Homework 3 - Central Perk

November 2, 2019

MSBA 6410: Exploratory Data Analysis & Visualization

Authors: Sarah Black, Michael DeGuire, Anthony Meyers, Danny Moncada, and Jonathan Watkins

Table of Contents

1 Problem Statement and Approach

1.1 Central Perk Objectives

Central Perk is a New York City coffee shop looking to increase profits by acting on the data that they have on sales and customers. They belive that they have a loyal base of customers that they can generate the additional revenue from rather than trying to acquire new customers. They also believe that buisness spikes at certain days and times and is slow at other times; they are interested in trying to smooth this out over their open hours

1.2 Our Approach

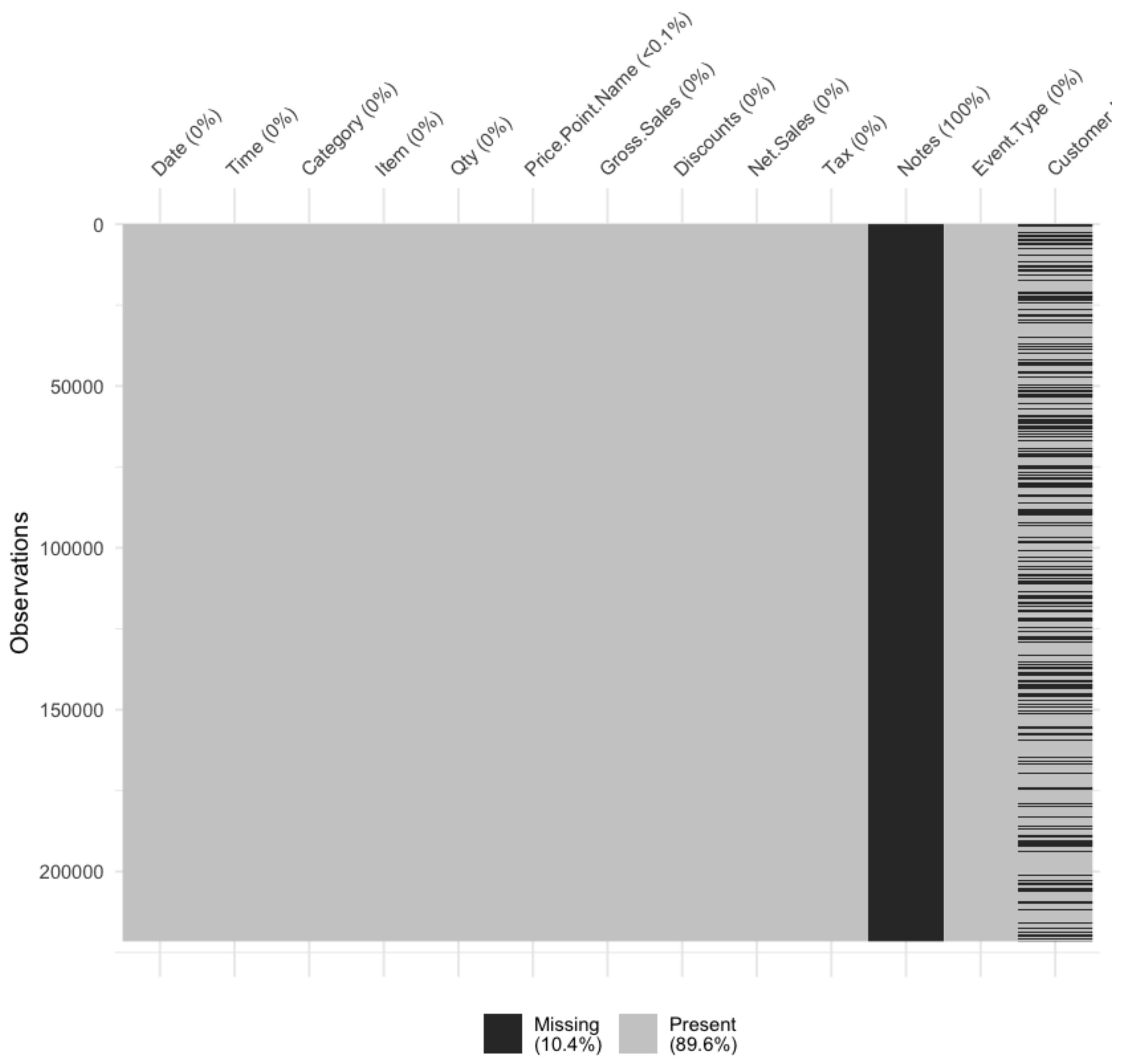
We want to evaluate how loyal the customer base is and how demand flucuates over the days and hours of operation. Once we have a solid understanding of those two pieces we want to look into what sales by product look like to see if there are any trends by hour, day, or season.
To help with increasing revenue from the existing customer base we will want to look into how the most loyal customers compare to the occational customers so that we can increase repeat business without alienating the base. We have also done some research to look at recommendations for coffee shops in general and will be comparing those findings with the data to help ground the recomendations in the current industry trajectory.

2 Data Preparation

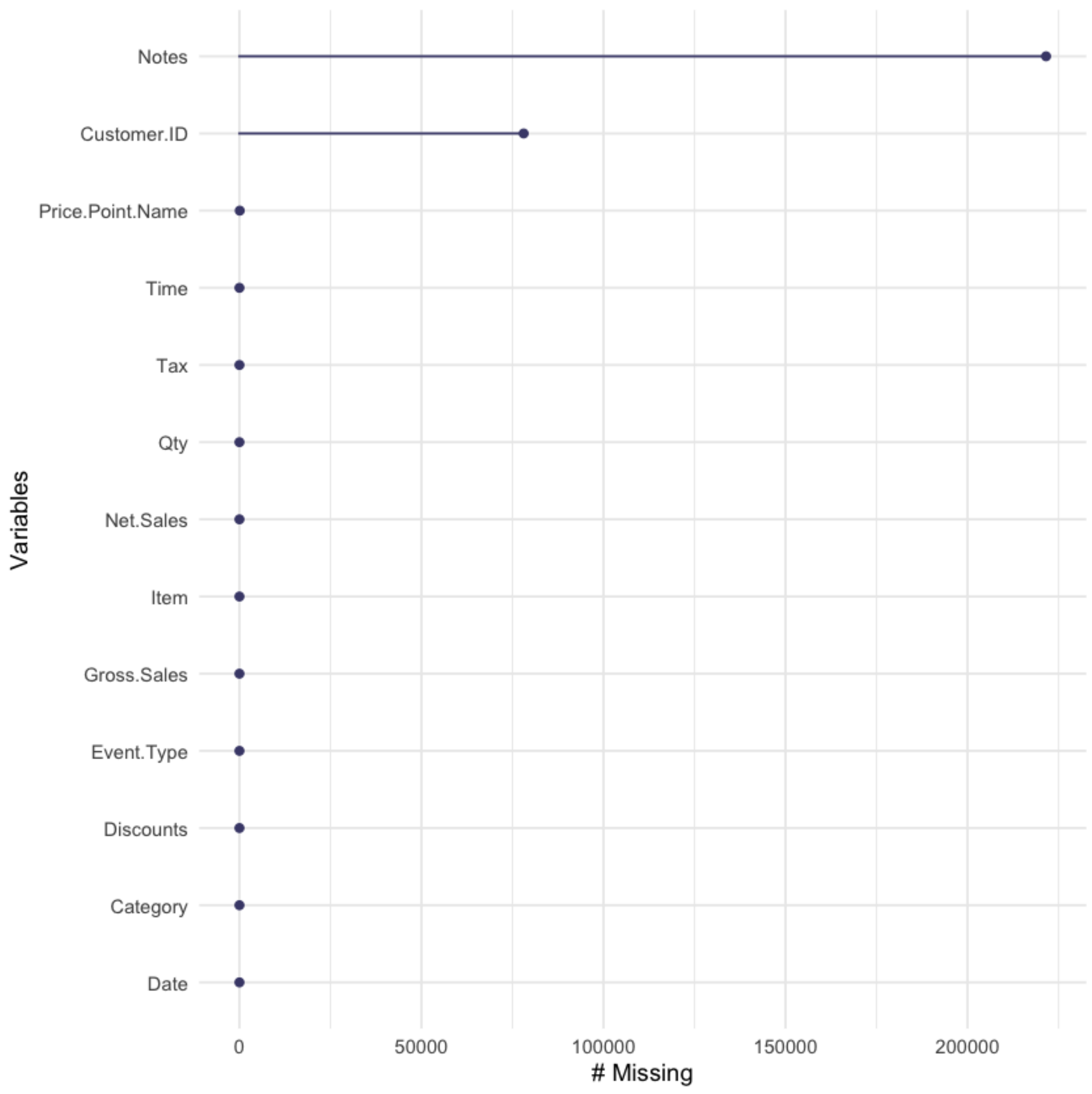
First we brought in our data and assessed what values were missing

```
[1]: suppressMessages(suppressWarnings(library(readr)))
suppressMessages(suppressWarnings(library(tidyverse)))
suppressMessages(suppressWarnings(library(ggrepel)))
suppressMessages(suppressWarnings(library(naniar)))
suppressMessages(suppressWarnings(library(arules)))
suppressMessages(suppressWarnings(library(arulesViz)))

options(scipen = 999)

df6 <- suppressMessages(suppressWarnings(read_csv("~/Desktop/HW 3/Central Perk/
  ↪Central Perk Item Sales Summary 2016.csv")))
df7 <- suppressMessages(suppressWarnings(read_csv("~/Desktop/HW 3/Central Perk/
  ↪Central Perk Item Sales Summary 2017.csv")))
df8 <- suppressMessages(suppressWarnings(read_csv("~/Desktop/HW 3/Central Perk/
  ↪Central Perk Item Sales Summary 2018.csv")))

df <- rbind(df6, df7, df8)
```

```
[2]: naniar::vis_miss(df, warn_large_data=FALSE)
```



[3]:
```
naniar::gg_miss_var(df)
```



Missing values don't appear to be a problem in the dataset, there is a field for notes which isn't being used (100% missing) and only some transactions have customer ids, otherwise data appears complete

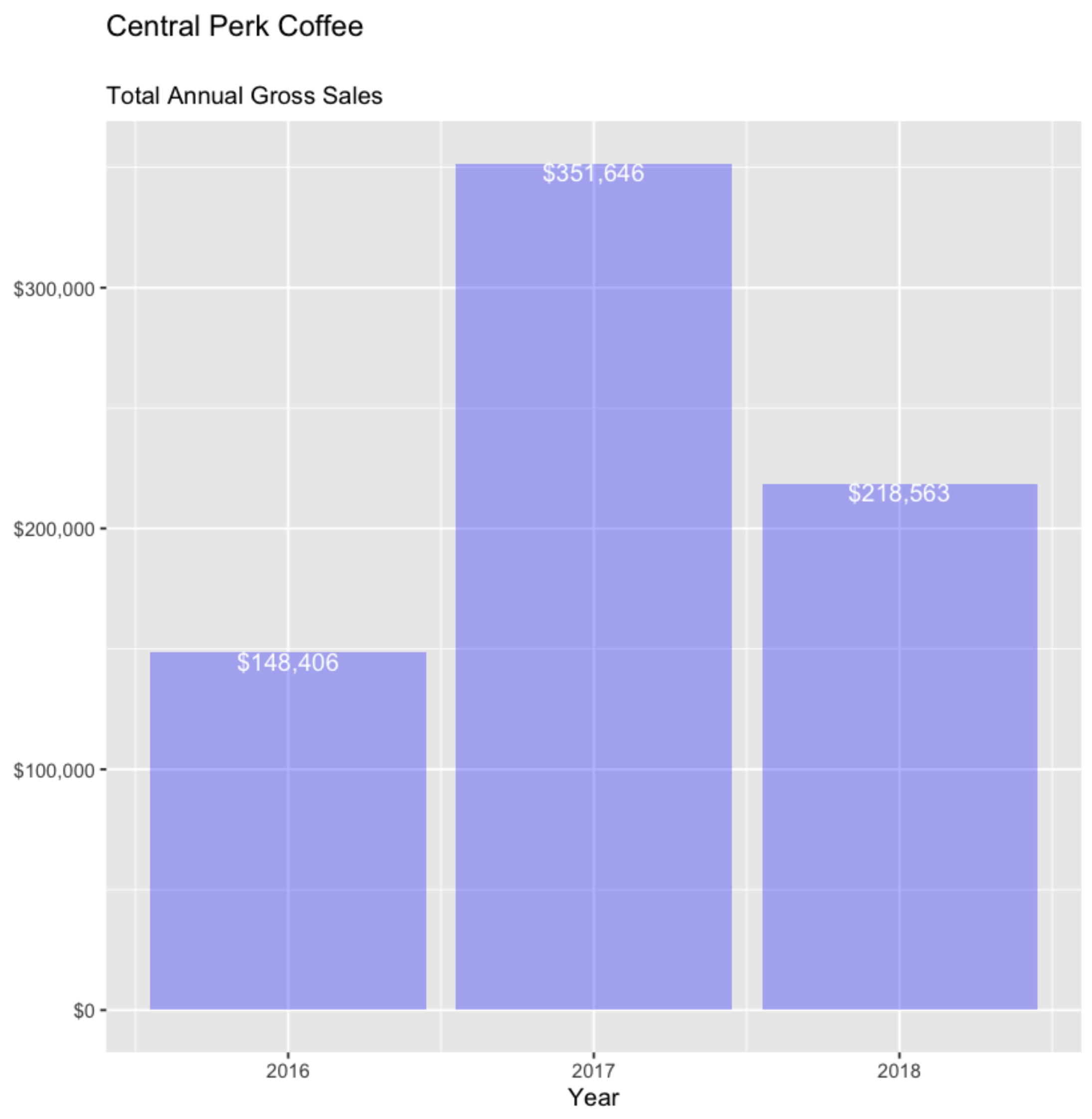
We then changed the formatting on fields to ensure that numbers and dates could be worked with

[4]:
```
df <- df %>%
  filter(Date != "Unknown Error") %>%
  mutate(Date = lubridate::mdy(Date),
         Gross.Sales = as.numeric(str_replace_all(Gross.Sales,"[\\(\\$\\)]","")),
         Discounts = as.numeric(str_replace_all(Discounts,"[\\(\\$\\)]","")),
         Net.Sales = as.numeric(str_replace_all(Net.Sales,"[\\(\\$\\)]","")),
         Tax = as.numeric(str_replace_all(Tax,"[\\(\\$\\)]","")),
         month_ = lubridate::month(Date),
         year_ = lubridate::year(Date)) %>%
  mutate(Item = gsub("[^[:alnum:][:blank:]?&/\\-]", "", Item)) %>%
  mutate(Item_Qty = paste0(Item, "-", Qty))

rm(df6, df7, df8)
```

2.1 Exploring gross sales

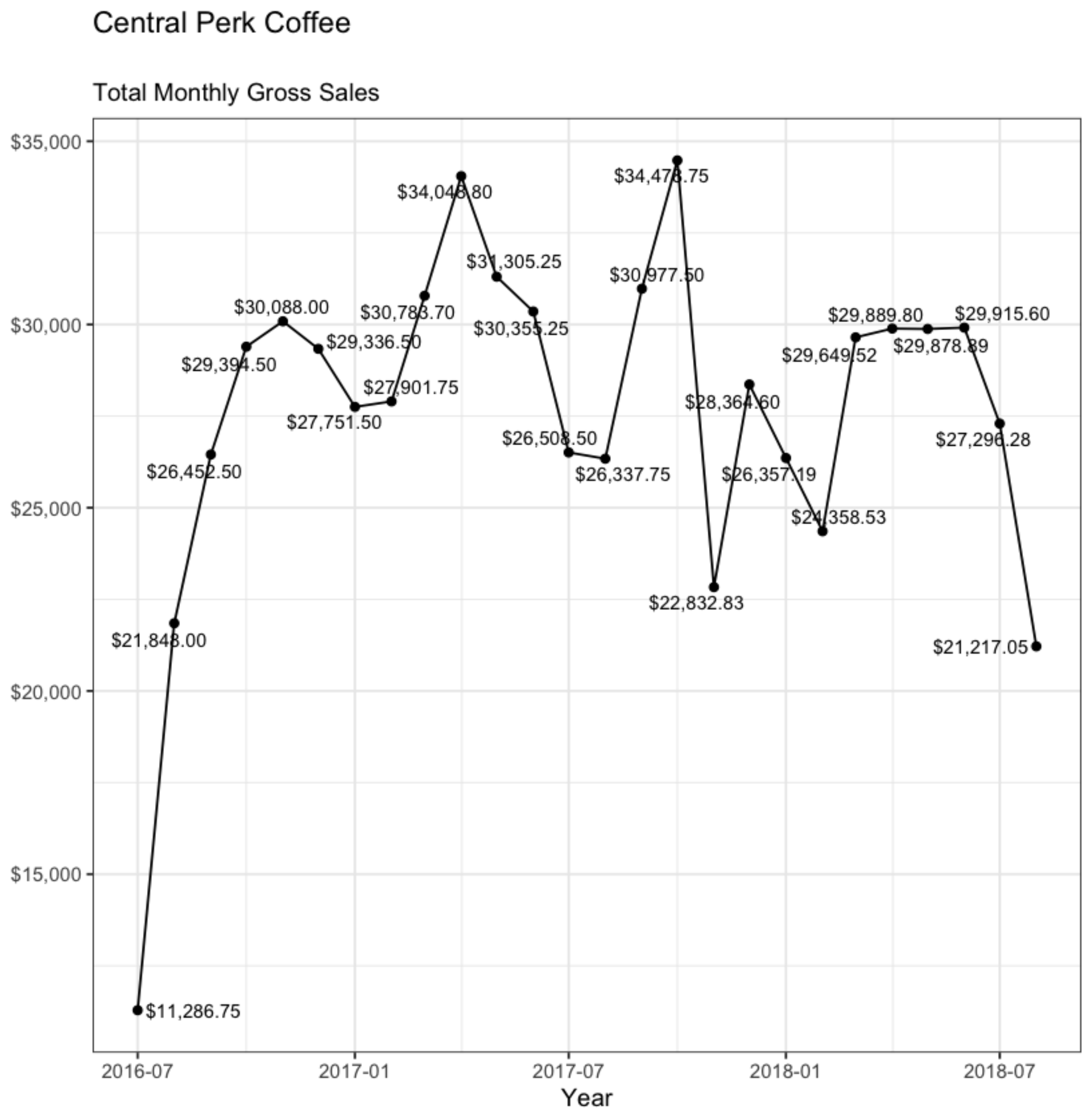
As a starting point we looked at gross sales by year

[5]:
```
TOT <- df %>%
  group_by(year_) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales),
            Net.Sales.TOT = sum(Net.Sales))

ggplot(TOT, aes(x=year_, y=Gross.Sales.TOT)) +
geom_col(fill="blue", alpha=.3) + scale_y_continuous(labels = scales::dollar) +
labs(title = "Central Perk Coffee\n", subtitle = "Total Annual Gross Sales",
     x = "Year", y = "") +
geom_text(label = scales::dollar(TOT$Gross.Sales.TOT), col="white", vjust=1)
```



Interestingly, 2017 has more sales. Given the composition of the data, we be able to extract more meaningful insight if we break the split down by month instead of only by year.

[6]:
```
month.TOT <- df %>%
  group_by(month_, year_) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales),
            Net.Sales.TOT = sum(Net.Sales)) %>%
  mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1")))

p1 <- ggplot(month.TOT, aes(x=new_date, y=Gross.Sales.TOT)) +
  geom_line() + geom_point() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Monthly Gross Sales",
     x = "Year", y = "") + scale_y_continuous(labels = scales::dollar) +
  geom_text_repel(label = scales::dollar(month.TOT$Gross.Sales.TOT), size=3) +
  theme_bw()

p2 <- ggplot(month.TOT, aes(x=new_date, y=Net.Sales.TOT)) +
  geom_line() + geom_point() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Monthly Net Sales",
     x = "Year", y = "") + scale_y_continuous(labels = scales::dollar) +
  geom_text_repel(label = scales::dollar(month.TOT$Net.Sales.TOT), size=3) +
  theme_bw()

p1
# cowplot::plot_grid(p1, p2, nrow = 2)
```



This explains the by year breakdown, it appears we only have July of 2016 through August of 2018. We do see some spikes which we should look at later for seasonal trends

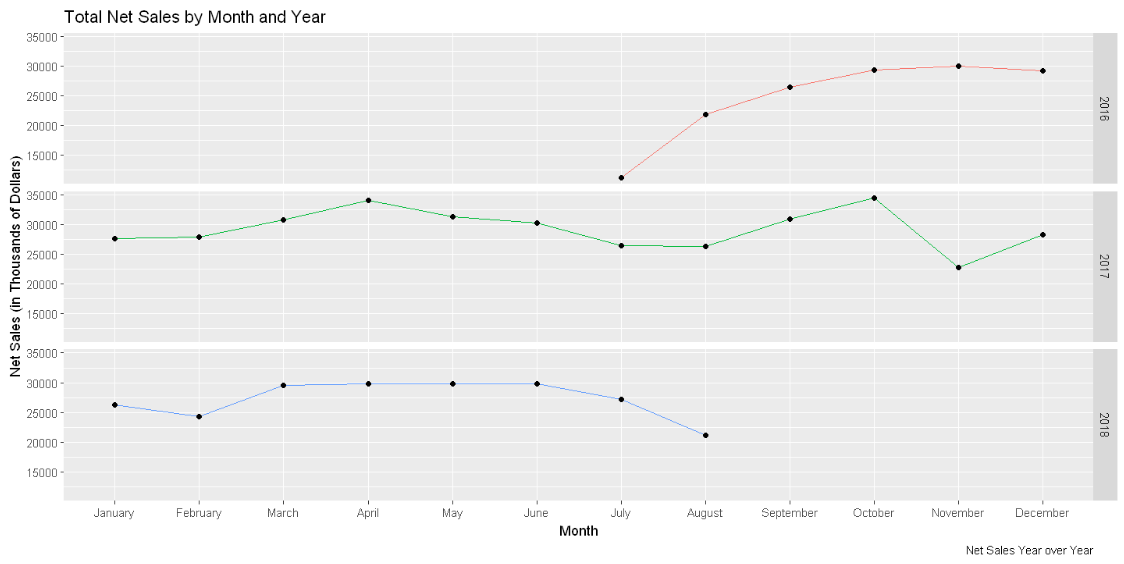
[18]:
```
# change plot size for the graphs below
library(repr)
options(repr.plot.width=12, repr.plot.height=6)
```

[11]:
```
## Now lets summarize NET sales by month and year.  We want to see what changes␣
↪happen over time.

net_sales_by_month_year <-
    centralperk_df %>%
        group_by(Year, Month) %>%
            summarise(NetSales = sum(Net.Sales))
```

[12]:
```
## Create more helper functions for structuring and ordering our data

year_levels = c("2016", "2017", "2018")
month_levels = c("January", "February", "March", "April", "May", "June", "July",␣
↪"August",
                "September", "October", "November", "December")


## Re-order our columns for our visualizations below
net_sales_by_month_year$Year <- factor(net_sales_by_month_year$Year, levels =␣
↪year_levels)

net_sales_by_month_year$Month <- factor(net_sales_by_month_year$Month, levels =␣
↪month_levels)
```

[13]:
```
## Create an abbreviation for the month for the second plot below
## Grab the first three letters for each month

net_sales_by_month_year$MonthAbv <- substr(net_sales_by_month_year$Month, 0, 3)

abv_levels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep",␣
↪"Oct",
              "Nov", "Dec")

net_sales_by_month_year$MonthAbv <- factor(net_sales_by_month_year$MonthAbv,␣
↪levels = abv_levels)
```

[19]:
```
## Create first plot to see month over month how much Net Sales change

ggplot(net_sales_by_month_year, aes ( x = Month, y = NetSales, group = Year)) +
    geom_line(aes (col = Year)) +
    geom_point() +
    facet_grid(Year ~ .) +
    theme(legend.position = "none") +
    labs(title = "Total Net Sales by Month and Year", y = "Net Sales (in␣
↪Thousands of Dollars)",
        caption = "Net Sales Year over Year")
```



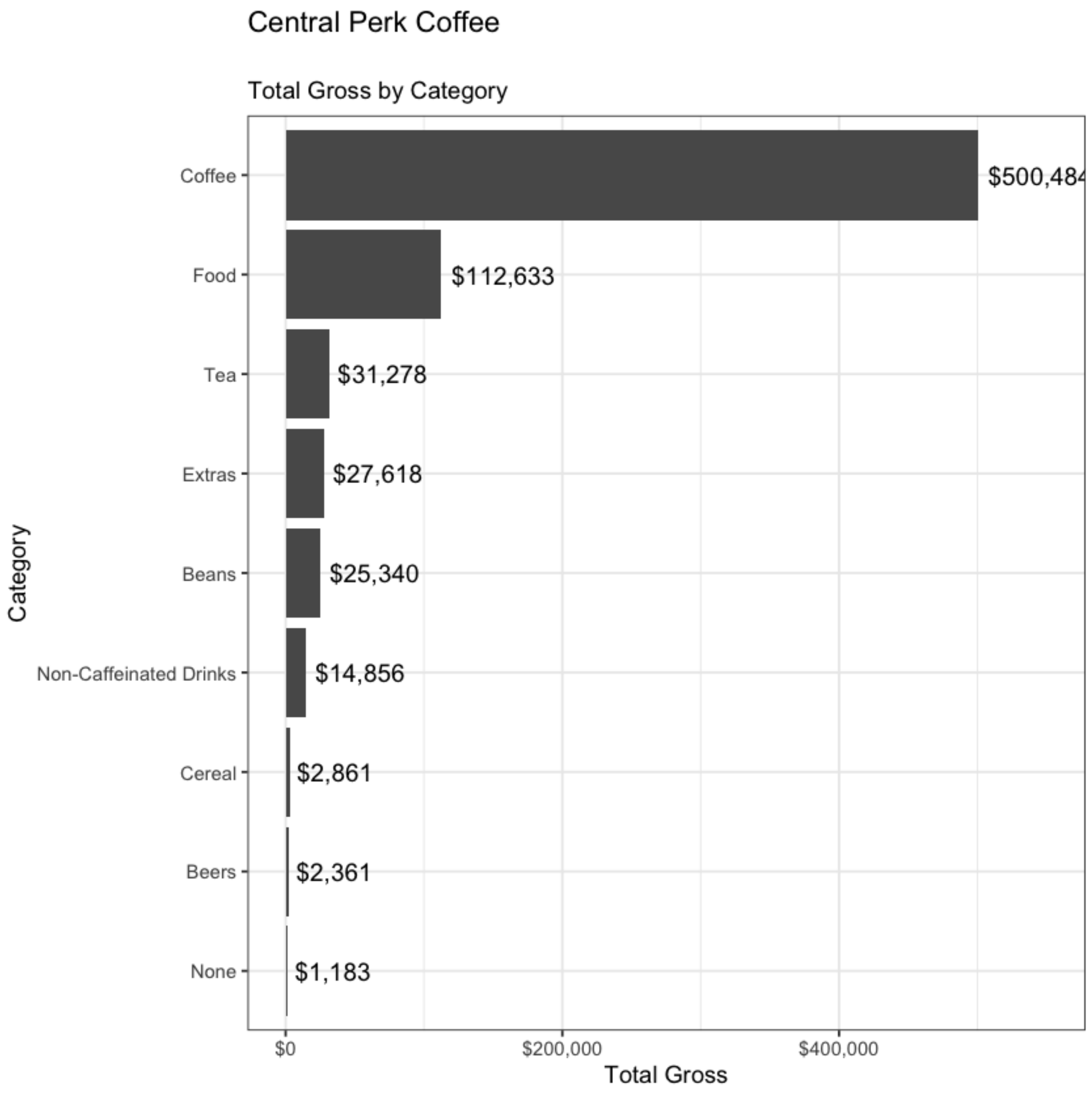
The year over year is telling. We can see that the March - June window is relatively stable between 2017 and 2018 with a rough average of $3, but it is hard to see how the years connect, so we joined them end to end for a continuous view

```
[1]: ## Generate a similar plot but now see how it changes over the life span of the␣
↪data

ggplot(net_sales_by_month_year, aes ( x = MonthAbv, y = NetSales, group = Year))␣
↪+
    geom_line(aes (col = Year)) +
    geom_point() +
    facet_grid(. ~ Year) +
    theme(legend.position = "none") +
    labs(title = "Total Net Sales by Month and Year", x = "Month", y = "Net␣
↪Sales (in $One's)",
        caption = "Net Sales Year over Year")
```

```
        Error in ggplot(net_sales_by_month_year, aes(x = MonthAbv, y = NetSales, :␣
↪could not find function "ggplot"
    Traceback:
```

Now that we have some idea of the when sales were occuring, we looked at what products were the biggest sellers:

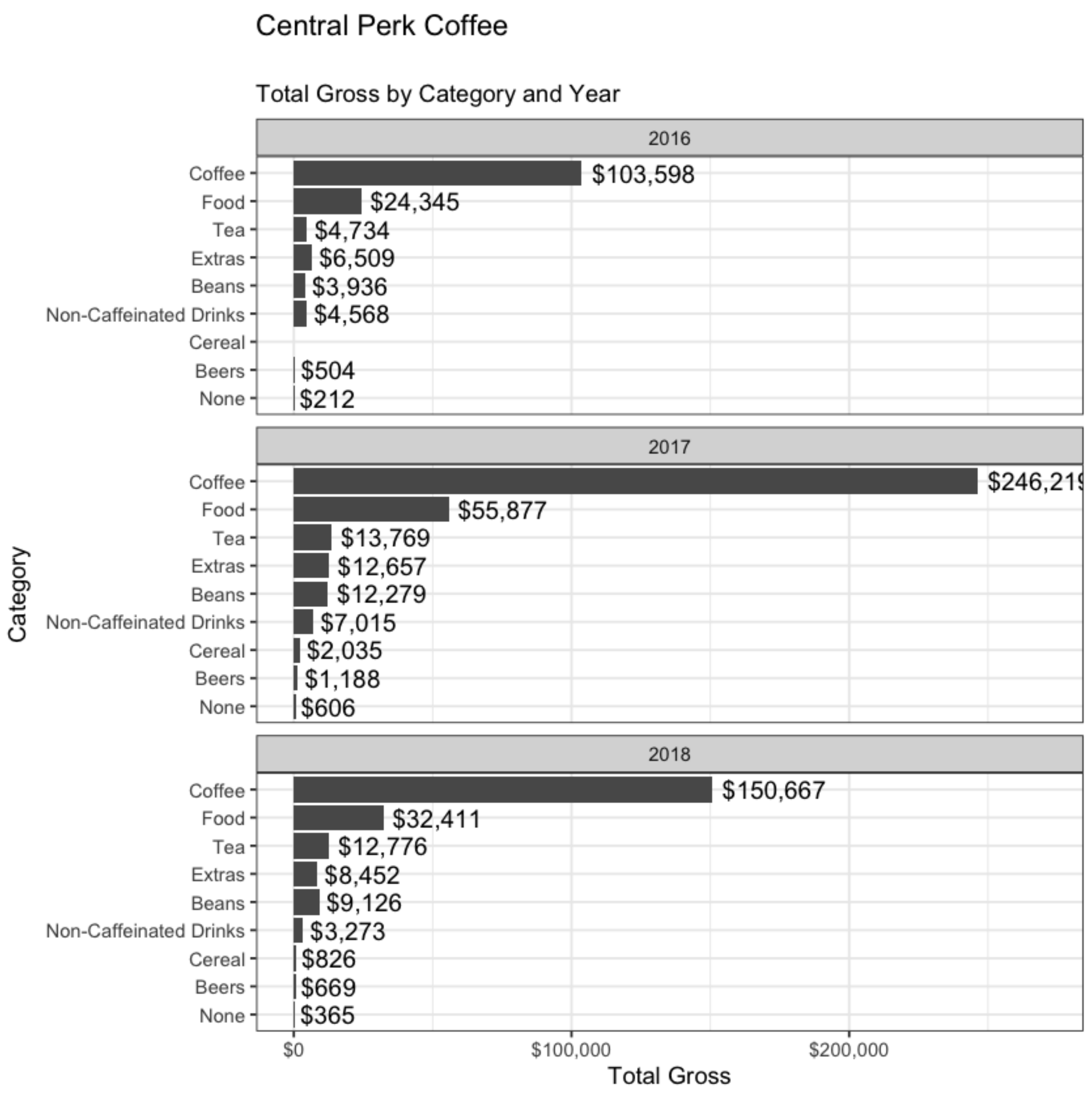
```
[7]: # Highest Grossing Category Overall
TOT <- df %>%
  group_by(Category) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales))

ggplot(TOT, aes(x=reorder(Category, Gross.Sales.TOT), y=Gross.Sales.TOT)) +
  geom_col() + coord_flip() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Gross by Category",
      x = "Category", y = "Total Gross") +
  scale_y_continuous(labels = scales::dollar, limits = c(0,1.1*max(TOT$Gross.
 ↪Sales.TOT))) +
  geom_text(label = scales::dollar(TOT$Gross.Sales.TOT), hjust=-.1, size=4) +
  theme_bw()
```



That good news is that coffee is the biggest seller at the coffee shop! We do see some other interesting components from the graph, such as the low sales totals of Cereal and Beers. These might be good areas for the coffee shop to explore eliminating to help reduce cost and overhead.

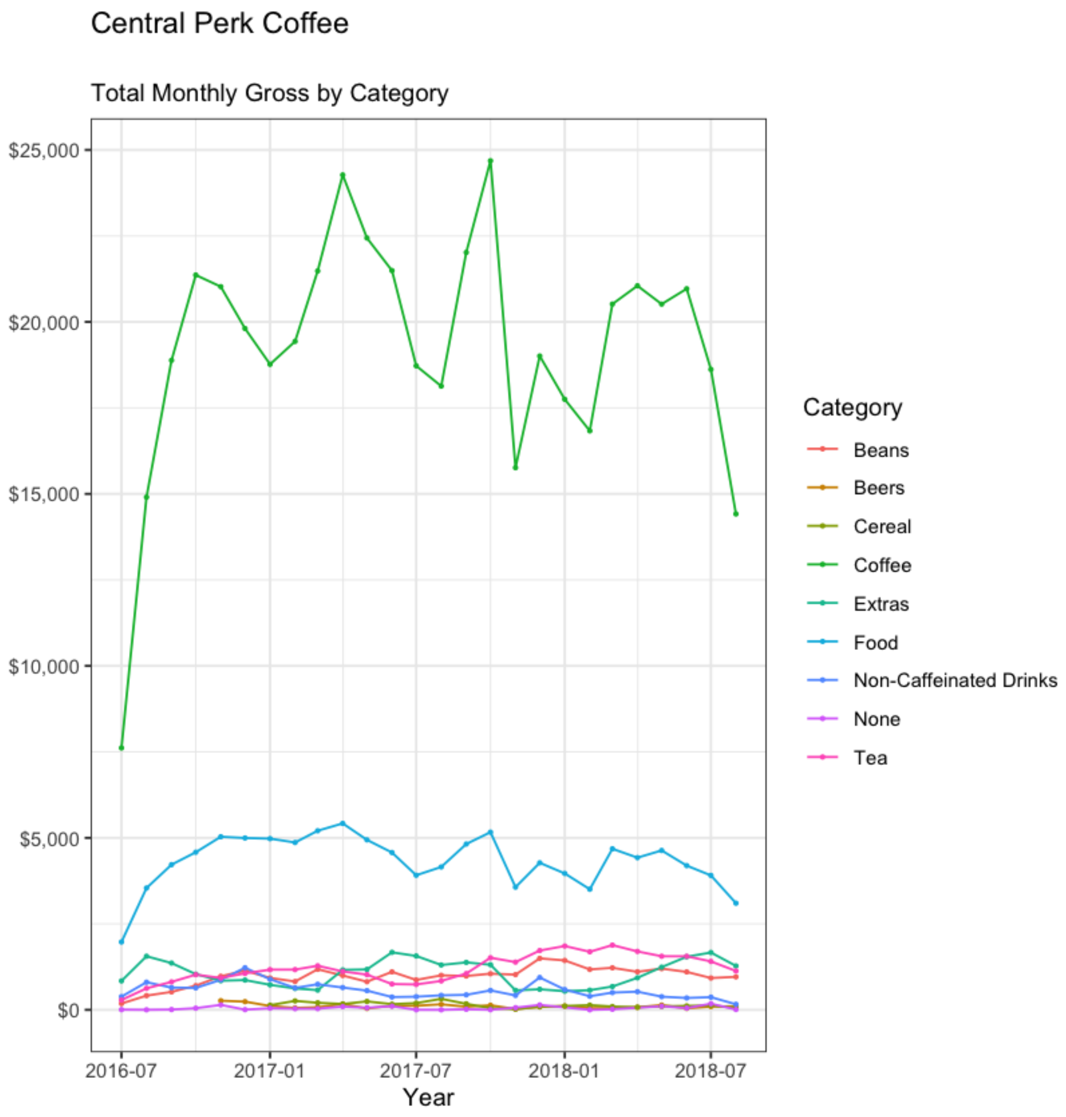
Are there any trends at the year by year level?

[8]:
```
# Highest Grossing Category Overall
TOT <- df %>%
  group_by(Category, year_) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales))

ggplot(TOT, aes(x=reorder(Category, Gross.Sales.TOT), y=Gross.Sales.TOT)) +
  geom_col() + coord_flip() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Gross by Category and␣
↪Year",
     x = "Category", y = "Total Gross") +
  scale_y_continuous(labels = scales::dollar, limits = c(0,1.1*max(TOT$Gross.
↪Sales.TOT))) +
  geom_text(label = scales::dollar(TOT$Gross.Sales.TOT), hjust=-.1, size=4) +
  theme_bw() + facet_wrap(~year_, nrow = 3)
```



There are some catagories here with very few sales, but perhaps it would make sense to visualize the differences

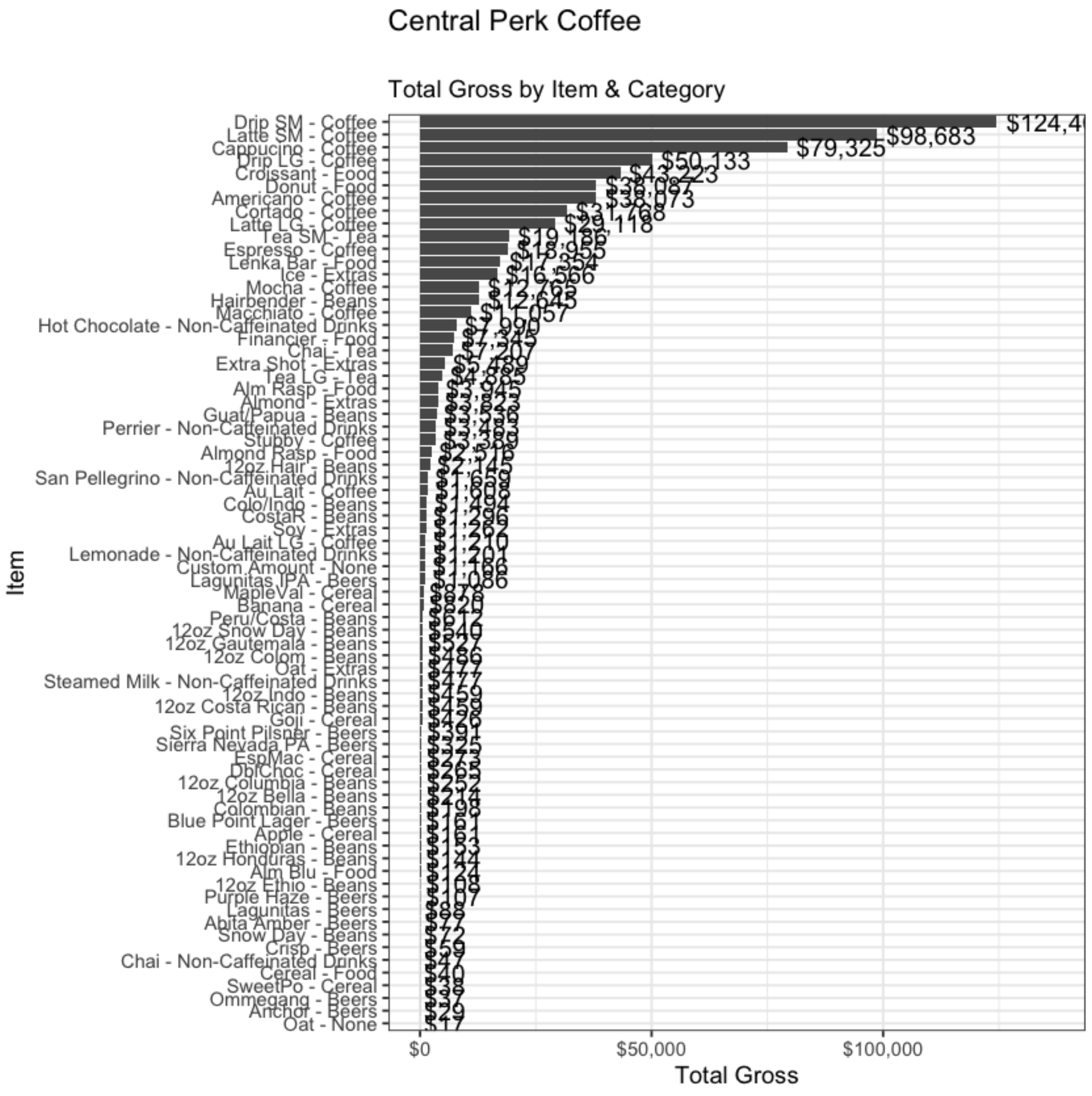
```
[9]: month.TOT <- df %>%
       group_by(Category, month_, year_) %>%
       summarise(Gross.Sales.TOT = sum(Gross.Sales),
                 Net.Sales.TOT = sum(Net.Sales)) %>%
       mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1")))

     ggplot(month.TOT, aes(x=new_date, y=Gross.Sales.TOT, color = Category)) +
       geom_line() + geom_point(size=.5) +
       labs(title = "Central Perk Coffee\n", subtitle = "Total Monthly Gross by␣
      ↪Category",
    x = "Year", y = "") + scale_y_continuous(labels = scales::dollar) +
  theme_bw()
```



Coffee and food are the biggest sellers and this really groups everything else at the bottom. There is some variation in the other catergories, however, cereal and beer hover at near zero continuously.

Moving beyond the categories into the individual items to see what we can learn.

[10]:
```
# Highest Grossing Item Overall
TOT <- df %>%
  group_by(Item, Category) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales)) %>%
  mutate(Item_Category = paste0(Item, " - ", Category))

ggplot(TOT, aes(x=reorder(Item_Category, Gross.Sales.TOT), y=Gross.Sales.TOT)) +
  geom_col() + coord_flip() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Gross by Item &␣
↪Category",
     x = "Item", y = "Total Gross") +
  scale_y_continuous(labels = scales::dollar, limits = c(0,1.1*max(TOT$Gross.
↪Sales.TOT))) +
  geom_text(label = scales::dollar(TOT$Gross.Sales.TOT), hjust=-.1, size=4) +
  theme_bw()
```



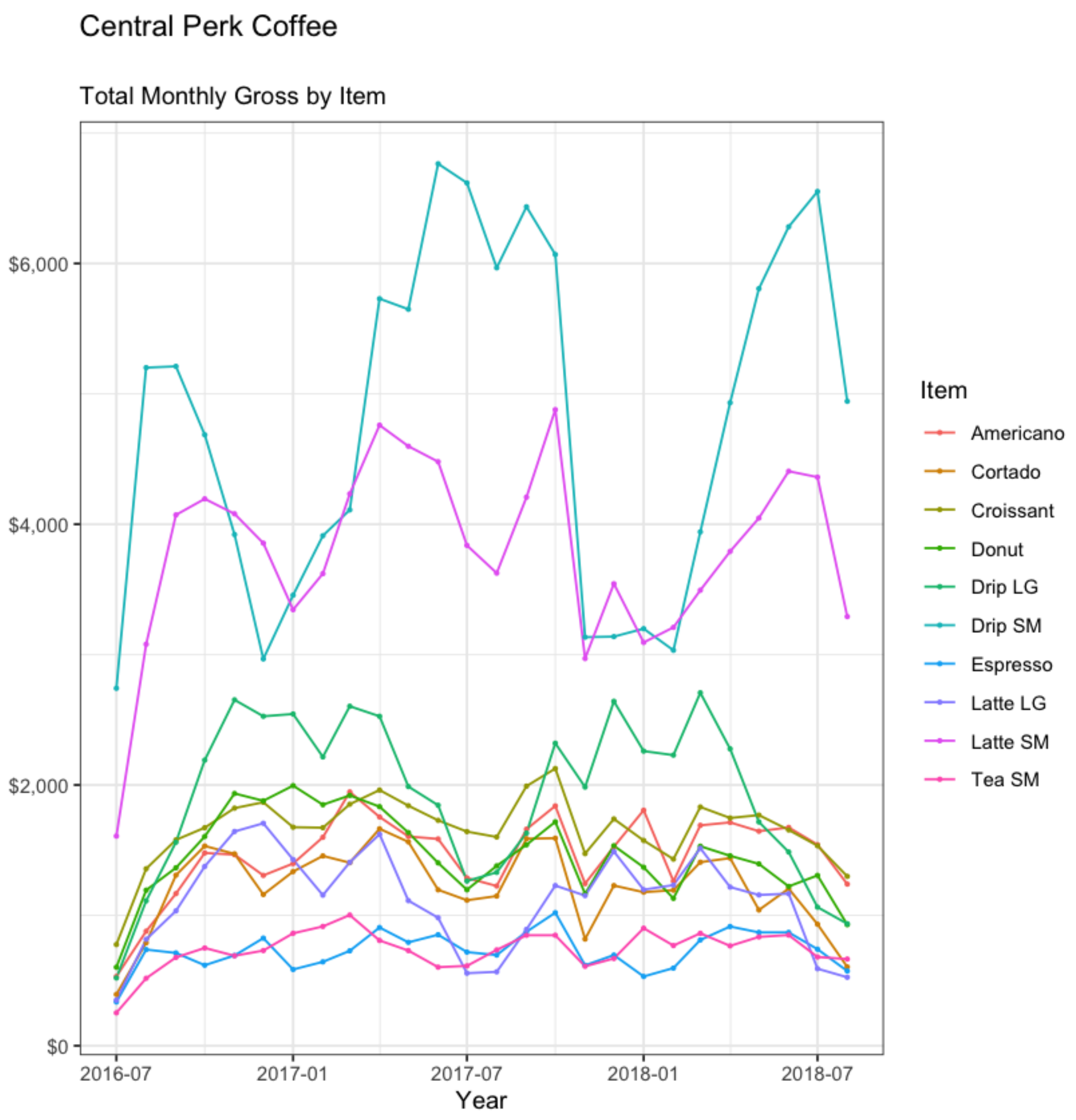
From the exhibit above, we can clearly see that drip coffee, lattes, and cappucinos are overwhelmingly the prefered menu item that customers are purchasing.

[2]:
```
# Highest Grossing Item Overall
TOT <- df %>%
  group_by(Item, year_) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales)) %>%
  group_by(year_) %>% top_n(10)

ggplot(TOT, aes(x=reorder(Item, Gross.Sales.TOT), y=Gross.Sales.TOT)) +
  geom_col() + coord_flip() +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Gross by Item and␣
↪Year",
     x = "Item", y = "Total Gross") +
  scale_y_continuous(labels = scales::dollar, limits = c(0,1.1*max(TOT$Gross.
↪Sales.TOT))) +
  geom_text(label = scales::dollar(TOT$Gross.Sales.TOT), hjust=-.1, size=4) +
  theme_bw() + facet_wrap(~year_, nrow = 3)
```

```
        Error in df %>% group_by(Item, year_) %>% summarise(Gross.Sales.TOT =␣
↪sum(Gross.Sales)) %>% : could not find function "%>%"
    Traceback:
```

We can see that the trend is top selling items is consistent year over year. This also gives us some confidence that the data generating process is relatively stable.

[12]:
```
month.TOT <- df %>%
  group_by(Item, month_, year_) %>%
  summarise(Gross.Sales.TOT = sum(Gross.Sales),
            Net.Sales.TOT = sum(Net.Sales)) %>%
  mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1"))) %>%
  filter(Item %in% c("Drip SM", "Latte SM", "Cappacino", "Drip LG", "Croissant",
                     "Donut", "Americano", "Cortado", "Latte LG", "Tea SM",
                     "Espresso"))

ggplot(month.TOT, aes(x=new_date, y=Gross.Sales.TOT, color = Item)) +
  geom_line() + geom_point(size=.5) +
  labs(title = "Central Perk Coffee\n", subtitle = "Total Monthly Gross by Item",
     x = "Year", y = "") + scale_y_continuous(labels = scales::dollar) +
  theme_bw()
```

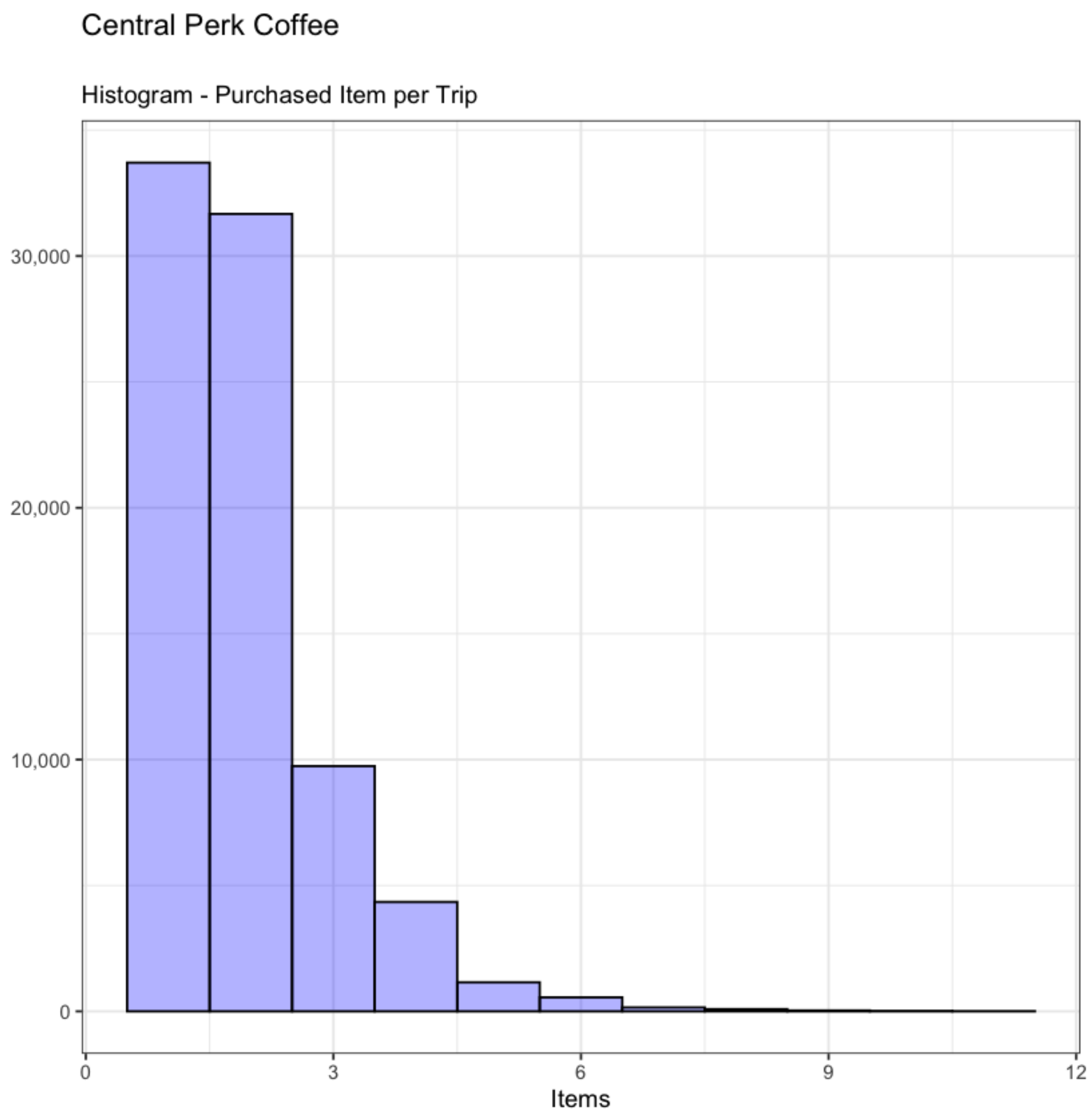


If we examine the by day plots, we see that there are likely to be seasonality trends. Intuitively, this makes sense. New York is in the northern hemisphere and can get quite cold during the winter months (November - April). Hot drinks are, usually, preferred when it's cold out over the hot days of the summer.

Given the distribution of sales and the variety of the items being sold, a basic question might be how many *things* are customers buying when they conduct a transaction?

[13]:
```
customers <- df %>%
  filter(is.na(Customer.ID)!=TRUE) %>%
  group_by(Customer.ID, Date, Time) %>%
  mutate(id = row_number(), total = n())
```

```
N_VALS <- customers %>%
  group_by(Customer.ID, Date, Time) %>%
  summarise(n_vals = sum(Qty)) %>% filter(n_vals < 11.5)

ggplot(N_VALS, aes(x=round(n_vals, 0))) +
  geom_histogram(binwidth = 1, color="black", fill="blue", alpha=.3) +
  labs(title = "Central Perk Coffee\n",
       subtitle = "Histogram - Purchased Item per Trip",
       x = "Items", y = "") + scale_y_continuous(labels = scales::comma) +
  theme_bw()
```

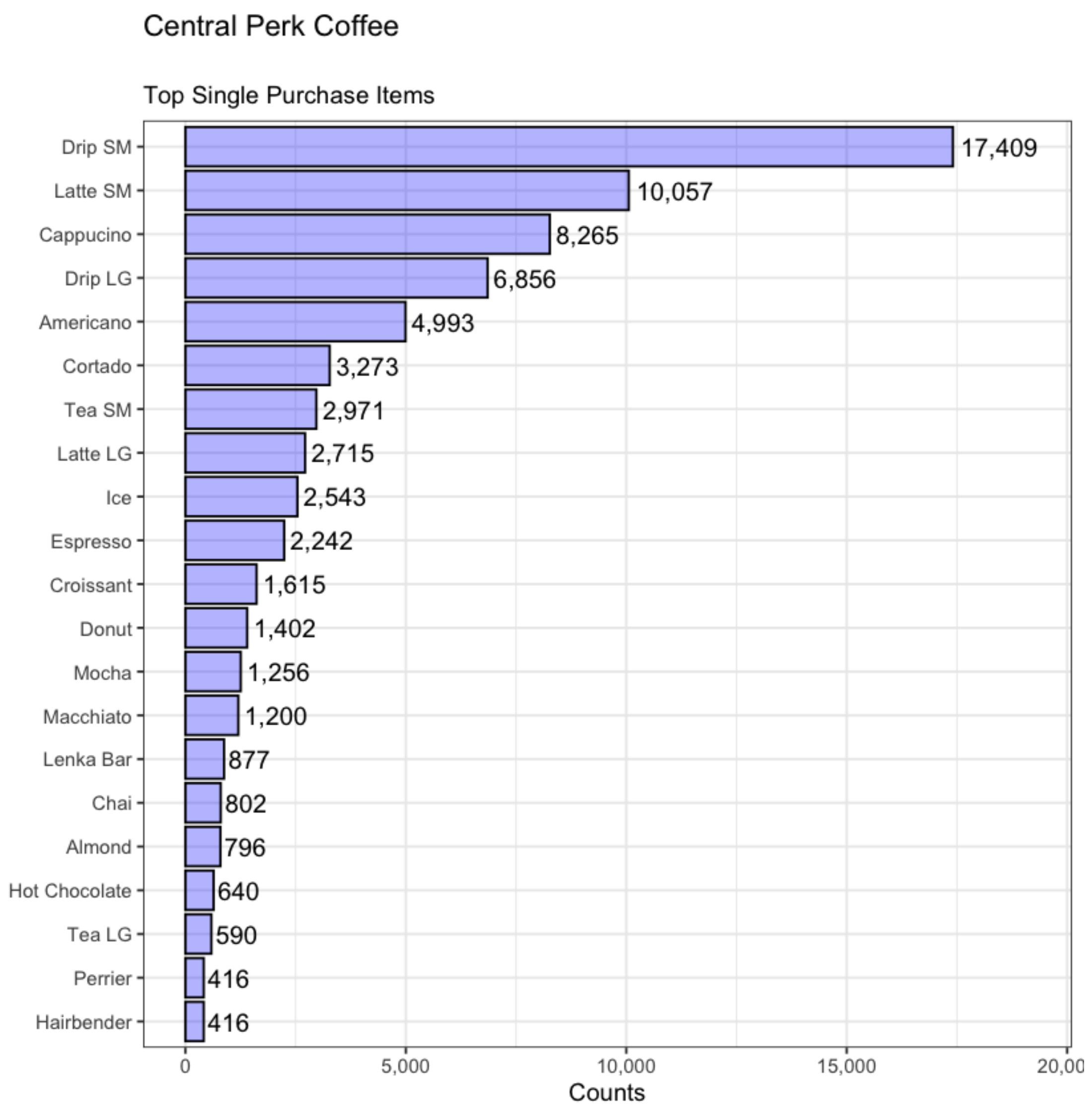


Most transactions are just one or two items. Again, intuitively this makes sense. You stop in, grab a cup of coffee, maybe a donut or a croissant and head on your way.

A follow up question to look at was *what* are customers buying both as single items and in multiple item transactions?

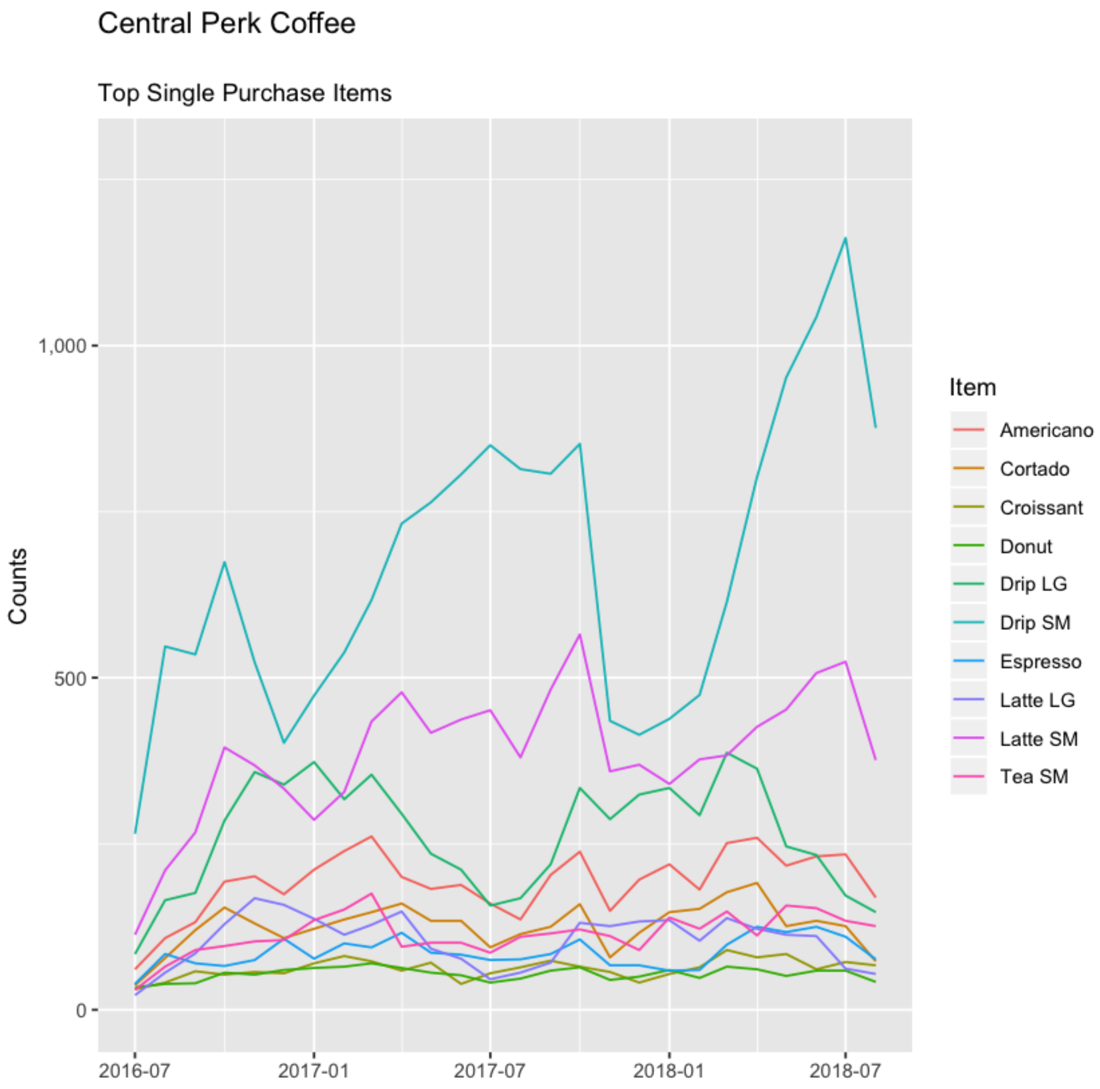
```
[14]: # Most Frequent Single Item Purchases
single_items <- customers %>%
  filter(id == 1, Qty == 1) %>%
  group_by(Item) %>%
  summarise(counts = n()) %>%
  arrange(desc(counts)) %>%
  top_n(20)

ggplot(single_items, aes(x=reorder(Item, counts), y=counts)) +
  geom_col(color="black", fill="blue", alpha=.3) + coord_flip() +
  labs(title="Central Perk Coffee\n", subtitle="Top Single Purchase Items",
       x="", y="Counts") +
  scale_y_continuous(labels = scales::comma, limits = c(0, 1.
↪1*max(single_items$counts))) +
  geom_text(label = scales::comma(single_items$counts), hjust=-.1, size=4) +
  theme_bw()
```

```
Selecting by counts
```



The top items purchased alone look similar to the top selling items in general.

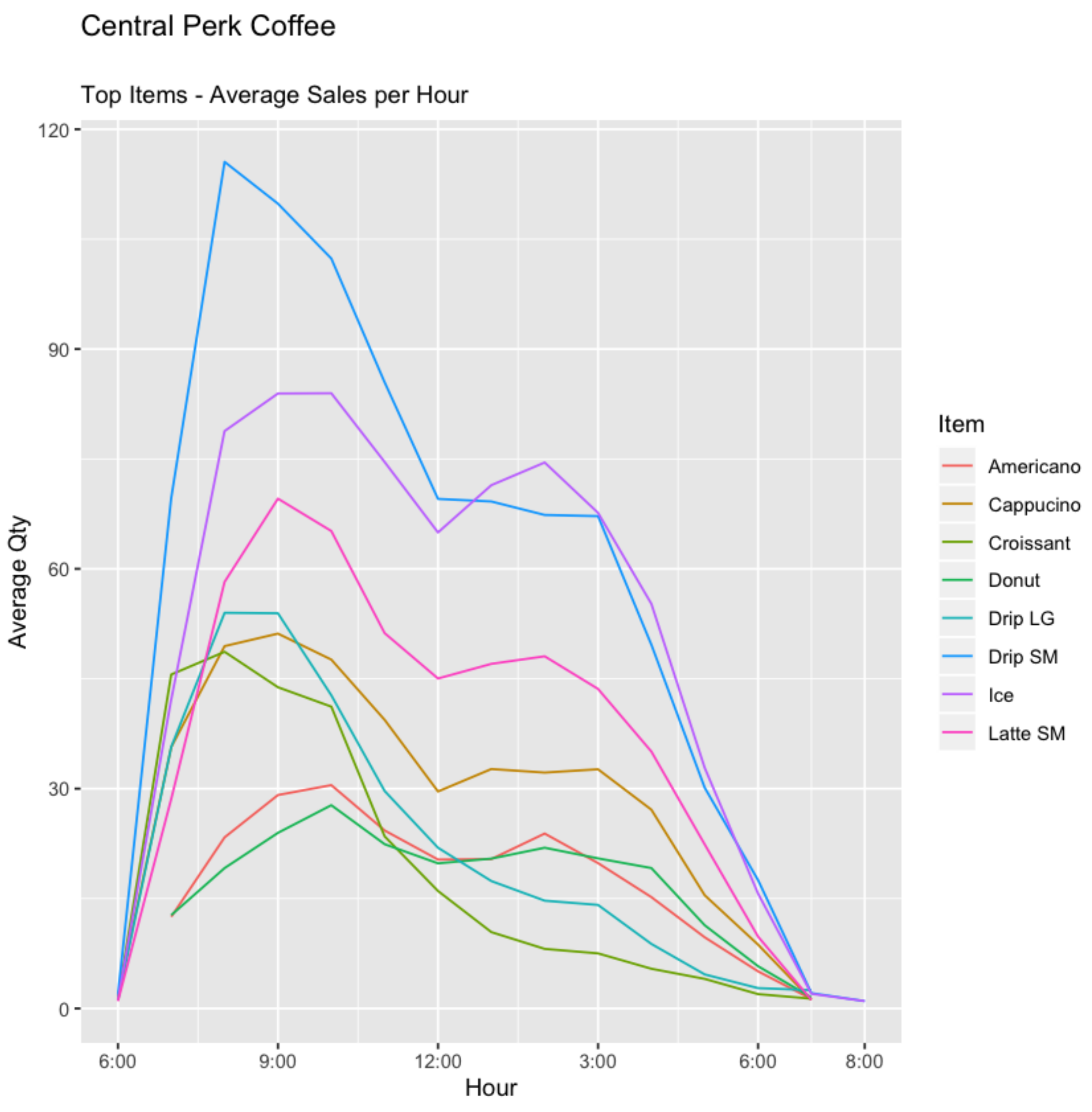
[15]:
```r
# Most Frequent Single Item Purchases
single_items <- customers %>%
  filter(id == 1, Qty == 1) %>%
  group_by(Item, month_, year_) %>%
  summarise(counts = n()) %>%
  mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1"))) %>%
  filter(Item %in% c("Drip SM", "Latte SM", "Cappacino", "Drip LG", "Croissant",
                     "Donut", "Americano", "Cortado", "Latte LG", "Tea SM",
                     "Espresso"))

ggplot(single_items, aes(x=new_date, y=counts, color=Item)) +
  geom_line() +
  labs(title="Central Perk Coffee\n", subtitle="Top Single Purchase Items",
      x="", y="Counts") +
  scale_y_continuous(labels = scales::comma, limits = c(0, 1.
↪1*max(single_items$counts)))
```



While drip coffee dominates sales overall, there still see the aforementioned seasonal trends.

To look into the claim that sales are inconsistent throughout the day, we looked at which items were sold at which time of day:

[17]:
```
customers <- df %>%
  mutate(day_ = lubridate::day(Date)) %>%
  filter(is.na(Customer.ID)==FALSE) %>%
  select(Date, Time, Item, Qty, Gross.Sales, Customer.ID, month_, year_, day_)␣
 ↪%>%
  group_by(Date, Time, Customer.ID, month_, year_, day_) %>%
  mutate(id = row_number(), total = n(), Total.Gross.Sales = sum(Gross.Sales),
         week_day = lubridate::wday(Date, label = TRUE, abbr = FALSE)) %>%
  ungroup() %>%
  select(-c("Gross.Sales", "Date"))

customers <- customers %>% mutate(hour_ = lubridate::hour(Time))


temp <- customers %>%
  select(Item, Qty, hour_, day_) %>%
  group_by(Item, hour_, day_) %>% summarise(totals = sum(Qty)) %>%
  ungroup() %>% group_by(hour_, Item) %>% summarise(avg = mean(totals)) %>%
  filter(Item %in% c("Drip SM", "Ice", "Latte SM", "Cappucino",
                         "Drip LG", "Americano", "Donut", "Croissant"))

ggplot(temp, aes(x=hour_, y=avg, color= Item)) + geom_line() +
  labs(title = "Central Perk Coffee\n", subtitle = "Top Items - Average Sales␣
 ↪per Hour",
       x = "Hour", y = "Average Qty") +
  scale_x_continuous(breaks = c("6:00" = 6, "9:00" = 9, "12:00" = 12, "3:00" =␣
 ↪15,
                                "6:00" = 18, "8:00" = 20))
```



Central Perk believes that sales vary over the course of the day, looking at the items sold by hour, that appears to be true. There is a rush for everything in the morning and then an overall decline through the day. There is a small uptick around 2pm.

To better understand the items purchased together, we looked into association rules. Association rules allow us to generate hypotheses from the data that we can test to investigate the claims from Central Perk.

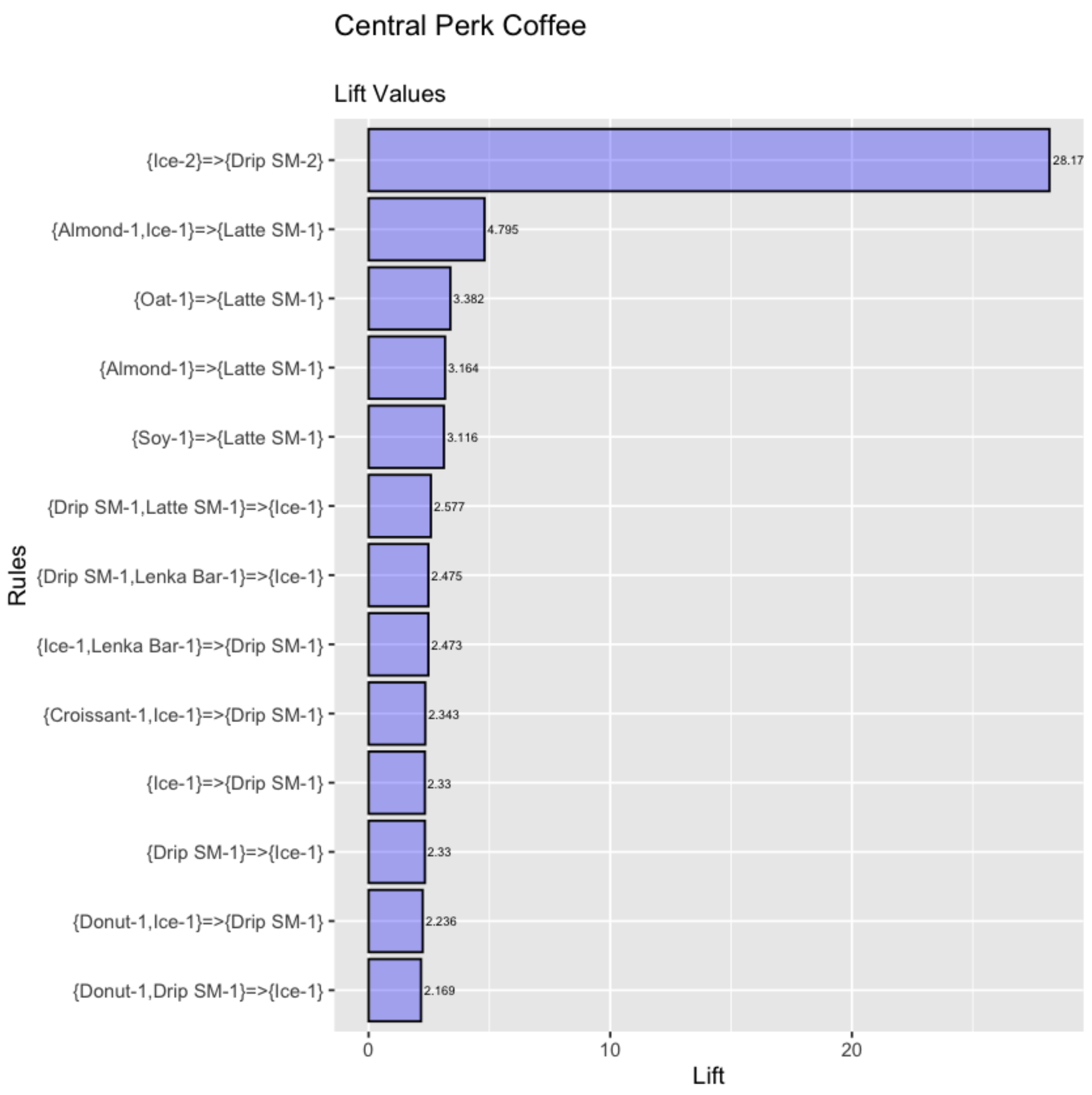
[23]:
```
rules <- apriori(all_data, parameter = list(supp = 0.001, conf = 0.3))
rules_conf <- sort(rules, by="confidence", decreasing = TRUE)

graph_lift(rules_conf)
```

```
Apriori
```

```
Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.3    0.1    1 none FALSE            TRUE       5   0.001      1
 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: 81

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[204 item(s), 81511 transaction(s)] done [0.02s].
sorting and recoding items ... [57 item(s)] done [0.00s].
creating transaction tree ... done [0.02s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [46 rule(s)] done [0.00s].
creating S4 object  ... done [0.01s].
     lhs                                rhs          support     confidence
[1]  {Ice-1,Soy-1}                   => {Latte SM-1} 0.003361509 0.8228228
[2]  {Almond-1,Ice-1}                => {Latte SM-1} 0.009667407 0.7880000
[3]  {Ice-1,Oat-1}                   => {Latte SM-1} 0.002183754 0.7705628
[4]  {Extra Shot-1,Ice-1}            => {Drip SM-1}  0.003717290 0.7112676
[5]  {Ice-1,Lenka Bar-1}             => {Drip SM-1}  0.005631142 0.6575931
[6]  {Almond-1,Drip SM-1,Latte SM-1} => {Ice-1}      0.001153219 0.6482759
[7]  {Almond-1,Drip SM-1,Ice-1}      => {Latte SM-1} 0.001153219 0.6482759
[8]  {Croissant-1,Ice-1}             => {Drip SM-1}  0.007360970 0.6230530
[9]  {Ice-1}                         => {Drip SM-1}  0.143968299 0.6194247
[10] {Ice-2}                         => {Drip SM-2}  0.009655139 0.6162882
[11] {Drip SM-1,Latte SM-1}          => {Ice-1}      0.006158678 0.5990453
[12] {Drip SM-1,Perrier-1}           => {Ice-1}      0.001128682 0.5974026
[13] {Donut-1,Ice-1}                 => {Drip SM-1}  0.008146140 0.5944494
[14] {Financier-1,Ice-1}             => {Drip SM-1}  0.001815706 0.5943775
[15] {Americano-1,Drip SM-1}         => {Ice-1}      0.002588608 0.5893855
[16] {Drip SM-1,Lenka Bar-1}         => {Ice-1}      0.005631142 0.5751880
[17] {Ice-1,Perrier-1}               => {Drip SM-1}  0.001128682 0.5679012
[18] {Drip LG-1,Ice-1}               => {Drip SM-1}  0.002723559 0.5563910
[19] {Oat-1}                         => {Latte SM-1} 0.005079069 0.5557047
[20] {Almond-1,Croissant-1}          => {Latte SM-1} 0.001607145 0.5550847
[21] {Almond-1,Donut-1}              => {Latte SM-1} 0.001668486 0.5440000
[22] {Drip SM-1}                     => {Ice-1}      0.143968299 0.5414321
[23] {Almond-1,Drip SM-1}            => {Latte SM-1} 0.001778901 0.5390335
[24] {Almond-1,Drip SM-1}            => {Ice-1}      0.001778901 0.5390335
[25] {Drip SM-1,Tea SM-1}            => {Ice-1}      0.002061072 0.5333333
[26] {Almond-1}                      => {Latte SM-1} 0.032891266 0.5198759
[27] {Soy-1}                         => {Latte SM-1} 0.010207211 0.5120000
```

```
[28] {Donut-1,Drip SM-1}            => {Ice-1}      0.008146140 0.5041762
[29] {Almond-1,Lenka Bar-1}         => {Latte SM-1} 0.001226828 0.4950495
[30] {Extra Shot-1}                 => {Drip SM-1}  0.009581529 0.4509238
[31] {Latte SM-1,Tea SM-1}          => {Ice-1}      0.001030536 0.4444444
[32] {Drip SM-2}                    => {Ice-2}      0.009655139 0.4413909
[33] {Drip LG-1,Drip SM-1}          => {Ice-1}      0.002723559 0.4319066
[34] {Latte SM-1,Oat-1}             => {Ice-1}      0.002183754 0.4299517
[35] {Drip SM-1,Espresso-1}         => {Ice-1}      0.001410853 0.4275093
[36] {Croissant-1,Drip SM-1}        => {Ice-1}      0.007360970 0.4255319
[37] {Cappucino-1,Drip SM-1}        => {Ice-1}      0.002686754 0.4171429
[38] {Cortado-1,Drip SM-1}          => {Ice-1}      0.001042804 0.4146341
[39] {Drip SM-1,Financier-1}        => {Ice-1}      0.001815706 0.4134078
[40] {Americano-1,Latte SM-1}       => {Ice-1}      0.002085608 0.3881279
[41] {Drip SM-1,Extra Shot-1}       => {Ice-1}      0.003717290 0.3879641
[42] {Cappucino-1,Ice-1}            => {Drip SM-1}  0.002686754 0.3876106
[43] {Latte SM-1,Soy-1}             => {Ice-1}      0.003361509 0.3293269
[44] {Donut-1,Latte SM-1}           => {Ice-1}      0.003447387 0.3282710
[45] {Latte SM-1}                   => {Ice-1}      0.053538786 0.3257932
[46] {Oat-1}                        => {Ice-1}      0.002833973 0.3100671
     lift      count
[1]   5.007026   274
[2]   4.795123   788
[3]   4.689014   178
[4]   2.674916   303
[5]   2.473059   459
[6]   2.789212    94
[7]   3.944876    94
[8]   2.343161   600
[9]   2.329516 11735
[10] 28.174013   787
[11]  2.577397   502
[12]  2.570329    92
[13]  2.235589   664
[14]  2.235319   148
[15]  2.535835   211
[16]  2.474750   459
[17]  2.135748    92
[18]  2.092460   222
[19]  3.381564   414
[20]  3.377791   131
[21]  3.310338   136
[22]  2.329516 11735
[23]  3.280116   145
[24]  2.319195   145
[25]  2.294671   168
[26]  3.163539  2681
[27]  3.115613   832
[28]  2.169222   664
```

```
[29]  3.012466   100
[30]  1.695822   781
[31]  1.912225    84
[32] 28.174013   787
[33]  1.858281   222
[34]  1.849870   178
[35]  1.839362   115
[36]  1.830854   600
[37]  1.794760   219
[38]  1.783966    85
[39]  1.778690   148
[40]  1.669923   170
[41]  1.669219   303
[42]  1.457716   219
[43]  1.416931   274
[44]  1.412388   281
[45]  1.401728  4364
[46]  1.334066   231
```



The association rules appear to mostly be drink modifications (ice and milk type) though there are some food/drink combos. Drip coffee dominates here as well, but now with ice.

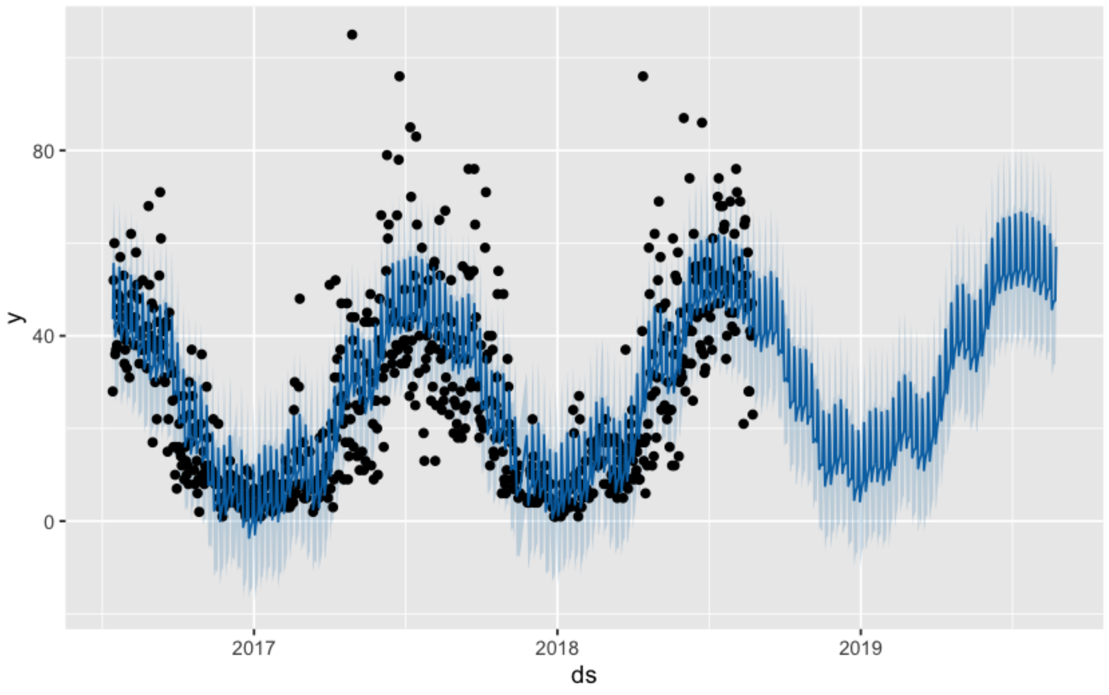
One might also notice the propensity with which 'Ice' appears in the association rule output. This is quite interesting, as a strategy of focusing on cold coffee, whether iced or cold brewed, could be utilized by Central Perk to help prevent the dip in sales during the summer months.

Cold Coffee and Seasonality

To build upon and balance the results of our association rules analysis, we also examined the time-series nature of iced coffee sales. Based on the combined results of these analyses, we believe the roll-out of our new cold coffee roster should be timed to coincide with the summer months, since this is where our iced coffee offering have peaked historically.

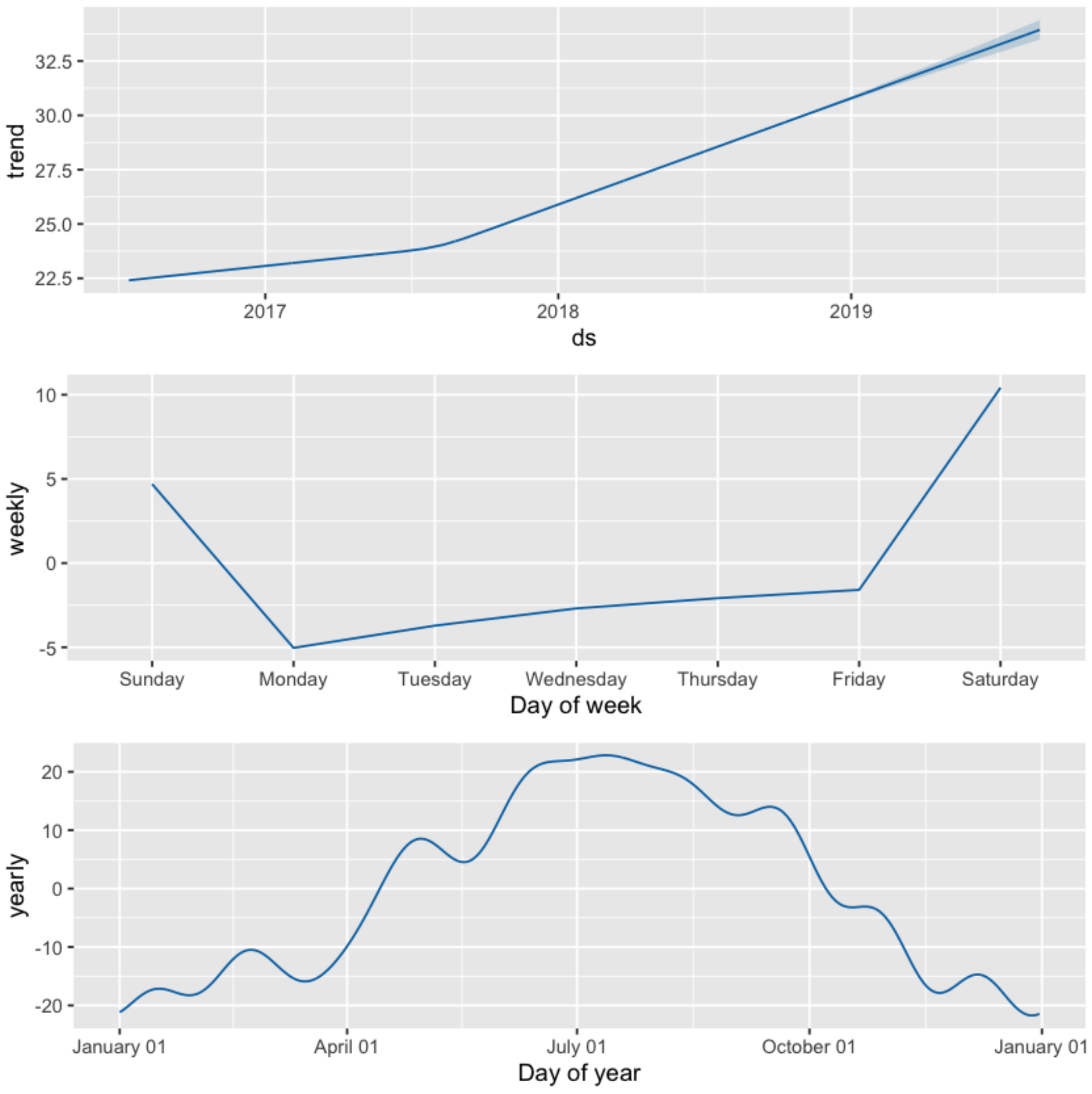
For this we choose to use the prophet library, opensourced by the Facebook Core Data Science team. Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. A key benefit of prophet is that it is robust to missing data and shifts in the trend, and typically handles outliers well.

[6]:
```
library(prophet)

# Please See Code Appendix A.
jxx <- jx %>% filter(grepl("Drip", Item_Sequence) & grepl("Ice", Item_Sequence))␣
 ↪%>%
  select(Item_Sequence, Date) %>% group_by(Date) %>% summarise(totals = n()) %>%
  ungroup() %>% select(ds = Date, y = totals)

m <- prophet(jxx)
future <- make_future_dataframe(m, periods = 365)
forecast <- predict(m, future)
plot(m, forecast)
```

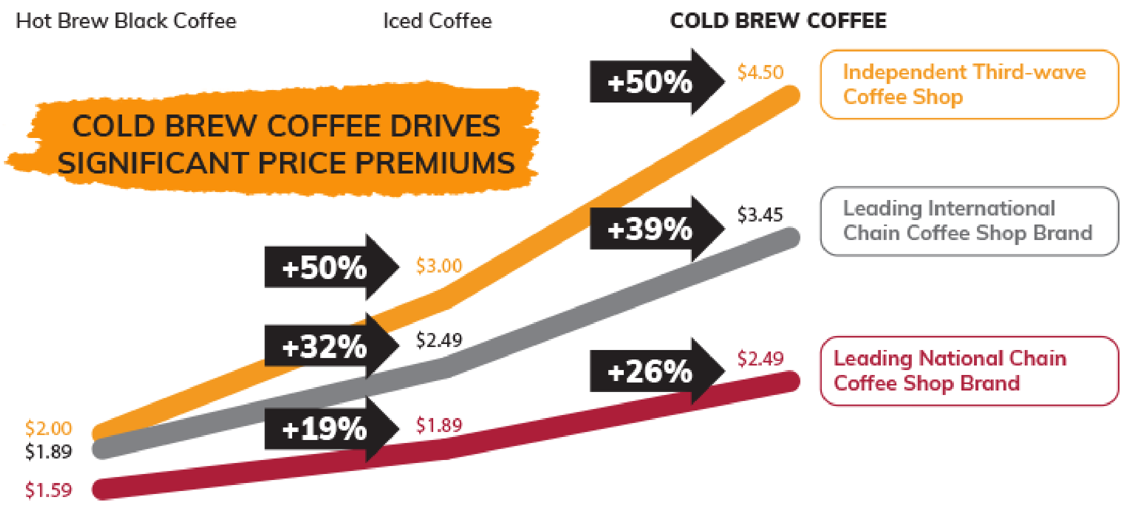


Simple forecasting can be applied with the Prophet package to project future sales from the time series data, including uncertainty metrics around the projections.

[7]:
```
prophet_plot_components(m, forecast)
```



Since, the sales of Ice and Drip SM coffee have performed particularly well over the summer months, peaking in July. We hope to exploit this pattern by introducing a more attractive/upscale range of easily implemented iced coffee options. Based on our market research, we believe the eventual introduction of cold press coffee options and dedicated iced coffee options would allow us to increase our sales margins during the warmest months of the year when sales have, otherwise, traditionally sagged.



Source: https://www.nestleprofessional.us/cold-brew-curious

Based on our current annual projections, we would currently expect to experience a similar, if not increased, amount of demand for iced coffee in 2019. While there are some residual questions surrounding price elasticity, we believe the anticipated increase in demand should soften the effect of any changes to our current pricing structure.

2.2 Looking a bit more closely at our top customers

Regulars and top customers drive a sustained portion of revenue and sales, so a deeper review of the data is wise.

[3]:
```
##  Pull in the packages required to perform our analyis
suppressPackageStartupMessages(library(tidyverse))
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(ggplot2))
suppressPackageStartupMessages(library(forcats))
suppressPackageStartupMessages(library(lubridate))
suppressPackageStartupMessages(library(data.table))

## Read it back in so we don't have to do all the pre-processing again

centralperk_df <- suppressMessages(readr::read_csv(file = "CentralParkAllYears.
 ↪csv"))

attach(centralperk_df)

## Now that we've created a unique identifier for every transaction, then we can␣
 ↪use this to mine the rules
```

```
centralperk_df$transancation_id = paste(centralperk_df$Date, centralperk_df$Time)
```

```
The following objects are masked from centralperk_df (pos = 14):

    Category, Customer.ID, Date, DayofWeek, Discounts, Event.Type,
    Gross.Sales, Hour, id, Item, Month, Net.Sales, Notes,
    Price.Point.Name, Qty, Tax, Time, Year
```

[4]:
```
## Being our analysis for repeate customers by removing any rows that have no␣
 ↪Customer ID
repeat_customers <- centralperk_df %>% filter(Customer.ID != 'NA')
```

[5]:
```
## Group by customers to get count of trips over the life span of data
## Arrange them in descending order to see how often they came in

grouped_repeat_customer <-
        repeat_customers %>%
            group_by(Customer.ID) %>%
                summarise(trip_count = n()) %>%
                    arrange(-trip_count)
```

[6]:
```
top_hundred_customers <- head(grouped_repeat_customer, 100)

top_hundred_customers_ids <- top_hundred_customers$Customer.ID
```

[7]:
```
## How many discounts did our top 100 customers receive over the three years␣
 ↪they visited Central Perk?

repeat_customers %>%
        filter(Customer.ID %in% top_hundred_customers_ids) %>%
            summarise(total_discount = sum(Discounts))

## A total of less than $16.00 over three years!   Not great!
```

total_discount
-15.99

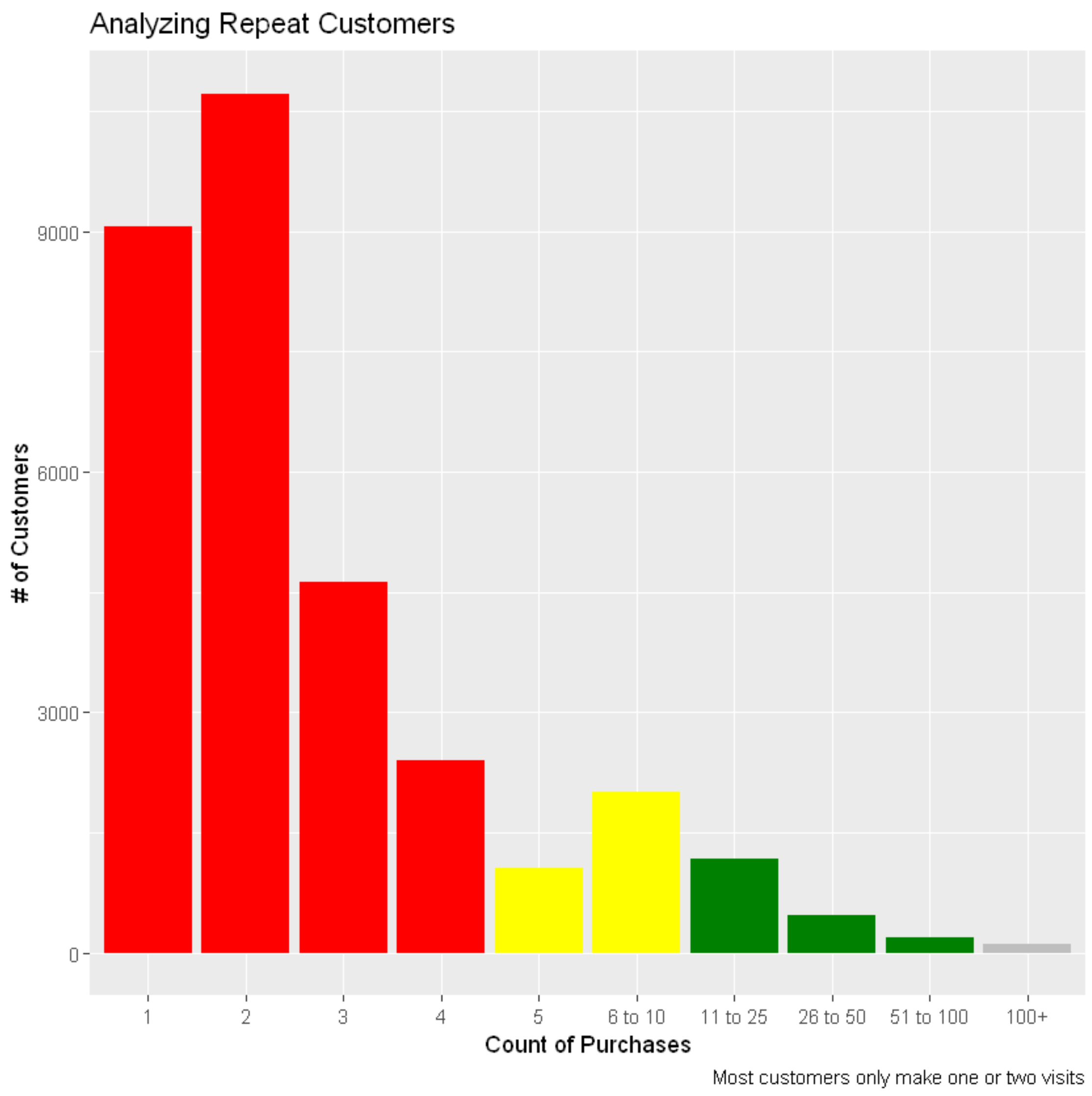
Our top 100 customers only received a total discount of $16.00 (total!) in the three years! That's not a good way to get people to return to our coffee shop. Regulars also fill seats and they are key nodes in the recommendation network. They are likely to recommend the shop to friends and family more than the pass-by customer.

[8]:
```
## Create categories to bucket customer segments into better groups
bins <- c(1, 2, 3, 4, 5, 6, 11, 26, 51, 101, 700)

binlabels <- c("1", "2", "3", "4", "5", "6 to 10", "11 to 25", "26 to 50", "51␣
 ↪to 100", "100+")
```

```
## Use a function to add a new column to our table to show these nice bucketed␣
 ↪groups
setDT(grouped_repeat_customer) [ , bins := cut(trip_count,
                                              breaks = bins,
                                              right = FALSE,
                                              labels = binlabels)]
```

[9]:
```
## Now group our visits with the appropriate label

customer_visits_binned <-
        grouped_repeat_customer %>%
            group_by(bins) %>%
                summarise(customer_visits = n())
```

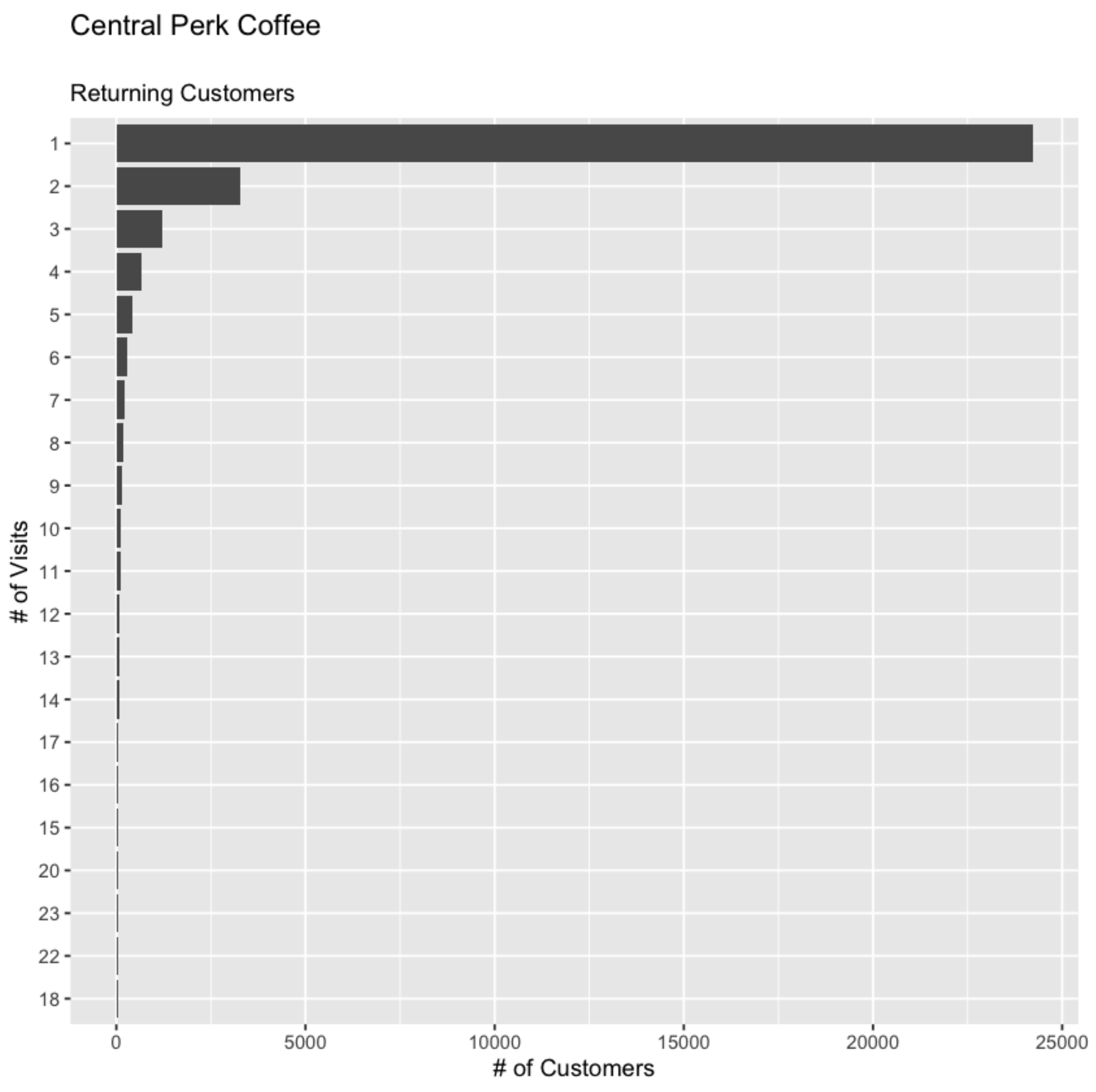
[10]:
```
## Create a nice visualization showing customer visits

ggplot(customer_visits_binned, aes ( x = bins, weight = customer_visits, fill =␣
 ↪bins)) +
    geom_bar() +
    scale_fill_manual("legend", values = c("1" = "red", "2" = "red", "3" =␣
 ↪"red", "4" ="red",
                                          "5" = "yellow", "6 to 10" = "yellow",␣
 ↪"11 to 25" = "#008000",
                                           "26 to 50" = "#008000", "51 to 100" =␣
 ↪"#008000", "100+" = "008000")) +
    theme(legend.position = "none") +
    labs(x = "Count of Purchases", title = "Analyzing Repeat Customers", y = "#␣
 ↪of Customers",
        caption = "Most customers only make one or two visits")
```



We can see that most customer only made one or two visits, based on the data collected. This isn't too surprising, given the location near central park, but we would like to see more customers in the right towers of the histogram. Seeing increased return visits by customers indicates an increased number of regular customers.

```
[38]: it %>%
        group_by(Customer.ID) %>%
        summarise(total = n()) %>%
        ungroup() %>% group_by(total) %>%
        summarise(totals = n()) %>% top_n(20) %>%
        ggplot(aes(x=reorder(as.factor(total), totals), y=totals)) + geom_col() +
        coord_flip() + labs(title = "Central Perk Coffee\n", subtitle = "Returning␣
      ↪Customers",
                    x = "# of Visits", y="# of Customers")
```

```
Selecting by totals
```



This shows a lot of customers with one visit, this runs counter to the idea that the customer base is loyal, but we should look at the impact of repeat customers on sales

[39]:
```
customers <- df %>%
  mutate(Customer.ID = case_when(is.na(Customer.ID) == TRUE ~ "Unknown",
                                 is.na(Customer.ID) != TRUE ~ Customer.ID)) %>%
  select(Date, Time, Item_Qty, Customer.ID, Gross.Sales, month_, year_, Qty) %>%
  group_by(Date, Time, month_, year_) %>%
  mutate(id = row_number(), total = n(), Total.Gross.Sales = sum(Gross.Sales),
```

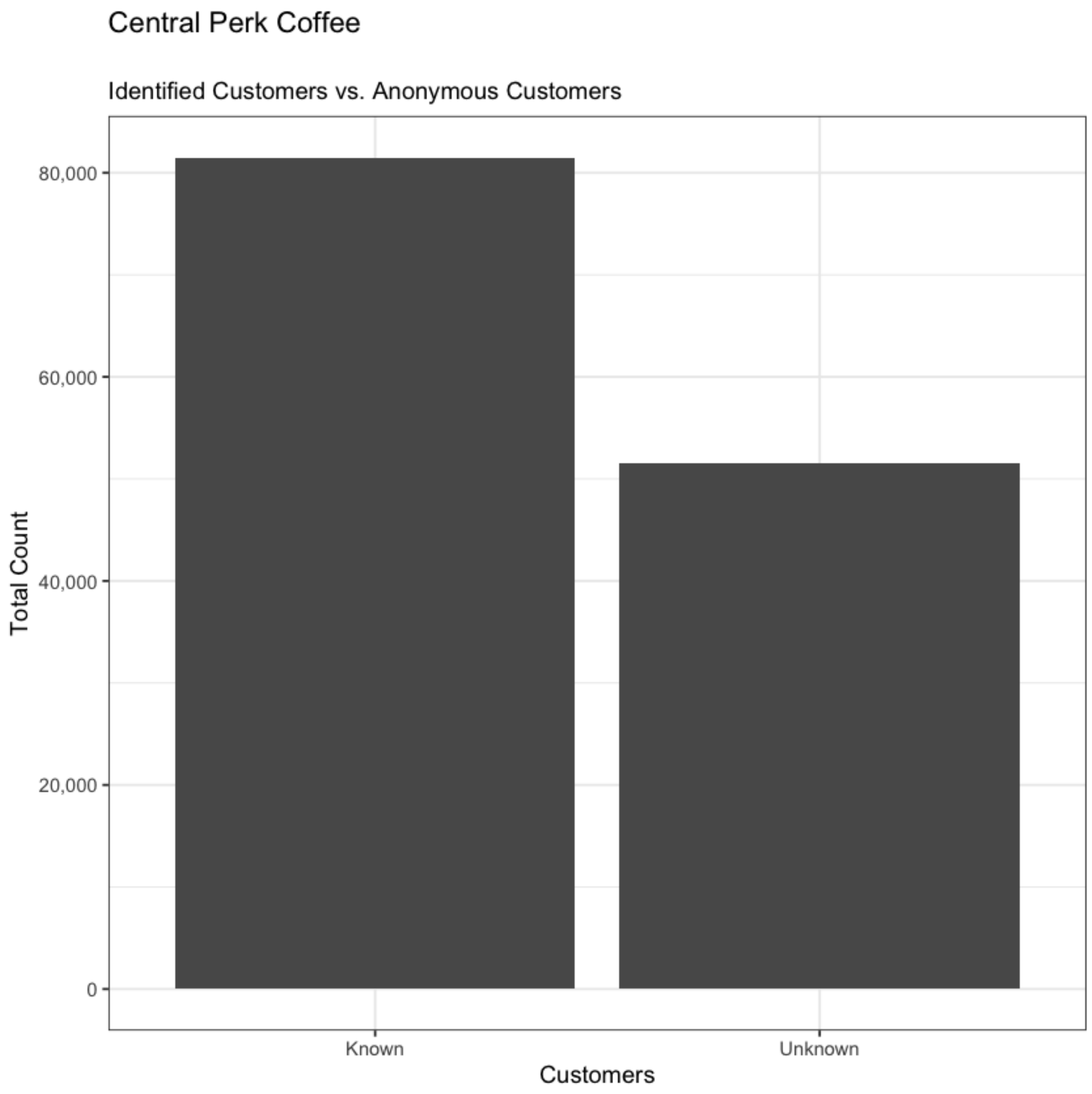
```r
          week_day = lubridate::wday(Date, label = TRUE, abbr = FALSE),
          total_qty = sum(Qty)) %>%
  ungroup() %>% select(-c("Gross.Sales", "Qty")) %>%
  mutate(hour_ = lubridate::hour(Time)) %>% group_by()

customers

reformat_data <- function(customers) {
  item1 <- customers %>% filter(id == 1) %>%
    select(-c("id")) %>% rename(Item1 = Item_Qty)
  item2 <- customers %>% filter(id == 2) %>%
    select(-c("id")) %>% rename(Item2 = Item_Qty)
  item3 <- customers %>% filter(id == 3) %>%
    select(-c("id")) %>% rename(Item3 = Item_Qty)
  item4 <- customers %>% filter(id == 4) %>%
    select(-c("id")) %>% rename(Item4 = Item_Qty)
  item5 <- customers %>% filter(id == 5) %>%
    select(-c("id")) %>% rename(Item5 = Item_Qty)
  item6 <- customers %>% filter(id == 6) %>%
    select(-c("id")) %>% rename(Item6 = Item_Qty)
  item7 <- customers %>% filter(id == 7) %>%
    select(-c("id")) %>% rename(Item7 = Item_Qty)
  item8 <- customers %>% filter(id == 8) %>%
    select(-c("id")) %>% rename(Item8 = Item_Qty)
  item9 <- customers %>% filter(id == 9) %>%
    select(-c("id")) %>% rename(Item9 = Item_Qty)
  item10 <- customers %>% filter(id == 10) %>%
    select(-c("id")) %>% rename(Item10 = Item_Qty)
  item11 <- customers %>% filter(id == 11) %>%
    select(-c("id")) %>% rename(Item11 = Item_Qty)
  item12 <- customers %>% filter(id == 12) %>%
    select(-c("id")) %>% rename(Item12 = Item_Qty)
  item13 <- customers %>% filter(id == 13) %>%
    select(-c("id")) %>% rename(Item13 = Item_Qty)
  item14 <- customers %>% filter(id == 14) %>%
    select(-c("id")) %>% rename(Item14 = Item_Qty)
  item15 <- customers %>% filter(id == 15) %>%
    select(-c("id")) %>% rename(Item15 = Item_Qty)
  item16 <- customers %>% filter(id == 16) %>%
    select(-c("id")) %>% rename(Item16 = Item_Qty)
  item17 <- customers %>% filter(id == 17) %>%
    select(-c("id")) %>% rename(Item17 = Item_Qty)

  concat_string <- c("Customer.ID", "month_", "year_", "hour_", "week_day",
                     "Total.Gross.Sales", "Time", "total", "total_qty", "Date")
  j1 <- left_join(item1, item2, by = concat_string)
  j2 <- left_join(j1, item3, by = concat_string)
  j3 <- left_join(j2, item4, by = concat_string)
  j4 <- left_join(j3, item5, by = concat_string)
  j5 <- left_join(j4, item6, by = concat_string)
  j6 <- left_join(j5, item7, by = concat_string)
  j7 <- left_join(j6, item8, by = concat_string)
  j8 <- left_join(j7, item9, by = concat_string)
  j9 <- left_join(j8, item10, by = concat_string)
  j10 <- left_join(j9, item11, by = concat_string)
  j11 <- left_join(j10, item12, by = concat_string)
  j12 <- left_join(j11, item13, by = concat_string)
  j13 <- left_join(j12, item14, by = concat_string)
  j14 <- left_join(j13, item15, by = concat_string)
  j15 <- left_join(j14, item16, by = concat_string)
  j16 <- left_join(j15, item17, by = concat_string)
  it <- j16 %>%
    select(Customer.ID, month_, year_, hour_, week_day, Total.Gross.Sales,
           Item1, Item2, Item3, Item4, Item5, Item6, Item7, Item8, Item9, Item10,
           Item11,Item12, Item13, Item14, Item15, Item16, Item17, total,␣
 ↪total_qty, Date)
  return(it)
  }
it <- reformat_data(customers)
```

A tibble: 221560 × 12

Date	Time	Item_Qty	Customer.ID
<date>	<drtn>	<chr>	<chr>
2017-12-31	16:57:33	Cappucino-1	Unknown
2017-12-31	16:37:29	Tea SM-1	Unknown
2017-12-31	16:33:08	Cappucino-1	2a90ca0a266d3e357b693cc366e59e91569726de
2017-12-31	16:33:08	Espresso-1	2a90ca0a266d3e357b693cc366e59e91569726de
2017-12-31	16:31:29	Espresso-1	Unknown
2017-12-31	16:11:06	Alm Rasp-1	398cfea2ef68cc372b4593ed54991917a3e1bba3
2017-12-31	16:11:06	Drip SM-1	398cfea2ef68cc372b4593ed54991917a3e1bba3
2017-12-31	16:01:41	Cappucino-1	Unknown
2017-12-31	15:34:21	Drip SM-1	Unknown
2017-12-31	15:23:51	Americano-1	49f2a6f79ab242c7c973a189bee75879be4d5b51
2017-12-31	15:21:17	Mocha-1	Unknown
2017-12-31	15:20:58	Drip LG-1	Unknown
2017-12-31	15:20:58	Tea SM-1	Unknown
2017-12-31	15:19:40	Drip SM-1	Unknown
2017-12-31	15:17:50	Drip SM-1	Unknown
2017-12-31	15:17:50	Latte SM-1	Unknown
2017-12-31	15:11:31	Latte SM-1	f7ec8686685b8b951c4241c269bb619ac194e281
2017-12-31	15:11:31	Almond-1	f7ec8686685b8b951c4241c269bb619ac194e281
2017-12-31	15:11:31	Latte SM-1	f7ec8686685b8b951c4241c269bb619ac194e281
2017-12-31	15:07:57	Cappucino-1	7d11e118c143f4182a275ccf6ae020963b134b7e
2017-12-31	15:05:49	Drip SM-2	b4f10dad52bdda286335c7f4351f9421afb9b673
2017-12-31	15:03:34	Drip LG-1	b4f10dad52bdda286335c7f4351f9421afb9b673
2017-12-31	15:03:34	Americano-1	b4f10dad52bdda286335c7f4351f9421afb9b673
2017-12-31	15:03:34	Latte SM-1	b4f10dad52bdda286335c7f4351f9421afb9b673
2017-12-31	14:55:44	Mocha-1	1df1fdf05c803f2f30830f41c9da6a644ce21df9
2017-12-31	14:51:44	Hot Chocolate-1	bebe4c52eb5385b31fe7658bc9a89d0b08bd2e67
2017-12-31	14:51:13	Drip LG-1	24000a943adb8c0752628292f5e1c2ccc7b962d0
2017-12-31	14:49:16	Donut-1	94bca045b50c32661141a83c15bd8ecc95b244a3
2017-12-31	14:49:16	Drip SM-1	94bca045b50c32661141a83c15bd8ecc95b244a3
2017-12-31	14:43:41	Latte SM-1	149adf224bd932cc0ec6ae1ef13e4d208f02ca2b
2018-01-02	08:55:34	Drip LG-1	a07d0b82f0638158f5e2fcd963a3a3a093665fac
2018-01-02	08:54:38	Latte LG-1	424ece3b329bad9dc6e9b802cb1d35d1190e01cd
2018-01-02	08:50:55	Drip SM-1	Unknown
2018-01-02	08:50:29	Drip LG-1	b341b0b37b46eda04e3cdf06a99849225fa99bd7
2018-01-02	08:50:29	Hot Chocolate-1	b341b0b37b46eda04e3cdf06a99849225fa99bd7
2018-01-02	08:50:29	Croissant-1	b341b0b37b46eda04e3cdf06a99849225fa99bd7
2018-01-02	08:50:29	Donut-2	b341b0b37b46eda04e3cdf06a99849225fa99bd7
2018-01-02	08:46:44	Espresso-1	695a73fc089126421c046d86048751c4fc72b20c
2018-01-02	08:44:31	Latte SM-1	dd6b911c4cdc7352f92520fbd1d99891f91b383c
2018-01-02	08:31:32	Drip SM-1	Unknown
2018-01-02	08:19:09	Latte LG-2	Unknown
2018-01-02	08:19:09	Latte SM-2	Unknown
2018-01-02	08:19:09	Almond-2	Unknown
2018-01-02	08:17:06	CostaR-1	a224c0d2ea95e0acdcdb3415a6a41972822b81f7
2018-01-02	08:14:47	Cappucino-2	d1ab921af75b562940bb8b95c28b2190e5ad9466
2018-01-02	08:13:12	Croissant-1	87ac3738c2f1243fddb830ed164d90533ab28d64
2018-01-02	08:02:25	Drip SM-1	Unknown
2018-01-02	08:00:34	Cappucino-1	5c42523471817023a341d5b768c6912835c43714
2018-01-02	07:55:47	Drip SM-1	af495134246a402d512789fa9f8b700dccb1142c
2018-01-02	07:55:47	Croissant-1	af495134246a402d512789fa9f8b700dccb1142c
2018-01-02	07:52:41	Drip LG-2	738b0b0ef184dd3e9d3e4f2e4b865dc551e08f8e

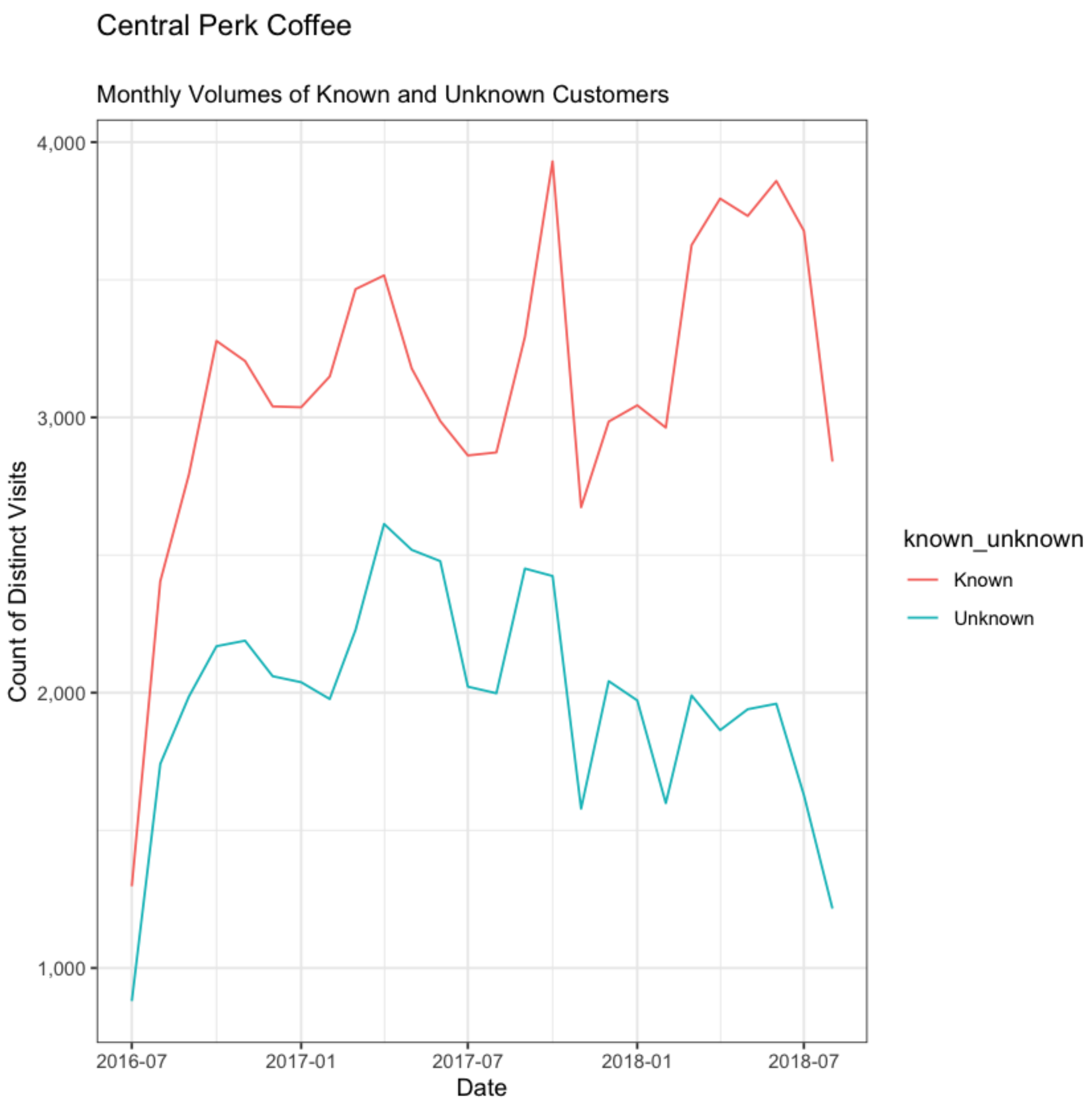
Continuing this look a returning customers, we looked at identified and anonymous customers

[40]:
```r
itx <- it %>%
  mutate(known_unknown = case_when(Customer.ID == "Unknown" ~ "Unknown",
                                   Customer.ID != "Unknown" ~ "Known"))

itx %>% group_by(known_unknown) %>% summarise(totals = n()) %>%
  ggplot(aes(x = known_unknown, y = totals)) + geom_col() +
  labs(title = "Central Perk Coffee\n",
       subtitle = "Identified Customers vs. Anonymous Customers",
       x = "Customers", y = "Total Count") +
  scale_y_continuous(labels = scales::comma) + theme_bw()
```



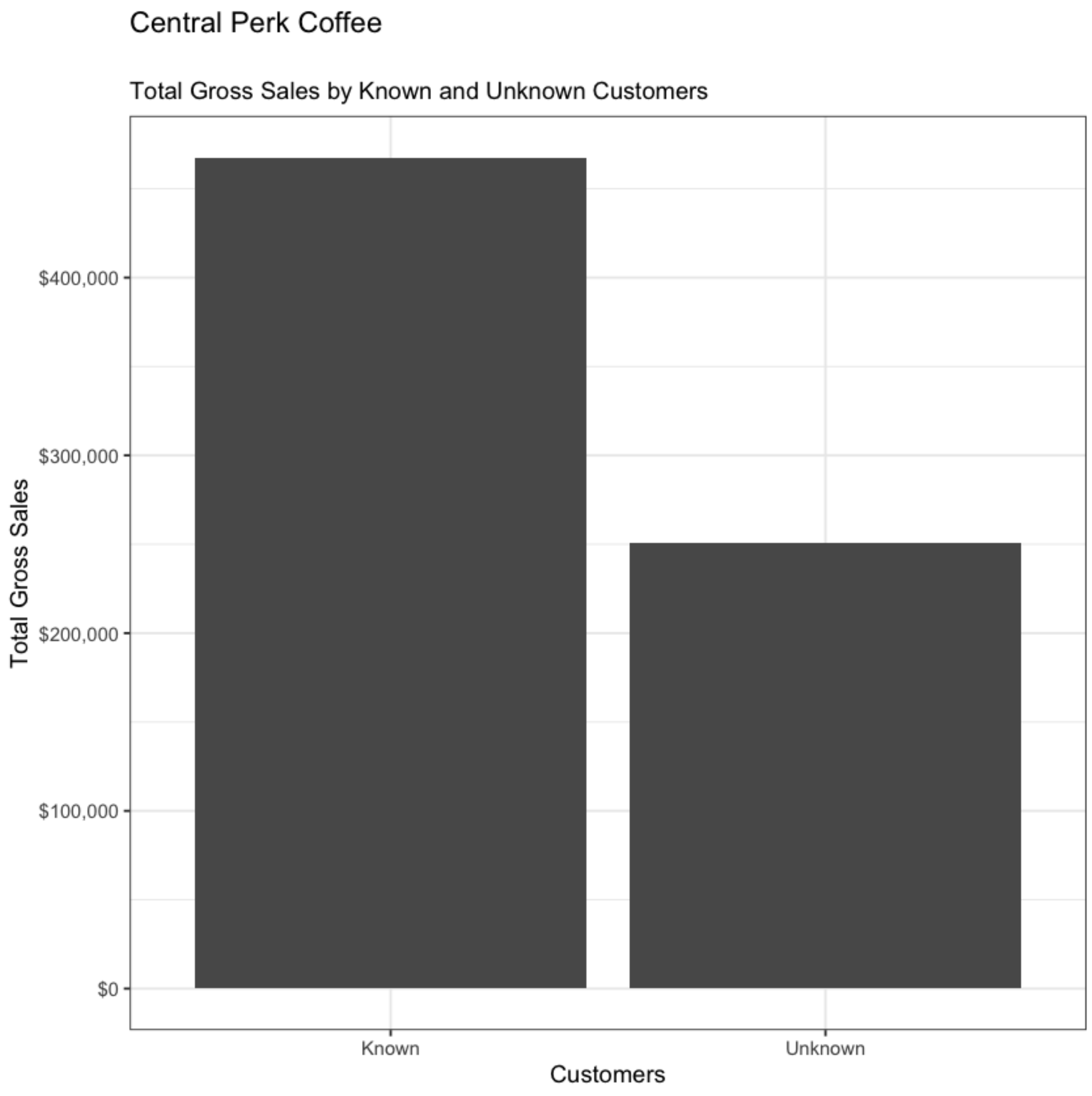
There are more known customers than unknown in the dataset. We then took a look at the monthly sales trends between the groups

```
[41]: itx %>% group_by(known_unknown, month_, year_) %>% summarise(totals = n()) %>%
        mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1"))) %>%
        ggplot(aes(x=new_date, y=totals, color = known_unknown)) + geom_line() +
        labs(title = "Central Perk Coffee\n",
             subtitle = "Monthly Volumes of Known and Unknown Customers",
             x = "Date", y = "Count of Distinct Visits") +
        scale_y_continuous(labels = scales::comma) + theme_bw()
```

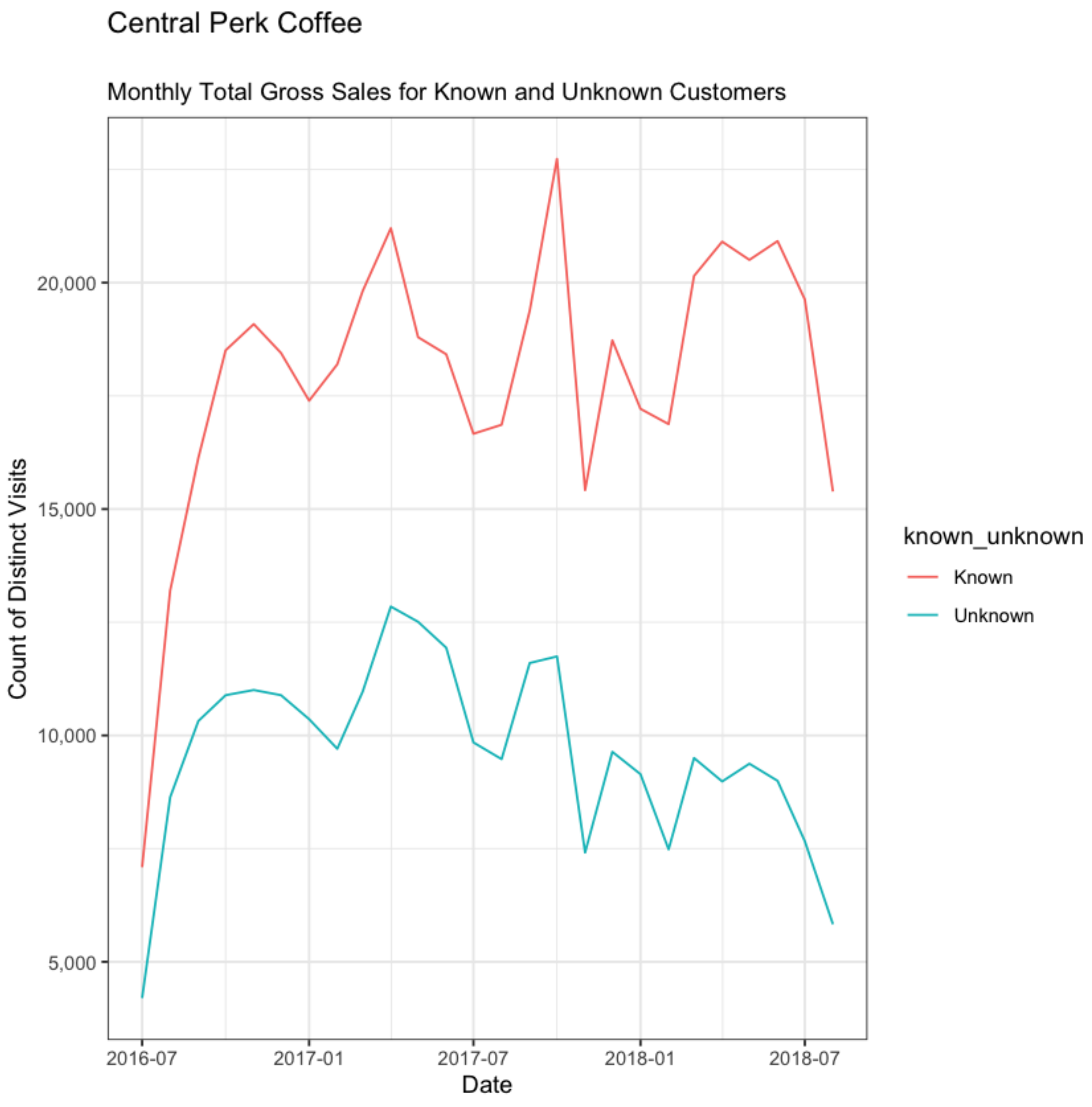


The trends are similar between the groups but the known customers are contributing more sales. This gives support to Central Perk's belief that the customers are loyal.

```
[42]: xx <- itx %>% group_by(known_unknown) %>% summarise(total = sum(Total.Gross.
      ↪Sales))
       ggplot(xx, aes(x = known_unknown, y = total)) + geom_col() +
       labs(title = "Central Perk Coffee\n",
            subtitle = "Total Gross Sales by Known and Unknown Customers",
            x = "Customers", y = "Total Gross Sales") +
         scale_y_continuous(labels = scales::dollar) + theme_bw()
```



```
[43]: itx %>% group_by(known_unknown, month_, year_) %>% summarise(totals = sum(Total.
      ↪Gross.Sales)) %>%
       mutate(new_date = lubridate::ymd(paste0(year_, "-", month_, "-1"))) %>%
       ggplot(aes(x=new_date, y=totals, color = known_unknown)) + geom_line() +
       labs(title = "Central Perk Coffee\n",
            subtitle = "Monthly Total Gross Sales for Known and Unknown Customers",
            x = "Date", y = "Count of Distinct Visits") +
       scale_y_continuous(labels = scales::comma) + theme_bw()
```

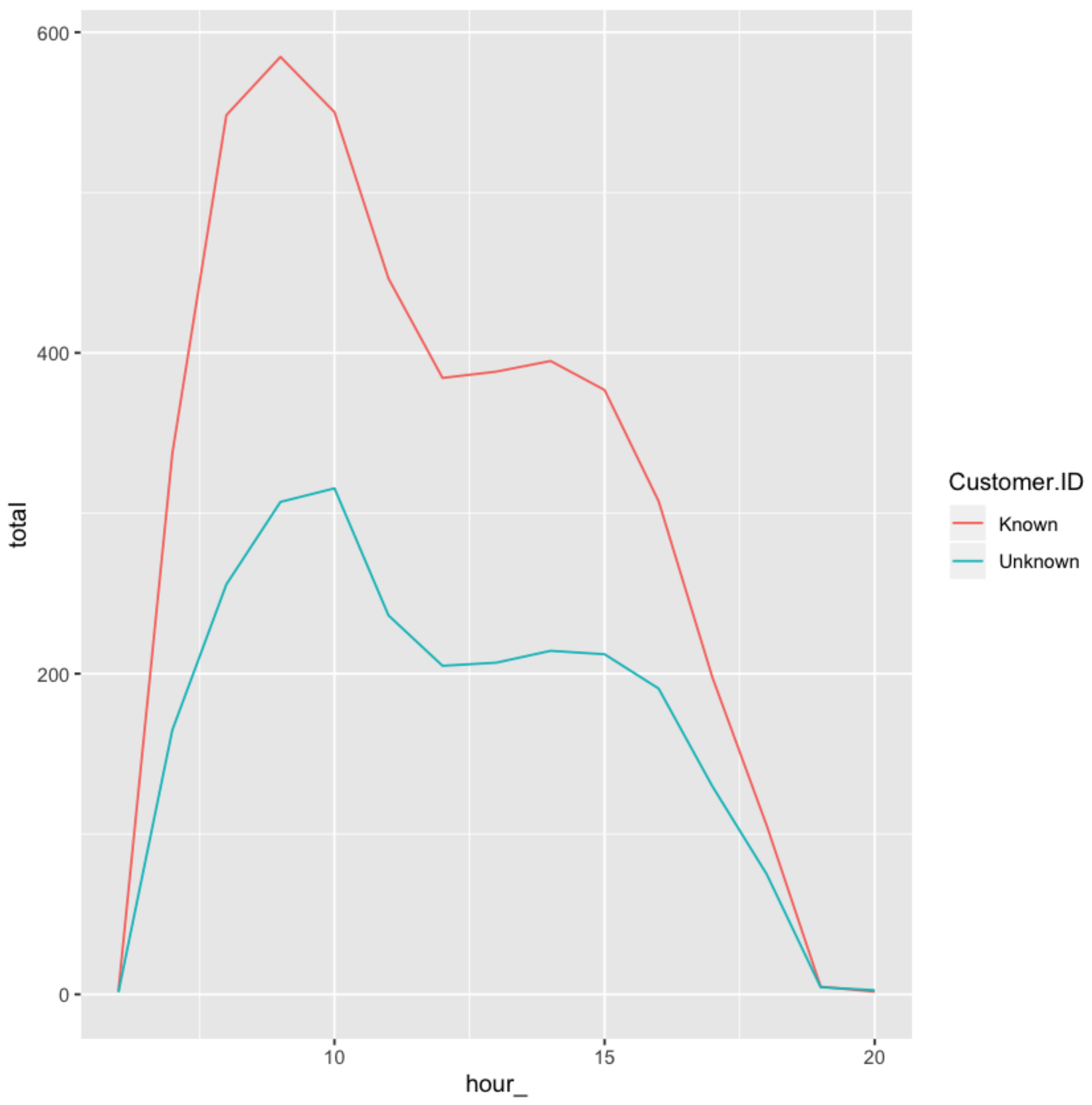


Repeating this plot with gross sales highlights the differences between the known and unknown are even greater than we initially expected.

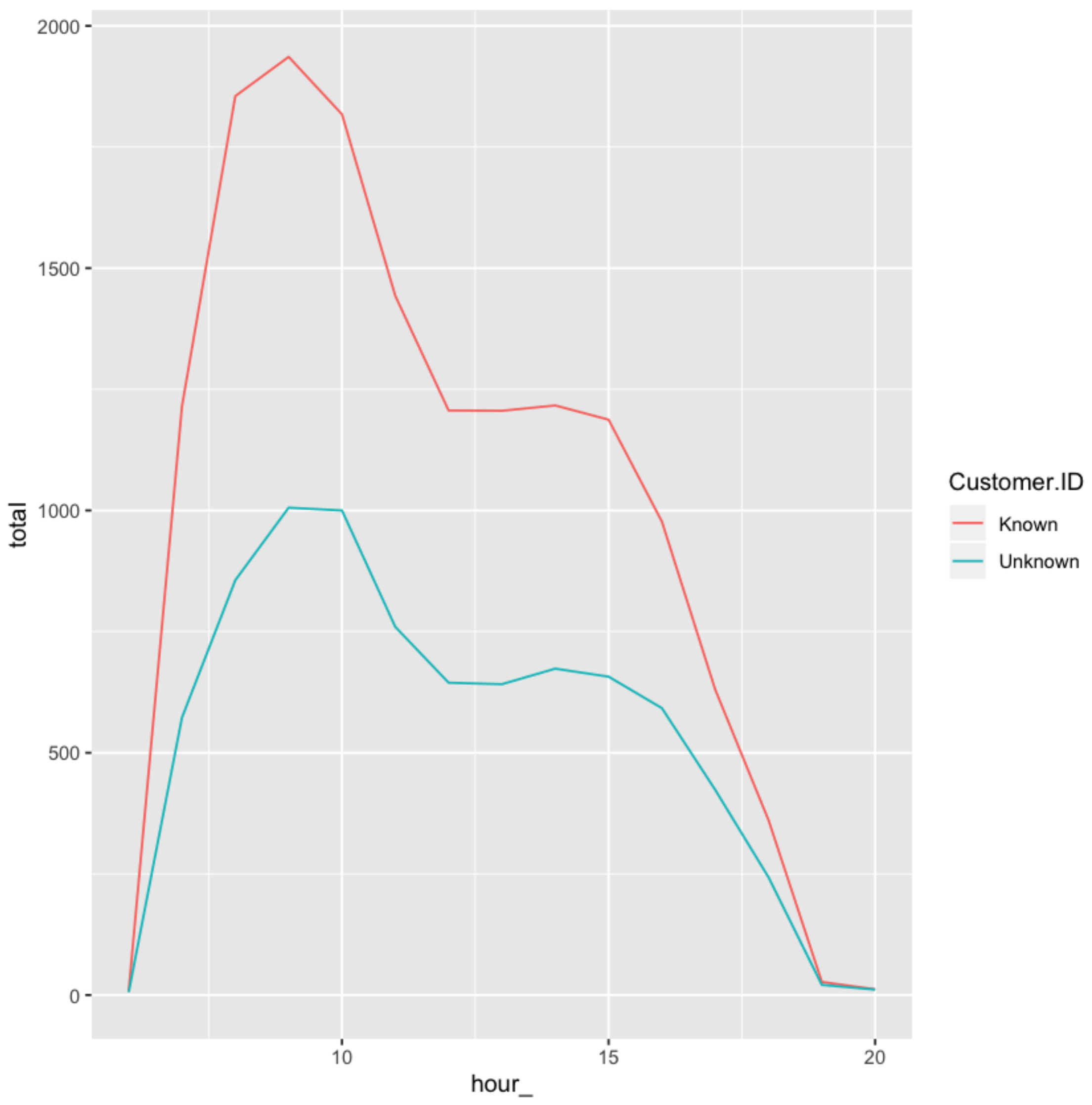
[50]:
```
reformat_data <- function(customers) {
  item1 <- customers %>% filter(id == 1) %>%
    select(-c("id")) %>% rename(Item1 = Item_Qty)
  item2 <- customers %>% filter(id == 2) %>%
    select(-c("id")) %>% rename(Item2 = Item_Qty)
  item3 <- customers %>% filter(id == 3) %>%
    select(-c("id")) %>% rename(Item3 = Item_Qty)
  item4 <- customers %>% filter(id == 4) %>%
    select(-c("id")) %>% rename(Item4 = Item_Qty)
  item5 <- customers %>% filter(id == 5) %>%
    select(-c("id")) %>% rename(Item5 = Item_Qty)
  item6 <- customers %>% filter(id == 6) %>%
    select(-c("id")) %>% rename(Item6 = Item_Qty)
  item7 <- customers %>% filter(id == 7) %>%
    select(-c("id")) %>% rename(Item7 = Item_Qty)
  item8 <- customers %>% filter(id == 8) %>%
    select(-c("id")) %>% rename(Item8 = Item_Qty)
  item9 <- customers %>% filter(id == 9) %>%
    select(-c("id")) %>% rename(Item9 = Item_Qty)
  item10 <- customers %>% filter(id == 10) %>%
    select(-c("id")) %>% rename(Item10 = Item_Qty)
  item11 <- customers %>% filter(id == 11) %>%
    select(-c("id")) %>% rename(Item11 = Item_Qty)
  item12 <- customers %>% filter(id == 12) %>%
    select(-c("id")) %>% rename(Item12 = Item_Qty)
  item13 <- customers %>% filter(id == 13) %>%
    select(-c("id")) %>% rename(Item13 = Item_Qty)
  item14 <- customers %>% filter(id == 14) %>%
    select(-c("id")) %>% rename(Item14 = Item_Qty)
  item15 <- customers %>% filter(id == 15) %>%
    select(-c("id")) %>% rename(Item15 = Item_Qty)
  item16 <- customers %>% filter(id == 16) %>%
    select(-c("id")) %>% rename(Item16 = Item_Qty)
  item17 <- customers %>% filter(id == 17) %>%
    select(-c("id")) %>% rename(Item17 = Item_Qty)

  concat_string <- c("Customer.ID", "month_", "year_", "hour_", "week_day",
                     "Total.Gross.Sales", "Time", "total", "total_qty", "Date")
  j1 <- left_join(item1, item2, by = concat_string)
  j2 <- left_join(j1, item3, by = concat_string)
  j3 <- left_join(j2, item4, by = concat_string)
  j4 <- left_join(j3, item5, by = concat_string)
  j5 <- left_join(j4, item6, by = concat_string)
  j6 <- left_join(j5, item7, by = concat_string)
  j7 <- left_join(j6, item8, by = concat_string)
  j8 <- left_join(j7, item9, by = concat_string)
  j9 <- left_join(j8, item10, by = concat_string)
  j10 <- left_join(j9, item11, by = concat_string)
  j11 <- left_join(j10, item12, by = concat_string)
  j12 <- left_join(j11, item13, by = concat_string)
  j13 <- left_join(j12, item14, by = concat_string)
  j14 <- left_join(j13, item15, by = concat_string)
  j15 <- left_join(j14, item16, by = concat_string)
```

```
  j16 <- left_join(j15, item17, by = concat_string)
  it <- j16 %>%
    select(Customer.ID, month_, year_, hour_, week_day, Total.Gross.Sales,
           Item1, Item2, Item3, Item4, Item5, Item6, Item7, Item8, Item9, Item10,
           Item11,Item12, Item13, Item14, Item15, Item16, Item17, total,␣
 ↪total_qty, Date)
  return(it)
  }
it <- reformat_data(customers)
```

[51]:
```
it %>% mutate(Customer.ID = case_when(Customer.ID == "Unknown" ~ "Unknown",
                                      Customer.ID != "Unknown" ~ "Known"),
             day_ = lubridate::day(Date)) %>%
  group_by(Customer.ID, hour_, day_) %>% summarise(total = sum(total)) %>%
  ungroup() %>% group_by(Customer.ID, hour_) %>% summarise(total = mean(total))␣
 ↪%>%
  ggplot(aes(x=hour_, y=total, color=Customer.ID)) + geom_line()
```



```
[53]: it %>% mutate(Customer.ID = case_when(Customer.ID == "Unknown" ~ "Unknown",
                                          Customer.ID != "Unknown" ~ "Known"),
                   day_ = lubridate::day(Date)) %>%
        group_by(Customer.ID, hour_, day_) %>%
        summarise(total = sum(Total.Gross.Sales)) %>%
        ungroup() %>%
        group_by(Customer.ID, hour_) %>%
        summarise(total = mean(total)) %>%
        ggplot(aes(x=hour_, y=total, color=Customer.ID)) + geom_line()
```



The known and unknown customers have similar habits by time of day, it might be possible to convert some of these unknown customers to known if loyalty promotions are added and executed effectively.

2.3 Looking at sales by day

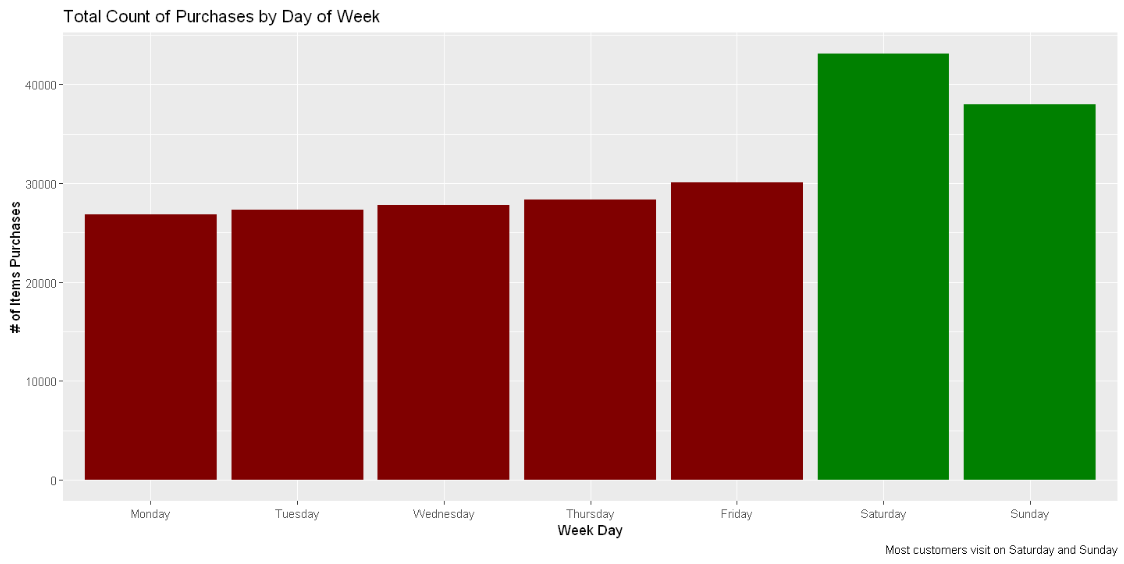
NYC, especially central park, is known to be a hub for tourists and locals alike. Do we see different trends or are there days where we see increased or decreased sales?

[21]:
```
## Finally, we group by count of purchases by day of week - do we see increased␣
↪traffic on different days?
```

```
purchases_grouped_by_day <-
    centralperk_df %>%
        group_by(DayofWeek) %>%
            summarise(purchases = n())
```

[22]:
```
## Create a levels factorization to help us sort our data labels more accurately
daylevels <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday",␣
 ↪"Saturday", "Sunday")

## Re-arrange sort order to generate a better visualization below
purchases_grouped_by_day$DayofWeek <- factor(purchases_grouped_by_day$DayofWeek,␣
 ↪levels = daylevels)
```

[23]:
```
### Create a visualization to see count of purchases by day of week

ggplot(purchases_grouped_by_day, aes ( x = DayofWeek, weight = purchases, fill =␣
 ↪DayofWeek)) +
    geom_bar() +
    scale_fill_manual("legend", values = c("Monday" = "#800000", "Tuesday" =␣
 ↪"#800000", "Wednesday" = "#800000",
                                          "Thursday" = "#800000", "Friday" =␣
 ↪"#800000", "Saturday" = "#008000",
                                          "Sunday" = "#008000")) +
    theme(legend.position = "none") +
    labs(x = "Week Day", title = "Total Count of Purchases by Day of Week", y =␣
 ↪"# of Items Purchases",
        caption = "Most customers visit on Saturday and Sunday")
```

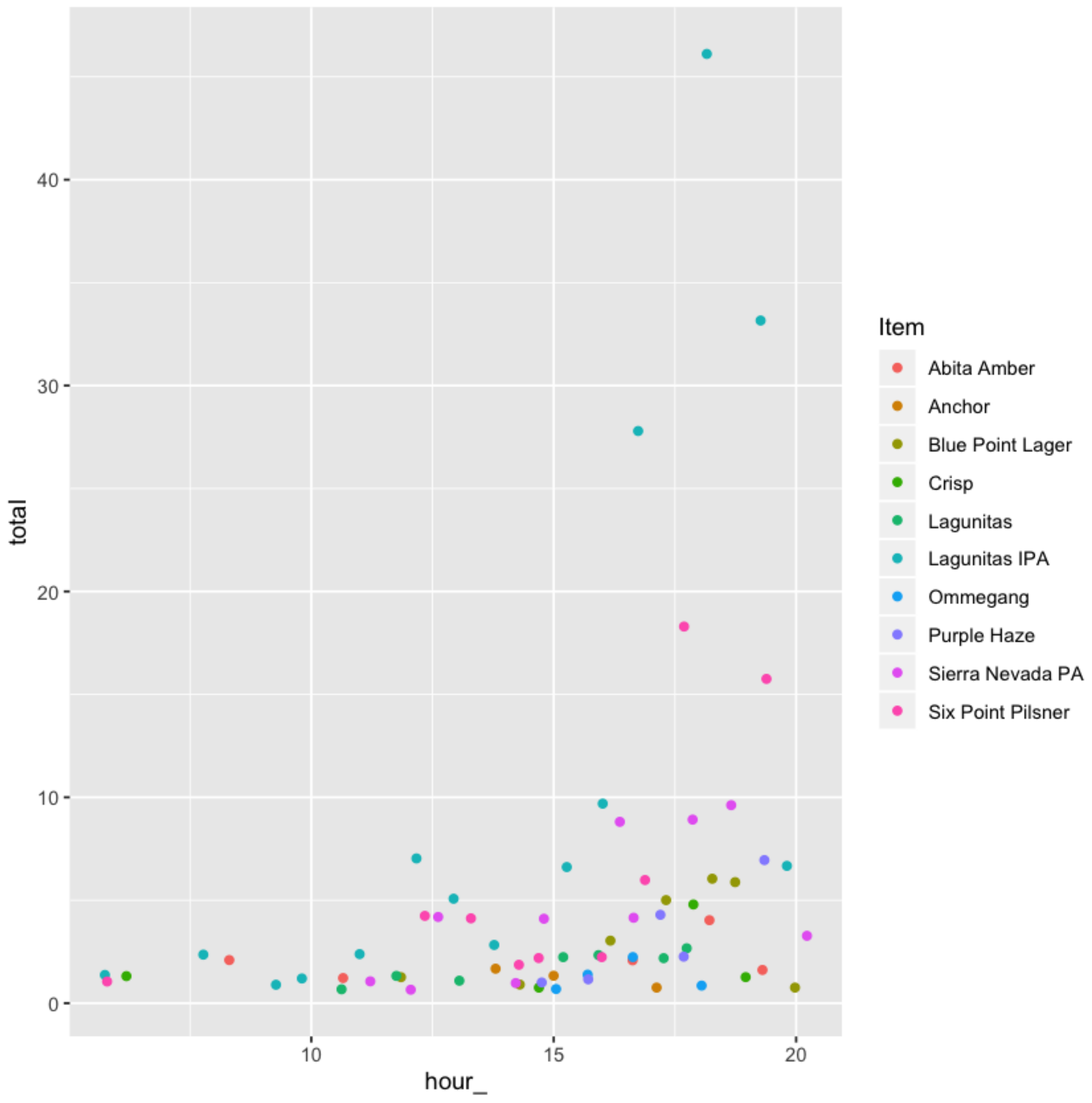


It appears that sales are higher on the weekends. If we want to smooth sales over the week, we will need to find a way to increase sales on weekdays.

2.4 Additional Analysis

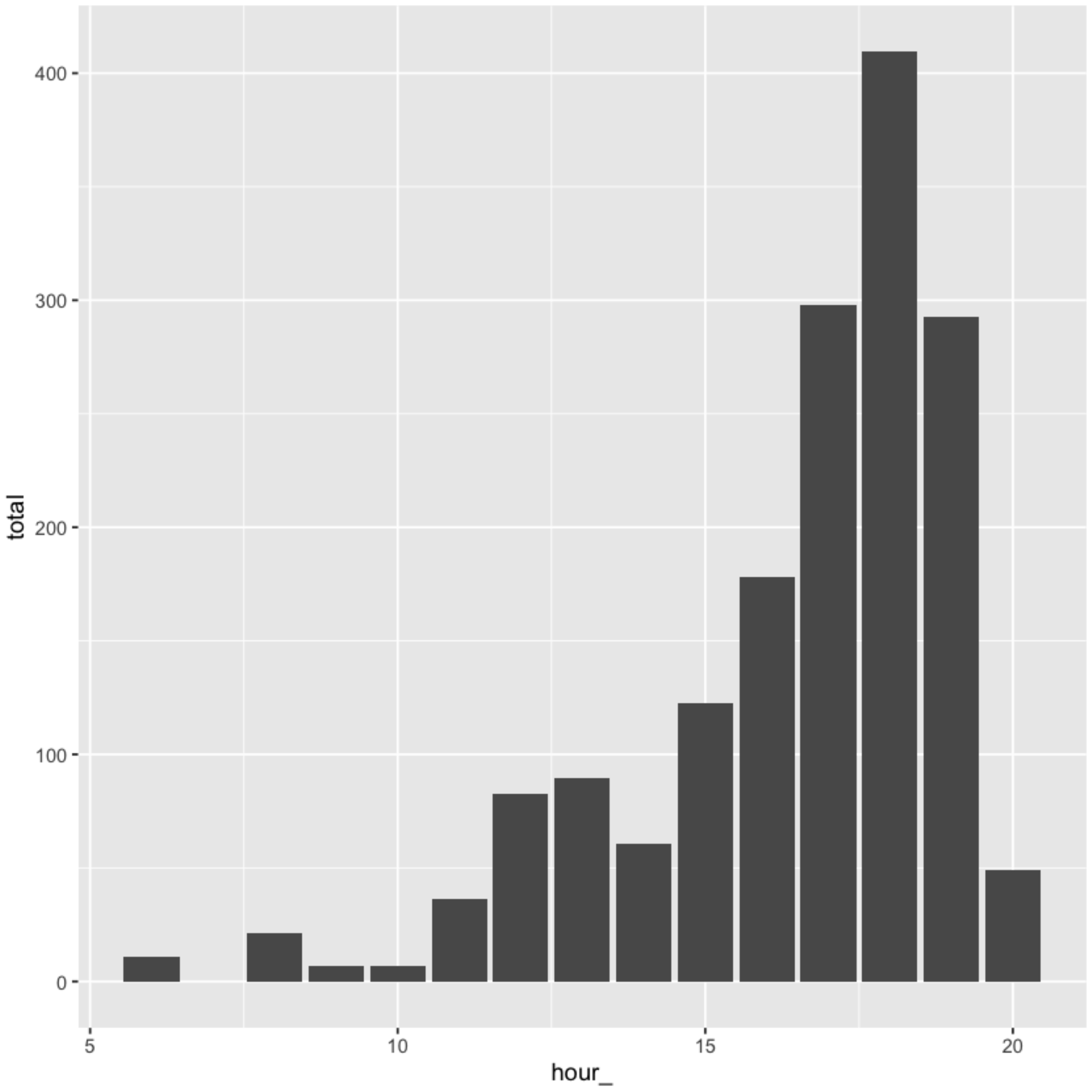
Beer makes up such a small percentage of sales, it is worth more exploration. Does it make sense to continue to offer beer? Maybe just some of the selection?

[54]:
```
dfx <- df %>% mutate(hour_ = lubridate::hour(Time))

# Beer Purchases
dfx %>% filter(Category == "Beers") %>%
  group_by(hour_, Item) %>% summarise(total = sum(Qty)) %>%
  ggplot(aes(x = hour_, y = total, color = Item)) + geom_jitter()
```



As individual products there are only a couple that appear popular, offering a wide selection is probably not worthwhile.

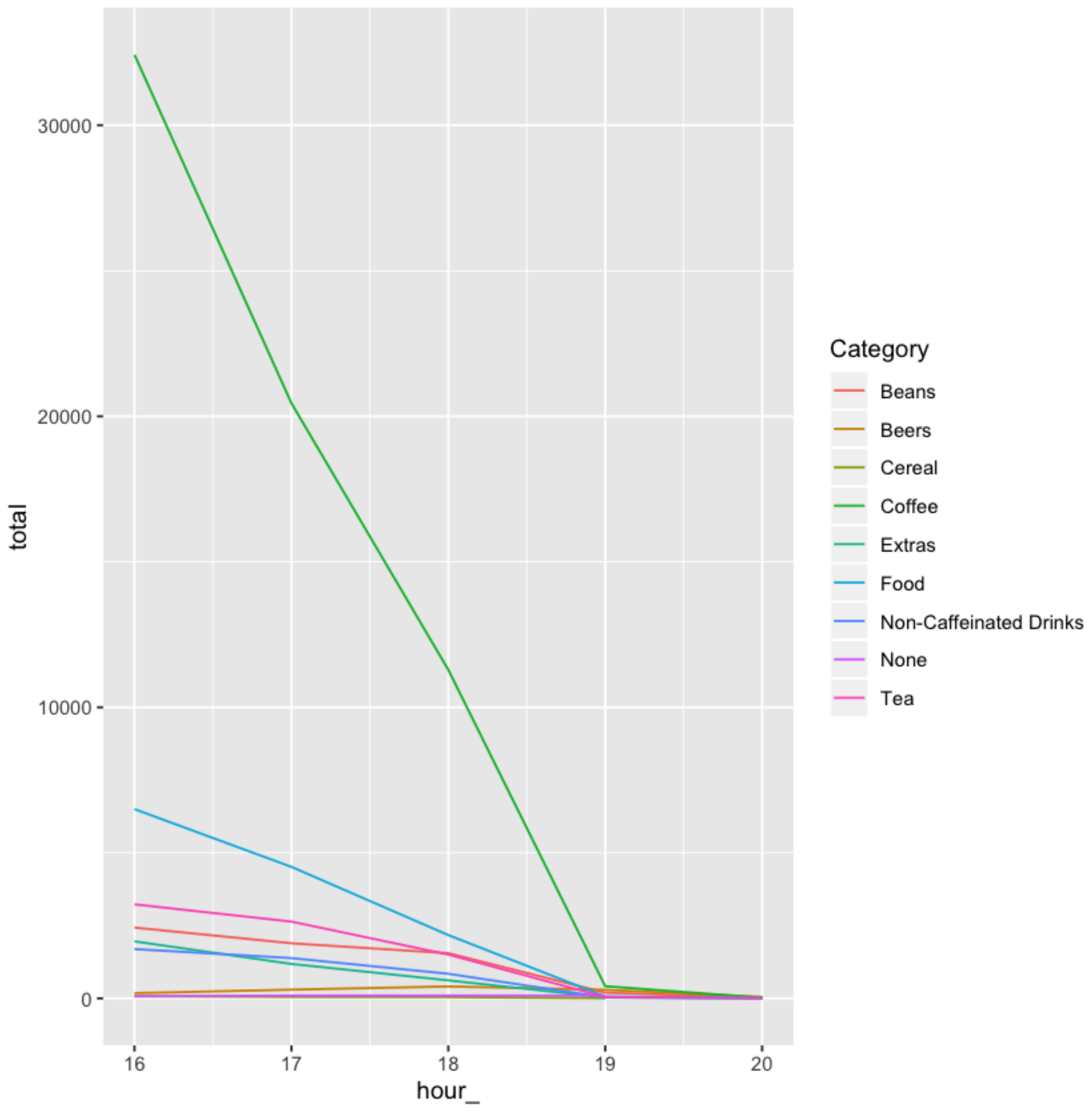
```
[56]: dfx %>% filter(Category == "Beers") %>%
        group_by(hour_) %>% summarise(total = sum(Net.Sales)) %>%
        ggplot(aes(x = hour_, y = total)) + geom_col()
```



Almost all beer sales occur in the late afternoon-evening time period.

```
[57]: dfx %>% filter(hour_ > 15) %>%
        group_by(hour_, Category) %>%
```

```r
  summarise(total = sum(Net.Sales)) %>%
  ggplot(aes(x = hour_, y = total , color = Category)) + geom_line()
```



The time trends makes sense, however overall sales are very low during the afternoon/evening hours. If beer is being looked at to drive customer sales in the afternoon, it doesn't seem particularly effective.

This might be an area worth divesting or reducing the offering. It won't directly increase sales or answer any of the questions that Central Perk has asked, but it is an area that could reduce cost and could be converted into a nitro-brew station that major coffee chains are starting to see a lot of success with:

https://www.wsj.com/articles/starbucks-posts-higher-profit-as-u-s-sales-strengthen-11572466893

3 Interpretation

Key insights from our exploration: - Most Central Perk customers are NOT regulars, however, regulars make up more of the sales - Sales are lower in the afternoon/evening - Sales are lower on weekdays - Sales of coffee with ice are high in the summer - Small drinks make up a majority of the sales, but there is still a demand for larger sizes

4 Conclusions

We looked into Central Perk's two assumptions that their customer base is loyal and that sales fluctuate over the day and week. We found support for both assumptions. Repeat customers are responsible for more sales than unknown customers, however, there aren't differences in when or what the groups are ordering, so we do think that a focus on increasing loyalty would prove profitable. Additionally, finding ways to increase afternoon visits should help smooth demand over the day. Smoothing demand by season may be possible with alterations to the product offerings.

If Central Perk follows our recommendations, sales across the day, week, and year will smooth to create more consistent operations for the company. In addition, there will be increased revenues from the methods that smooth demand and by adding a medium sized drink offering. This meets the goals set forth by the project and will not alienate any customers, allowing all who come to Central Perk to feel like "friends."

5 Recommendations

Central Perk should introduce a loyalty program that rewards repeat customers with incentives to have them continue to come back. In addition, this program could tip "maybe" repeat customers into becoming definite repeat customers.

Central Perk should implement a "happy hour" during the weekdays from 4pm to 6pm. The happy hour during the week will normalize sales throughout the day and increase overall weekday sales.

Central Perk should introduce iced/cold coffee options, we would recommend timing this rollout for the summer. Iced coffee drinks in the summer will give that season the boost it needs to come in line with the rest of the year. Further, information from the coffee industry shows that iced/cold drinks are able to be sold at a premium, which should drive revenue.

Central Perk should introduce a medium drink size. Sales for small sizes are by far the highest, but there are enough purchases of large drinks that we know the demand for bigger drinks is there. By introducing a medium, Central Perk can grow revenues from customers who would purchase a medium at a higher price, but not marginalize customers who still want to purchase a large

Central Perk should stop selling beer. Total sales over the 3-year period is ~$2000, the cost to stock this, maintain a liquor license, and train employees must exceed the profits on beer sales. It would be better to focus on other ways to drive afternoon business and to increase summer coffee sales by focusing on the iced coffee trend that has been successful domestically.

Code Appendix A

```
[ ]: customers <- df %>%
       mutate(Customer.ID = case_when(is.na(Customer.ID) == TRUE ~ "Unknown",
                                      is.na(Customer.ID) != TRUE ~ Customer.ID)) %>%
       select(Date, Time, Item_Qty, Customer.ID, Gross.Sales, month_, year_, Qty) %>%
       group_by(Date, Time, month_, year_) %>%
       mutate(id = row_number(), total = n(), Total.Gross.Sales = sum(Gross.Sales),
              week_day = lubridate::wday(Date, label = TRUE, abbr = FALSE),
              total_qty = sum(Qty)) %>%
       ungroup() %>% select(-c("Gross.Sales", "Qty")) %>%
       mutate(hour_ = lubridate::hour(Time)) %>% group_by()

     reformat_data <- function(customers) {
       item1 <- customers %>% filter(id == 1) %>%
         select(-c("id")) %>% rename(Item1 = Item_Qty)
       item2 <- customers %>% filter(id == 2) %>%
         select(-c("id")) %>% rename(Item2 = Item_Qty)
       item3 <- customers %>% filter(id == 3) %>%
         select(-c("id")) %>% rename(Item3 = Item_Qty)
       item4 <- customers %>% filter(id == 4) %>%
         select(-c("id")) %>% rename(Item4 = Item_Qty)
       item5 <- customers %>% filter(id == 5) %>%
         select(-c("id")) %>% rename(Item5 = Item_Qty)
       item6 <- customers %>% filter(id == 6) %>%
         select(-c("id")) %>% rename(Item6 = Item_Qty)
       item7 <- customers %>% filter(id == 7) %>%
         select(-c("id")) %>% rename(Item7 = Item_Qty)
       item8 <- customers %>% filter(id == 8) %>%
         select(-c("id")) %>% rename(Item8 = Item_Qty)
       item9 <- customers %>% filter(id == 9) %>%
         select(-c("id")) %>% rename(Item9 = Item_Qty)
       item10 <- customers %>% filter(id == 10) %>%
         select(-c("id")) %>% rename(Item10 = Item_Qty)
       item11 <- customers %>% filter(id == 11) %>%
         select(-c("id")) %>% rename(Item11 = Item_Qty)
       item12 <- customers %>% filter(id == 12) %>%
         select(-c("id")) %>% rename(Item12 = Item_Qty)
       item13 <- customers %>% filter(id == 13) %>%
         select(-c("id")) %>% rename(Item13 = Item_Qty)
       item14 <- customers %>% filter(id == 14) %>%
         select(-c("id")) %>% rename(Item14 = Item_Qty)
       item15 <- customers %>% filter(id == 15) %>%
         select(-c("id")) %>% rename(Item15 = Item_Qty)
       item16 <- customers %>% filter(id == 16) %>%
         select(-c("id")) %>% rename(Item16 = Item_Qty)
       item17 <- customers %>% filter(id == 17) %>%
         select(-c("id")) %>% rename(Item17 = Item_Qty)

       concat_string <- c("Customer.ID", "month_", "year_", "hour_", "week_day",
                          "Total.Gross.Sales", "Time", "total", "total_qty", "Date")
       j1 <- left_join(item1, item2, by = concat_string)
       j2 <- left_join(j1, item3, by = concat_string)
       j3 <- left_join(j2, item4, by = concat_string)
       j4 <- left_join(j3, item5, by = concat_string)
       j5 <- left_join(j4, item6, by = concat_string)
       j6 <- left_join(j5, item7, by = concat_string)
       j7 <- left_join(j6, item8, by = concat_string)
       j8 <- left_join(j7, item9, by = concat_string)
       j9 <- left_join(j8, item10, by = concat_string)
       j10 <- left_join(j9, item11, by = concat_string)
       j11 <- left_join(j10, item12, by = concat_string)
       j12 <- left_join(j11, item13, by = concat_string)
       j13 <- left_join(j12, item14, by = concat_string)
       j14 <- left_join(j13, item15, by = concat_string)
       j15 <- left_join(j14, item16, by = concat_string)
       j16 <- left_join(j15, item17, by = concat_string)
       it <- j16 %>%
         select(Customer.ID, month_, year_, hour_, week_day, Total.Gross.Sales,
                Item1, Item2, Item3, Item4, Item5, Item6, Item7, Item8, Item9, Item10,
                Item11,Item12, Item13, Item14, Item15, Item16, Item17, total,␣
      ↪total_qty, Date)
       return(it)
       }
     it <- reformat_data(customers)

     jx <- as_tibble(it) %>% unite("Item_Sequence", Item1:Item17, sep = ",", na.rm =␣
      ↪TRUE)
     jx$Item_Sequence <- sapply(lapply(strsplit(jx$Item_Sequence, ","),␣
      ↪sort),paste,collapse=",")
```