Business Plan for Central Perk

Pranav Srinivas Nagavelli, Samira Palagiri, Tarun Singh, William Thomas, Yuni Suter

03 November, 2018

Table of Contents

Business Problem

Central Perk owners believe that their customer base is loyal and that their sales have been a consistent year on year. However, these beliefs are based on intuition and anecdotal information; not from a rigorous exploration of their sales data. Additionally, they don't have a good sense of their customer's purchase details, purchase patterns, nor general volume distribution. Gaining this knowledge is critical to developing plans to increase revenue and smoothing demand patterns without alienating the existing customer base or relying on new customers.

Approach

We want to approve the Central Perk's hypothesis by investigating the data to conclude the following aspects:

- Are the sales increasing for Central Perk year-on-year?

- Whether the existing customer base is "LOYAL"?
- What type of customer segments exists in our data?
- Can we infer any product preferences based on the customer segments?

Data Exploration

Loading required libraries and associated packages.

```
#rm(list = ls())
#install.packages('naniar')
#install.packages('Amelia')
#install.packages('tidyverse')

library(stringr)
library(lubridate)
library(Amelia)
library(naniar)
library(dplyr)
library(magrittr)
library(ggplot2)
library(reshape2)
library(plyr)
library(arules)
library(arulesViz)
library(zoo)
library(tidyverse)
library(gridExtra)
library(DescTools)
```

Importing the data

The Central Perk data is spread across three CSV files. Importing the data from three files and summarizing the data.

```
knitr::opts_knit$set(root.dir = "C:\\Users\\Pranav\\Desktop\\MSBA\\6410 -
Exploratory Data Analytics and Visualization\\HW 4\\HW 4\\Central Perk")
#knitr::opts_chunk$set(cache=FALSE)

filenames <- dir("C:\\Users\\Pranav\\Desktop\\MSBA\\6410 - Exploratory Data
Analytics and Visualization\\HW 4\\HW 4\\Central Perk")
#Reading all files in the working folder
dfs <- do.call(rbind, lapply(filenames, function(x) cbind(read.csv(x,
stringsAsFactors = TRUE, na.strings = c('', NA, ' ')))))

summary(dfs)

##        Date              Time                        Category
##  4/29/17 :   570   12:36:11:    27   Coffee                :129648
##  10/7/17 :   559   10:12:27:    25   Extras                : 46022
```

```
##  6/25/17 :   559   10:37:03:    25   Food                    : 27852
##  04/14/18:   539   10:39:44:    25   Tea                     : 10749
##  6/3/17  :   534   10:08:35:    24   Non-Caffeinated Drinks:  4853
##  9/9/17  :   521   (Other) :221434   (Other)                 :  2436
##  (Other) :218279   NA's    :     1   NA's                    :     1
##         Item            Qty              Price.Point.Name  Gross.Sales
##  Drip SM  :39937   Min.   :-2.000   LG             :20343   $0.50   : 17424
##  Ice      :32250   1st Qu.: 1.000   Plain / Choc :10021   $3.00   : 16919
##  Latte SM :21824   Median : 1.000   Regular        :87249   $0.46   : 13780
##  Cappucino:18388   Mean   : 1.076   Regular Price:34632   $2.76   : 12300
##  Drip LG  :13450   3rd Qu.: 1.000   SM             :69230   $4.00   : 11089
##  (Other)  :95711   Max.   :30.000   NA's           :   86   (Other):150048
##  NA's     :    1   NA's   :1                                NA's    :     1
##    Discounts        Net.Sales          Tax
##  $0.00  :115219   $0.50  : 17416   $0.00  :93862
##  $0.00  :105592   $3.00  : 16897   $0.00  :46274
##  $-0.50 :    66   $0.46  : 13734   $0.04  :13764
##  ($3.50):    64   $2.76  : 12274   $0.37  : 7100
##  $-3.50 :    44   $4.00  : 11079   $0.35  : 6809
##  (Other):   575   (Other):150160   (Other):53751
##  NA's   :     1   NA's   :     1   NA's   :    1
##                Notes            Event.Type
##  Accidental Charge:    68   Payment:221473
##  Canceled Order   :    11   Refund :    87
##  Tips             :     4   NA's   :     1
##  Returned Goods   :     3
##  1 skim           :     1
##  (Other)          :     3
##  NA's             :221471
##                                   Customer.ID
##  588ffc8ace2a17134c64d10a8f0cad0ffb1441e6:   619
##  af495134246a402d512789fa9f8b700dccb1142c:   602
##  5254f7a15c667a76046812ac3735cc6feddf5769:   518
##  216e58487a41161707f827b4dbefac67880a8db2:   487
##  453e71627d5904df0318f6f5f0529acc8dd1f46a:   430
##  (Other)                                 :140828
##  NA's                                    : 78077
```

```r
head(dfs)
```

```
##       Date     Time Category      Item Qty Price.Point.Name Gross.Sales
## 1 12/31/17 16:57:33   Coffee Cappucino   1          Regular      $3.90
## 2 12/31/17 16:37:29      Tea    Tea SM   1               SM      $2.30
## 3 12/31/17 16:33:08   Coffee Cappucino   1          Regular      $3.90
## 4 12/31/17 16:33:08   Coffee  Espresso   1          Regular      $3.22
## 5 12/31/17 16:31:29   Coffee  Espresso   1          Regular      $3.22
## 6 12/31/17 16:11:06     Food  Alm Rasp   1    Regular Price      $4.13
##   Discounts Net.Sales   Tax Notes Event.Type
## 1     $0.00     $3.90 $0.35  <NA>    Payment
## 2     $0.00     $2.30 $0.20  <NA>    Payment
```

```
## 3    $0.00     $3.90  $0.35   <NA>    Payment
## 4    $0.00     $3.22  $0.28   <NA>    Payment
## 5    $0.00     $3.22  $0.29   <NA>    Payment
## 6    $0.00     $4.13  $0.37   <NA>    Payment
##                                 Customer.ID
## 1                                      <NA>
## 2                                      <NA>
## 3 2a90ca0a266d3e357b693cc366e59e91569726de
## 4 2a90ca0a266d3e357b693cc366e59e91569726de
## 5                                      <NA>
## 6 398cfea2ef68cc372b4593ed54991917a3e1bba3
```

Conclusion The data has 221561 rows and 13 columns with missing values which have to be treated and few of the columns assigned to different class.

Cleaning

Based on the summary of the data, we can want to explore the following issues:

- We see that few of the numeric columns have characters which have to be treated
- There are missing values that have to be treated

Further crucial data cleaning steps include:

- Check for duplicate records
- Assign right class types to the columns

The following pattern on the left is replaced with their appropriate item in the data:

pattern1	pattern2
Colom, Columbia, Colombianbian	Columbian
CostaR	Costa Rican
Ethio, Ethiopianpian	Ethiopian
Hair	Hairbender
Alm Rasp	Almond Rasp
Chai	Tea

```
# Create a copy of the data frame
df <- dfs

# Checking for class of variables
sapply(df, class)

##             Date             Time         Category             Item
##         "factor"         "factor"         "factor"         "factor"
##              Qty Price.Point.Name      Gross.Sales        Discounts
##        "integer"         "factor"         "factor"         "factor"
##        Net.Sales              Tax            Notes       Event.Type
##         "factor"         "factor"         "factor"         "factor"
```

```
##      Customer.ID
##         "factor"

# Converting Date column to date class format
df$Date <- as.Date(df$Date,"%m/%d/%y")

# Extracting Year from Date
#df$year <-  as.factor(year(df$Date))

# Filtering the factor class columns that has to be numeric
cl <- c("Gross.Sales","Net.Sales","Tax", "Discounts")

# Replacing the characters from the above columns and converting them to
numeric
df[,cl] <- sapply(df[,cl], function(x) str_replace_all(x, "[$)]", "" ))
df[,cl] <- sapply(df[,cl], function(x) str_replace_all(x, "[(]", "-" ))
df[,cl] <- sapply(df[,cl], as.numeric)

unique(df$Category)

##  [1] Coffee                 Tea                    Food
##  [4] Extras                 Non-Caffeinated Drinks Beans
##  [7] Cereal                 Beers                  None
## [10] <NA>
## 9 Levels: Beans Beers Cereal Coffee Extras ... Tea

# Removing the NA category
df <- df[!is.na(df$Category)& (df$Category != "None"),]

# Clubbing the "Regular" and "Regular Price" in Price.Point.Name into
"Regular"

df$Price.Point.Name <- as.factor(gsub("Regular Price", "Regular",
df$Price.Point.Name))

unique(sort(df$Item))

##  [1] 12oz Costa Rican  12oz Hair         12oz Snow Day
##  [4] Abita Amber       Alm Rasp          Almond
##  [7] Almond Rasp       Americano         Apple
## [10] Au Lait           Au Lait LG        Banana
## [13] Blue Point Lager  Cappucino         Cereal
## [16] Chai              Colo/Indo         Colombian
## [19] Cortado           CostaR            Croissant
## [22] DblChoc           Donut             Drip LG
## [25] Drip SM           EspMac            Espresso
## [28] Ethiopian         Extra Shot        Financier
## [31] Goji              Guat/Papua        Hairbender
## [34] Hot Chocolate     Ice               Lagunitas IPA
## [37] Latte LG          Latte SM          Lenka Bar
```

```
## [40] Macchiato         MapleVal          Mocha
## [43] Perrier           Peru/Costa        Purple Haze
## [46] San Pellegrino    Sierra Nevada PA  Six Point Pilsner
## [49] Snow Day          Soy               Steamed Milk
## [52] Stubby            SweetPo           Tea LG
## [55] Tea SM            12oz Columbia     12oz Ethio
## [58] 12oz Gautemala    ð\215Lemonadeð\215  12oz Bella
## [61] 12oz Colom        12oz Honduras     12oz Indo
## [64] Alm Blu           Anchor            Crisp
## [67] Lagunitas         Oat               Ommegang
## 70 Levels: 12oz Costa Rican 12oz Hair 12oz Snow Day ... Ommegang
```

```
# Removing Size details from Item list and editing the text
df$Item <- as.factor(df$Item %>% gsub("\\bSM\\b", "", .) %>% gsub("\\bLG\\b",
"", .) %>% gsub("\\b12oz\\b", "", .) %>% gsub("\\bCostaR\\b", "Costa Rican",
.) %>% gsub("\\bColom\\b", "Colombian", .) %>% gsub("\\bColombianbian\\b",
"Colombian", .) %>% gsub("\\bColumbia\\b", "Colombian", .) %>%
gsub("\\bEthio\\b", "Ethiopian", .) %>% gsub("\\bEthiopianpian\\b",
"Ethiopian", .) %>% gsub("\\bHair\\b", "Hairbender", .) %>%
gsub("ðY\u008d<LemonadeðY\u008d<", "Lemonade", .) %>% gsub("\\bAlm Rasp\\b",
"Almond Rasp", .) %>% gsub("\\bChai\\b", "Tea", .) %>% trimws())

unique(sort(df$Item))
```

```
##  [1] Abita Amber       Alm Blu           Almond
##  [4] Almond Rasp       Americano         Anchor
##  [7] Apple             Au Lait           Banana
## [10] Bella             Blue Point Lager  Cappucino
## [13] Cereal            Colo/Indo         Colombian
## [16] Cortado           Costa Rican       Crisp
## [19] Croissant         DblChoc           Donut
## [22] Drip              ð\215Lemonadeð\215  EspMac
## [25] Espresso          Ethiopian         Extra Shot
## [28] Financier         Gautemala         Goji
## [31] Guat/Papua        Hairbender        Honduras
## [34] Hot Chocolate     Ice               Indo
## [37] Lagunitas         Lagunitas IPA     Latte
## [40] Lenka Bar         Macchiato         MapleVal
## [43] Mocha             Oat               Ommegang
## [46] Perrier           Peru/Costa        Purple Haze
## [49] San Pellegrino    Sierra Nevada PA  Six Point Pilsner
## [52] Snow Day          Soy               Steamed Milk
## [55] Stubby            SweetPo           Tea
## 57 Levels: Abita Amber Alm Blu Almond Almond Rasp Americano ... Tea
```

```
# Creating the date time field
df$DateTime <- with(df, as.POSIXct(paste(as.Date(Date, format="%m/%d/%y"),
Time)))

# Dropping refunds from the data and removing the redundant data
```

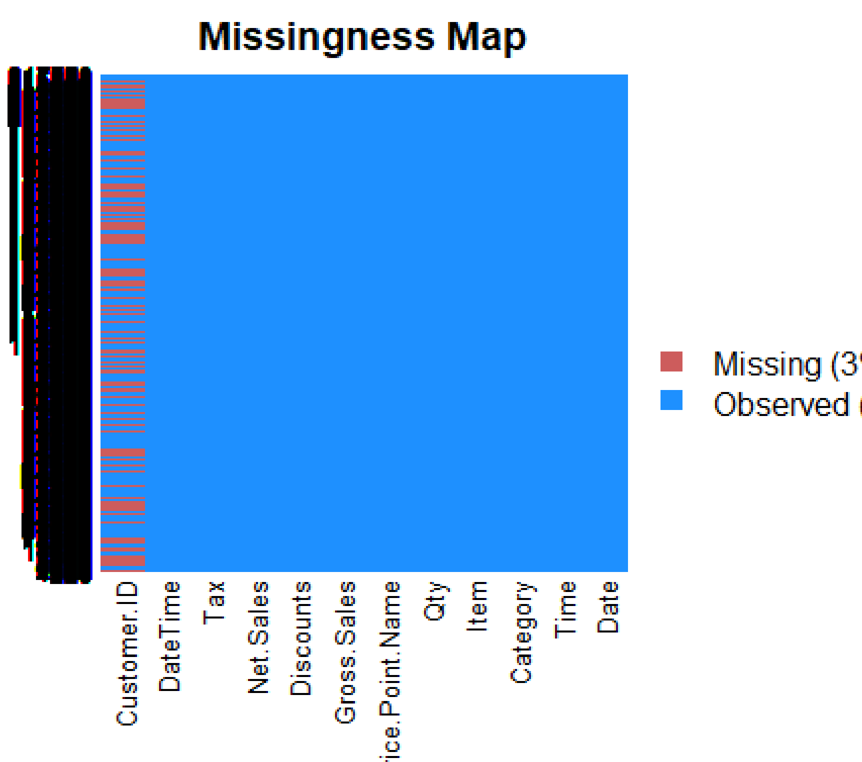
```r
df <- df[(df$Event.Type != "Refund"),]
df$Event.Type <- NULL

# Check for non-NA notes in the dataframe
sum(!is.na(df$Notes))
```

```
## [1] 3
```

```r
# The notes are mostly available for refunds with only 3 notes for non-
refunds
df$Notes <- NULL

# Checking for duplicated rows
sum(duplicated(df)) >1
```

```
## [1] TRUE
```

```r
# Filtering for non-duplicated rows
DF <-  df[!duplicated(df), ]

missmap(DF)
```




```r
summary(DF)
```

```
##        Date                 Time                              Category
##  Min.   :2016-07-15   10:12:27:    25    Coffee                   :128227
##  1st Qu.:2017-02-09   10:39:44:    25    Extras                   : 43943
##  Median :2017-08-10   10:08:35:    24    Food                     : 27767
```

```
## Mean   :2017-08-12  15:13:45:    24  Tea                      : 10710
## 3rd Qu.:2018-03-02  10:13:36:    23  Non-Caffeinated Drinks:  4833
## Max.   :2018-08-24  10:18:39:    23  Beans                    :  1466
##                     (Other) :217650  (Other)                  :   848
##        Item           Qty            Price.Point.Name   Gross.Sales
## Drip      :52613   Min.   : 1.000   LG          : 20289   Min.   : 0.460
## Ice       :30451   1st Qu.: 1.000   Plain / Choc:  9987   1st Qu.: 2.500
## Latte     :26699   Median : 1.000   Regular     :119460   Median : 3.250
## Cappucino:18299    Mean   : 1.078   SM          : 68058   Mean   : 3.265
## Americano:10917    3rd Qu.: 1.000                          3rd Qu.: 4.000
## Tea       :10721   Max.   :30.000                          Max.   :90.000
## (Other)   :68094
##   Discounts            Net.Sales          Tax
## Min.   :-10.500000  Min.   : 0.230   Min.   :0.0000
## 1st Qu.:  0.000000  1st Qu.: 2.500   1st Qu.:0.0000
## Median :  0.000000  Median : 3.250   Median :0.0000
## Mean   : -0.005899  Mean   : 3.259   Mean   :0.1018
## 3rd Qu.:  0.000000  3rd Qu.: 4.000   3rd Qu.:0.2400
## Max.   :  0.000000  Max.   :90.000   Max.   :6.1200
##
##                                   Customer.ID
## 588ffc8ace2a17134c64d10a8f0cad0ffb1441e6:   619
## af495134246a402d512789fa9f8b700dccb1142c:   601
## 5254f7a15c667a76046812ac3735cc6feddf5769:   514
## 216e58487a41161707f827b4dbefac67880a8db2:   481
## 4a0e1d6c4643262eb250de6e591b999de4b85217:   416
## (Other)                                 :138045
## NA's                                    : 77118
##    DateTime
## Min.   :2016-07-15 10:59:47
## 1st Qu.:2017-02-09 15:30:36
## Median :2017-08-10 12:12:25
## Mean   :2017-08-13 08:50:02
## 3rd Qu.:2018-03-02 16:15:10
## Max.   :2018-08-24 10:13:13
##
```

```r
rm(df)
```

Assumption

- We ignored refund transactions for this particular exercise as the data is not enough to conclude anything based on the refunds
- We found an overlap between notes and refunds in the data. After removing the records with refunds only 3 notes are left in the data. We removed them from our analysis

Inference:

- Duplicate records were found in the data and removed

- None and Null categories are found and removed from the data
- Removed the redundant size information (SM, LG, 12oz) from the Item

- Following are the items on the left which are clubbed to the right one based on their right fit
- 35% of the records have a missing Customer ID

Visualization

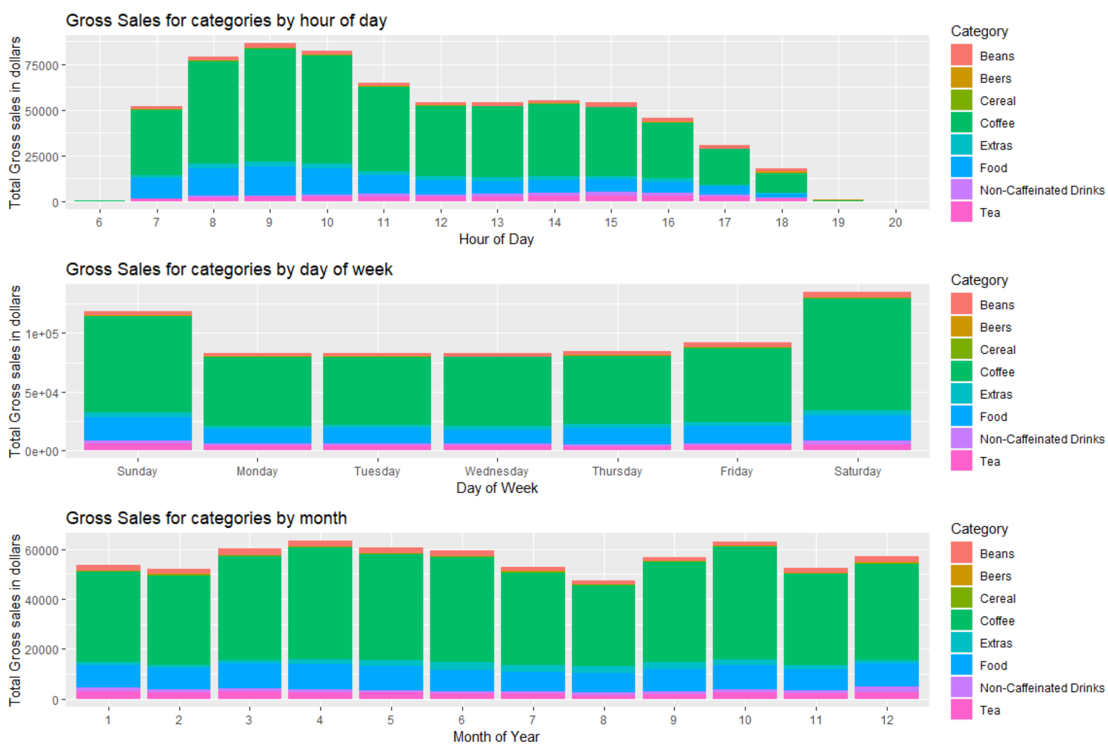
We want to explore the data further to evaluate the impact of sales based on the following factors:

- Month of the Year
- Day of the Week
- Hour of the Day

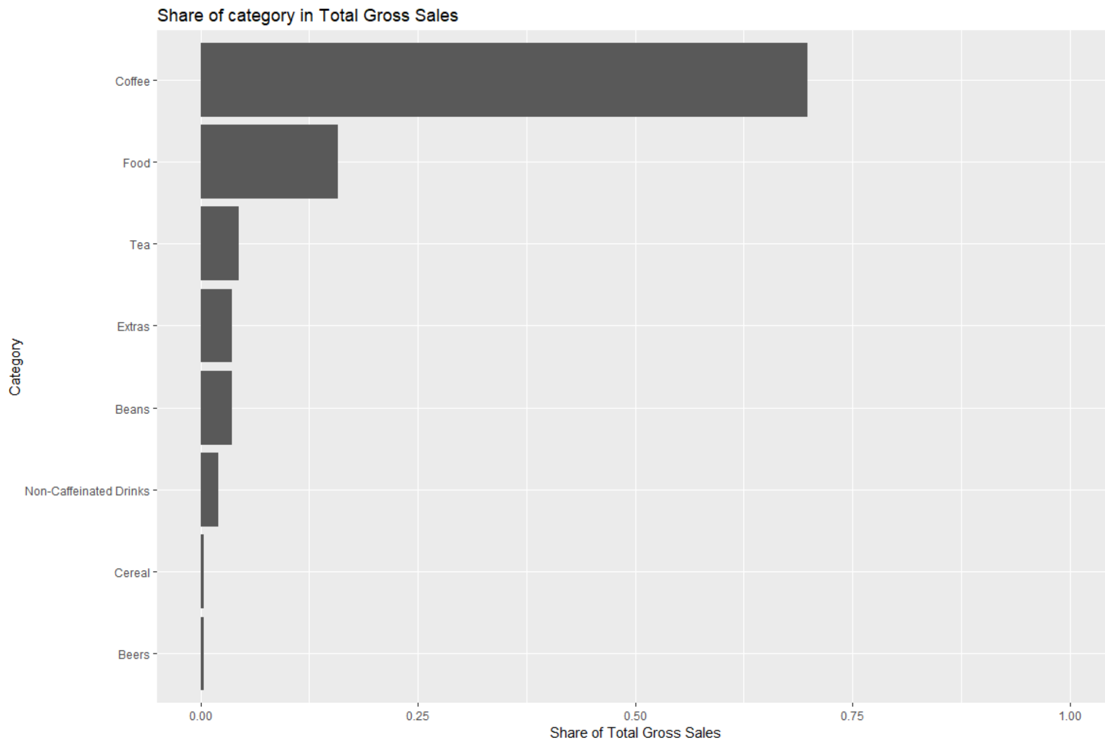
Also, we want to see the share of different categories in the total gross sales for Central Perk.

```
# Copy of the cleaned data
cp <- DF

## See the range of the data
range(DF$Date)

## [1] "2016-07-15" "2018-08-24"

y1_start <- as.Date("20160801","%Y%m%d")
y1_end <- as.Date("20170731","%Y%m%d")
y2_start <- as.Date("20170801","%Y%m%d")
y2_end <- as.Date("20180731","%Y%m%d")

cp <- cp[(cp$Date >= y1_start) & (cp$Date <= y2_end),]

cp$yid <- as.factor(ifelse(((cp$Date >= y1_start) & (cp$Date <=
y1_end)),"y1","y2"))

# Extracting the month, day of week and hour of the day level details for
analysis
cp$Hour <- as.factor(hour(hms(cp$Time)))
cp$Month <- as.factor(month((cp$Date)))
cp$Day <- as.factor(wday((cp$Date))) # 1 is Sunday

p1 <- ggplot(cp, aes(x = Hour, y = Gross.Sales, fill = Category)) +
geom_bar(stat = "identity")  + labs(x="Hour of Day", y = "Total Gross sales
in dollars",
       title="Gross Sales for categories by hour of day")

cp$Day_Name <- mapvalues(cp$Day,
```

```
          from=c("1","2","3","4","5","6","7"),

to=c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"))

p2 <- ggplot(cp, aes(x = Day_Name, y = Gross.Sales, fill = Category)) +
geom_bar(stat = "identity")  + labs(x="Day of Week", y = "Total Gross sales
in dollars",
       title="Gross Sales for categories by day of week")

cp$Day_Name <- NULL

p3 <- ggplot(cp, aes(x = Month, y = Gross.Sales, fill = Category)) +
geom_bar(stat = "identity")  + labs(x="Month of Year", y = "Total Gross sales
in dollars",
       title="Gross Sales for categories by month")

grid.arrange(p1, p2, p3, nrow=3)
```




```
cat_data <- cp %>% group_by(Category) %>% dplyr::summarise(s =
sum(Gross.Sales))
cat_data$s <- cat_data$s/sum(cp$Gross.Sales)
ggplot(cat_data, aes(x = reorder(Category,s), y = s)) + ylim(0,1) +
coord_flip()+ geom_bar(stat = "identity")  + labs(x="Category", y = "Share of
Total Gross Sales",
       title="Share of category in Total Gross Sales") + theme_update()
```




```
rm(cat_data)
```

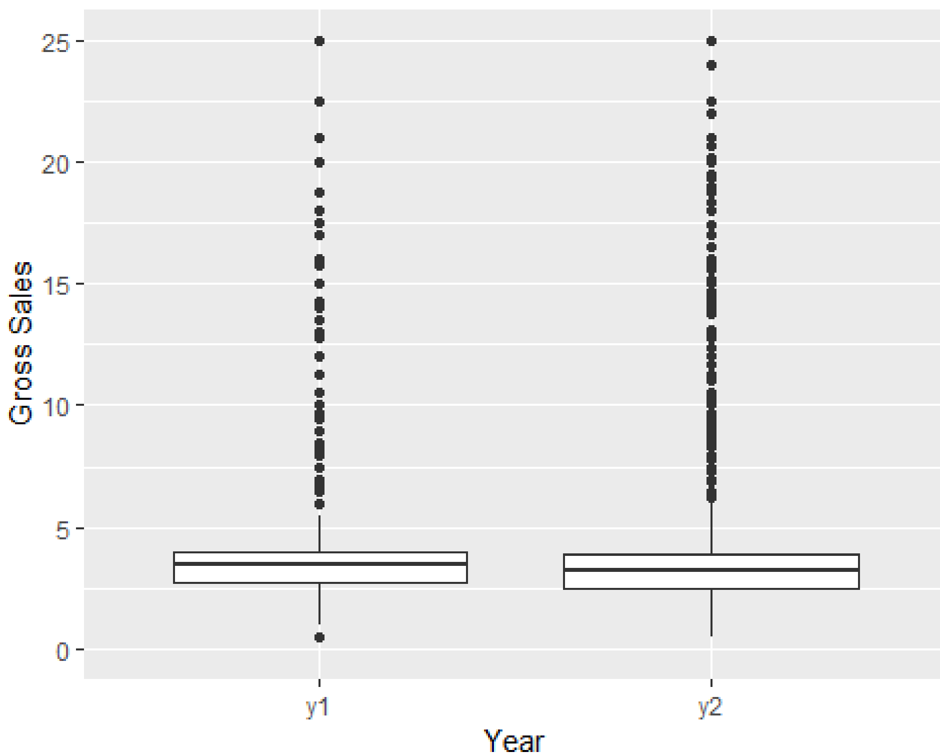
Inference

- The gross sales were prominent in the mornings from the window 8-11 and thereafter the sales dipped as the day progresses
- The demand for coffee and food follows the same trend as the hour of the day sales trends
- The sales on weekends were more than the weekdays
- Coffee has the major share of gross sales which is 70% of the total gross sales

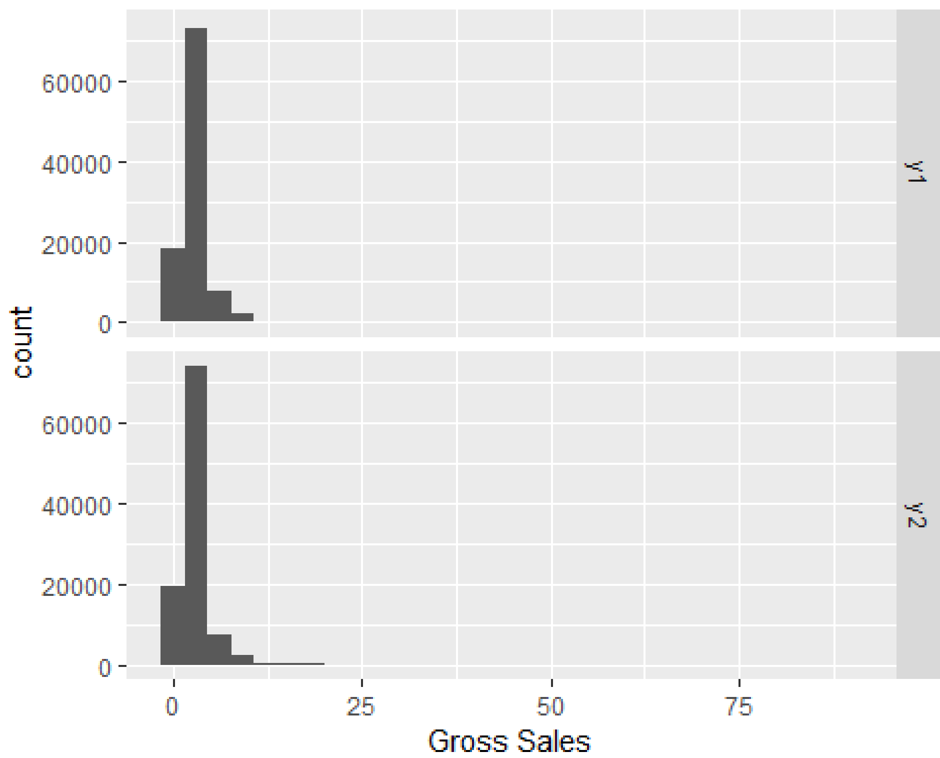
Statistical techniques to test the sales growth YoY

We want to check the hypothesis of the central Perk that *the sales for the coffee shops are increasing year on year* by performing statistical tests

```
ggplot(cp, aes(x = yid, y = Gross.Sales)) + geom_boxplot() + ylim(0,25) +
labs(x = "Year",y = "Gross Sales")

## Warning: Removed 109 rows containing non-finite values (stat_boxplot).
```

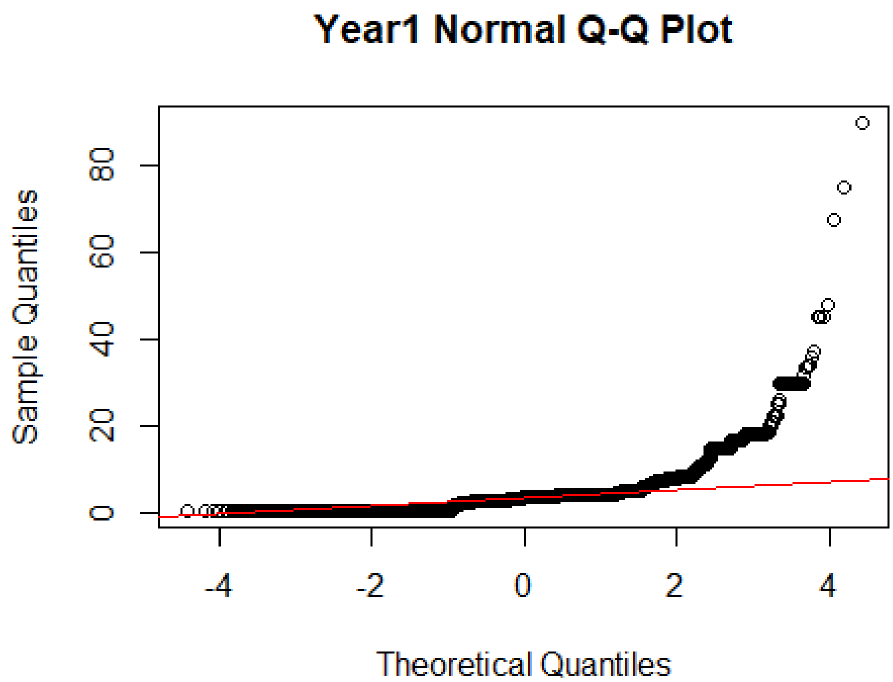


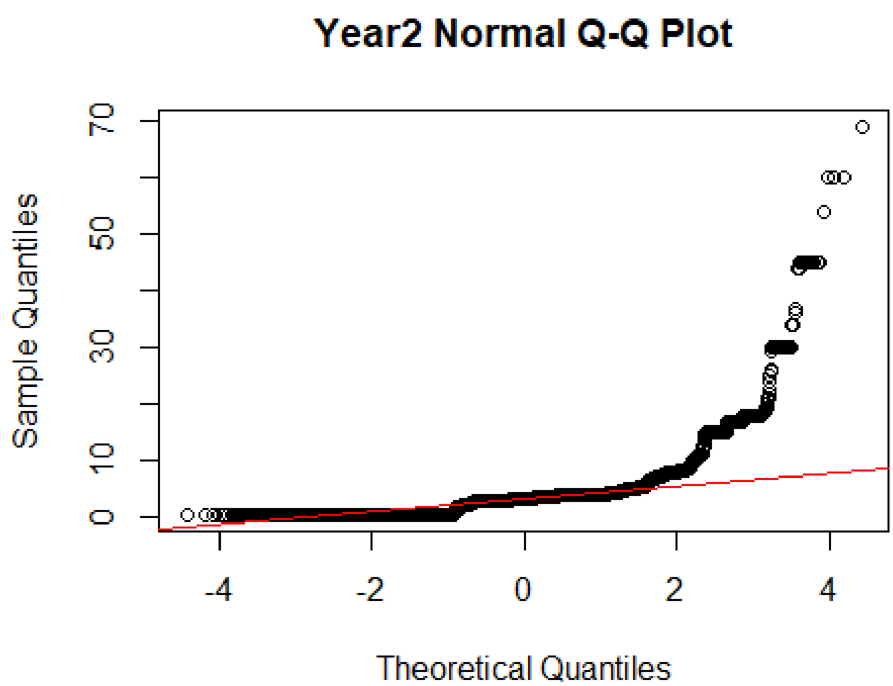

```r
y1 <- cp[cp$yid == "y1",]$Gross.Sales
y2 <- cp[cp$yid == "y2",]$Gross.Sales

cp %>% group_by(yid) %>%
  dplyr::summarise(
    count = n(),
    mean = mean(Gross.Sales, na.rm = TRUE),
    sd = sd(Gross.Sales, na.rm = TRUE)
  )

## # A tibble: 2 x 4
##   yid    count  mean    sd
##   <fct>  <int> <dbl> <dbl>
## 1 y1    102084  3.34  2.15
## 2 y2    103927  3.25  2.23

# Histogram to check normality
ggplot(cp, aes(x=Gross.Sales)) + geom_histogram() + labs(x = "Gross Sales") +
facet_grid(yid~.)

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Normality check based on Kolmogorov-Smirnov test
ks.test(y1,y='pnorm',alternative='two.sided');ks.test(y2,y='pnorm',alternativ
e='two.sided')

## Warning in ks.test(y1, y = "pnorm", alternative = "two.sided"): ties
should
## not be present for the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  y1
## D = 0.79688, p-value < 2.2e-16
## alternative hypothesis: two-sided

## Warning in ks.test(y2, y = "pnorm", alternative = "two.sided"): ties
should
## not be present for the Kolmogorov-Smirnov test

##
##  One-sample Kolmogorov-Smirnov test
##
## data:  y2
## D = 0.78076, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

```r
# Normal Q-Q plot
qqnorm(y1, main = "Year1 Normal Q-Q Plot");qqline(y1,col='red');qqnorm(y2,
main = "Year2 Normal Q-Q Plot");qqline(y2,col='red')
```





```
#Wilcoxon Test
wilcox.test(y2, y1,exact = FALSE, alternative = "less")

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  y2 and y1
## W = 4.885e+09, p-value < 2.2e-16
## alternative hypothesis: true location shift is less than 0
```

Assumptions Since the data is from Mid July 2016 to partial month on August 2018, we took the start of the year from August to July for our analysis.

Year	Period
y1	August 2016 - July 2017
y2	August 2017 - July 2018

Inference

- From the box plot we can see that the year2 (y2) mean sales are slightly less than the year1 (y1) gross sales
- The p values based on Kolmogorov-Smirnov test suggests that the sales from both years are not normal. This can be backed up by the evidence of histogram and Q-Q plots. Thus, we can't use t-test here
- We used Wilcoxon statistical technique to test our hypothesis. The null and alternative hypothesis for our test is as follows:

- Null Hypothesis: The sales in the y2 are greater than or equal to sales in y1
- Alternative Hypothesis: The sales in the y2 are less than sales in y1

Conclusion Based on the p-value < 2.2e-16 suggests there is enough evidence to reject the null hypothesis

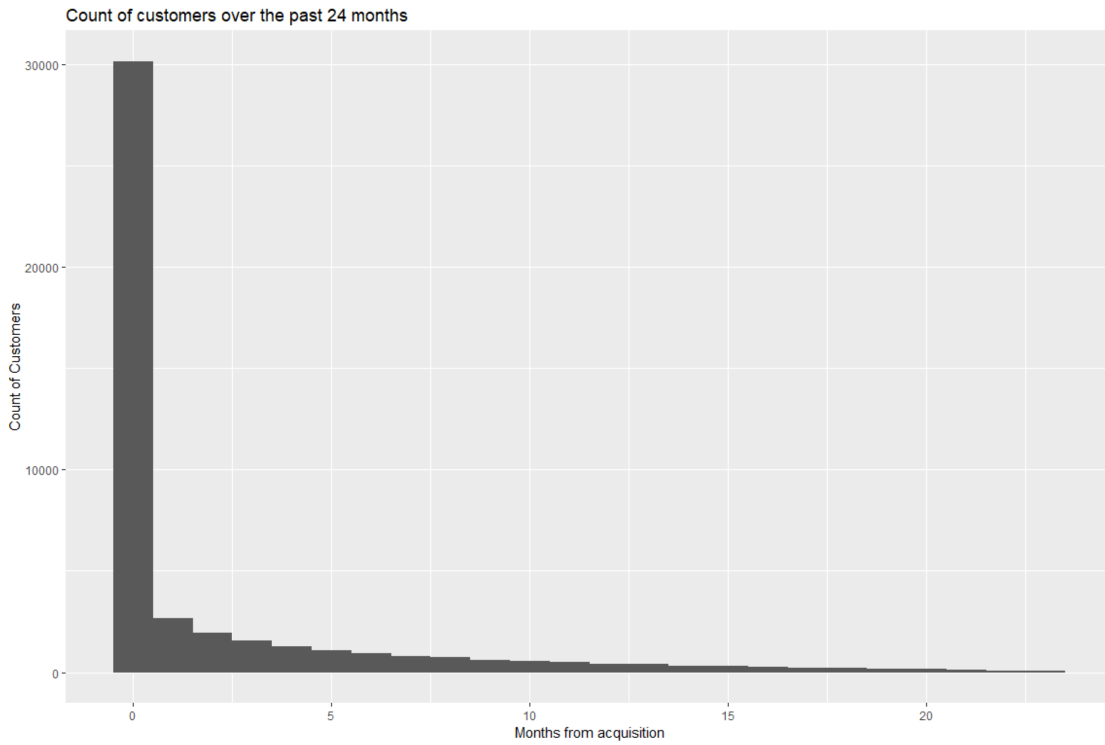
Loyalty check for existing customer base through retention analysis

Data Transformation

After our preliminary analysis on the given dataset, we went onto explore the customer information and answer the question "whether the existing customer base is loyal?"

```
## Getting customers with non null customer id
customers <- cp[!is.na(cp$Customer.ID),]

unique_cust <- customers %>% group_by(Customer.ID) %>%
dplyr::summarise(Acq_date = min(Date)) %>%
  mutate(Acq_month = format(Acq_date,"%Y-%m"))
```

```r
customer_wise_order <- customers %>%
  mutate(ym = format(Date, "%Y-%m")) %>%
  group_by(Customer.ID, ym) %>%
  dplyr::summarise(Orders = n(),
            Gross_Sales = sum(Gross.Sales),
            Net_Sales = sum(Net.Sales),
            Qty = sum(Qty))

final_cust <- inner_join(unique_cust, customer_wise_order, by="Customer.ID")
final_cust$x <- as.yearmon(final_cust$ym,"%Y-%m")
final_cust$y <- as.yearmon(final_cust$Acq_month,"%Y-%m")

final_cust$diff <- round((final_cust$x - final_cust$y) * 12)
str(final_cust)

## Classes 'tbl_df', 'tbl' and 'data.frame':    45631 obs. of  11 variables:
##  $ Customer.ID: Factor w/ 31822 levels
"0000063491abb5e5068c300be93c56c0841250da",..: 1 1 2 3 4 5 6 7 8 9 ...
##  $ Acq_date   : Date, format: "2016-09-14" "2016-09-14" ...
##  $ Acq_month  : chr  "2016-09" "2016-09" "2017-04" "2017-04" ...
##  $ ym         : chr  "2016-09" "2017-02" "2017-04" "2017-04" ...
##  $ Orders     : int  3 1 1 4 3 1 1 2 2 3 ...
##  $ Gross_Sales: num  9.5 3.75 18 14.5 8.26 3 5 4.75 3.5 7.75 ...
##  $ Net_Sales  : num  9.5 3.75 18 14.5 8.26 3 5 4.75 3.5 7.75 ...
##  $ Qty        : int  3 1 1 4 3 1 1 2 2 3 ...
##  $ x          : 'yearmon' num  Sep 2016 Feb 2017 Apr 2017 Apr 2017 ...
##  $ y          : 'yearmon' num  Sep 2016 Sep 2016 Apr 2017 Apr 2017 ...
##  $ diff       : num  0 5 0 0 0 0 0 0 0 0 ...

ggplot(final_cust, aes(x=diff)) + geom_histogram(bins = 24) + labs(title =
"Count of customers over the past 24 months", x = "Months from acquisition",
y = "Count of Customers") + theme_update()
```



Assumption

- All the records with NA customer ID which are 35% of the records in the original data are customer's who are not loyalty members for the coffee shop. Therefore, we removed those records from our analysis.
- Since July 2016 and August 2018 month don't have the complete month data, we analyzed the data from August 2017 to July 2018 which is a two-year data

Year	Period
y1	August 2016 - July 2017
y2	August 2017 - July 2018

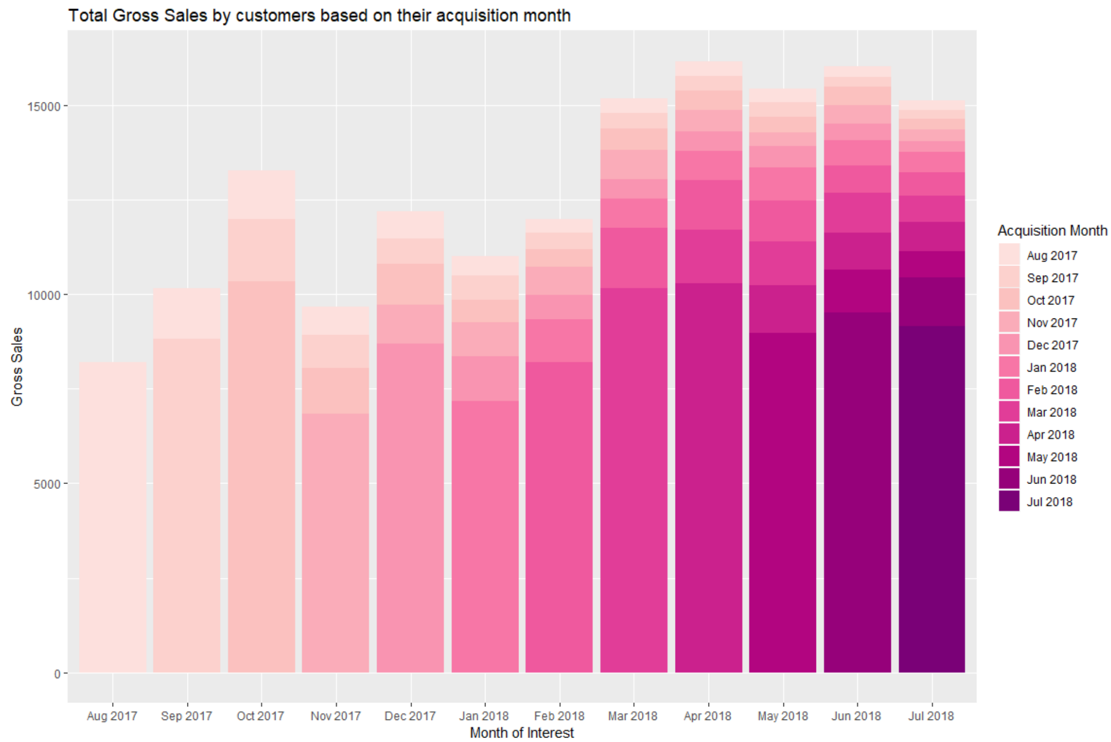
Conclusion The new customer level data has 30151 unique customers. The drop of customers is evident from the graph "Count of customers over the past 24 months". The customers acquired in the month-0 have dropped from 30151 to less than 2500 in the next month which is close to a 90% drop in their customer base.

Total gross sales by customers based on the acquisition month

We looked at the last 12 month data to see the customers acquisition and retention patterns. Also, the contribution of sales for each month was tracked based on the month they were acquired.

```
# Filtering the customers who were acquired after August 2017
final_cust_y2 <- final_cust[final_cust$Acq_month > '2017-07',]
```

```
#Pivot up on to get repeat users
pivot_cust <- final_cust_y2 %>%
            select (Customer.ID, Acq_month, diff) %>%
        group_by(Acq_month, diff) %>% dplyr::summarise(repeat_users =
n_distinct(Customer.ID))

pivot_cust$cmonth <- as.Date(as.yearmon(pivot_cust$Acq_month))

pivot_cust$cmonth <- as.yearmon(AddMonths(pivot_cust$cmonth,
pivot_cust$diff))
pivot_cust$Acq_month <- as.yearmon(pivot_cust$Acq_month)
pivot_cust$diff <- NULL

pivot_cust <- pivot_cust[,c(1,3,2)]
pivot_cust1 <- spread(pivot_cust, cmonth,repeat_users)

# Revenue for retained customers

pivot_cust <- final_cust_y2 %>% select(Net_Sales, Acq_month, ym) %>%
group_by(Acq_month, ym) %>%
  dplyr::summarise(repeat_purchases = sum(Net_Sales))

pivot_cust$Acq_month <- as.yearmon(pivot_cust$Acq_month)
pivot_cust$ym <- as.yearmon(pivot_cust$ym)

pivot_cust2 <- spread(pivot_cust, ym,repeat_purchases)
#we need to melt data
cohort_fig1 <- melt(pivot_cust2, id.vars = "Acq_month")

colnames(cohort_fig1) <- c('cohort', 'month', 'revenue')
cohort_fig1[is.na(cohort_fig1$revenue),]$revenue <- 0

palette1 <-
colorRampPalette(c('#fde0dd','#fcc5c0','#fa9fb5','#f768a1','#dd3497','#ae017e
','#7a0177'))

ggplot(cohort_fig1, aes(x= as.factor(month), y=revenue, fill =
as.factor(cohort)))+ geom_bar(stat="identity") +
  scale_fill_manual(values = palette1(nrow(pivot_cust2))) +
  ggtitle('Total Gross Sales by customers based on their acquisition month')
+ labs(x= "Month of Interest", y = "Gross Sales") + labs(fill = "Acquisition
Month")
```



Assumption Only sales by customers acquired after July 2017 are considered for the analysis. The gross sales amount doesn't reflect the entire sales amount in that particular month.

Interpretation For our analysis, we considered the period starting from August 2017 to July 2018. In the first month of our dataset, the entire sales were from the customers acquired in that month ONLY. In the subsequent months the contribution of sales come from two sources; First, from customers acquired in the earlier month starting August 2017 and returning to the store to make a purchase. Second, sales of customers acquired in that month.

For example, sales in the Sep 2017 $8K sales were from customers acquired in Sep 2017 and $2k sales were from customers acquired in July 2017 and returned in Sep 2017 to make a purchase.

Exploring the customer segments that exist in the data

We would like to use the customer level data to find homogeneous groups who distinguish themselves from other groups. We used clustering on customer data to analyze the groups. We introduced new variables that can help us find better clusters. The following are the variables of interest to us:

- Number of Visits
- Maximum Date of Transaction
- Minimum Date of Transaction

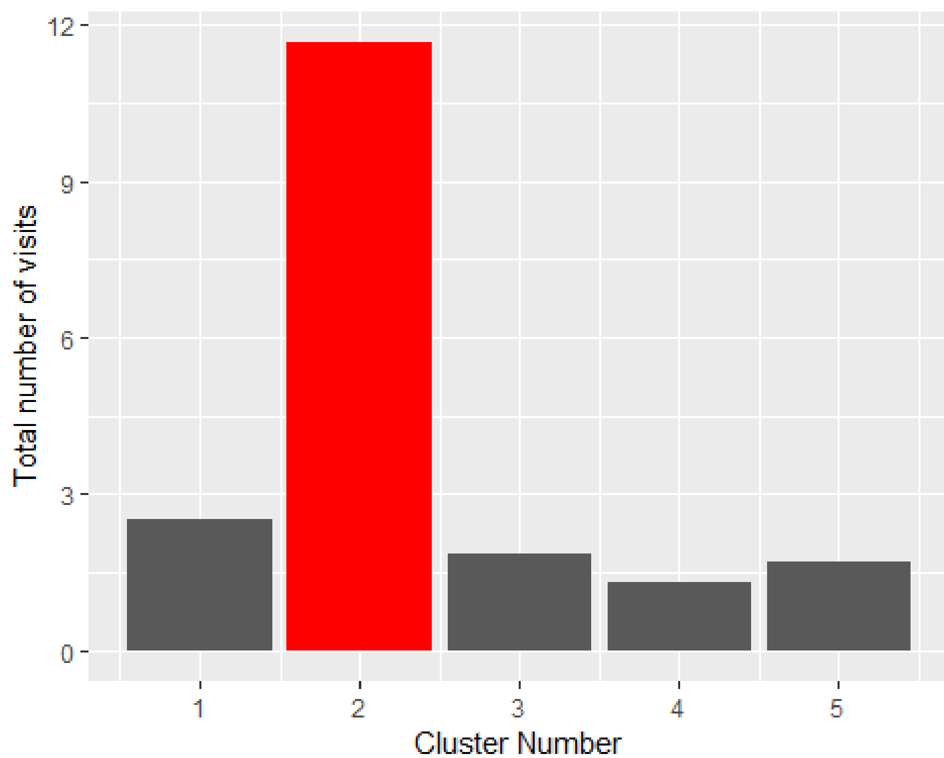
- Weekly Spend
- Weekly Visits
- Popular Items
- Weekend Sales
- Popular Hour

```
cp2 <- customers %>% filter(Category != 'Extras')

cp2_basket <- cp2 %>% group_by(Customer.ID,DateTime) %>%
  dplyr::summarise(basket_size = sum(Qty),
            sum_sales = sum(Gross.Sales)) %>% ungroup() %>%
group_by(Customer.ID) %>%
  dplyr::summarise(avg_basket = mean(basket_size),
            num_visits = length(DateTime),
            min_date = min(DateTime),
            max_date = max(DateTime),
            total_sales = sum(sum_sales)) %>%
  mutate(num_weeks =  if_else(num_visits ==
1,1,as.numeric(difftime(max_date,min_date,units='weeks'))),
         num_weeks = if_else(num_weeks < 1, 1, num_weeks),
         weekly_spend = total_sales/num_weeks,
         weekly_visits = num_visits/num_weeks)

cp2_popular <- cp2 %>% group_by(Customer.ID,Item) %>%
dplyr::summarise(total_qty = sum(Qty)) %>%
  ungroup() %>% group_by(Customer.ID) %>% arrange(desc(total_qty)) %>%
  dplyr::summarise(popular_item = first(Item))

cp2_time <- cp2 %>% group_by(Customer.ID,DateTime) %>%
dplyr::summarise(total_qty = sum(Qty)) %>%
  mutate(hour = hour(DateTime)) %>% ungroup() %>%
  group_by(Customer.ID,hour) %>% dplyr::summarize(num = length(total_qty))
%>% ungroup() %>% group_by(Customer.ID) %>%
  arrange(desc(num)) %>% dplyr::summarise(popular_hour = first(hour))

cp2_weekend <- cp2 %>% mutate(day = wday(DateTime)) %>% filter(day %in%
c(1,7)) %>%
  group_by(Customer.ID) %>% dplyr::summarize(weekend_sales =
sum(Gross.Sales))

'%ni%' <- Negate('%in%')
cp2_coffee <- cp2 %>% mutate(o_cat_flag = if_else(Category %ni%
c("Coffee","Tea"),1,0),
                             o_qty = o_cat_flag * Qty) %>%
  group_by(Customer.ID) %>%
  dplyr::summarise(total_qty = sum(Qty),
            total_cats = n_distinct(Category),
            other_qty = sum(o_qty),
            per_other = other_qty/total_qty)
```

```r
first2 <- merge(cp2_basket, cp2_popular, by='Customer.ID')
second2 <- merge(first2, cp2_time, by='Customer.ID')
third2 <- merge(second2, cp2_coffee, by = 'Customer.ID')
final2 <- merge(third2,cp2_weekend,by='Customer.ID',all.x=TRUE)

clust2_data <- final2 %>% mutate(weekend_sales =
if_else(is.na(weekend_sales),0,weekend_sales),
                                per_weekend = weekend_sales/total_sales) %>%

select(Customer.ID,avg_basket,num_weeks,num_visits,weekly_spend,weekly_visits
,popular_item,total_cats,popular_hour,per_weekend,per_other)

## Normalizing data
normalize <- function(x){
  return ((x - min(x))/(max(x) - min(x)))}

sapply(clust2_data, class)

##   Customer.ID    avg_basket     num_weeks    num_visits  weekly_spend
##      "factor"     "numeric"     "numeric"     "integer"     "numeric"
## weekly_visits  popular_item    total_cats  popular_hour   per_weekend
##     "numeric"      "factor"     "integer"     "integer"     "numeric"
##     per_other
##     "numeric"

ix <- (sapply(clust2_data, class) == "numeric") | (sapply(clust2_data, class)
== "integer")
clust2_data2 <- clust2_data
clust2_data2[,ix] <- sapply(clust2_data2[,ix], normalize)

## Removing the Customer ID field
clust2_data3 <- clust2_data2[,-1]

## Clustering
clust2_kmeans4 <- clust2_data3[,-c(2,6)] ##Removing the num_weeks variable
## Removing the popular item variable

SSE_curve <- c()
for (k in 1:10) {
  kcluster <- kmeans(clust2_kmeans4, k)
  sse <- sum(kcluster$withinss)
  SSE_curve[k] <- sse
}
plot(1:10, SSE_curve, type="b", xlab="Number of Clusters", ylab="SSE")
```


```
## The steepest dip in SSE occurs when k=5
num_clusters4 = 5

kcluster4 <- kmeans(clust2_kmeans4, num_clusters4)

## Obtaining cluster number from kmeans to join it to original data
clustered_data4 <- cbind(clust2_data,kcluster4$cluster)
#write.csv(clustered_data4,"clustered_data4.csv")

clustered_data4 <- read.csv("clustered_data4.csv")

# Number of Visits by cluster
tapply(clustered_data4$num_visits, clustered_data4$Cluster.number, mean)

##         1         2         3         4         5
##  2.504751 11.660816  1.863012  1.323939  1.703808

#Popular hour by cluster
tapply(clustered_data4$popular_hour, clustered_data4$Cluster.number, mean)

##         1         2         3         4         5
## 11.928728  9.720816 12.454217 13.064900 12.501661

#Cluster who spend well on weekends
tapply(clustered_data4$per_weekend, clustered_data4$Cluster.number, mean)
```

```
##           1           2           3           4           5
## 0.012940217 0.462818995 0.005561891 0.993415570 0.984058500

#Num of weeks with the coffee shop by cluster
tapply(clustered_data4$num_weeks, clustered_data4$Cluster.number, mean)

##         1         2         3         4         5
##  6.052839 28.617837  3.998209  3.178687  4.424115

c1 <- clustered_data4 %>% group_by(Cluster.number) %>%
dplyr::summarise(num_visits = mean(num_visits))

c1 %>%
  dplyr::mutate(highlight_flag = ifelse(Cluster.number == 2, T, F)) %>%
  ggplot(aes(x = Cluster.number, y = num_visits)) +
    geom_bar(stat = "identity", aes(fill = highlight_flag)) +
    scale_fill_manual(values = c('#595959', 'red')) + theme(legend.position =
'none') + labs(x= "Cluster Number", y = "Total number of visits")
```




```
c2 <- clustered_data4 %>% group_by(Cluster.number) %>%
dplyr::summarise(per_weekend = mean(per_weekend))
```

Interpretation

- The elbow chart suggests that when the number of clusters equal to 5 there is a steep decrease in standard squared error (SSE). So we took 5 clusters and profiles them on various variables

- Cluster 2 has made more visits to the coffee shops on an average when compared to other clusters
- The people in cluster 4 and 5 spend more on weekends than the other clusters

		Cluster number				
	Variables	1	2 = Loyalty customers	3	4	5
Size of cluster	#Customers	5400	2400	11000	8900	3900
Clustering variables	Day of visit	Week	Equally distributed	Week	Weekend	Weekend
	%Other items bought	50%	15%	0%	0%	50%
	Categories bought	2	2	1	1	2
	#Visits	1	11	1	1	1
	Popular hours	12	10	12	13	12
	Weekly visits	1	1	1	1	1
	Weekly spend	8	5	5	5	8
Other variables	#Weeks	Low	High	Low	Low	Low
	Spend/visit	8	5	5	5	8

Comparison of key characteristics for the clusters

Analyzing the customer preference for most loyal customers

Based on the clustering results, it is evident that cluster 2 has high visits to the coffee shop and contains all characteristics to be defined as the loyal customer segment. We want to know what makes this particular segment a loyal customer base by looking at their purchase patterns and by analyzing their sales basket. We used the unsupervised technique, association rules to further understand the data.

```
cp_cleaned <- cp[, c("Customer.ID","Item")]
cp_cleaned <- cp_cleaned[complete.cases(cp_cleaned),]

ddf <- ddply(cp_cleaned,"Customer.ID", function(df1) paste(df1$Item,collapse
= ","))
names(ddf) <- c("Customer.ID", "Items")

cluster_user <- read.csv("clustered_data4.csv", header = TRUE)

merged_data <- merge(ddf, cluster_user[,c("Customer.ID", "Cluster.number")],
by= "Customer.ID")
merged_data <- merged_data %>% filter(Cluster.number == 2)
merged_data$Customer.ID <- NULL
merged_data$Cluster.number <- NULL

write.table(merged_data, file = "association_data.txt",quote = F, row.names =
F, col.names = F)

txn_df = read.transactions("association_data.txt", format = "basket",
sep=",")

## Warning in asMethod(object): removing duplicated items in transactions
```

```r
#summary(txn_df)

basket_rules <- apriori(txn_df,parameter = list(sup = 0.1, conf = 0.1, minlen
= 2))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.1    0.1    1 none FALSE            TRUE       5     0.1      2
##  maxlen target   ext
##      10  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 242
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[48 item(s), 2423 transaction(s)] done [0.00s].
## sorting and recoding items ... [11 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [62 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

subrules <- head(basket_rules, n = 15, by = "lift")
#plot(subrules, method="graph", control=list(type="items"), main =
"Association Rules", cex = 2)
inspect(subrules)

##      lhs                   rhs         support   confidence lift     count
## [1]  {Latte}            => {Almond}    0.1551795 0.3421292  1.674705  376
## [2]  {Almond}           => {Latte}     0.1551795 0.7595960  1.674705  376
## [3]  {Drip,Latte}       => {Croissant} 0.1031779 0.4058442  1.385015  250
## [4]  {Drip,Latte}       => {Ice}       0.1890219 0.7435065  1.378360  458
## [5]  {Drip,Tea}         => {Ice}       0.1238135 0.7142857  1.324188  300
## [6]  {Donut,Drip}       => {Ice}       0.1163846 0.7103275  1.316850  282
## [7]  {Donut,Ice}        => {Drip}      0.1163846 0.8571429  1.302105  282
## [8]  {Croissant,Ice}    => {Drip}      0.1427982 0.8459658  1.285125  346
## [9]  {Cappucino}        => {Croissant} 0.1151465 0.3720000  1.269515  279
## [10] {Croissant}        => {Cappucino} 0.1151465 0.3929577  1.269515  279
## [11] {Ice,Tea}          => {Drip}      0.1238135 0.8356546  1.269461  300
## [12] {Croissant,Drip}   => {Ice}       0.1427982 0.6641075  1.231165  346
## [13] {Ice}              => {Drip}      0.4341725 0.8048967  1.222737 1052
## [14] {Drip}             => {Ice}       0.4341725 0.6595611  1.222737 1052
## [15] {Drip,Ice}         => {Donut}     0.1163846 0.2680608  1.218596  282
```

Interpretation We found the following set of products have high lift values:

- Almond and Latte
- Drip, Latte and Croissant

Summary

We believe that the following recommendations will increase customer loyalty, increase revenue, and smooth the daily sales curve.

Create Mobile Application and Loyalty Program Coffee shops are building loyal customers with a mobile app that simplify ordering with pre-ordering and favorite purchases, managing loyalty points, target marketing. For example, Starbucks reports generating 15% of its sales volume through their mobile app. Additionally, customers using a loyalty program are more likely to refer others and bring friends and co-workers to the shop. Building a mobile app does not require an IT department, there are companies who specialize in creating mobile apps for coffee shops.

Increase Food Offerings Daily sales of food items follow the same pattern as coffee sales, with pastries being most popular. We believe that diversifying food offerings to include healthy, pre-packaged, "grab and go" food will give customers a reason to visit other than for their morning coffee. This will increase demand during time of day where demand is lagging from its peak.

Happy Hour Sales from 4:00 to close is lower than any other period of the day. Many people do not drink coffee in the late afternoon or evening. Creating a "happy hour" from 4:00 to close with a 10% discount on all beverages for loyalty member customers will incent customers to visit the shop late in the day. We don't expect this to raise sales to the morning or early afternoon levels but expect it to increase traffic from current levels.

Cross-Selling Opportunity Drip coffee is the most popular product and customers often purchase a donut or croissant with their drip coffee. This is a cross selling opportunity for the associates and the mobile app. When a customer purchases a drip coffee offer them a donut or croissant.