Instacart Exploratory Analysis

Mohab Diab

April 2, 2019

Instacart Market Basket Analysis

Which products will an Instacart consumer purchase again? The objective of this Kaggle competition is to use the anonymized data on customer orders over time to predict which previously purchased products will be in a user's next order.

The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. For each user, 4 to 100 of their prior orders are given, with the sequence of products purchased in each order.

The data has been provided in 6 csv files:

-aisles.csv -departments.csv -order_products__prior.csv -order_products__train.csv -orders.csv -products.csv

Let's load some Backages firstly:

```r
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------------------
------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------------------
------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggplot2)
```

Let's load the data and look at the first few rows of the files to understand the data better.

```r
aisles <- read.csv('aisles.csv')
departments <- read.csv('departments.csv')
order_products__prior <- read.csv('order_products__prior.csv')
order_products__train <- read.csv('order_products__train.csv')
orders <- read.csv('orders.csv')
products <- read.csv('products.csv')
```

```r
head(aisles, 1)
```

	aisle_id <int>	aisle <fctr>
1	1	prepared soups salads

1 row

```r
head(aisles, 1)
```

	aisle_id <int>	aisle <fctr>
1	1	prepared soups salads

1 row

```r
head(departments, 1)
```

	department_id <int>	department <fctr>
1	1	frozen

1 row

```r
head(orders, 3)
```

	order_id <int>	user_id <int>	eval_set <fctr>	order_number <int>	order_dow <int>	order_hour_of_day <int>	days_since_prior_order <dbl>
1	2539329	1	prior	1	2	8	NA
2	2398795	1	prior	2	3	7	15
3	473747	1	prior	3	3	12	21

3 rows

```r
head(products, 1)
```

	product_id <int>	product_name <fctr>	aisle_id <int>	department_id <int>
1	1	Chocolate Sandwich Cookies	61	19

1 row

```r
head(order_products__prior, 1)
```

	order_id <int>	product_id <int>	add_to_cart_order <int>	reordered <int>
1	2	33120	1	1

1 row

```r
head(order_products__train, 1)
```

	order_id <int>	product_id <int>	add_to_cart_order <int>	reordered <int>
1	1	49302	1	1

1 row

As we could see, orders.csv has all the information about the given order, like the user who has purchased the order, when was it purchased, days since prior order and so on. The columns present in order_products_train and order_products_prior are same. Then what is the difference between these files?

In this dataset, 4 to 100 orders of a customer are given, and we need to predict the products that will be re-ordered. So the last order of the user has been taken out and divided into train and test sets. All the prior order information of the customer is present in order_products_prior file. We can also note that there is a column in orders.csv file called eval_set which tells us as to which of the three datasets (prior, train or test) the given row goes to. Order_products*csv file has more detailed information about the products that been bought in the given order along with the re-ordered status. The products ordered in the last order of the training set has been provided in the Order_products_train.csv

Let us first get the count of rows in each of the three sets.

```r
orders %>%
        group_by(eval_set) %>%
        summarise(users=n_distinct(user_id))
```

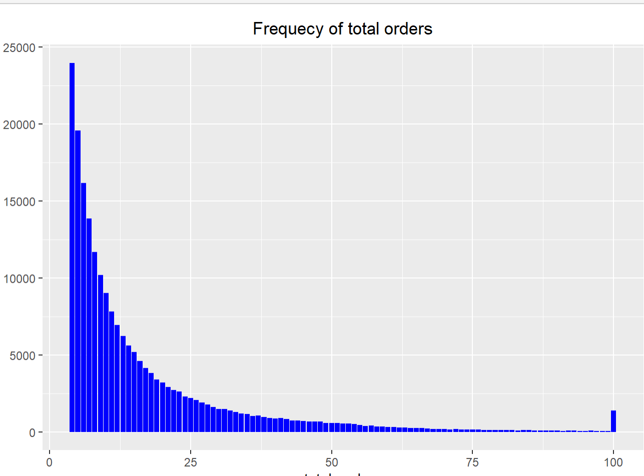
eval_set <fctr>	users <int>
prior	206209
test	75000
train	131209

3 rows

So there are 206,209 customers in total. Out of which, the last purchase of 131,209 customers is given as train set and we need to predict for 75,000 customers belonging to the test set.

Let's validate Number of orders Range:

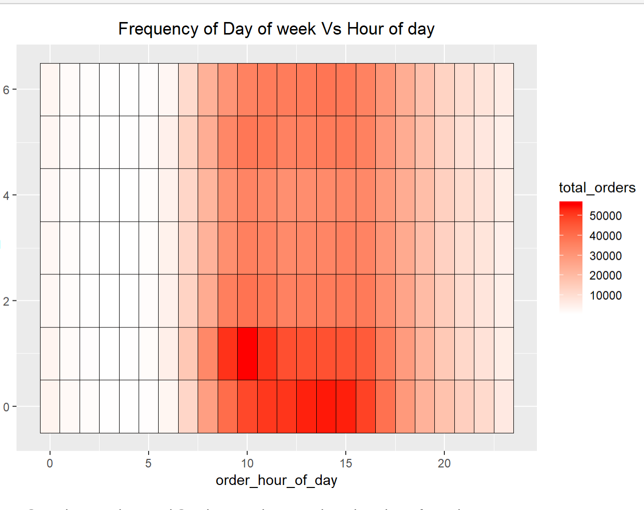
```r
grouped_df <- orders %>%
                group_by(user_id) %>%
                summarise(total_orders= max(order_number))
ggplot(grouped_df, aes(total_orders)) + geom_bar(fill="blue") + ggtitle("Frequecy of total orders") + theme(plo
t.title = element_text(hjust = 0.5))
```



The total number of orders are indeed between 4-100 per customer in a decreasing trend with a spike at 100.

Let's now look at the ordering pattern based on the day of the week and the hour of the day.

```r
df <- orders %>%
                        group_by(order_dow, order_hour_of_day) %>%
                        summarise(total_orders=n())
ggplot(df, aes(order_hour_of_day,order_dow)) + geom_tile(aes(fill = total_orders), colour = "black") + scale_fi
ll_gradient(low = "white",
     high = "red") + ggtitle("Frequency of Day of week Vs Hour of day") + theme(plot.title = element_text(hjust
= 0.5))
```



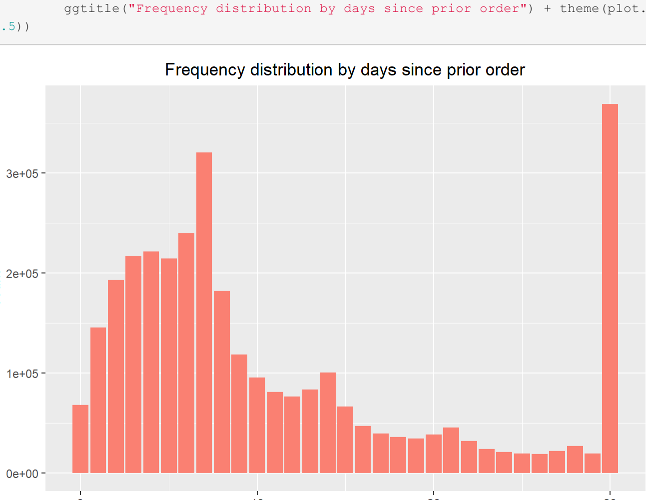
Seems Saturday evenings and Sunday mornings are the prime times for orders.

Now let us check the time interval between the orders.

```r
max(orders$days_since_prior_order, na.rm = T)
```

```
## [1] 30
```

```r
grouped_df <- orders %>% drop_na() %>%
                         group_by(days_since_prior_order) %>%
                         summarise(count=n(), na.rm = T)
ggplot(grouped_df, aes(days_since_prior_order, count)) + geom_bar(stat="identity",fill="salmon") +
        ggtitle("Frequency distribution by days since prior order") + theme(plot.title = element_text(hjust =
0.5))
```



Looks like customers order once in every week (check the peak at 7 days) or once in a month (peak at 30 days). We could also see smaller peaks at 14, 21 and 28 days (weekly intervals).

Since our objective is to figure out the re-orders, *##let us check out the re-order percentage in prior set and train set.*

```r
sum(order_products__prior$reordered)/nrow(order_products__prior)
```

```
## [1] 0.5896975
```

```r
sum(order_products__train$reordered)/nrow(order_products__train)
```

```
## [1] 0.5985944
```

On an average, about 59% of the products in an order are re-ordered products.

Let's now find the percentage of orders with no reordered products.

```r
grouped_df <- order_products__prior %>%
                group_by(order_id) %>%
                summarise(reordered_pr = sum(reordered==1))
sum(grouped_df$reordered_pr==0)/nrow(grouped_df)
```

```
## [1] 0.1208486
```

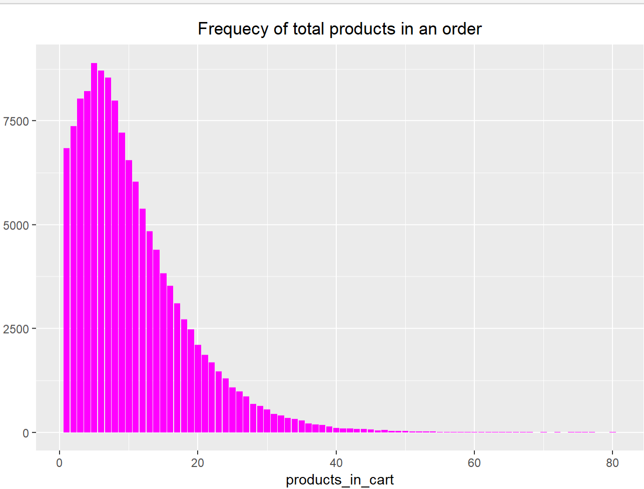
```r
grouped_df <- order_products__train %>%
                group_by(order_id) %>%
                summarise(reordered_tr = sum(reordered==1))
sum(grouped_df$reordered_tr==0)/nrow(grouped_df)
```

```
## [1] 0.06555953
```

About 12% of the orders in prior set have no re-ordered items while in the train set, 6.5% of the orders have no reordered items.

Now let us see the number of products bought in each order.

```r
grouped_df <- order_products__train %>%
                group_by(order_id) %>%
                summarise(products_in_cart = max(add_to_cart_order))
ggplot(grouped_df, aes(products_in_cart)) + geom_bar(fill="magenta") + ggtitle("Frequecy of total products in a
n order") + theme(plot.title = element_text(hjust = 0.5))
```



A right tailed distribution with the maximum value at 5!

Let's now merge the product,aisles and department details with the order_prior details.

What are the top selling products?

```r
orderp_product <- order_products__prior %>%
                 inner_join(products) %>%
                 inner_join(aisles) %>%
                 inner_join(departments)
```

```
## Joining, by = "product_id"
```

```
## Joining, by = "aisle_id"
```

```
## Joining, by = "department_id"
```

```r
rev(sort(table(orderp_product$product_name)))[1:20]
```

```
##
##                 Banana   Bag of Organic Bananas     Organic Strawberries
##                 472565                   379450                   264683
##    Organic Baby Spinach     Organic Hass Avocado          Organic Avocado
##                 241921                   213584                   176815
##             Large Lemon              Strawberries                    Limes
##                 152657                   142951                   140627
##      Organic Whole Milk      Organic Raspberries     Organic Yellow Onion
##                 137905                   137057                   113426
##          Organic Garlic         Organic Zucchini      Organic Blueberries
##                 109778                   104823                   100060
##          Cucumber Kirby       Organic Fuji Apple           Organic Lemon
##                  97315                    89632                    87746
## Apple Honeycrisp Organic   Organic Grape Tomatoes
##                  85020                    84255
```

Most of them are organic products.! Also majority of them are fruits.

Now let us look at the important aisles.

```r
rev(sort(table(orderp_product$aisle)))[1:20]
```

```
##
##                  fresh fruits              fresh vegetables
##                       3642188                       3418021
##    packaged vegetables fruits                        yogurt
##                       1765313                       1452343
##               packaged cheese                          milk
##                        979763                        891015
## water seltzer sparkling water                chips pretzels
##                        841533                        722470
##                soy lactosefree                        bread
##                        638253                        584834
##                  refrigerated                frozen produce
##                        575881                        522654
##                 ice cream ice                      crackers
##                        498425                        458838
##           energy granola bars                          eggs
##                        456386                        452134
##                    lunch meat                  frozen meals
##                        395130                        390299
##            baby food formula                    fresh herbs
##                        382456                        377741
```

Let us now check the department wise distribution.

```r
grouped_df <- orderp_product %>%
                group_by(department) %>%
                summarise(count_percentage = n()/nrow(orderp_product)*100)
grouped_df[rev(order(grouped_df$count_percentage)),]
```

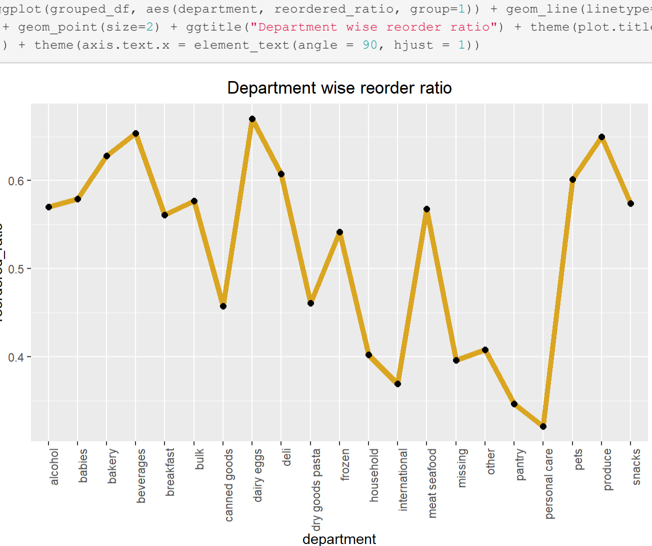
department <fctr>	count_percentage <dbl>
produce	29.2259607
dairy eggs	16.6921575
snacks	8.9027146
beverages	8.2940385
frozen	6.8952281
pantry	5.7826624
bakery	3.6281965
canned goods	3.2929700
deli	3.2411456
dry goods pasta	2.6719305

1-10 of 21 rows   Previous 1 2 3 Next

Produce is the largest department.

Now let us check the reordered percentage of each department.

```r
grouped_df <- orderp_product %>%
                group_by(department) %>%
                summarise(reordered_ratio = sum(reordered)/n())
ggplot(grouped_df, aes(department, reordered_ratio, group=1)) + geom_line(linetype=1, color="goldenrod", size=2
)+ geom_point(size=2) + ggtitle("Department wise reorder ratio") + theme(plot.title = element_text(hjust = 0.5
)) + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Personal care has lowest reorder ratio and dairy eggs have highest reorder ratio.

Aisle - Reorder ratio

```r
grouped_df <- orderp_product %>%
                group_by(aisle) %>%
                summarise(reordered_ratio = sum(reordered)/n())
grouped_df[rev(order(grouped_df$reordered_ratio))[1:10],]
```

aisle <fctr>	reordered_ratio <dbl>
milk	0.7814279
water seltzer sparkling water	0.7295935
fresh fruits	0.7181038
eggs	0.7053661
soy lactosefree	0.6925514
packaged produce	0.6907343
yogurt	0.6864893
cream	0.6850460
bread	0.6701679
refrigerated	0.6633020

1-10 of 10 rows

The aisles for milk, water seltzer sparkling water, fresh fruits and eggs have the highest reorder ratio.

Add to Cart - Reorder ratio:

```r
grouped_df <- orderp_product %>%
                group_by(add_to_cart_order) %>%
                summarise(reordered_ratio = sum(reordered)/n())
ggplot(grouped_df[1:70,], aes(add_to_cart_order, reordered_ratio, group=1)) + geom_line(linetype=1, color="medi
umorchid", size=2)+ geom_point(size=1) + ggtitle("Add to Cart reorder ratio") + theme(plot.title = element_text
(hjust = 0.5)) + theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

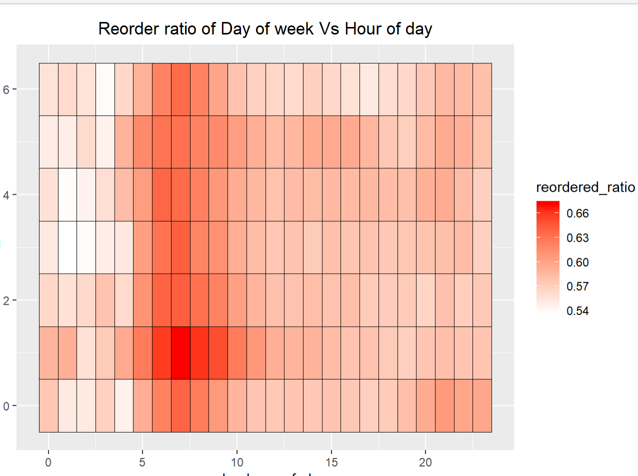
Looks like the products that are added to the cart initially are more likely to be reordered again compared to the ones added later.

and finally let's check reordering intensity per time

```r
orderp_product <- orderp_product %>%
                        inner_join(orders)
```

```
## Joining, by = "order_id"
```

```r
grouped_df <- orderp_product %>%
                group_by(order_dow, order_hour_of_day) %>%
                summarise(reordered_ratio = sum(reordered)/n())
ggplot(grouped_df, aes(order_hour_of_day,order_dow)) + geom_tile(aes(fill = reordered_ratio), colour = "black")
+ scale_fill_gradient(low = "white",
     high = "red") + ggtitle("Reorder ratio of Day of week Vs Hour of day") + theme(plot.title = element_text(h
just = 0.5))
```



Looks like reorder ratios are quite high during the early mornings compared to later half of the day.