Next product in the basket prediction

Sapna

2/18/2020

```
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

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

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

```
library(ggplot2)
library(tidyr)
library(arules)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

##
## Attaching package: 'arules'

## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
library(arulesViz)
```

```
## Loading required package: grid
```

```
library(colorspace)
```

Project

```
# Reading files
aisles <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data/aisle
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   aisle_id = col_double(),
##   aisle = col_character()
## )
```

```
departments <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data//depa
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   department_id = col_double(),
##   department = col_character()
## )
```

```
order_products_prior <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data//orde
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   order_id = col_double(),
##   product_id = col_double(),
##   add_to_cart_order = col_double(),
##   reordered = col_double()
## )
```

```
order_products_train <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data//orde
```

```
##
## -- Column specification --------------------------------------------------------
```

```
## cols(
##   order_id = col_double(),
##   product_id = col_double(),
##   add_to_cart_order = col_double(),
##   reordered = col_double()
## )
```

```
orders <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data//orde
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   order_id = col_double(),
##   user_id = col_double(),
##   eval_set = col_character(),
##   order_number = col_double(),
##   order_dow = col_double(),
##   order_hour_of_day = col_character(),
##   days_since_prior_order = col_double()
## )
```

```
products <-
  readr::read_csv("/Users/sapnasharma/Documents/MSDS_Subjects/IDMP/IDMP_Project/IDMP_Project_Data//produ
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   product_id = col_double(),
##   product_name = col_character(),
##   aisle_id = col_double(),
##   department_id = col_double()
## )
```

Observing the data

There are 2 CSV files, namely **order__products__train** and **order__products__prior**, that specify which products were purchased in each order.order__products__prior contains previous order products for all customers and order__products__train contains the latest order products for some customers only.

```
str(order_products_train)
```

```
## tibble [1,384,617 x 4] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ order_id         : num [1:1384617] 1 1 1 1 1 1 1 1 36 36 ...
##  $ product_id       : num [1:1384617] 49302 11109 10246 49683 43633 ...
##  $ add_to_cart_order: num [1:1384617] 1 2 3 4 5 6 7 8 1 2 ...
##  $ reordered        : num [1:1384617] 1 1 0 0 1 0 0 1 0 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   order_id = col_double(),
##   ..   product_id = col_double(),
##   ..   add_to_cart_order = col_double(),
##   ..   reordered = col_double()
##   .. )
```

```r
str(order_products_prior)
```

```
## tibble [32,434,489 x 4] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ order_id         : num [1:32434489] 2 2 2 2 2 2 2 2 2 3 ...
##  $ product_id       : num [1:32434489] 33120 28985 9327 45918 30035 ...
##  $ add_to_cart_order: num [1:32434489] 1 2 3 4 5 6 7 8 9 1 ...
##  $ reordered        : num [1:32434489] 1 1 0 1 0 1 1 1 0 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   order_id = col_double(),
##   ..   product_id = col_double(),
##   ..   add_to_cart_order = col_double(),
##   ..   reordered = col_double()
##   .. )
```

There are 13,84,617 products in the order_products_train file and 3,24,34,489 products in the order_products_prior file. Both files have 4 feature columns: The ID of the order (order_id) The ID of the product (product_id) The ordering of that product in the order (add_to_cart_order) Whether that product was reordered (reordered).

```r
#we combine both files to find the unique items
order_products <- rbind(order_products_train,order_products_prior)
#str(order_products)
#unique customers
order_products %>% distinct(order_id) %>% count()
```

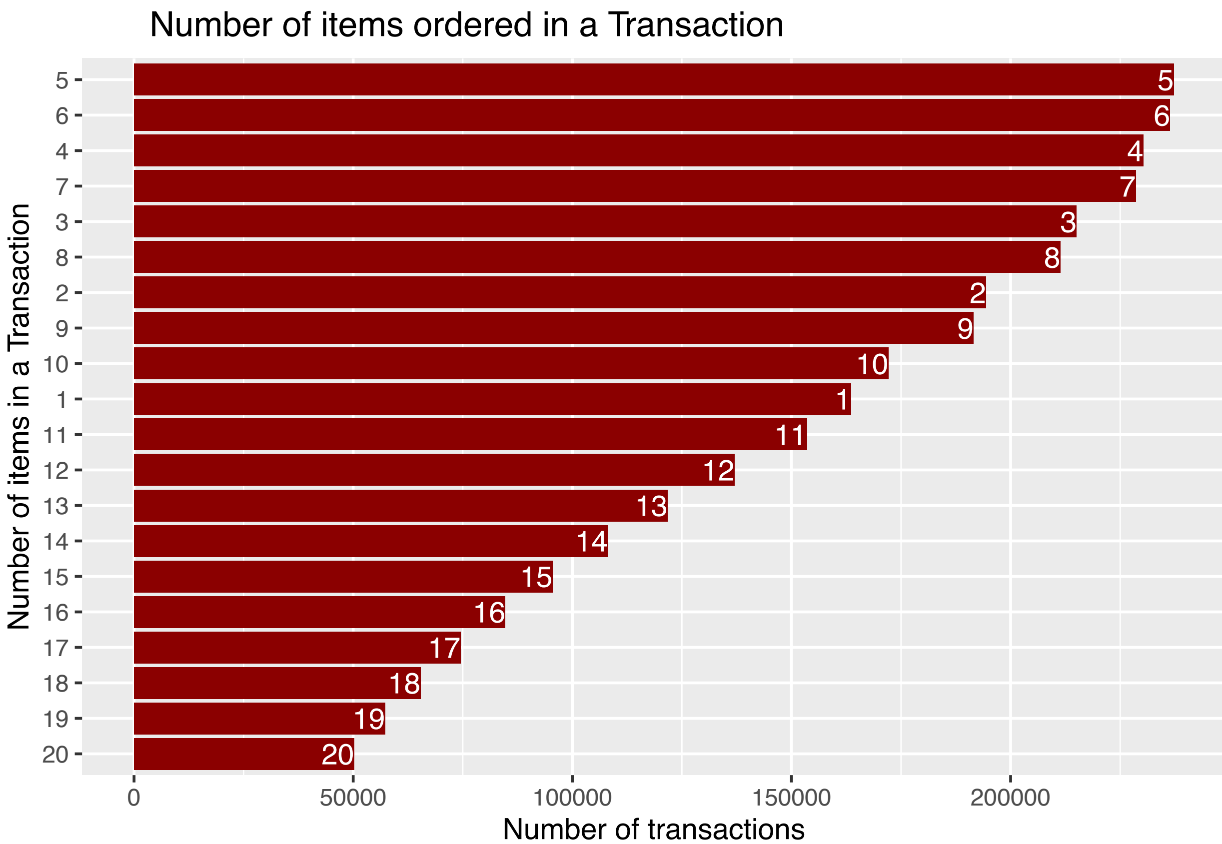
```
## # A tibble: 1 x 1
##         n
##     <int>
## 1 3346083
```

```r
#unique products
order_products %>% distinct(product_id) %>% count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1 49685
```

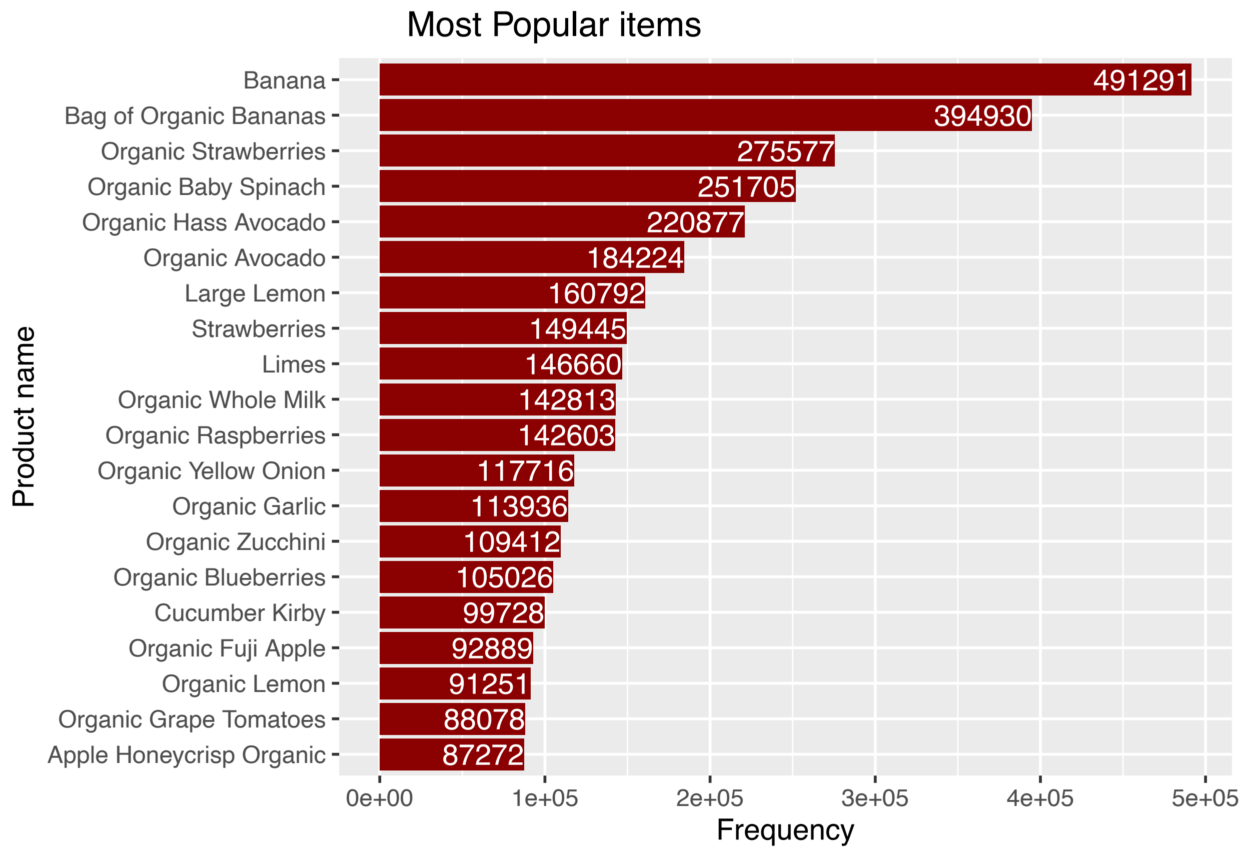
Overall, there are 33,46,083 unique orders for 49,685 unique products.

```r
# How many items were ordered the most
order_products %>% group_by(order_id)  %>% count()%>% group_by(n) %>% count() %>% head(20)%>%
  ggplot() + aes(x = reorder(n,nn),nn) +
  geom_col(stat = "identity",fill = "darkred") +
    coord_flip() +
  labs(title = "       Number of items ordered in a Transaction ") +
 xlab("Number of items in a Transaction") +ylab("Number of transactions")  +
  geom_text(aes(label=n), hjust=1, color = "white")
```



We can observe in the plot above that people usually order around 5 or 6 products in an order.

```
order_products %>% left_join(products) %>%
  group_by(product_name) %>% count() %>% arrange(desc(n))%>%
  head(20) %>%ggplot() + aes(x = reorder(product_name,n),n) + geom_col(stat = "identity",fill = "darkred
  geom_text(aes(label=n), hjust=1, color = "white")
```



Seeing the count

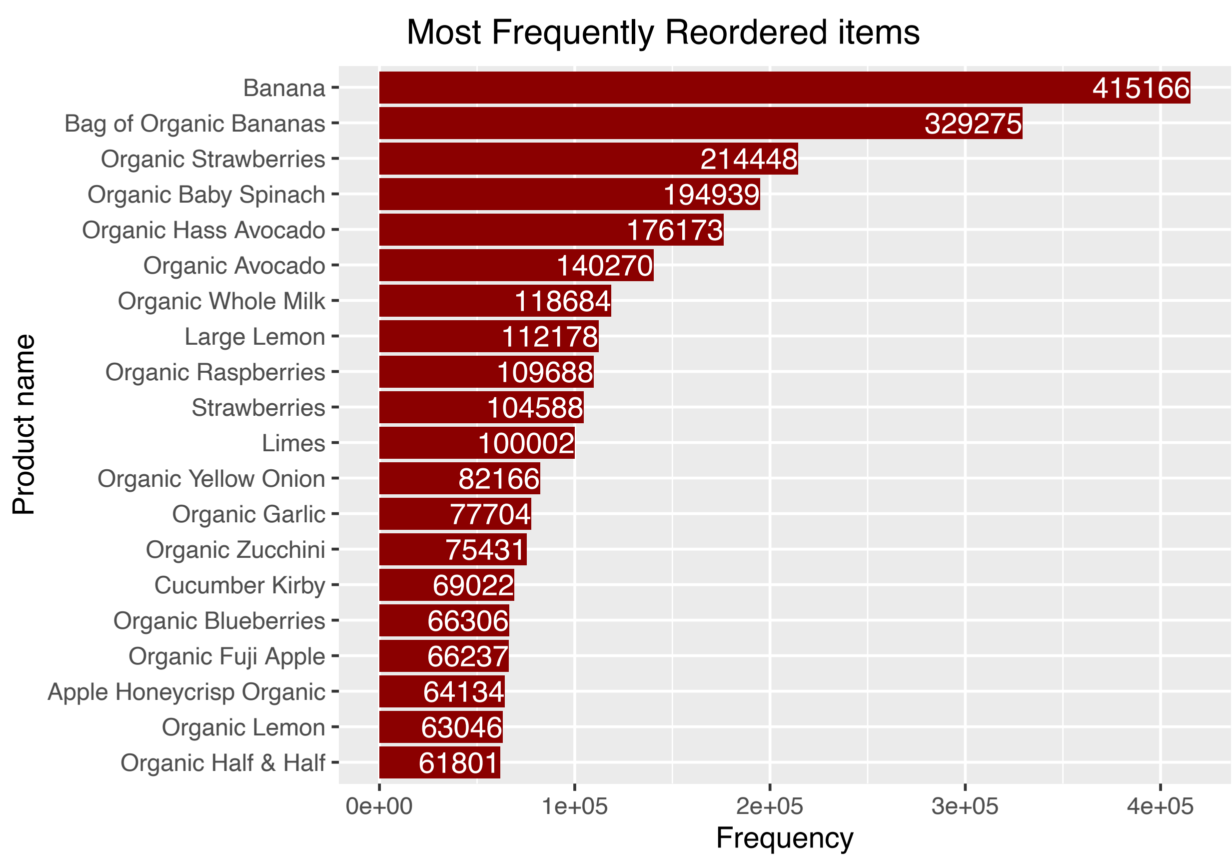
```
print(order_products %>% left_join(products) %>%
  group_by(product_name) %>% count() %>% arrange(desc(n))%>%
  head(20))
```

```
## # A tibble: 20 x 2
## # Groups:   product_name [20]
##    product_name                 n
##    <chr>                    <int>
##  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
## 11 Organic Raspberries     142603
## 12 Organic Yellow Onion    117716
## 13 Organic Garlic          113936
## 14 Organic Zucchini        109412
## 15 Organic Blueberries     105026
## 16 Cucumber Kirby           99728
## 17 Organic Fuji Apple       92889
```

```
## 18 Organic Lemon              91251
## 19 Organic Grape Tomatoes     88078
## 20 Apple Honeycrisp Organic   87272
```

Top five popular item happens to be :
Banana (491291),
Bag of Organic Bananas (394930),
Organic Strawberries (275577),
Organic Baby Spinach (251705)
Organic Hass Avocado (220877)

```
# most reordered items
order_products %>%
  group_by(product_id )%>%
  summarise(x=sum(reordered))%>% arrange(desc(x) ) %>%
  left_join(products) %>%
  head(20) %>%ggplot() + aes(x = reorder(product_name,x),x) + geom_col(stat = "identity",fill = "darkre
  geom_text(aes(label=x), hjust=1, color = "white")
```




```
str(orders)
```

```
## tibble [3,421,083 x 7] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ order_id              : num [1:3421083] 2539329 2398795 473747 2254736 431534 ...
##  $ user_id               : num [1:3421083] 1 1 1 1 1 1 1 1 1 1 ...
##  $ eval_set              : chr [1:3421083] "prior" "prior" "prior" "prior" ...
##  $ order_number          : num [1:3421083] 1 2 3 4 5 6 7 8 9 10 ...
```

```
##  $ order_dow             : num [1:3421083] 2 3 3 4 4 2 1 1 1 4 ...
##  $ order_hour_of_day     : chr [1:3421083] "08" "07" "12" "07" ...
##  $ days_since_prior_order: num [1:3421083] NA 15 21 29 28 19 20 14 0 30 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   order_id = col_double(),
##   ..   user_id = col_double(),
##   ..   eval_set = col_character(),
##   ..   order_number = col_double(),
##   ..   order_dow = col_double(),
##   ..   order_hour_of_day = col_character(),
##   ..   days_since_prior_order = col_double()
##   .. )
```

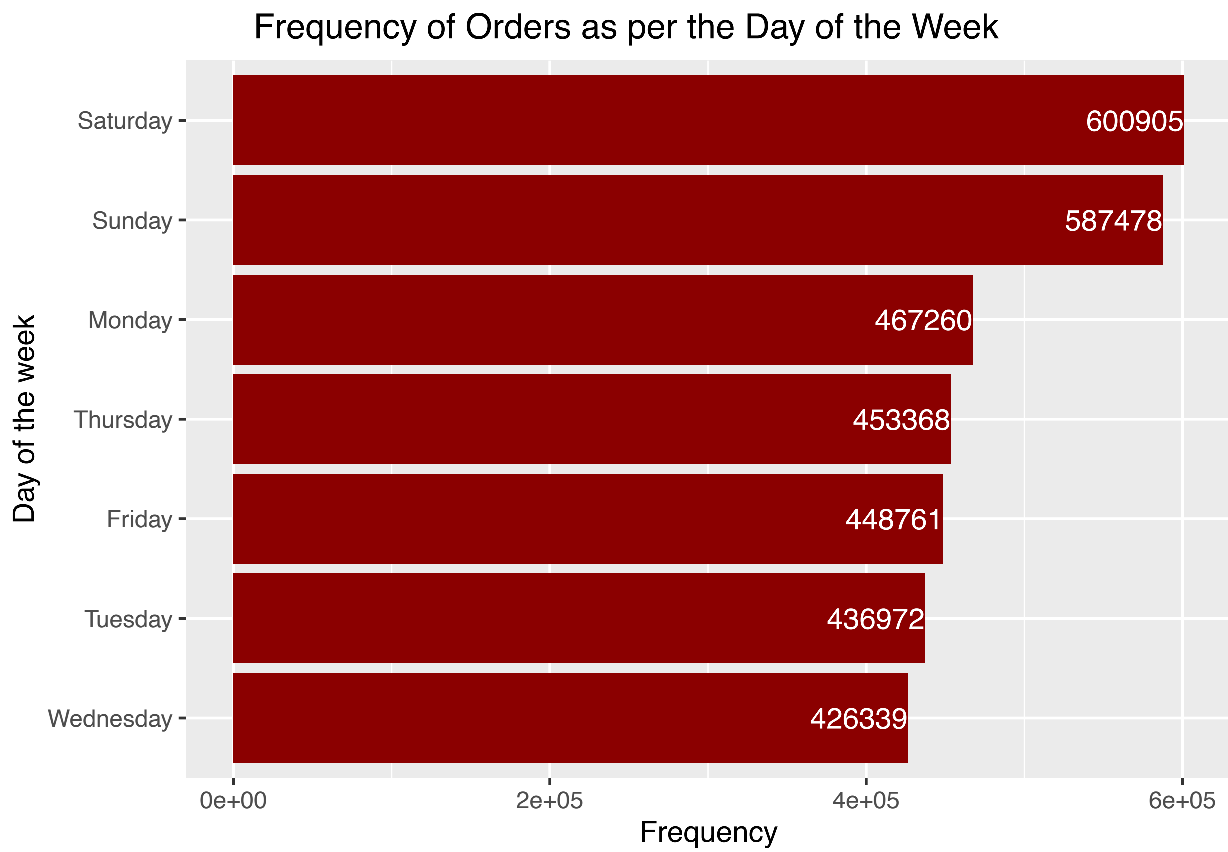
```r
head(orders,20)
```

```
## # A tibble: 20 x 7
##    order_id user_id eval_set order_number order_dow order_hour_of_d~
##       <dbl>   <dbl> <chr>           <dbl>     <dbl> <chr>
##  1  2539329       1 prior               1         2 08
##  2  2398795       1 prior               2         3 07
##  3   473747       1 prior               3         3 12
##  4  2254736       1 prior               4         4 07
##  5   431534       1 prior               5         4 15
##  6  3367565       1 prior               6         2 07
##  7   550135       1 prior               7         1 09
##  8  3108588       1 prior               8         1 14
##  9  2295261       1 prior               9         1 16
## 10  2550362       1 prior              10         4 08
## 11  1187899       1 train              11         4 08
## 12  2168274       2 prior               1         2 11
## 13  1501582       2 prior               2         5 10
## 14  1901567       2 prior               3         1 10
## 15   738281       2 prior               4         2 10
## 16  1673511       2 prior               5         3 11
## 17  1199898       2 prior               6         2 09
## 18  3194192       2 prior               7         2 12
## 19   788338       2 prior               8         1 15
## 20  1718559       2 prior               9         2 09
## # ... with 1 more variable: days_since_prior_order <dbl>
```

The orders.csv file has 3,421,083 orders and 7 feature columns: The ID of the order (order_id) The ID of the customer (user_id) Which evaluation datasets that the order is in — prior, train, or test (eval_set) The number of the order (order_number) The day of the week when that order occurred (order_dow) The hour of the day when that order occurred (order_hour_of_day) The number of days since the previous order (days_since_prior_order)

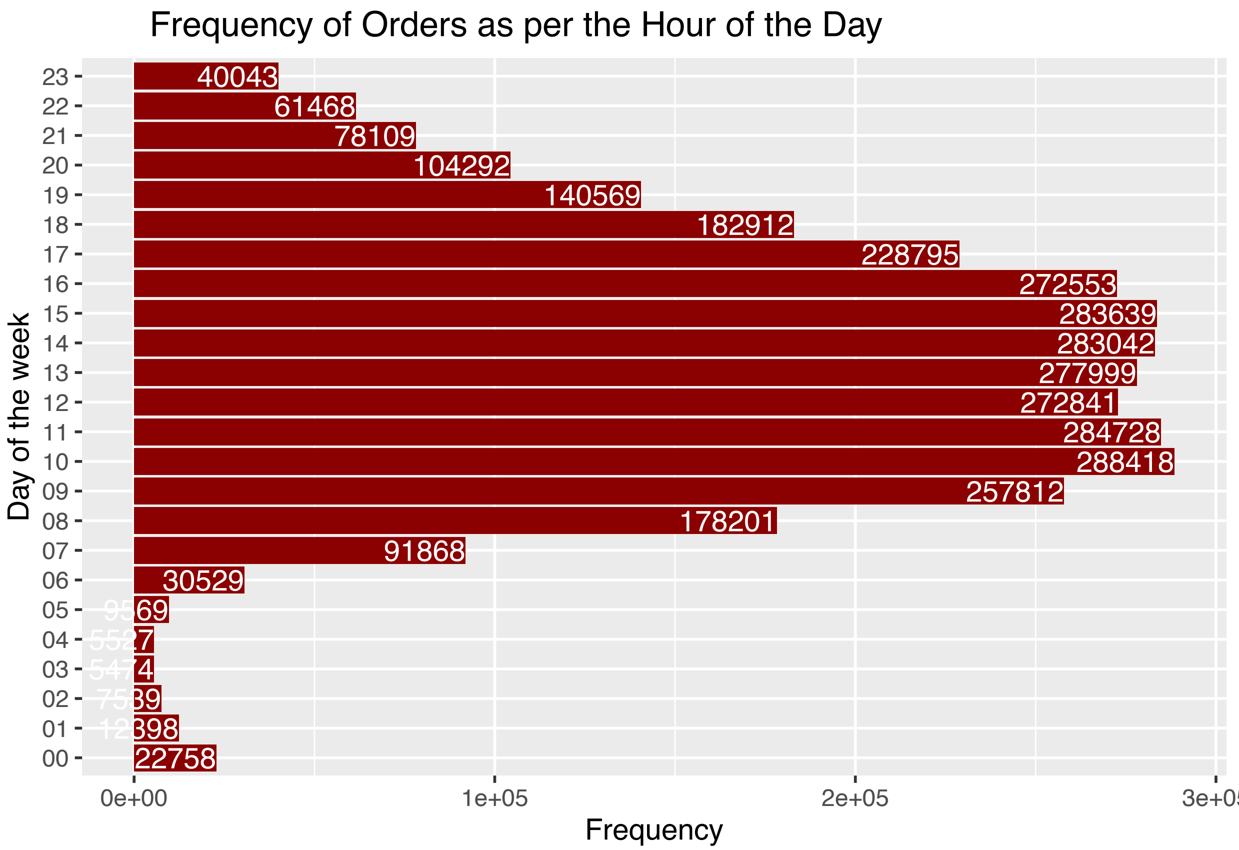
```r
# Day of the week
orders_n <- orders
orders_n$order_dow[orders_n$order_dow == 0] <- "Saturday"
orders_n$order_dow[orders_n$order_dow == 1] <- "Sunday"
orders_n$order_dow[orders_n$order_dow == 2] <- "Monday"
orders_n$order_dow[orders_n$order_dow == 3] <- "Tuesday"
```

```
orders_n$order_dow[orders_n$order_dow == 4] <- "Wednesday"
orders_n$order_dow[orders_n$order_dow == 5] <- "Thursday"
orders_n$order_dow[orders_n$order_dow == 6] <- "Friday"
orders_n %>% group_by(order_dow) %>% count() %>%
  ggplot() + aes(x = reorder(order_dow,n),n) + geom_col(stat = "identity",fill = "darkred") + coord_fli
  labs(title = "       Frequency of Orders as per the Day of the Week     ") +
  xlab("Day of the week") +ylab("Frequency") +
  geom_text(aes(label=n), hjust=1, color = "white")
```



Clearly most orders are made on Saturday followed by Sunday.

```
orders %>% group_by(order_hour_of_day) %>% count() %>%
  ggplot() + aes(x = (order_hour_of_day),n) + geom_col(stat = "identity",fill = "darkred") + coord_flip
  labs(title = "       Frequency of Orders as per the Hour of the Day     ") +
  xlab("Day of the week") +ylab("Frequency") +
  geom_text(aes(label=n), hjust=1, color = "white")
```



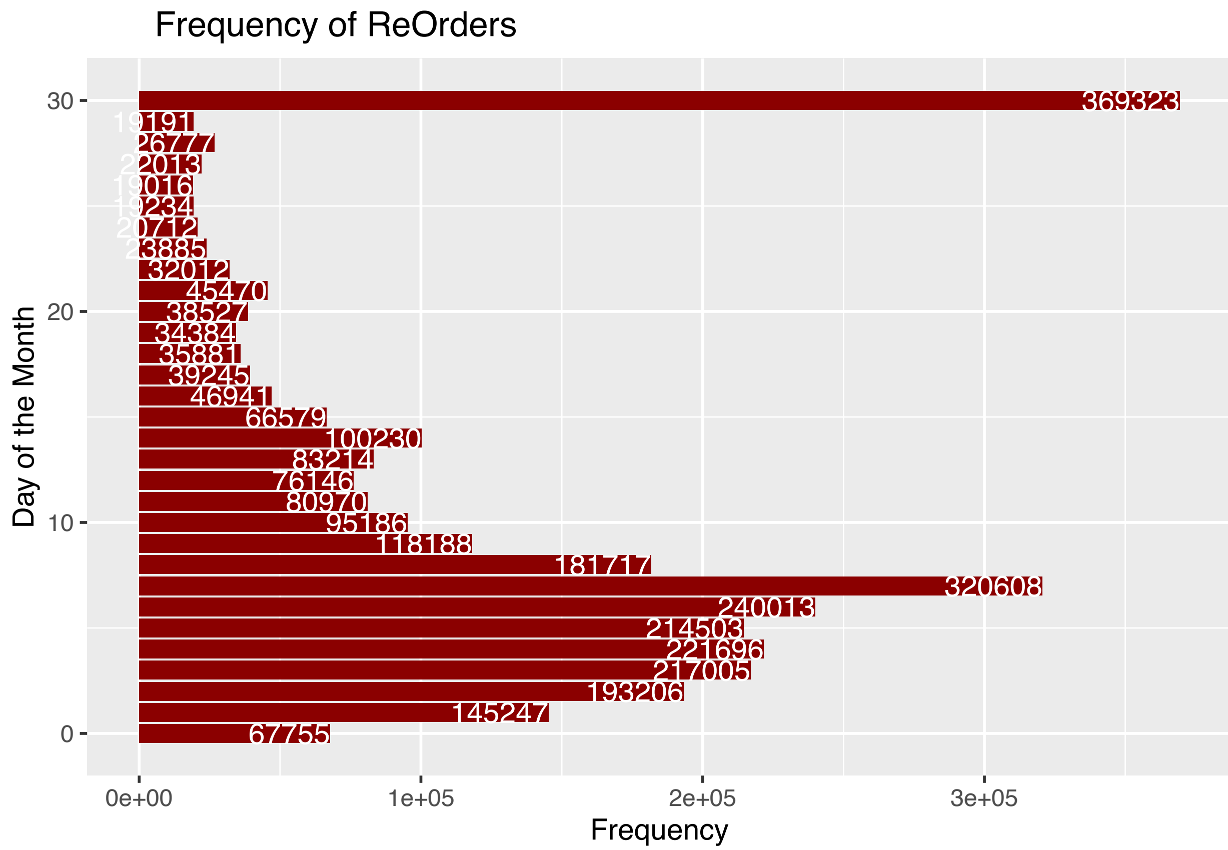
It is seen that most orders are placed between 10 am to 4pm.

```
#When do they order again?
orders %>% group_by(days_since_prior_order) %>% count() %>%
  ggplot() + aes(x = (days_since_prior_order),n) + geom_col(stat = "identity",fill = "darkred") + coord_
  labs(title = "        Frequency of ReOrders      ") +
  xlab("Day of the Month") +ylab("Frequency") + geom_text(aes(label=n), hjust=1, color = "white")
```

```
## Warning: Ignoring unknown parameters: stat
```

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

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




```
orders
```

```
## # A tibble: 3,421,083 x 7
##    order_id user_id eval_set order_number order_dow order_hour_of_d~
##       <dbl>   <dbl> <chr>           <dbl>     <dbl> <chr>
##  1  2539329       1 prior               1         2 08
##  2  2398795       1 prior               2         3 07
##  3   473747       1 prior               3         3 12
##  4  2254736       1 prior               4         4 07
##  5   431534       1 prior               5         4 15
##  6  3367565       1 prior               6         2 07
##  7   550135       1 prior               7         1 09
##  8  3108588       1 prior               8         1 14
##  9  2295261       1 prior               9         1 16
## 10  2550362       1 prior              10         4 08
## # ... with 3,421,073 more rows, and 1 more variable:
## #   days_since_prior_order <dbl>
```

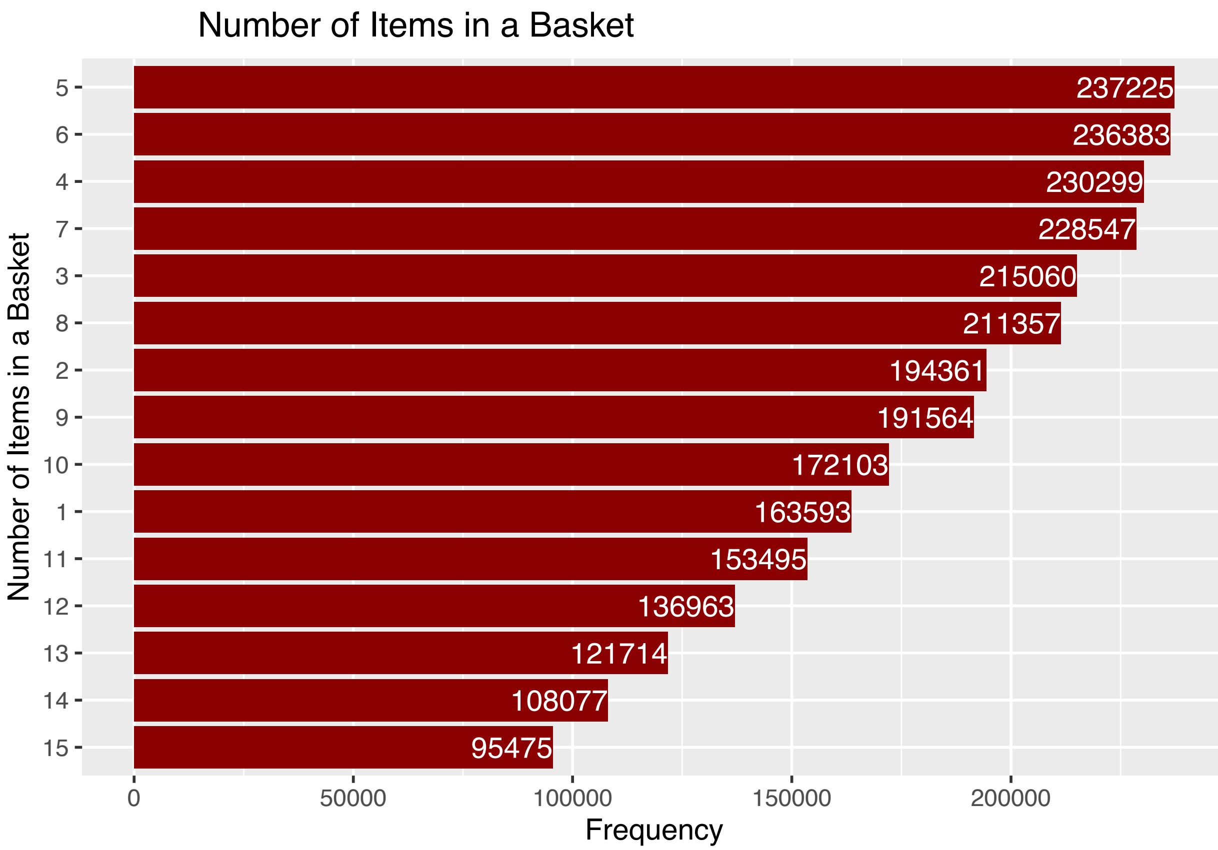
We see that things are either reordered after a month or after 7 days.

Prediction

```
# get the shopping baskets
order_baskets <- order_products %>%
  group_by(order_id) %>% count()
#ordet_baskets
```

```
order_baskets %>% group_by(n) %>% count %>%
  head(15) %>%ggplot() + aes(reorder(n,nn),nn)+geom_col(fill = "darkred")  + coord_flip() +
  geom_text(aes(label=nn), hjust=1, color = "white")+
  labs(title = "            Number of Items in a Basket") +
  xlab("Number of Items in a Basket") +
  ylab("Frequency")
```

```
## Storing counts in `nn`, as `n` already present in input
## i Use `name = "new_name"` to pick a new name.
```



We see that mostly peoply buy 3 to 7 items in an order.

```
order_baskets <- order_products %>%
  inner_join(products, by="product_id") %>%
  group_by(order_id) %>%
  summarise(basket = as.vector(list(product_name)))
```

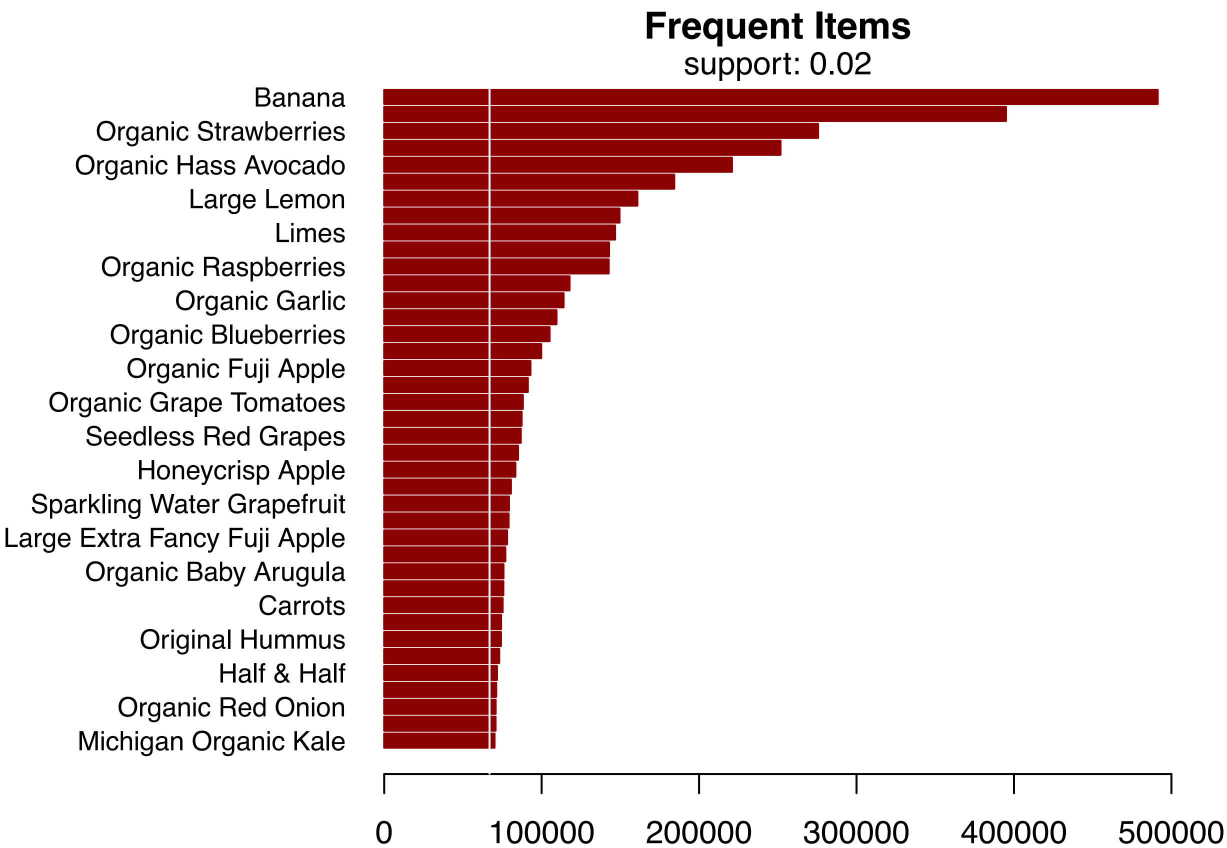
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
transactions <- as(order_baskets$basket, "transactions")
```

We determine which items are frequent. We set the support threshold to 0.02, that means an item will be considered as frequent iff at least 2 percent of all the baskets contain it. So in our case, an item will be considered as being frequent if it is contained in more than 64,000 baskets.

```
item_frequencies <- itemFrequency(transactions, type="a")
support <- 0.02
freq_items <- sort(item_frequencies, decreasing = F)
freq_items <- freq_items[freq_items>support*length(transactions)]

par(mar=c(2,10,2,2)); options(scipen=5)
barplot(freq_items, horiz=T, las=1, main="Frequent Items", cex.names=.8, xlim=c(0,500000),col = "darkre
mtext(paste("support:",support), padj = .8)
abline(v=support*length(transactions), col="white")
```



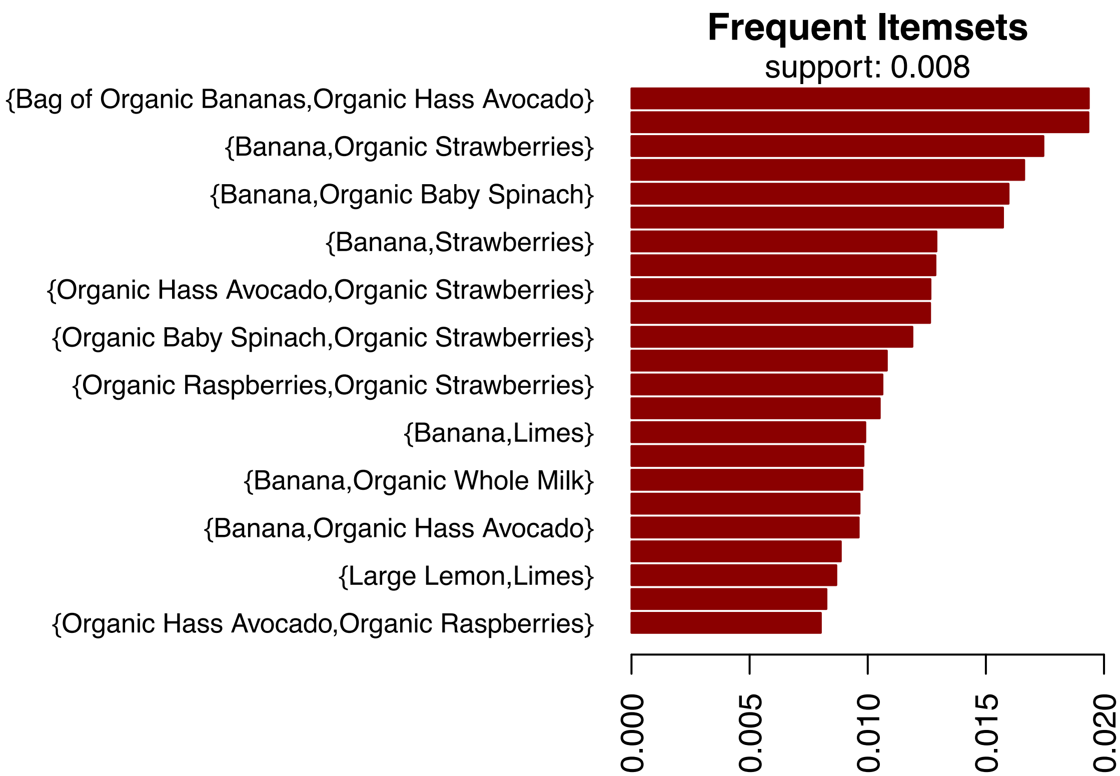

```
#glimpse(item_frequencies)
```

Frequent Itemsets*

Now, lets compute the frequent itemsets. We decrease the support threshold to take into account the small probability of observing a frequent itemset of at least size 2.
With a support threshold of 0.008 (~25k baskets), we observe frequent pairs

```
support <- 0.008
itemsets <- apriori(transactions, parameter = list(target = "frequent itemsets", supp=support, minlen=2

par(mar=c(5,18,2,2)+.1)
sets_order_supp <- DATAFRAME(sort(itemsets, by="support", decreasing = F))
barplot(sets_order_supp$support, names.arg=sets_order_supp$items, xlim=c(0,0.02), horiz = T, las = 2, c
mtext(paste("support:",support), padj = .8)
```



We observe that Bananas/Bag of Organic Bananas are most paired up items! Each of the eight pairs with highest support contains bananas. Nearly all of the items are either fruits or vegetables. There is just one frequent pair that contains milk or spinach.

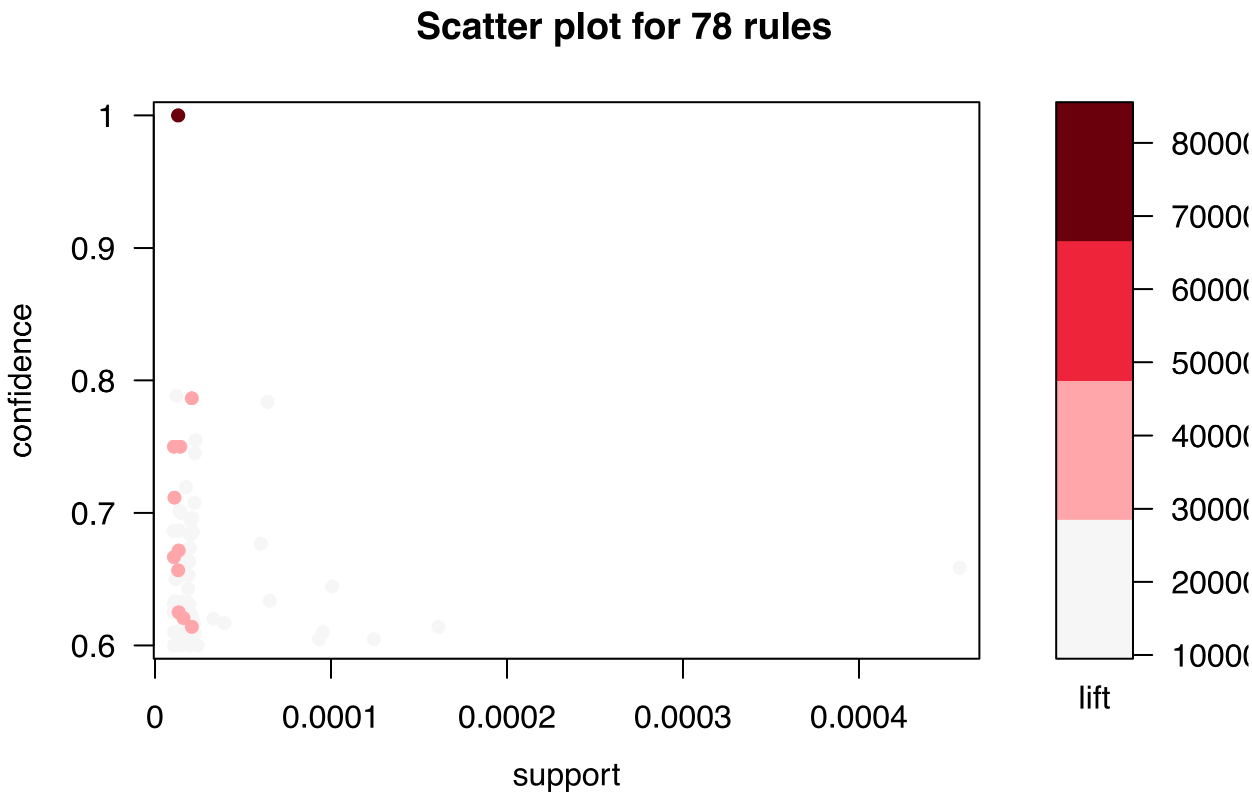
Association Rules

We use a low support threshold and a high confidence to generate strong rules even for items that are less frequent.

```
rules1 <- apriori(transactions, parameter = list(supp = 0.00001, conf = 0.6, maxlen=3), control = list(
summary(quality(rules1))
```

```
##     support             confidence        coverage              lift
##  Min.    :0.00001046   Min.   :0.6000   Min.   :0.00001315   Min.   :    4.26
##  1st Qu.:0.00001345   1st Qu.:0.6167   1st Qu.:0.00002002   1st Qu.:  342.90
##  Median :0.00001928   Median :0.6549   Median :0.00002929   Median : 3486.51
##  Mean   :0.00003084   Mean   :0.6661   Mean   :0.00004741   Mean   : 8952.09
##  3rd Qu.:0.00002114   3rd Qu.:0.6852   3rd Qu.:0.00003258   3rd Qu.:13925.88
##  Max.   :0.00045725   Max.   :1.0000   Max.   :0.00069424   Max.   :76047.34
##      count
##  Min.   :  35.00
##  1st Qu.:  45.00
##  Median :  64.50
##  Mean   : 103.18
##  3rd Qu.:  70.75
##  Max.   :1530.00
```

```
plot(rules1, col=sequential_hcl(4, palette = "Reds 3"),jitter=0)
```



We see some rules with a large lift value ,indicating a strong association between the items. Let's see the top 5 rules by lift.

```
inspect(sort(rules1, by="lift")[1:5])
```

```
##     lhs                                                rhs
## [1] {Moisturizing Facial Wash}                      => {Moisturizing Non-Drying Facial Wash}
## [2] {Moisturizing Non-Drying Facial Wash}           => {Moisturizing Facial Wash}
## [3] {Prepared Meals Simmered Beef Entree Dog Food}  => {Prepared Meals Beef & Chicken Medley Dog Foo
## [4] {Prepared Meals Beef & Chicken Medley Dog Food} => {Prepared Meals Simmered Beef Entree Dog Food
## [5] {Ocean Whitefish}                               => {Premium Classic Chicken Recipe Cat Food}
```

Its odd that we do not see any rules with bananas as expected . As we saw earlier that Bananas were present in top 8 frequest itemsets. Let's see the top 5 rules by confidence.

```
inspect(sort(rules1, by="confidence")[1:5])
```

```
##     lhs                                      rhs                                        support con
## [1] {Moisturizing Facial Wash}            => {Moisturizing Non-Drying Facial Wash} 0.00001314970  1.0
## [2] {Moisturizing Non-Drying Facial Wash} => {Moisturizing Facial Wash}            0.00001314970  1.0
## [3] {Extra Virgin Olive Oil Spray}        => {All-Purpose Unbleached Flour}        0.00001225313  0.7
## [4] {Raspberry Vinaigrette Salad Snax}    => {Thousand Island Salad Snax}          0.00002091998  0.7
## [5] {2nd Foods Turkey Meat}               => {2nd Foods Chicken & Gravy}           0.00006395538  0.7
```

Its odd that, again, we do not see any rules with bananas.

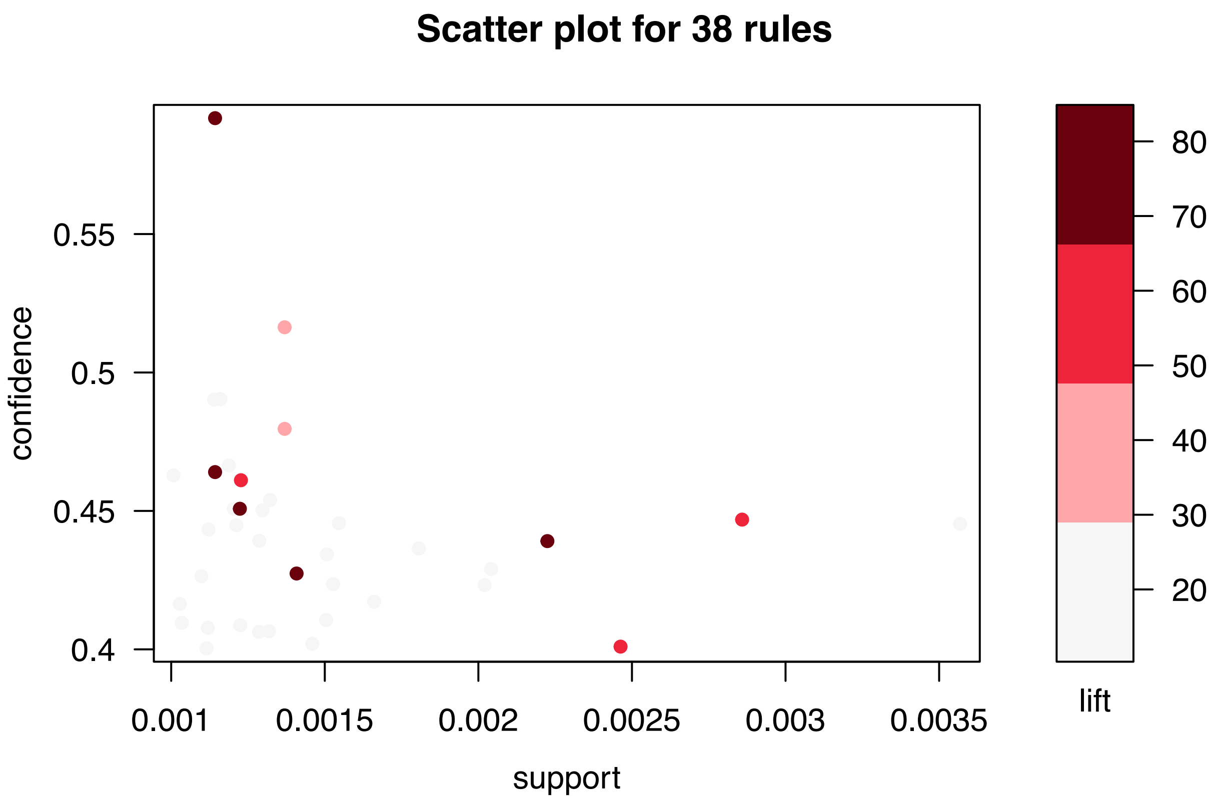
```r
plot(head(sort(rules1 , by="lift"),10), method="graph", control=list(type="items"))
```

```
## Warning: Unknown control parameters: type
```

```
## Available control parameters (with default values):
## main  =  Graph for 10 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot  =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE
```

We will try some other sets of rules: Here Next, we increase the support and decrease confidence to get rules of some more frequent items but with less confidence.

```r
rules2 <- apriori(transactions, parameter = list(supp = 0.001, conf = 0.4, maxlen=3), control = list(ve
plot(rules2, col=sequential_hcl(4, palette = "Reds 3"),jitter=0)
```




```r
inspect(sort(rules2, by="lift")[1:5])
```

```
##     lhs                                                   rhs
## [1] {Non Fat Acai & Mixed Berries Yogurt}              => {Icelandic Style Skyr Blueberry Non-fa
## [2] {Non Fat Raspberry Yogurt}                         => {Icelandic Style Skyr Blueberry Non-fa
## [3] {Total 2% Lowfat Greek Strained Yogurt with Peach,
##      Total 2% with Strawberry Lowfat Greek Strained Yogurt} => {Total 2% Lowfat Greek Strained Yogur
## [4] {Nonfat Icelandic Style Strawberry Yogurt}         => {Icelandic Style Skyr Blueberry Non-fa
## [5] {Total 2% Lowfat Greek Strained Yogurt With Blueberry,
##      Total 2% Lowfat Greek Strained Yogurt with Peach}     => {Total 2% with Strawberry Lowfat Greel
```

checking by confidence

```r
inspect(sort(rules2, by="confidence")[1:5])
```

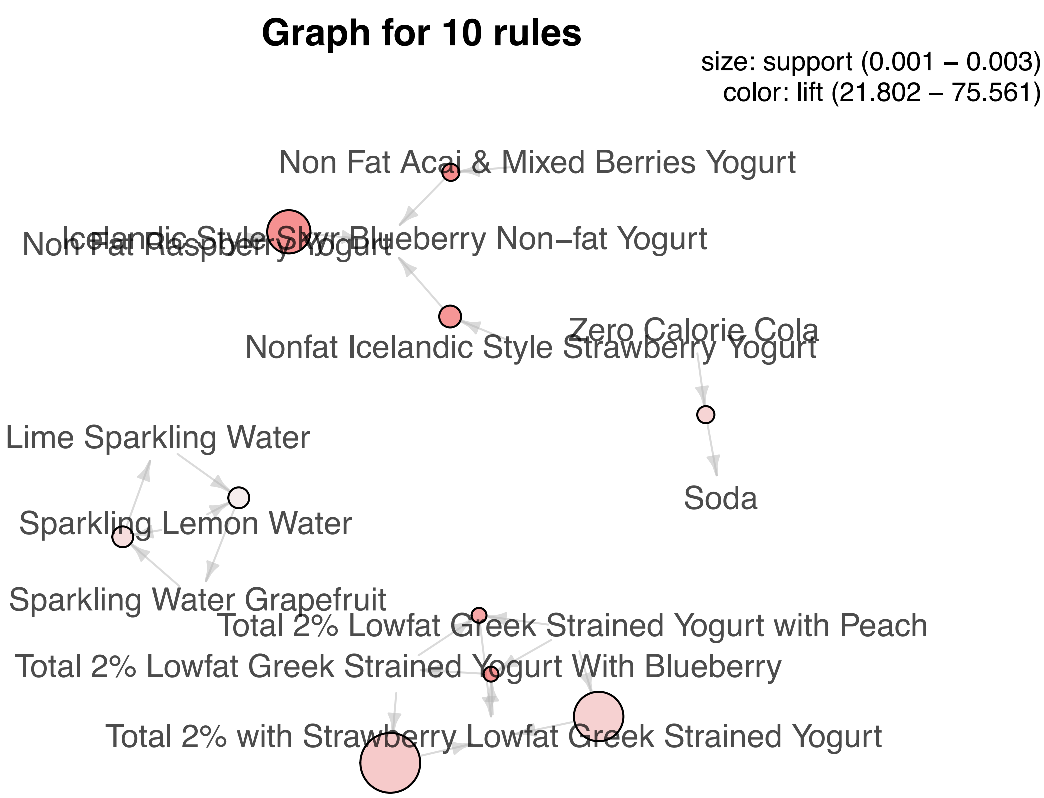
```
##     lhs                                                  rhs
## [1] {Total 2% Lowfat Greek Strained Yogurt With Blueberry,
##      Total 2% Lowfat Greek Strained Yogurt with Peach}    => {Total 2% with Strawberry Lowfat Greek
## [2] {Lime Sparkling Water,
##      Sparkling Lemon Water}                              => {Sparkling Water Grapefruit}
## [3] {Honeycrisp Apple,
##      Strawberries}                                       => {Banana}
## [4] {Organic Fuji Apple,
```

```
##      Strawberries}                                        => {Banana}
## [5] {Sparkling Lemon Water,
##      Sparkling Water Grapefruit}                          => {Lime Sparkling Water}
```

```
plot(head(sort(rules2 , by="lift"),10), method="graph", control=list(type="items"))
```

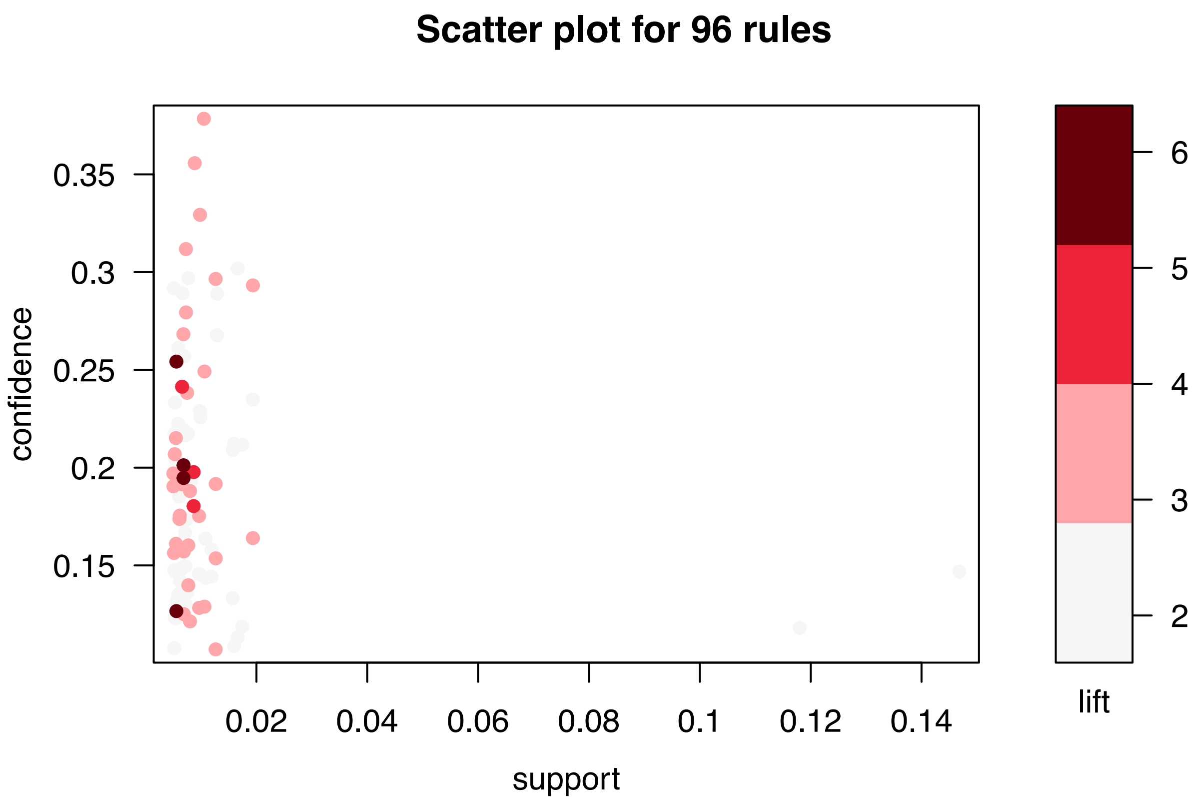
```
## Warning: Unknown control parameters: type

## Available control parameters (with default values):
## main  =  Graph for 10 rules
## nodeColors    =  c("#66CC6680", "#9999CC80")
## nodeCol   =  c("#EE0000FF", "#EE0303FF", "#EE0606FF", "#EE0909FF", "#EE0C0CFF", "#EE0F0FFF", "#EE121
## edgeCol   =  c("#474747FF", "#494949FF", "#4B4B4BFF", "#4D4D4DFF", "#4F4F4FFF", "#515151FF", "#53535
## alpha     =  0.5
## cex   =  1
## itemLabels    =  TRUE
## labelCol  =  #000000B3
## measureLabels     =  FALSE
## precision     =  3
## layout    =  NULL
## layoutParams  =  list()
## arrowSize     =  0.5
## engine    =  igraph
## plot  =  TRUE
## plot_options  =  list()
## max   =  100
## verbose   =  FALSE
```



Finally, lets further increase support and decrease confidence

```
rules3 <- apriori(transactions, parameter = list(supp = 0.005, conf = 0.1, maxlen=3), control = list(ve
plot(rules3, col=sequential_hcl(4, palette = "Reds 3"),jitter=0)
```




```
inspect(sort(rules3, by="lift")[1:5])
```

```
##     lhs                         rhs                     support     confidence
## [1] {Organic Cilantro}      => {Limes}                 0.005550370 0.2542368
## [2] {Limes}                 => {Organic Cilantro}      0.005550370 0.1266330
## [3] {Organic Yellow Onion} => {Organic Garlic}        0.006850697 0.1947314
## [4] {Organic Garlic}        => {Organic Yellow Onion} 0.006850697 0.2011919
## [5] {Limes}                 => {Large Lemon}           0.008666252 0.1977226
##     coverage   lift     count
## [1] 0.02183150 5.800474 18572
## [2] 0.04383035 5.800474 18572
## [3] 0.03518024 5.718889 22923
## [4] 0.03405056 5.718889 22923
## [5] 0.04383035 4.114610 28998
```

```
inspect(sort(rules3, by="confidence")[1:5])
```

```
##     lhs                                        rhs                          support confidence   cove
## [1] {Organic Fuji Apple}                    => {Banana}                 0.010505717  0.3784409 0.0277
## [2] {Honeycrisp Apple}                      => {Banana}                 0.008857820  0.3557249 0.0249
## [3] {Cucumber Kirby}                        => {Banana}                 0.009814461  0.3292957 0.0298
## [4] {Organic Large Extra Fancy Fuji Apple} => {Bag of Organic Bananas} 0.007273280  0.3117890 0.0233
## [5] {Organic Avocado}                       => {Banana}                 0.016619731  0.3018662 0.0550
```

```
#plot(head(sort(rules3 , by="lift"),10), method="graph", control=list(type="items"))
```