.75);
    df$titles <- factor(df$titles, levels=df$titles) # Reorder the columns
    gp = ggplot(df,aes(y=vals,x=titles))
    gp + geom_bar(position=dodge,stat="identity",col=1,fill="lightblue",width=.75) +
      labs(title="Turf Analysis", x="Flavors") +
      theme(axis.text.x = element_text(angle = 90, hjust = 1))
  }
}
```

We will measure Reach and Frequency with and without the original set.

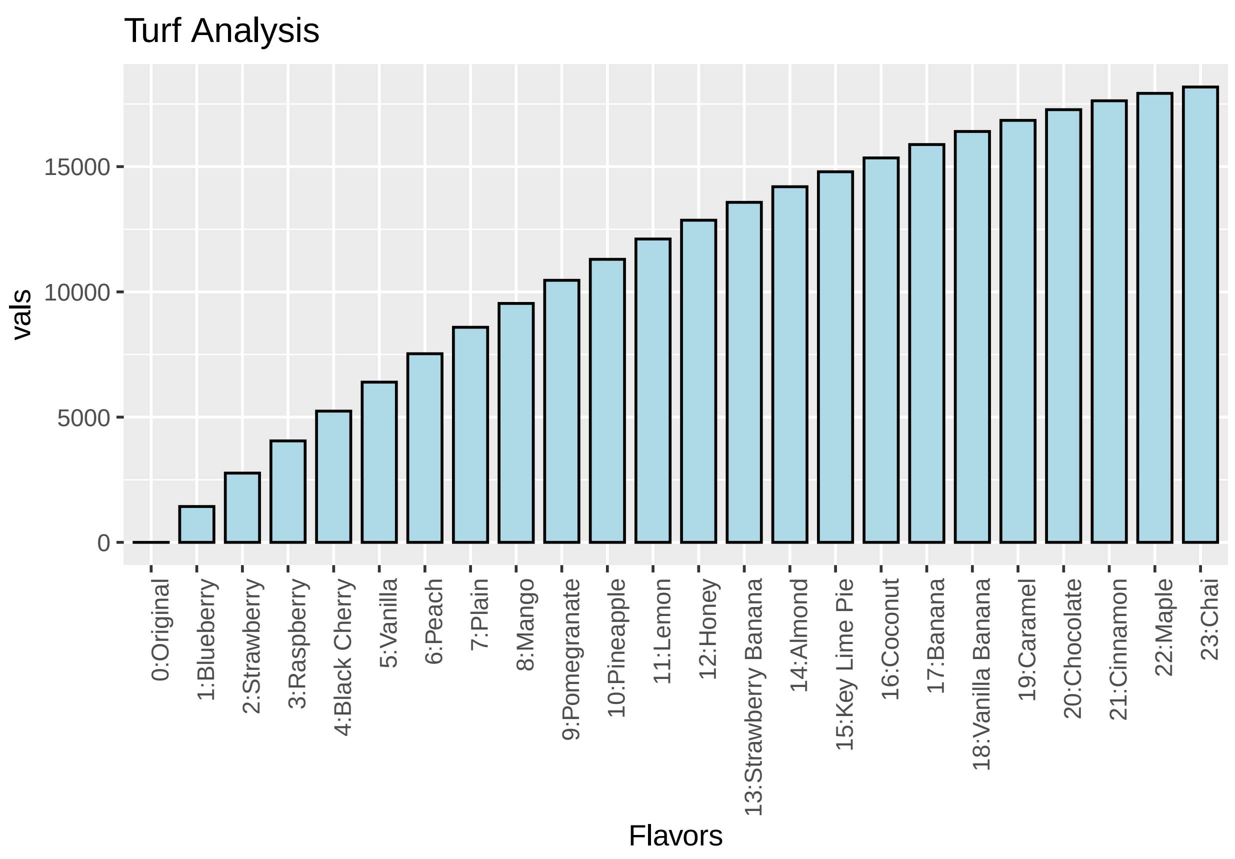
```
################################
#      OUTPUT FOR SLIDE 5      #
################################

originSet <- which(colnames(flavorPurchase) %in% c("Blueberry", "Honey", "Peach",
                                                   "Plain", "Strawberry", "Vanilla"))
fullSet <- 1:length(colnames(flavorPurchase))

# Reach with no original set
turf.1 = evalFull(fullSet, flavorPurchase, measure=measReach)
plot.TURF(turf.1)
```



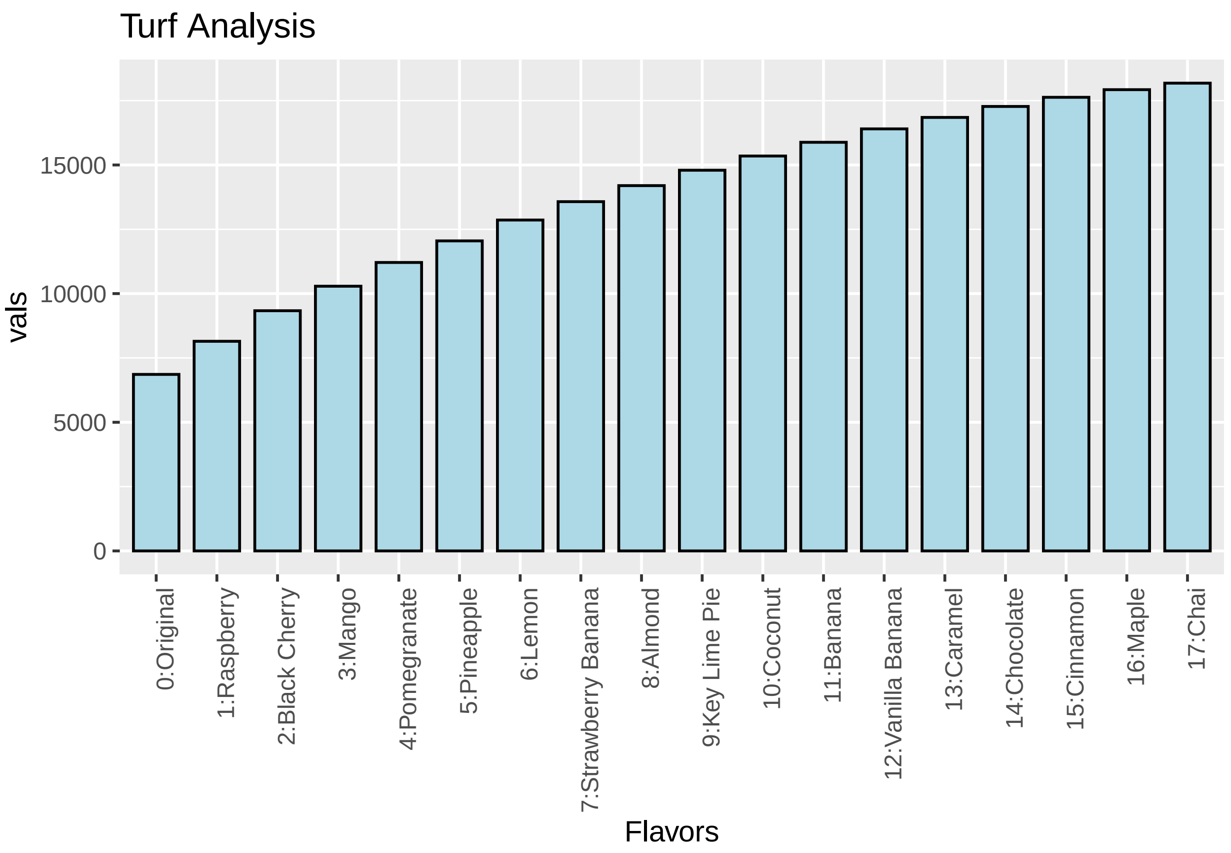
```r
# Reach with the original set
turf.2 = evalFull(fullSet, flavorPurchase, originSet, measure=measReach)
plot.TURF(turf.2)
```



```
# Frequency with no original set
turf.3 = evalFull(fullSet, flavorPurchase, measure=measFreq)
plot.TURF(turf.3)
```



```
# Frequency with the original set
turf.4 = evalFull(fullSet, flavorPurchase, originSet, measure=measFreq)
plot.TURF(turf.4)
```



Turf Analysis
vals
15000
10000
5000
0
0:Original
1:Raspberry
2:Black Cherry
3:Mango
4:Pomegranate
5:Pineapple
6:Lemon
7:Strawberry Banana
8:Almond
9:Key Lime Pie
10:Coconut
11:Banana
12:Vanilla Banana
13:Caramel
14:Chocolate
15:Cinnamon
16:Maple
17:Chai
Flavors