Instacart Market Basket Analysis EDA

Xi Liang

6/26/2017

Contents


```r
library(data.table)
library(dplyr)
```

```
## -------------------------------------------------------------------------

## data.table + dplyr code now lives in dtplyr.
## Please library(dtplyr)!

## -------------------------------------------------------------------------

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tm)
```

```
## Loading required package: NLP
```

```r
library(SnowballC)
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```r
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':
##
##     annotate
```

```r
library(treemap)
```

Loading files

```r
aisles <- fread("data/aisles.csv", stringsAsFactors = T)
dept <- fread("data/departments.csv", stringsAsFactors = T)
orders <- fread("data/orders.csv", stringsAsFactors = T)
products <- fread("data/products.csv", stringsAsFactors = T)
order_products_prior <- fread("data/order_products__prior.csv", stringsAsFactors = T)
```

```
##
Read 23.8% of 32434489 rows
Read 54.6% of 32434489 rows
Read 83.7% of 32434489 rows
Read 32434489 rows and 4 (of 4) columns from 0.538 GB file in 00:00:05
```

```r
order_products_train <- fread("data/order_products__train.csv", stringsAsFactors = T)
```

This dataset contains total six files, we will take a look at them one by one.

Departments

```r
dept$department
```

```
##  [1] frozen          other           bakery          produce
##  [5] alcohol         international   beverages       pets
##  [9] dry goods pasta bulk            personal care   meat seafood
## [13] pantry          breakfast       canned goods    dairy eggs
## [17] household       babies          snacks          deli
## [21] missing
## 21 Levels: alcohol babies bakery beverages breakfast bulk ... snacks
```

"Dept" data contains the names for 20 different departments, mostly are department names we see in our day to day life in a grocery store. This file also contain a category, "missing", to describe items that are not associated to any departments listed above.

Aisles

```
aisles$aisle
```

```
##   [1] prepared soups salads         specialty cheeses
##   [3] energy granola bars           instant foods
##   [5] marinades meat preparation    other
##   [7] packaged meat                 bakery desserts
##   [9] pasta sauce                   kitchen supplies
##  [11] cold flu allergy              fresh pasta
##  [13] prepared meals                tofu meat alternatives
##  [15] packaged seafood              fresh herbs
##  [17] baking ingredients            bulk dried fruits vegetables
##  [19] oils vinegars                 oral hygiene
##  [21] packaged cheese               hair care
##  [23] popcorn jerky                 fresh fruits
##  [25] soap                          coffee
##  [27] beers coolers                 red wines
##  [29] honeys syrups nectars         latino foods
##  [31] refrigerated                  packaged produce
##  [33] kosher foods                  frozen meat seafood
##  [35] poultry counter               butter
##  [37] ice cream ice                 frozen meals
##  [39] seafood counter               dog food care
##  [41] cat food care                 frozen vegan vegetarian
##  [43] buns rolls                    eye ear care
##  [45] candy chocolate               mint gum
##  [47] vitamins supplements          breakfast bars pastries
##  [49] packaged poultry              fruit vegetable snacks
##  [51] preserved dips spreads        frozen breakfast
##  [53] cream                         paper goods
##  [55] shave needs                   diapers wipes
##  [57] granola                       frozen breads doughs
##  [59] canned meals beans            trash bags liners
##  [61] cookies cakes                 white wines
##  [63] grains rice dried goods       energy sports drinks
##  [65] protein meal replacements     asian foods
##  [67] fresh dips tapenades          bulk grains rice dried goods
##  [69] soup broth bouillon           digestion
##  [71] refrigerated pudding desserts condiments
##  [73] facial care                   dish detergents
##  [75] laundry                       indian foods
##  [77] soft drinks                   crackers
##  [79] frozen pizza                  deodorants
##  [81] canned jarred vegetables      baby accessories
##  [83] fresh vegetables              milk
##  [85] food storage                  eggs
##  [87] more household                spreads
```

```
##  [89] salad dressing toppings       cocoa drink mixes
##  [91] soy lactosefree               baby food formula
##  [93] breakfast bakery              tea
##  [95] canned meat seafood           lunch meat
##  [97] baking supplies decor         juice nectars
##  [99] canned fruit applesauce       missing
## [101] air fresheners candles        baby bath body care
## [103] ice cream toppings            spices seasonings
## [105] doughs gelatins bake mixes    hot dogs bacon sausage
## [107] chips pretzels                other creams cheeses
## [109] skin care                     pickled goods olives
## [111] plates bowls cups flatware    bread
## [113] frozen juice                  cleaning products
## [115] water seltzer sparkling water frozen produce
## [117] nuts seeds dried fruit        first aid
## [119] frozen dessert                yogurt
## [121] cereal                        meat counter
## [123] packaged vegetables fruits    spirits
## [125] trail mix snack mix           feminine care
## [127] body lotions soap             tortillas flat bread
## [129] frozen appetizers sides       hot cereal pancake mixes
## [131] dry pasta                     beauty
## [133] muscles joints pain relief    specialty wines champagnes
## 134 Levels: air fresheners candles asian foods ... yogurt
```

This data contains 134 observations that describe the locations of a specific product in the market.

Products

```
products %>% glimpse()
```

```
## Observations: 49,688
## Variables: 4
## $ product_id    <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1...
## $ product_name  <fctr> Chocolate Sandwich Cookies, All-Seasons Salt, R...
## $ aisle_id      <int> 61, 104, 94, 38, 5, 11, 98, 116, 120, 115, 31, 1...
## $ department_id <int> 19, 13, 7, 1, 13, 11, 7, 1, 16, 7, 7, 1, 11, 17,...
```

"Products" contains 49688 unqiue observations (products) and each of the product is linked with specific aisle ID and department ID. Since we already have the files that contian department and aisle information, we could consolidate those information into this file to gain a better understanding the inventory of our grocery.

```
products_w_desc <- products

products_w_desc[, aisles_description := aisles$aisle[products$aisle_id]]
products_w_desc[, depart_description := dept$department[products$department_id]]
```

Here is what our data frame looks like after integratin the aisle and department information:

```
products_w_desc
```

```
##        product_id
##     1:          1
##     2:          2
##     3:          3
```

```
##     4:          4
##     5:          5
##    ---
## 49684:      49684
## 49685:      49685
## 49686:      49686
## 49687:      49687
## 49688:      49688
##                                                               product_name
##     1:                                            Chocolate Sandwich Cookies
##     2:                                                      All-Seasons Salt
##     3:                                  Robust Golden Unsweetened Oolong Tea
##     4: Smart Ones Classic Favorites Mini Rigatoni With Vodka Cream Sauce
##     5:                                             Green Chile Anytime Sauce
##    ---
## 49684:                             Vodka, Triple Distilled, Twist of Vanilla
## 49685:                                    En Croute Roast Hazelnut Cranberry
## 49686:                                                      Artisan Baguette
## 49687:                            Smartblend Healthy Metabolism Dry Cat Food
## 49688:                                                Fresh Foaming Cleanser
##        aisle_id department_id         aisles_description
##     1:       61            19               cookies cakes
##     2:      104            13           spices seasonings
##     3:       94             7                         tea
##     4:       38             1                frozen meals
##     5:        5            13 marinades meat preparation
##    ---
## 49684:      124             5                     spirits
## 49685:       42             1    frozen vegan vegetarian
## 49686:      112             3                       bread
## 49687:       41             8               cat food care
## 49688:       73            11                 facial care
##        depart_description
##     1:             snacks
##     2:             pantry
##     3:          beverages
##     4:             frozen
##     5:             pantry
##    ---
## 49684:            alcohol
## 49685:             frozen
## 49686:             bakery
## 49687:               pets
## 49688:      personal care
```

Products with missing information

Let's first take a look how many products in our data that don't have aisle and department information, and we would like to know what those products are. This will be a bit similar to text mining, so let's handle it with wordcloud.

```
products_missing_info <- products_w_desc[aisles_description == "missing" | depart_description == "missi
products_missing_info %>% dim
```

```
## [1] 1258    6
```

```r
products_corpus <- VCorpus(VectorSource(products_missing_info$product_name))
```

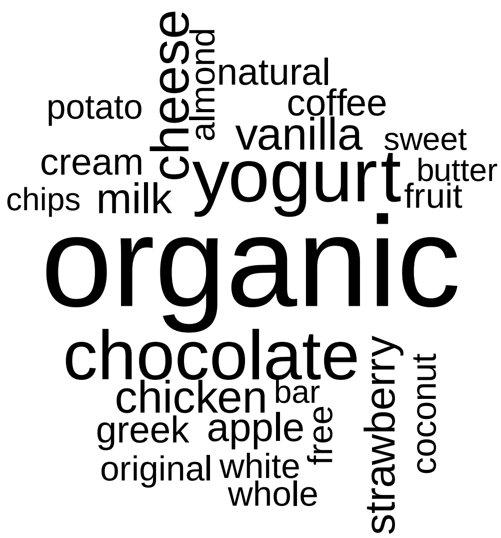
```r
#cleaning text info
products_corpus_clean <- tm_map(products_corpus,
                                content_transformer(tolower))

#check if it worked
products_corpus_clean[[1]] %>% as.character
```

```
## [1] "ultra antibacterial dish liquid"
```

```r
#remove numbers from corpus
products_corpus_clean <- tm_map(products_corpus_clean, removeNumbers)

#remove stop words (assuming if ther are any)
products_corpus_clean <- tm_map(products_corpus_clean, removeWords, stopwords())

#remove punctuation
products_corpus_clean <- tm_map(products_corpus_clean, removePunctuation)

#remove white spaces
products_corpus_clean <- tm_map(products_corpus_clean, stripWhitespace)
```

We will visualize the most frequent appeared words (at least appeared 25 times in the data) with wordcloud.

```r
wordcloud(products_corpus_clean, min.freq = 25, random.order = F)
```

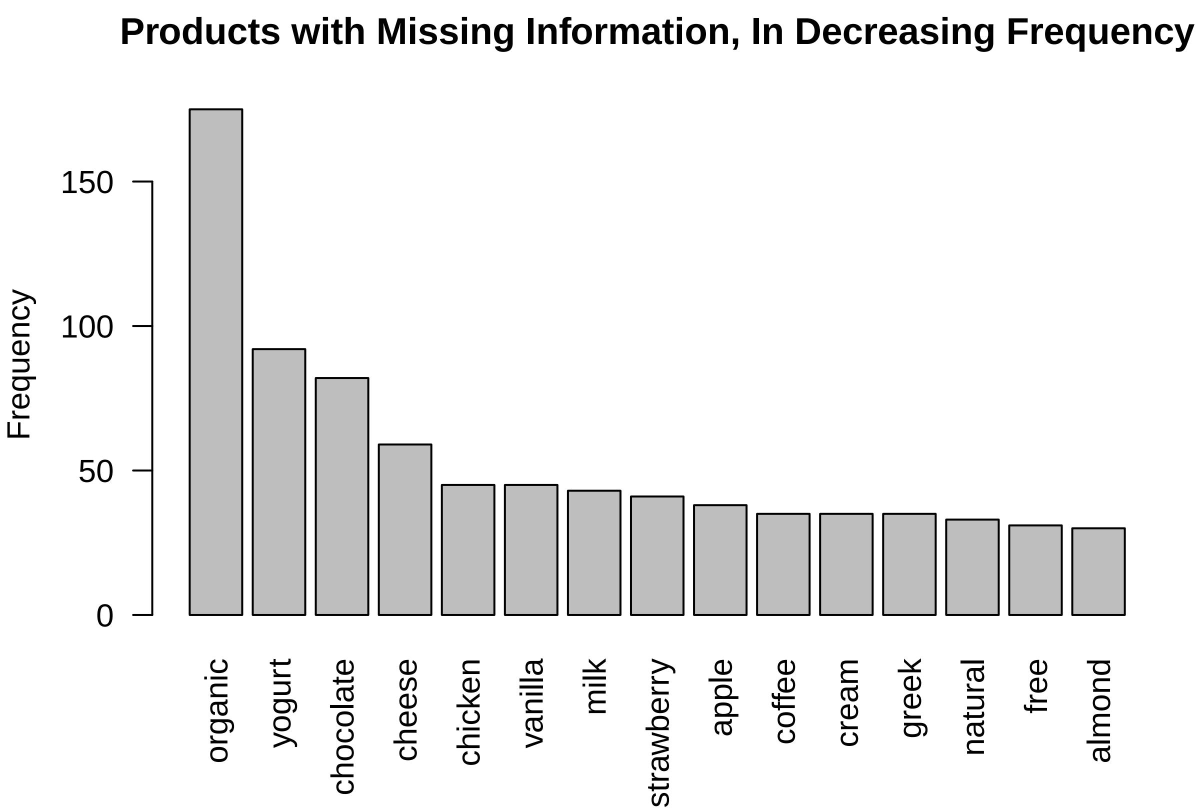



```r
products_corpus_clean_dtm <- TermDocumentMatrix(products_corpus_clean)
mat <- as.matrix(products_corpus_clean_dtm)
v <- sort(rowSums(mat), decreasing = TRUE)
d <- data.frame(word = names(v), freq = v)
head(d,20)
```

```
##                  word freq
## organic       organic  175
## yogurt         yogurt   92
## chocolate   chocolate   82
## cheese         cheese   59
```

```
## chicken       chicken   45
## vanilla       vanilla   45
## milk             milk   43
## strawberry strawberry   41
## apple           apple   38
## coffee         coffee   35
## cream           cream   35
## greek           greek   35
## natural       natural   33
## free             free   31
## almond         almond   30
## coconut       coconut   30
## original     original   30
## potato         potato   30
## fruit           fruit   29
## white           white   28
```

```r
barplot(d[1:15,]$freq, las = 2 ,names.arg = d[1:15,]$word,
        main = "Products with Missing Information, In Decreasing Frequency",
        ylab = "Frequency")
```



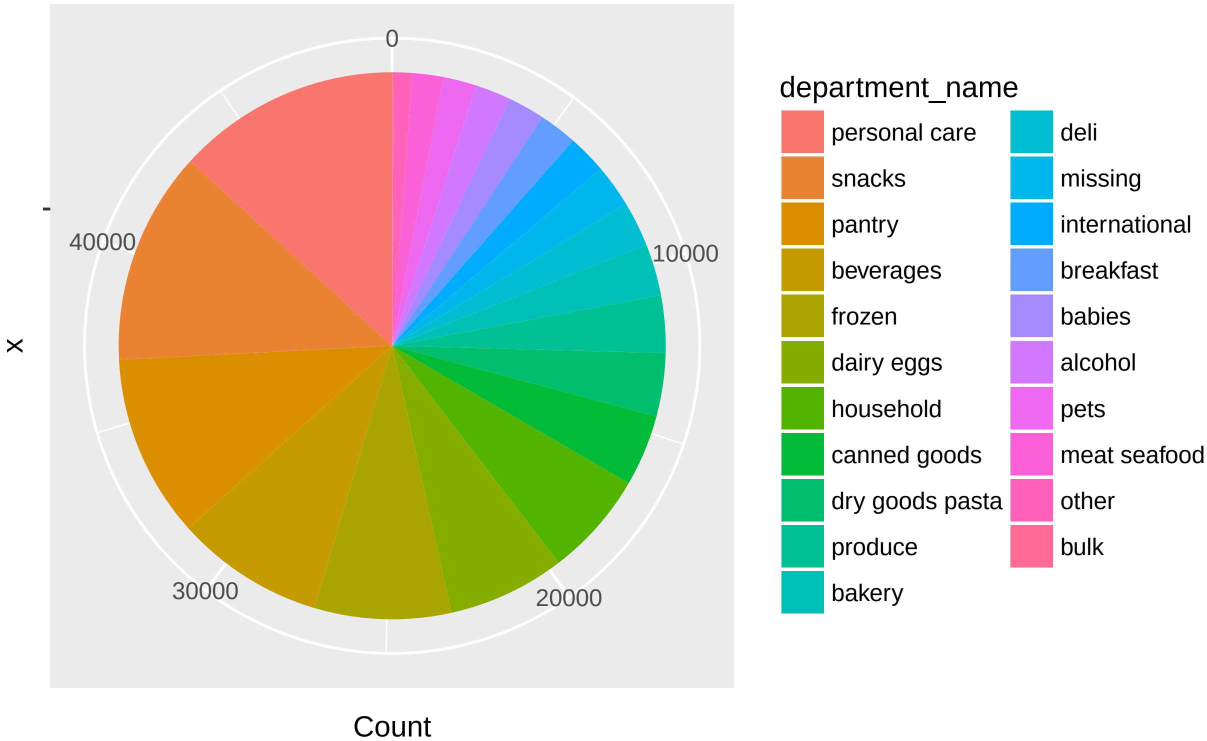
After analyzing what are the products that have highest rate of missing information, we will analyze the rest of the products in the grocery that have both of the aisle and department information. Through this analysis, we will know which departments contain the most products and what are those products.

```r
count_by_depart <- table(products_w_desc$depart_description) %>%
                   sort(decreasing = T) %>%
                   as.data.frame()

colnames(count_by_depart) <- c("department_name","Count")
```

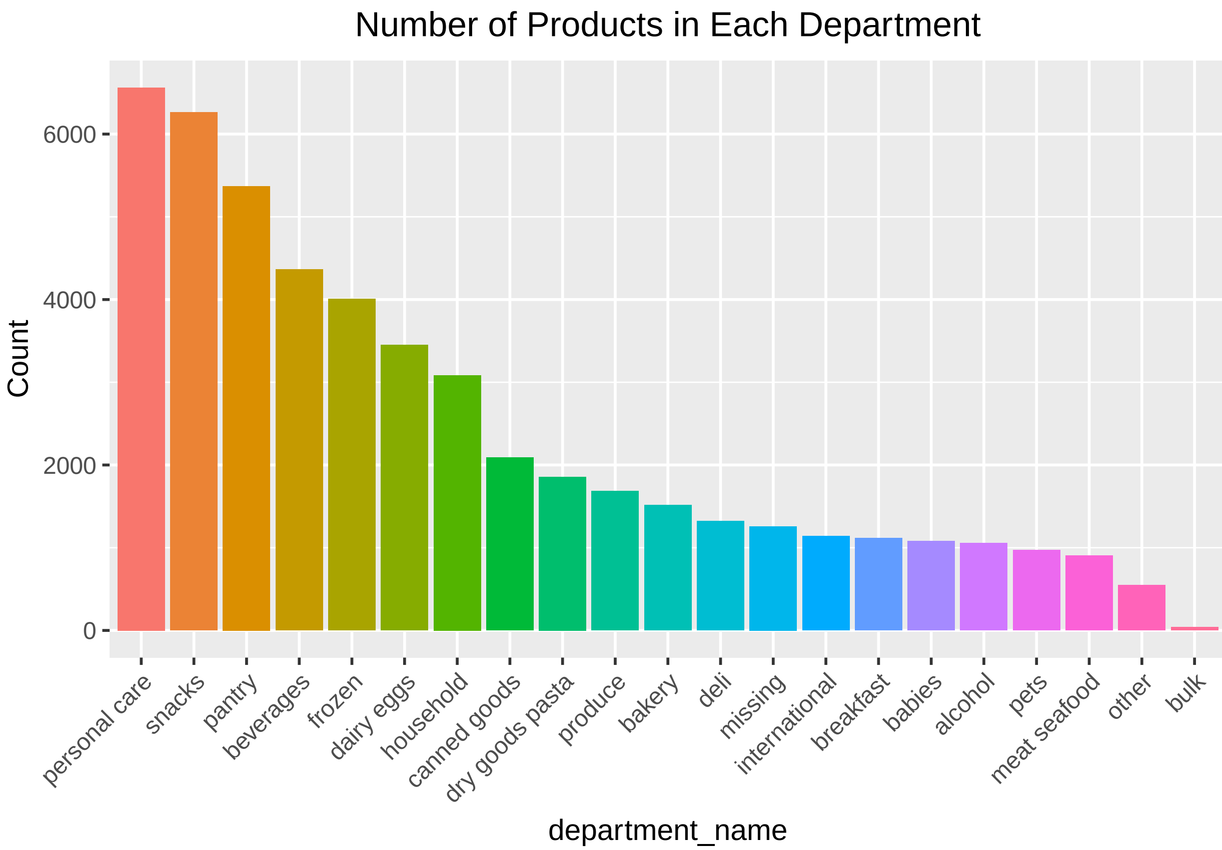
```r
count_by_depart$pct <- prop.table(count_by_depart$Count) %>%
                       round(3) * 100

count_by_depart
```

```
##    department_name Count  pct
## 1    personal care  6563 13.2
## 2           snacks  6264 12.6
## 3           pantry  5371 10.8
## 4        beverages  4365  8.8
## 5           frozen  4007  8.1
## 6       dairy eggs  3449  6.9
## 7        household  3085  6.2
## 8     canned goods  2092  4.2
## 9  dry goods pasta  1858  3.7
## 10         produce  1684  3.4
## 11          bakery  1516  3.1
## 12            deli  1322  2.7
## 13         missing  1258  2.5
## 14   international  1139  2.3
## 15       breakfast  1115  2.2
## 16          babies  1081  2.2
## 17         alcohol  1054  2.1
## 18            pets   972  2.0
## 19    meat seafood   907  1.8
## 20           other   548  1.1
## 21            bulk    38  0.1
```

```r
par(mfrow = c(1,2))
ggplot(count_by_depart, aes(x = "", y = Count, fill = department_name)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y",start = 0)
```




```r
ggplot(count_by_depart, aes(department_name, Count)) +
  geom_bar(aes(fill = department_name), stat = "identity") +
  labs(title = "Number of Products in Each Department") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(hjust = 0.5),
        legend.position = "none")
```



From the above result, we see department that has the most product is the personal care department (>6000 products), followlling by snacks. We will make a wild assumption by assuming that the number of product that a department carries should be correlated to the sales. For now, we will move on to the next data.

Orders

This data contains detail purchase history of customers. There are approximately 3.4 million transaction history and 206,209 customers Variables include "order_id", "user_id", "eval_set", "order_number", "order_dow", "order_hour_of_day", and "days_since_prior_order".

```
orders %>% dim
```

```
## [1] 3421083       7
```

```
unique(orders$user_id) %>% length
```

```
## [1] 206209
```

```
str(orders)
```

```
## Classes 'data.table' and 'data.frame':   3421083 obs. of  7 variables:
##  $ order_id              : int  2539329 2398795 473747 2254736 431534 3367565 550135 3108588 2295261
##  $ user_id               : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ eval_set              : Factor w/ 3 levels "prior","test",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ order_number          : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ order_dow             : int  2 3 3 4 4 2 1 1 1 4 ...
##  $ order_hour_of_day     : int  8 7 12 7 15 7 9 14 16 8 ...
##  $ days_since_prior_order: num  NA 15 21 29 28 19 20 14 0 30 ...
```

```
## - attr(*, ".internal.selfref")=<externalptr>
```

Before doing anything, we want to set a few variables in the data as factors.

```
orders$order_id <- as.factor(orders$order_id)
orders$user_id <- as.factor(orders$user_id)
orders$order_hour_of_day <- as.factor(orders$order_hour_of_day)
```

Among all the variables, we see a few interesting things. From the summary, we know that all the order IDs are unique; the maximum transaction history that one has is 100; "days_since_prior_order" has almost 200K records are NAs.

```
summary(orders)
```

```
##     order_id          user_id          eval_set        order_number
##  1      :      1   210    :    100   prior:3214874   Min.   :  1.00
##  2      :      1   310    :    100   test :  75000   1st Qu.:  5.00
##  3      :      1   313    :    100   train: 131209   Median : 11.00
##  4      :      1   690    :    100                   Mean   : 17.15
##  5      :      1   786    :    100                   3rd Qu.: 23.00
##  6      :      1   964    :    100                   Max.   :100.00
##  (Other):3421077   (Other):3420483
##    order_dow      order_hour_of_day days_since_prior_order
##  Min.   :0.000   10     : 288418   Min.   : 0.00
##  1st Qu.:1.000   11     : 284728   1st Qu.: 4.00
##  Median :3.000   15     : 283639   Median : 7.00
##  Mean   :2.776   14     : 283042   Mean   :11.11
##  3rd Qu.:5.000   13     : 277999   3rd Qu.:15.00
##  Max.   :6.000   12     : 272841   Max.   :30.00
##                  (Other):1730416   NA's   :206209
```

NA values

Let's find out what those missing values have in common.

```
missing_index <- which(is.na(orders$days_since_prior_order))
missing_df <- orders[missing_index,] %>% tbl_df
missing_df %>% summary
```

```
##     order_id         user_id          eval_set       order_number
##  20     :     1   1      :     1   prior:206209   Min.   :1
##  35     :     1   2      :     1   test :     0   1st Qu.:1
##  37     :     1   3      :     1   train:     0   Median :1
##  57     :     1   4      :     1                  Mean   :1
##  75     :     1   5      :     1                  3rd Qu.:1
##  100    :     1   6      :     1                  Max.   :1
##  (Other):206203   (Other):206203
##    order_dow      order_hour_of_day days_since_prior_order
##  Min.   :0.000   15     : 17264    Min.   : NA
##  1st Qu.:1.000   16     : 17094    1st Qu.: NA
##  Median :3.000   14     : 17035    Median : NA
##  Mean   :2.754   12     : 16995    Mean   :NaN
##  3rd Qu.:5.000   11     : 16916    3rd Qu.: NA
##  Max.   :6.000   13     : 16912    Max.   : NA
##                  (Other):103993    NA's   :206209
```

As we see from the summary above, all these missing values come from order number is one. Which make sense, as there will not be transaction history of a new customer. Next, let us see the transaction quantity and its distribution.

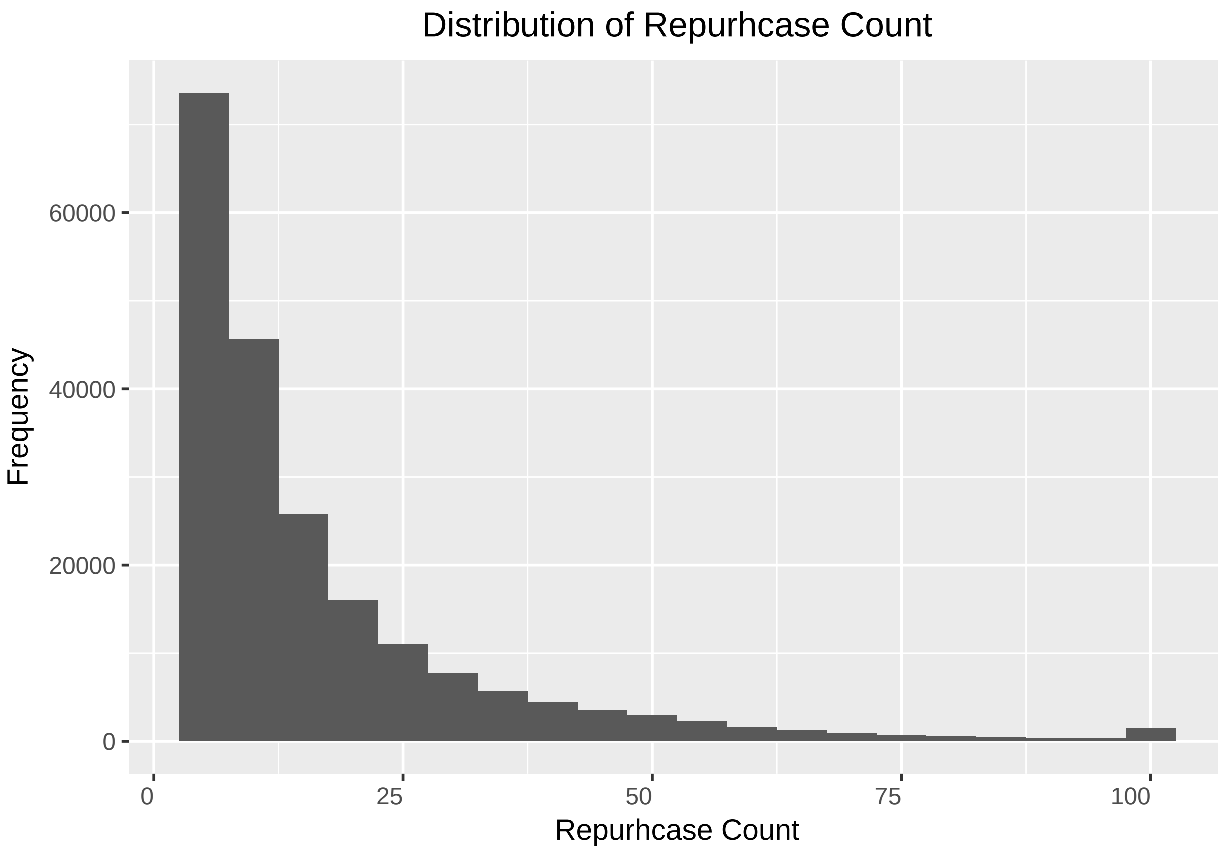
Number of transaction made

```
transaction_count <- orders[, .(transaction_count = .N), by = user_id]
transaction_count %>% summary
```

```
##     user_id        transaction_count
##  1      :     1   Min.   :  4.00
##  2      :     1   1st Qu.:  6.00
##  3      :     1   Median : 10.00
##  4      :     1   Mean   : 16.59
##  5      :     1   3rd Qu.: 20.00
##  6      :     1   Max.   :100.00
##  (Other):206203
```

We see that the minimum transaction made by specific customers in this data is 4, and the maximum is 100, with mean equals to 16.59 and median equals to 10. Following is the distribution of the transaction count frequency:

```
ggplot(transaction_count, aes(transaction_count)) +
  geom_histogram(binwidth = 5) +
  labs(title = "Distribution of Repurhcase Count") +
  xlab("Repurhcase Count") +
  ylab("Frequency") +
  theme(axis.text.x = element_text(hjust = 1),
        plot.title = element_text(hjust = 0.5))
```



A large portion of customer made less than 25 transactiosn, and that number continue to descrease as the transaction count increases. The population of customers that made 4 to 5 purchases from the store is the largest.

```
transaction_count[transaction_count == 100] %>% nrow
```

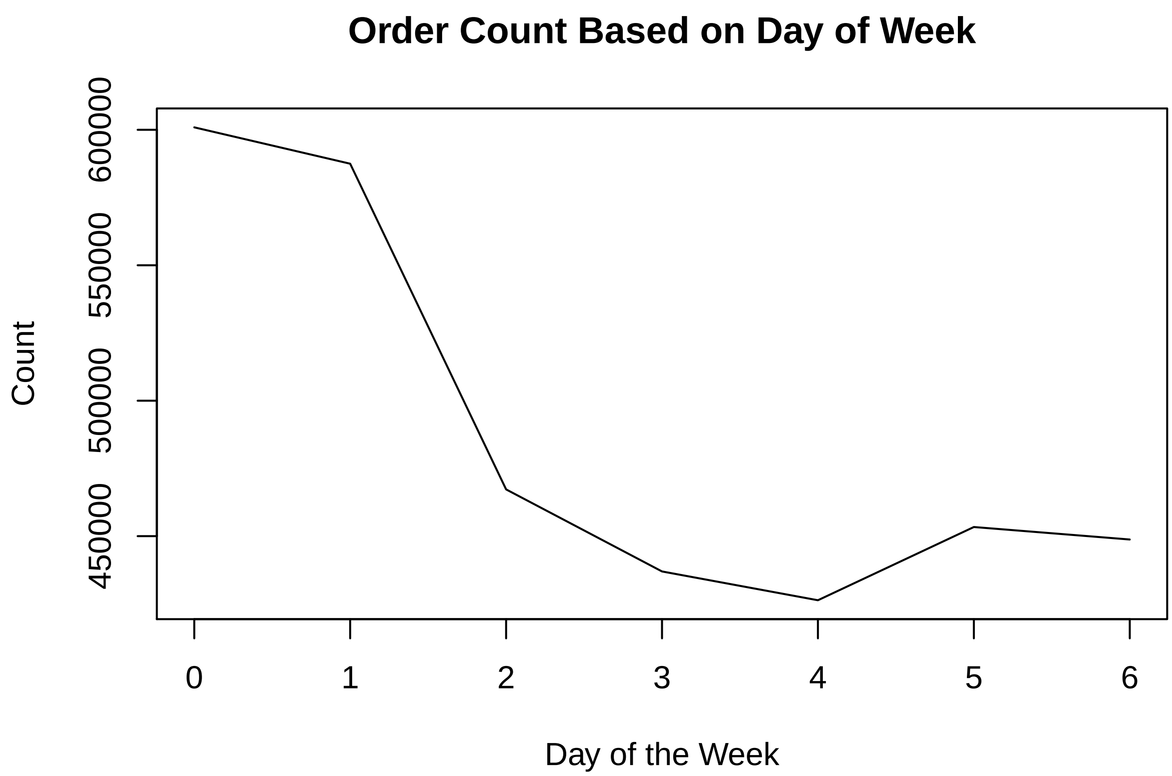
```
## [1] 1374
```

We have 1374 customers that made 100 transactions. It will be interesting to see what kind of products they bought and try to learn thier purhcasing behaviors.

Order day of week and hour of the day

Our data also includes what day of the week and what time of the day that a specific transaction happened. Let's take a look how these number fluctuate throughout the week.

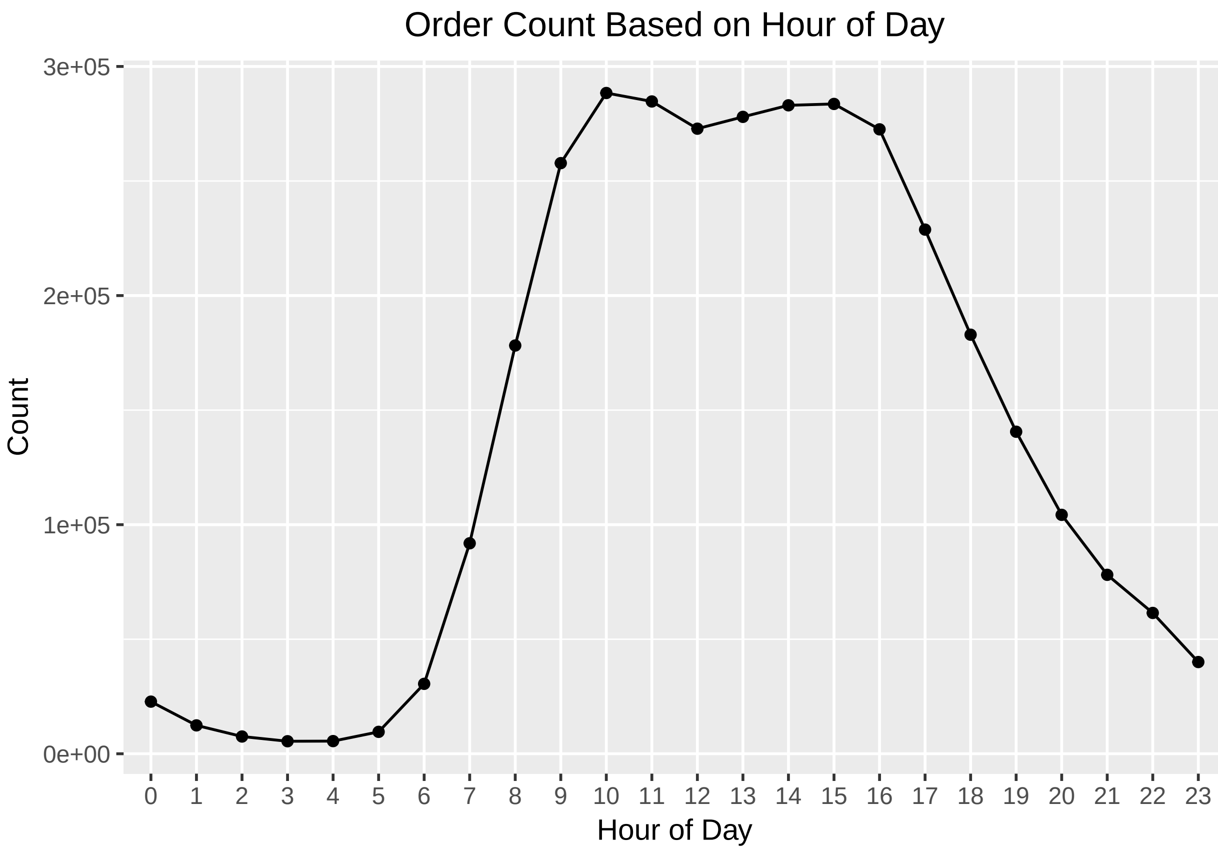
```
dow <- orders[, .N, by = order_dow][order(order_dow)]
plot(dow, type = "l",
     main = "Order Count Based on Day of Week",
     xlab = "Day of the Week",
     ylab = "Count")
```



From the plot above, we can see that day 0 and day 1 of the week are the peak days where the customers made most of their purchases, and the count decreases as the week develops, and eventually bounce back on day 5.

Order Hour of the Day

```
hod <- orders[, .N, by = order_hour_of_day][order(order_hour_of_day)]
ggplot(hod, aes(x= order_hour_of_day, y= N, group = 1)) +
  geom_point() +
  geom_line() +
  xlab("Hour of Day") +
  ylab("Count") +
  ggtitle("Order Count Based on Hour of Day") +
  theme(plot.title = element_text(hjust = 0.5))
```

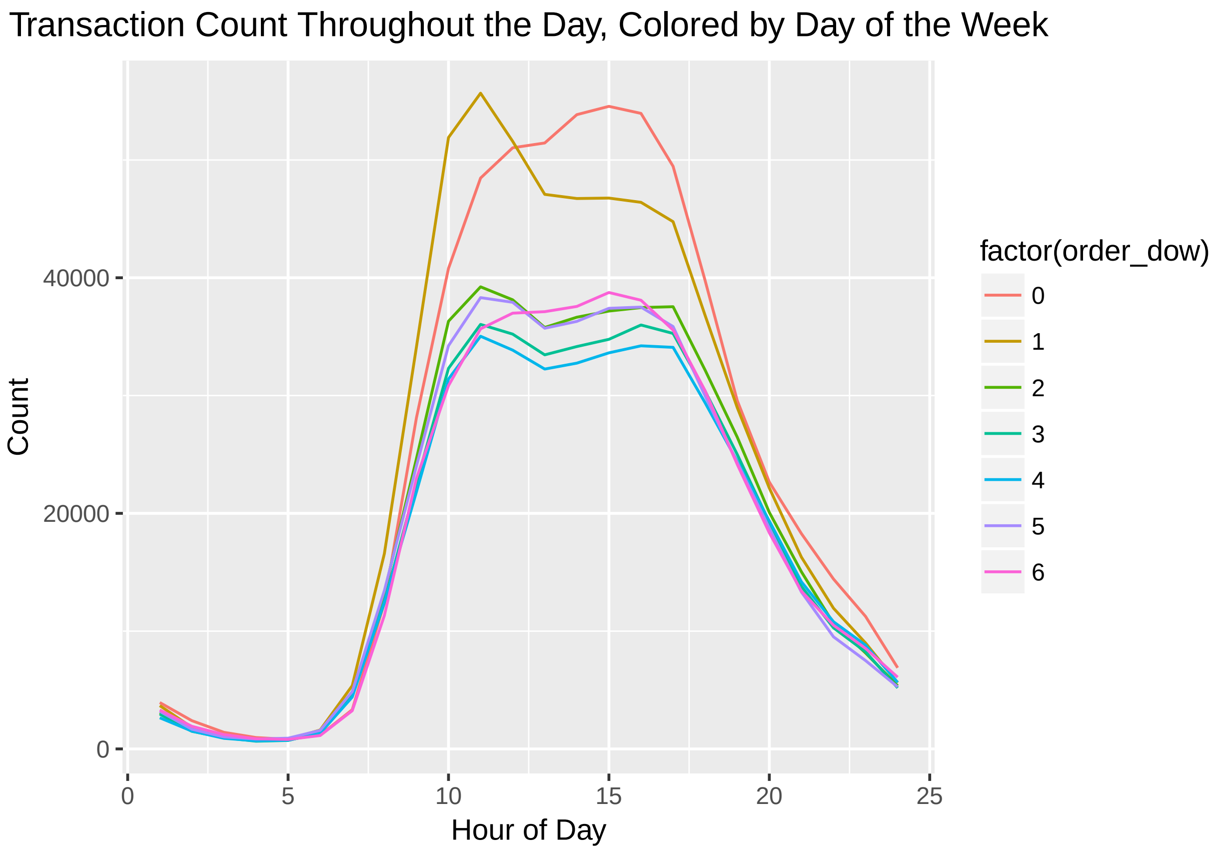


The peak hours are from 10AM to 4PM everyday. Since we don't have the locations where the customers are, so we wouldn't know if there is a difference in peak hours across the naiton.

Combining Day of Week and Hour of the Day

```
order_dow_hod <- orders[, .N, by = .(order_dow, order_hour_of_day)]
```

```
ggplot(order_dow_hod, aes(as.numeric(order_hour_of_day), N, color = factor(order_dow))) +
  geom_line() +
  labs(title = "Transaction Count Throughout the Day, Colored by Day of the Week") +
  theme(plot.title = element_text(hjust = 0.5)) +
  xlab("Hour of Day")+
  ylab("Count")
```



Combining all days in one week, we can see how they are different and similar to each other. As we can see, the pattern from 1AM to 7AM are pretty similar no matter which day it is throughout the week. After 7AM, both day 0 and day 1 start to pick up speed and eventually reach approximately 600K transactions at 3PM (day 0) and 11AM (day 1). We can also see that day 0 and day 5 have similar pattern at peak hours, despite the vast difference in transaction counts, once the transaction quantity go up after 7AM, they don't drop down until it reach the daily maximum. Comparing to the other days (all days beside day 1), we see these bumps created by decline in sales volume after 11AM, and start to bounce back up after 1PM, and eveutally died down around 5PM.

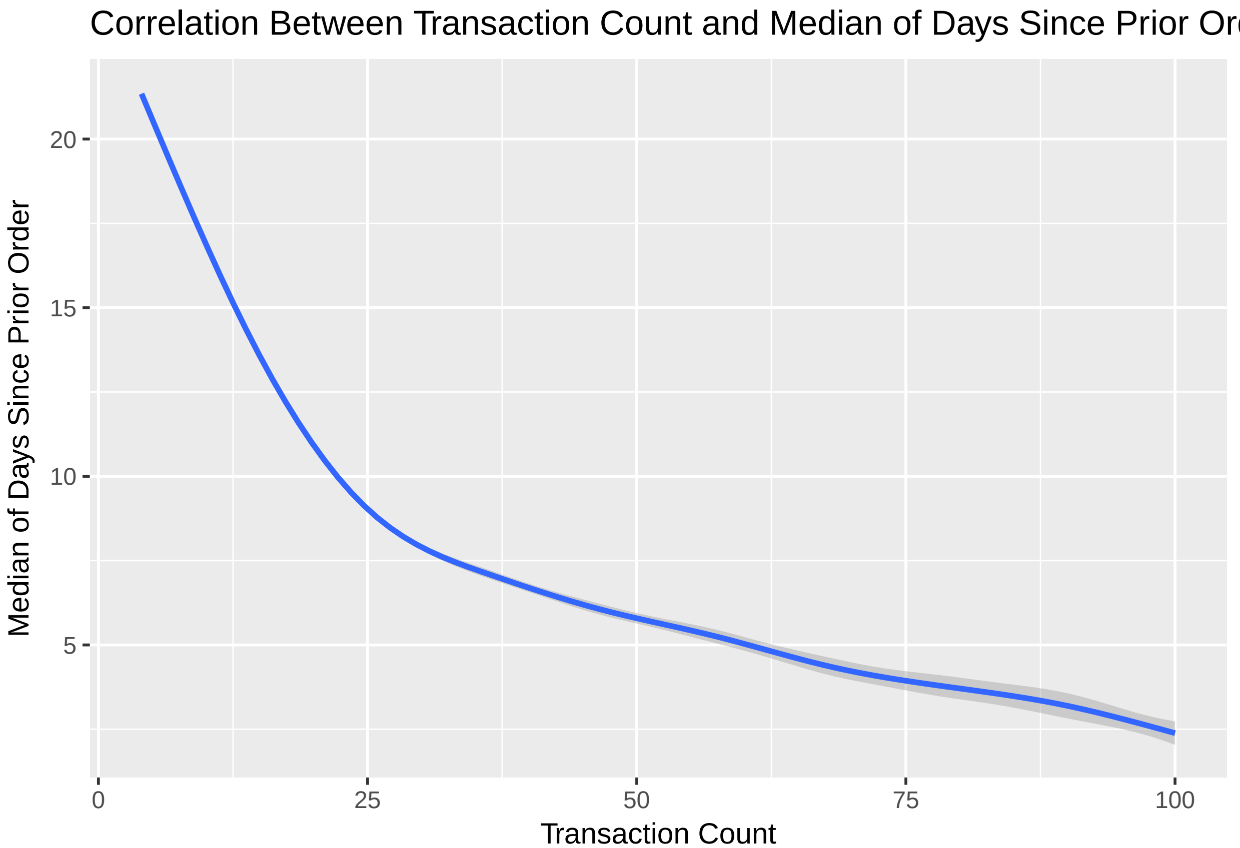
Combining "Days_since_prior_order" and Transaction Made

Since we can get the number of transaction of a specific customer made based on the number of occurance of a specific "user_id", and because that variable "days_since_prior_order" indicates the purhcase frequnecy of a customer, it will be interesting to look at the relationship between these two varaibles.

```
count_freq <- transaction_count[order(user_id)]

freq_df <- orders[, .(freq = median(days_since_prior_order, na.rm = T)), by = user_id]

count_freq$freq <- freq_df$freq
```

```
ggplot(count_freq, aes(transaction_count, freq)) +
  geom_smooth() +
  labs(title= "Correlation Between Transaction Count and Median of Days Since Prior Order") +
  xlab("Transaction Count") +
  ylab("Median of Days Since Prior Order")
```

```
## `geom_smooth()` using method = 'gam'
```

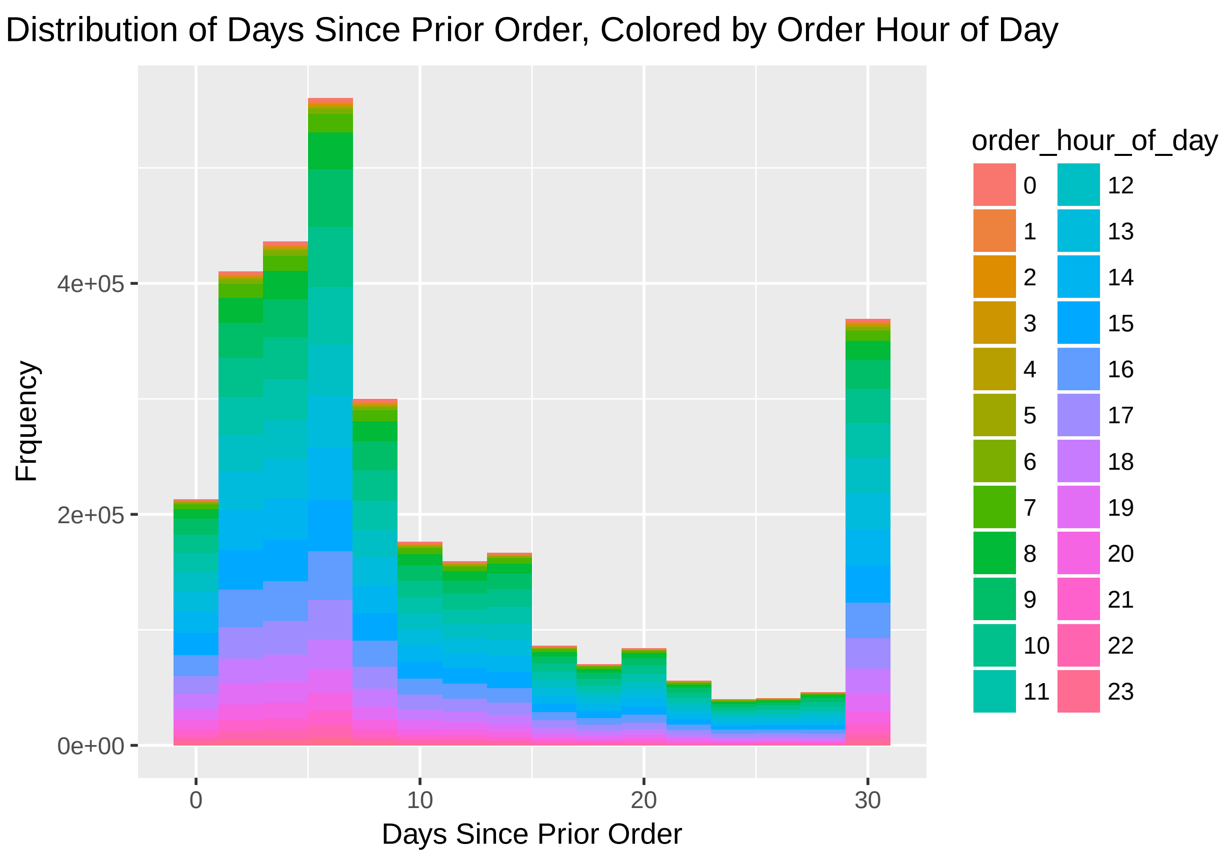


We clearly see that customers who have more transactions tend to shop more.

Distribution of Days Since Prior Order

```r
ggplot(orders, aes(days_since_prior_order, fill = order_hour_of_day)) +
  geom_histogram(binwidth = 2) +
  labs(title = "Distribution of Days Since Prior Order, Colored by Order Hour of Day") +
  xlab("Days Since Prior Order") +
  ylab("Frquency") +
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning: Removed 206209 rows containing non-finite values (stat_bin).
```




```r
#transaction made monthly apart
prop.table(table(orders$days_since_prior_order == 30))
```

```
##
##     FALSE      TRUE
## 0.8851205 0.1148795
```

```r
#transaction made within interval 0 days to 10 days
prop.table(table(orders$days_since_prior_order >0 & orders$days_since_prior_order <= 10))
```

```
##
##     FALSE      TRUE
## 0.3942627 0.6057373
```

From the plot above, we see a bimodal distribution. We can see that there are around 380K (11.48%) transactions made 30 days (one month) after the prior transaction. Around 63% of the customers make purhcase ranging from 0 day to 10 days (around 2% of customer make repurhcase on the same day).

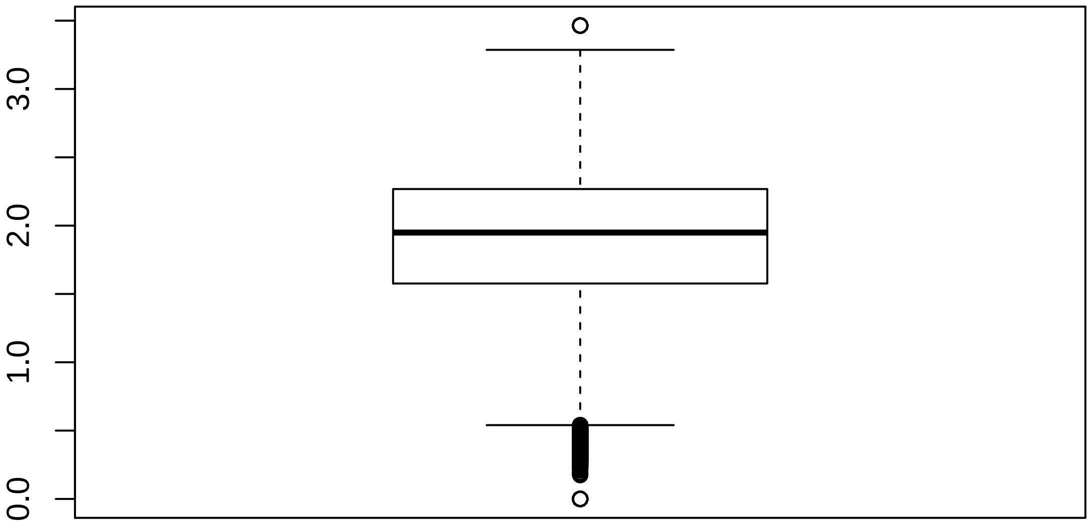
When customers come back, do they make purhcase around the same time (day/hour) ?

We also wonder when customers come back to make another purhcase, is there a consistent trend? For example, a customer made purhcase this Monday at 10AM, will this time frame be somewhat similar the next time this customer come back to make a purhcase again ? We will find out in this section.

We will use standard deviation as a metric to gauge the consistency of specific customer.

```r
#Calculate standard deviation to both of the "order_dow" and "order_hour_of_day", based on user_id, pas
order_patterns <- orders %>% select(user_id, order_dow, order_hour_of_day) %>%
                  group_by(user_id) %>%
                  summarise(dow_sd = sd(order_dow), order_hour_of_day_sd = sd(as.numeric(order_hour_of_
```

```r
boxplot(order_patterns$dow_sd)
```



```r
quantile(order_patterns$dow_sd)
```

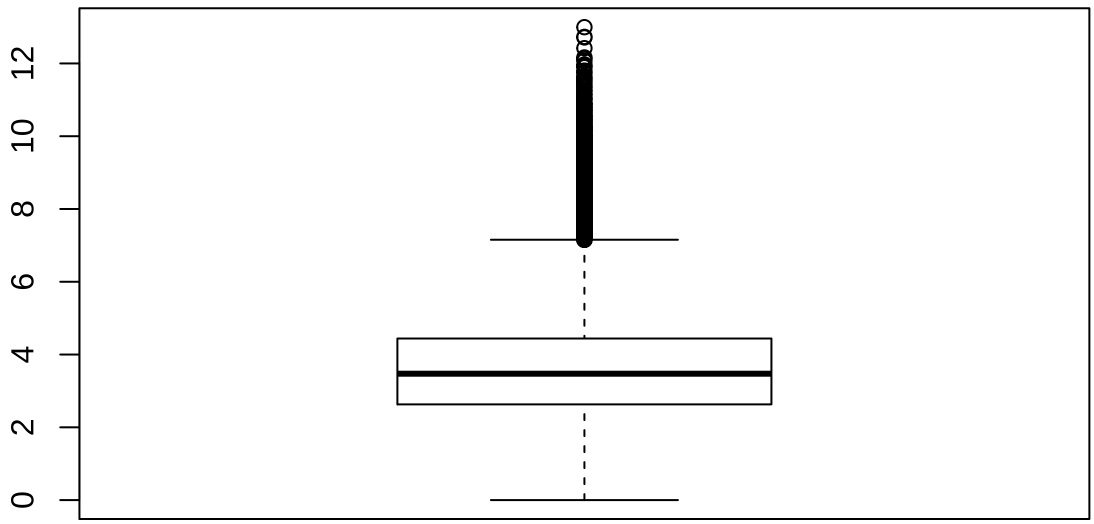
```
##       0%      25%      50%      75%     100%
## 0.000000 1.576482 1.949359 2.267787 3.464102
```

50% of the customer make purhcase 1.58 to 2.27 standard deviation of their day of the week means. We have two hypotheses to explain this phenomenon:

1. Only a small percentage of customer make purhcase on a specific day of the week, most customers would just make purhcase whenever they need, and products they purchase are different every time (reordering is low).
2. Only a small percentage of customer subscribe to a certain type of products and schduled delivery for certain interval of time, most customers make purhcase when products they often buy are exhausted, however, because most people would wait a day or two before placing new orders, which expalins the standard deviation.

We will try to address this observation in later section, when we combine the product information and the order history into one data frame.

```r
boxplot(order_patterns$order_hour_of_day_sd)
```




```
quantile(order_patterns$order_hour_of_day_sd)
```

```
##        0%       25%       50%       75%      100%
##  0.000000  2.629956  3.473342  4.440077 12.996794
```

Comparing to the day of the week standard devations, the standard deviation for hour of the day is higher, 50% of the customers make purhcase between 2.63 to 4.44 standard devations of their means in hour of the day. This huge standard devation make sense since we don't expect customers would keep track when they place order last time, and try to place the next order around the same time.

Orders_products (prior and train)

There are different data files containing the order history in every trasaction, file "order_products_prior" and file "order_products_train". These two files contain variable "order_id", "product_id", "add_to_cart_order", and a binary variable "reordered", with 1 indicates yes and 0 indicates no. In order to get a full grasp of all the transactions, we will combine these two data at the moment.

```
#combining of the prior and train files
combined <- rbind(order_products_prior, order_products_train)
dim(combined)
```

```
## [1] 33819106        4
```

```
str(combined)
```

```
## Classes 'data.table' and 'data.frame':   33819106 obs. of  4 variables:
##  $ order_id         : int  2 2 2 2 2 2 2 2 2 3 ...
##  $ product_id       : int  33120 28985 9327 45918 30035 17794 40141 1819 43668 33754 ...
##  $ add_to_cart_order: int  1 2 3 4 5 6 7 8 9 1 ...
##  $ reordered        : int  1 1 0 1 0 1 1 1 0 1 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

We will then use the product ID to get the product names, and store them under variable "product".

```
combined$product <- products$product_name[combined$product_id]
combined
```

```
##          order_id product_id add_to_cart_order reordered
##       1:        2      33120                 1         1
```

```
##        2:       2      28985                 2          1
##        3:       2       9327                 3          0
##        4:       2      45918                 4          1
##        5:       2      30035                 5          0
##      ---
## 33819102: 3421063      14233                 3          1
## 33819103: 3421063      35548                 4          1
## 33819104: 3421070      35951                 1          1
## 33819105: 3421070      16953                 2          1
## 33819106: 3421070       4724                 3          1
##                                  product
##        1:             Organic Egg Whites
##        2:          Michigan Organic Kale
##        3:                  Garlic Powder
##        4:                 Coconut Butter
##        5:              Natural Sweetener
##      ---
## 33819102:          Natural Artesian Water
## 33819103:            Twice Baked Potatoes
## 33819104: Organic Unsweetened Almond Milk
## 33819105:            Creamy Peanut Butter
## 33819106:              Broccoli Florettes
```

Most popular products

We will then sort the the product and see which products are the most popular.

```
product_count <- data.frame(product_name = table(combined$product) %>% sort(decreasing = T) %>%
                              names,
                            count = table(combined$product) %>% sort(decreasing = T) %>% unlist
                            %>% as.numeric())
```

Here are the ten most popular products in the grocery:

```
product_count %>% head(10)
```

```
##              product_name  count
## 1                  Banana 491291
## 2  Bag of Organic Bananas 394930
## 3    Organic Strawberries 275577
## 4    Organic Baby Spinach 251705
## 5    Organic Hass Avocado 220877
## 6         Organic Avocado 184224
## 7             Large Lemon 160792
## 8            Strawberries 149445
## 9                   Limes 146660
## 10     Organic Whole Milk 142813
```

People really love vegetables, fruits, and organic products.

First product add to the cart

Using the variable “add_to_cart_order”, we can find out which items usually get to added to the cart first.

```
combined[add_to_cart_order == 1, .N, by = product][order(-N)] %>% head(10)
```

```
##                    product      N
##  1:                 Banana 115521
##  2: Bag of Organic Bananas  82877
##  3:     Organic Whole Milk  32071
##  4:   Organic Strawberries  28875
##  5:   Organic Hass Avocado  24913
##  6:   Organic Baby Spinach  24412
##  7:        Organic Avocado  23393
##  8:           Spring Water  17552
##  9:           Strawberries  17073
## 10:    Organic Raspberries  14950
```

We can see that bananas are still the winner.

Which products get reordered most

```
combined %>%
  select(product, reordered) %>%
  group_by(product) %>%
  summarise(reorder_sum = sum(reordered)) %>%
  arrange(desc(reorder_sum)) %>%
  head(10)
```

```
## # A tibble: 10 x 2
##                   product reorder_sum
##                    <fctr>       <int>
##  1                 Banana      415166
##  2 Bag of Organic Bananas      329275
##  3   Organic Strawberries      214448
##  4   Organic Baby Spinach      194939
##  5   Organic Hass Avocado      176173
##  6        Organic Avocado      140270
##  7     Organic Whole Milk      118684
##  8            Large Lemon      112178
##  9    Organic Raspberries      109688
## 10           Strawberries      104588
```

This top 10 list is pretty similar to the most popular products, which make sense, given the fact that nearly 60% of the products are from reordered transactions.

In-depth Analysis

Combining everything

From this point forward, we will try to combined all the information we have at this point and try to gain a deeper insight of the transaction history.

```
combined$order_id <- as.factor(combined$order_id)
```

```
order_hist_detail <- merge(combined, orders, by = "order_id", all.x = TRUE)
#remove repeated variable
```

```r
order_hist_detail$product <- NULL

order_hist_detail %>% head(10)
```

```
##     order_id product_id add_to_cart_order reordered user_id eval_set
##  1:        1      49302                 1         1  112108    train
##  2:        1      11109                 2         1  112108    train
##  3:        1      10246                 3         0  112108    train
##  4:        1      49683                 4         0  112108    train
##  5:        1      43633                 5         1  112108    train
##  6:        1      13176                 6         0  112108    train
##  7:        1      47209                 7         0  112108    train
##  8:        1      22035                 8         1  112108    train
##  9:        2      33120                 1         1  202279    prior
## 10:        2      28985                 2         1  202279    prior
##     order_number order_dow order_hour_of_day days_since_prior_order
##  1:            4         4                10                      9
##  2:            4         4                10                      9
##  3:            4         4                10                      9
##  4:            4         4                10                      9
##  5:            4         4                10                      9
##  6:            4         4                10                      9
##  7:            4         4                10                      9
##  8:            4         4                10                      9
##  9:            3         5                 9                      8
## 10:            3         5                 9                      8
```

Combining transaction history with product information. Since we have almost fifty thousand products, adding the aisle and department information can help use to narrow down customer's preferences.

```r
order_hist_detail <- merge(order_hist_detail, products_w_desc, by = "product_id")[order(order_id)]
order_hist_detail %>% head(10)
```

```
##     product_id order_id add_to_cart_order reordered user_id eval_set
##  1:      10246        1                 3         0  112108    train
##  2:      11109        1                 2         1  112108    train
##  3:      13176        1                 6         0  112108    train
##  4:      22035        1                 8         1  112108    train
##  5:      43633        1                 5         1  112108    train
##  6:      47209        1                 7         0  112108    train
##  7:      49302        1                 1         1  112108    train
##  8:      49683        1                 4         0  112108    train
##  9:       1819        2                 8         1  202279    prior
## 10:       9327        2                 3         0  202279    prior
##     order_number order_dow order_hour_of_day days_since_prior_order
##  1:            4         4                10                      9
##  2:            4         4                10                      9
##  3:            4         4                10                      9
##  4:            4         4                10                      9
##  5:            4         4                10                      9
##  6:            4         4                10                      9
##  7:            4         4                10                      9
##  8:            4         4                10                      9
##  9:            3         5                 9                      8
## 10:            3         5                 9                      8
```

```
##                                      product_name aisle_id department_id
##  1:                          Organic Celery Hearts       83             4
##  2: Organic 4% Milk Fat Whole Milk Cottage Cheese      108            16
##  3:                         Bag of Organic Bananas       24             4
##  4:                    Organic Whole String Cheese       21            16
##  5:          Lightly Smoked Sardines in Olive Oil       95            15
##  6:                           Organic Hass Avocado       24             4
##  7:                               Bulgarian Yogurt      120            16
##  8:                                 Cucumber Kirby       83             4
##  9:         All Natural No Stir Creamy Almond Butter       88            13
## 10:                                  Garlic Powder      104            13
##      aisles_description depart_description
##  1:    fresh vegetables            produce
##  2: other creams cheeses         dairy eggs
##  3:         fresh fruits            produce
##  4:       packaged cheese         dairy eggs
##  5:   canned meat seafood       canned goods
##  6:         fresh fruits            produce
##  7:               yogurt         dairy eggs
##  8:    fresh vegetables            produce
##  9:              spreads             pantry
## 10:    spices seasonings             pantry
```

Most popular products in each department

There are twenty departments total (excluding "missing" department), we wonder what are the most popular (top three) products in each department, and how do their transaction count fluctuate as a day progress.

```r
top_3_each_dept = list()
for (i in 1:(nrow(dept)-1)) {
  tmp_df <- order_hist_detail[depart_description == dept$department[i]] %>%
    select(aisles_description) %>%
    group_by(aisles_description) %>%
    summarise(count = n()) %>%
    arrange(desc(count))

  top_3_each_dept[[i]] = tmp_df$aisles_description[1:3]
}
```

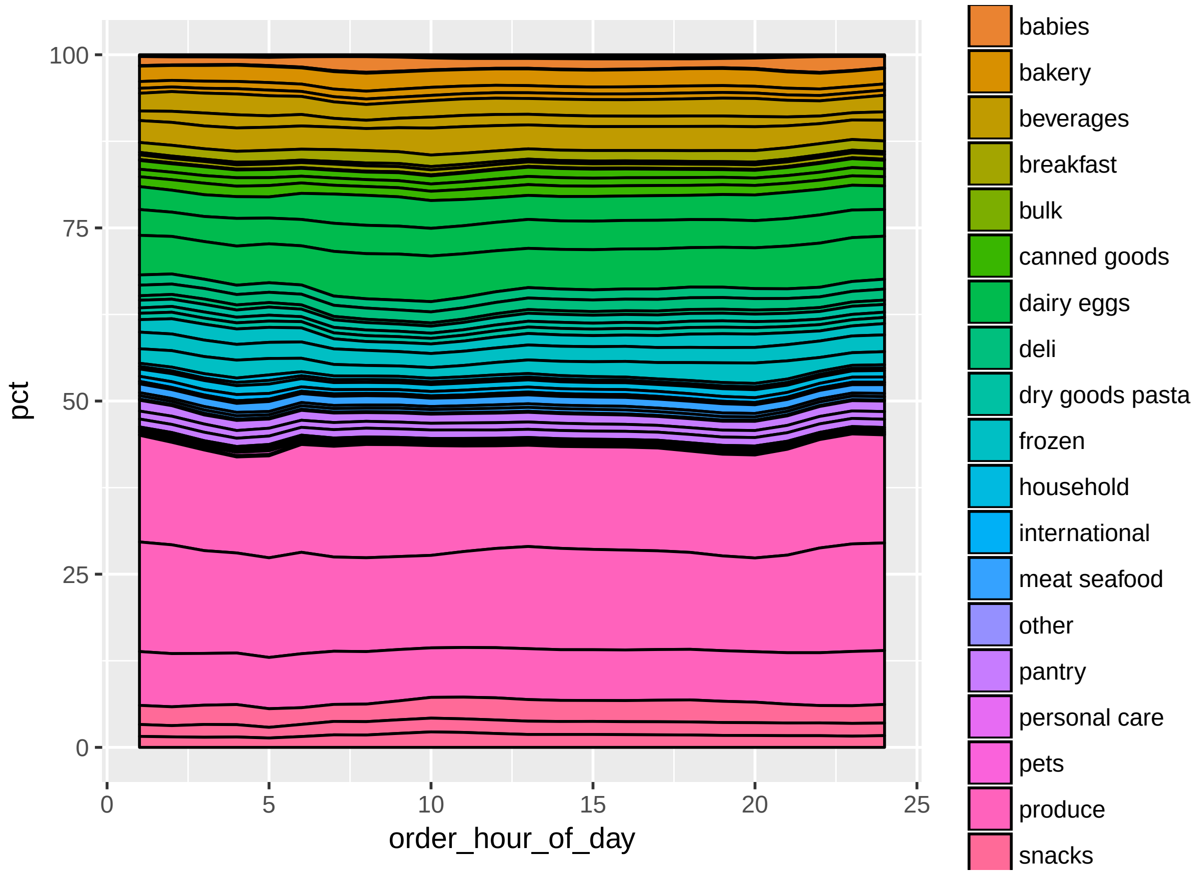
```r
tmp = top_3_each_dept %>% unlist
tmp = tmp[!is.na(tmp)]
tmp
```

```
##  [1] frozen produce               ice cream ice
##  [3] frozen meals                 other
##  [5] bread                        breakfast bakery
##  [7] tortillas flat bread         fresh fruits
##  [9] fresh vegetables             packaged vegetables fruits
## [11] beers coolers                red wines
## [13] white wines                  asian foods
## [15] latino foods                 indian foods
## [17] water seltzer sparkling water refrigerated
## [19] soft drinks                  cat food care
## [21] dog food care                dry pasta
```

```
## [23] pasta sauce                    instant foods
## [25] bulk dried fruits vegetables  bulk grains rice dried goods
## [27] soap                          oral hygiene
## [29] vitamins supplements          hot dogs bacon sausage
## [31] poultry counter               packaged poultry
## [33] baking ingredients            spreads
## [35] oils vinegars                 cereal
## [37] hot cereal pancake mixes      granola
## [39] soup broth bouillon           canned jarred vegetables
## [41] canned meals beans            yogurt
## [43] packaged cheese               milk
## [45] paper goods                   cleaning products
## [47] laundry                       baby food formula
## [49] diapers wipes                 baby bath body care
## [51] chips pretzels                crackers
## [53] energy granola bars           lunch meat
## [55] fresh dips tapenades          tofu meat alternatives
## 134 Levels: air fresheners candles asian foods ... yogurt
```

```r
index <- which(order_hist_detail$aisles_description %in% tmp) %>% unlist %>% as.numeric()
tmp2 <- order_hist_detail[index,]

hourly_total = tmp2[, .(hourly_total = .N), by = order_hour_of_day]

tmp3 <- tmp2[, .N, by = .(aisles_description, depart_description, order_hour_of_day)]
tmp3 <- merge(tmp3, hourly_total, by = "order_hour_of_day")
tmp3 <- tmp3 %>% mutate(pct = N/hourly_total*100)

tmp3$order_hour_of_day <- as.numeric(tmp3$order_hour_of_day)
```

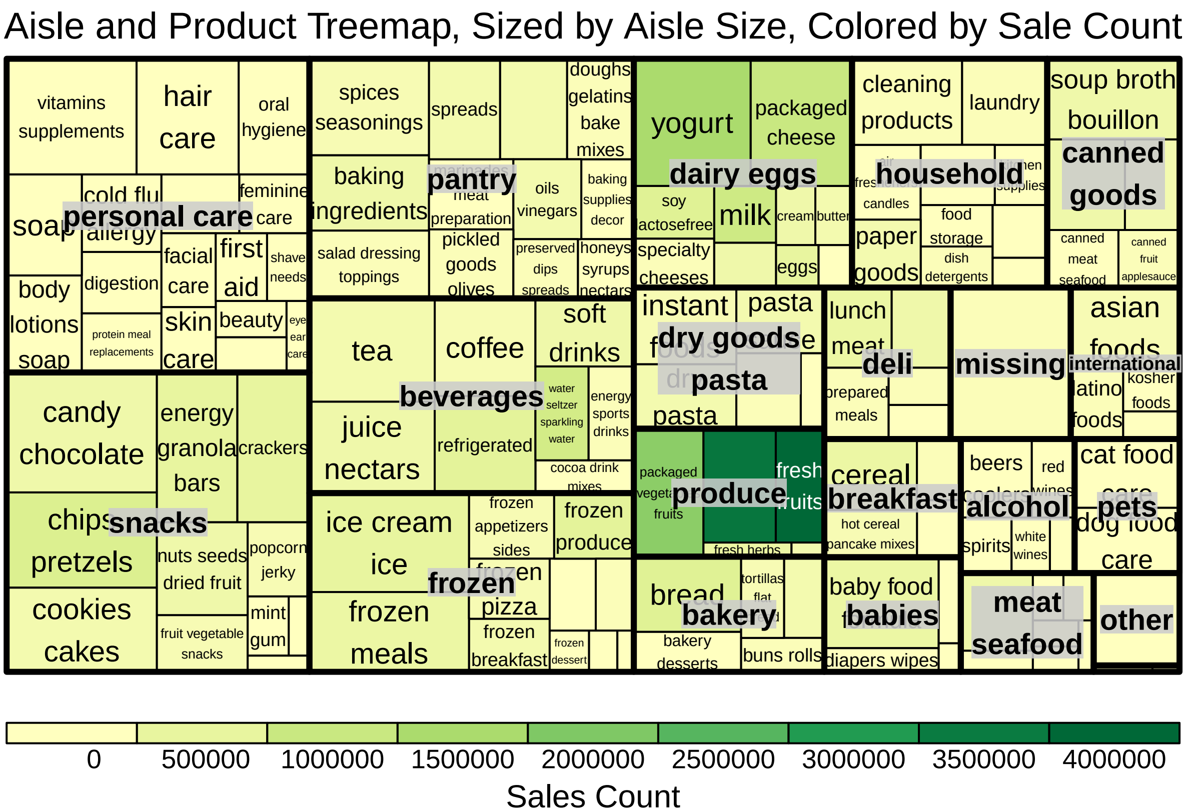
```r
ggplot(tmp3, aes(order_hour_of_day, pct, group = interaction(aisles_description, depart_description))) +
  geom_area(color = "black", aes(fill = depart_description))
```



From the plot above we can see how the percentage of sales in different aisels move as the day progress. Generally speaking, we don't see the percentage of sales shift too much. In addtion, we also see that aisles produce, snacks take up approximately 50% of the sales, among all the sales of top 3 products in each department.

```
tmp <- products_w_desc
depart_size <- tmp[, .(depart_size = .N), by = depart_description]
depart_aisles_size <- tmp[, .(aisles_size = .N), by = .(aisles_description, depart_description)]

depart_aisles_size <- merge(depart_aisles_size, depart_size, by ="depart_description", all.x = TRUE)

item_sales_count <- order_hist_detail[, .(item_sale_count = .N), by = aisles_description]

treemap_df <- merge(depart_aisles_size, item_sales_count, by = "aisles_description")
```

```
treemap(treemap_df,
        index = c("depart_description", "aisles_description"),
        vSize = "aisles_size",
        vColor = "item_sale_count",
        type = "value",
        title = "Aisle and Product Treemap, Sized by Aisle Size, Colored by Sale Count",
        title.legend = "Sales Count")
```



Earlier we've made an assumption that because "personal care" and "snacks" are the department carry most prodcuts in the grocery store, they should have the highest sale, however, we were wrong. From the treemap, we see that department that sold most products were the produce department (thanks to those bananas).