Instacart Product Recommender

Jyothirmayee Nagireddy - November 2020

1. The Problem and The Client

“Creating a Grocery Product Recommender for Instacart”

In the retail eCommerce shopping experience product recommendations come in many forms: they may be used to recommend products to add to one’s cart (Amazon’s “Frequently bought together” feature for instance) or they may be used on the checkout page to show customers products they may be interested in based on their total order or which movies they would like to watch based on the movies that they have watched and liked.

Instacart is an online grocery delivery service that allows users to place grocery orders through their website or app which are then fulfilled and delivered by a personal shopper- very similar to Uber Eats but for grocery stores. In 2017 they released a year of their data composed of about 3.3 million orders from about 200,000 customers.

Through the machine learning model and EDA, my best hope is to answer the following question:

- What product will the user add to cart given the user’s purchase history?

Further, recommendations may be more helpful if they are targeted towards a specific segment of customers, rather than made uniformly. For instance, if one group of customers tends to buy a lot of non-dairy milk substitutes and another group tends to buy traditional milk, it may make sense to make different recommendations to go along with that box of Cheerios. To make tailored recommendations, Instacart users must be segmented based on their purchase history using K-Means clustering and then made recommenders based on the product association rules within those clusters.

2. The Data

The dataset is a relational set of files describing customers' orders over time. The goal is to predict which products will be in the user's cart in 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, it is provided between 4 and 100 of their orders, with the sequence of products purchased in each order. We also provide the week and hour of the day the order was placed and a relative measure of time between orders.

Dataset: https://www.kaggle.com/c/instacart-market-basket-analysis/data

Each entity (customer, product, order, aisle, etc.) has an associated unique id. Most of the files and variable names should be self-explanatory.

- aisles.csv
- departments.csv
- order_products__*.csv

  These files specify which products were purchased in each order. order_products__prior.csv contains previous order contents for all customers. 'reordered' indicates that the customer has a previous order that contains the product. Note that some orders will have no reordered items. You may predict an explicit 'None' value for orders with no reordered items.
- orders.csv

  This file tells to which set (prior, train, test) an order belongs. You are predicting reordered items only for the test set orders. 'order_dow' is the day of the week.
- products.csv
- sample_submission.csv

2.1 Data Wrangling

As was mentioned above that the dataset is a relational set of files describing customer's orders over time. As expected, the dataset has no missing entries. Data wrangling on each of the files produced the following results:

- **Aisles**:

  There are 134 aisles. This table comprises 2 columns - unique aisle id and its corresponding aisle name.
- **Departments:**

  Every aisle is further grouped into 21 distinct departments. There is a department named "missing". Which is to be further examined later as to what products are in that department. There are departments such as frozen, bakery, dairy eggs, canned goods, personal care, produce, alcohol, etc.,.
- **Order_products__prior and order_products__train**

Order_products__prior is the largest file among the existing files. It comprises 32434489 rows × 4 columns. And order_products__train contains 1384617 rows × 4 columns. These two files have the same column names.

The difference between the two files being that the prior file comprises the prior data of customer's orders whereas the order_products__train file corresponds to the current customer's purchase.

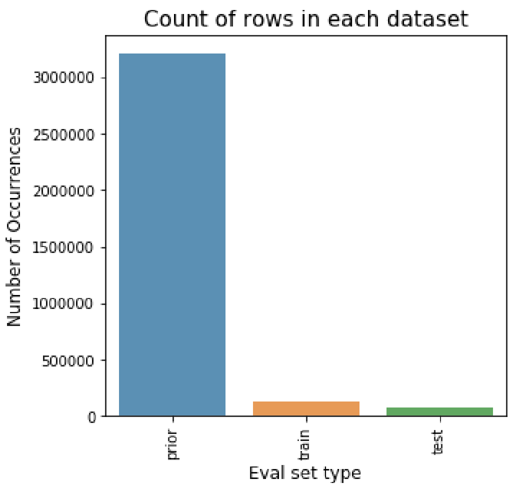
The 4 columns for the order_products__* tables are:

- Order_id
- Product_id
- Add_to_cart_order
- Reordered (1 for yes and 0 for no)

- **Orders**

  Orders file is comprised of 3421083 rows × 7 columns. The 7 columns are:

  - Order_id
  - User_id
  - Eval_set
  - Order_no
  - Order_dow (day of the week)
  - Order_hour_of_day
  - Days_since_prior_order



The Eval_set has three outcomes - prior, train, or test. The prior and train discussed earlier. The test outcome corresponds to the data for which the recommendation has to be made. There are 206,209 customers in total. Out of which, the last purchase of 131,209 customers is given a train set and we need to predict for the rest 75,000 customers.

2.2 Data Visualization and EDA

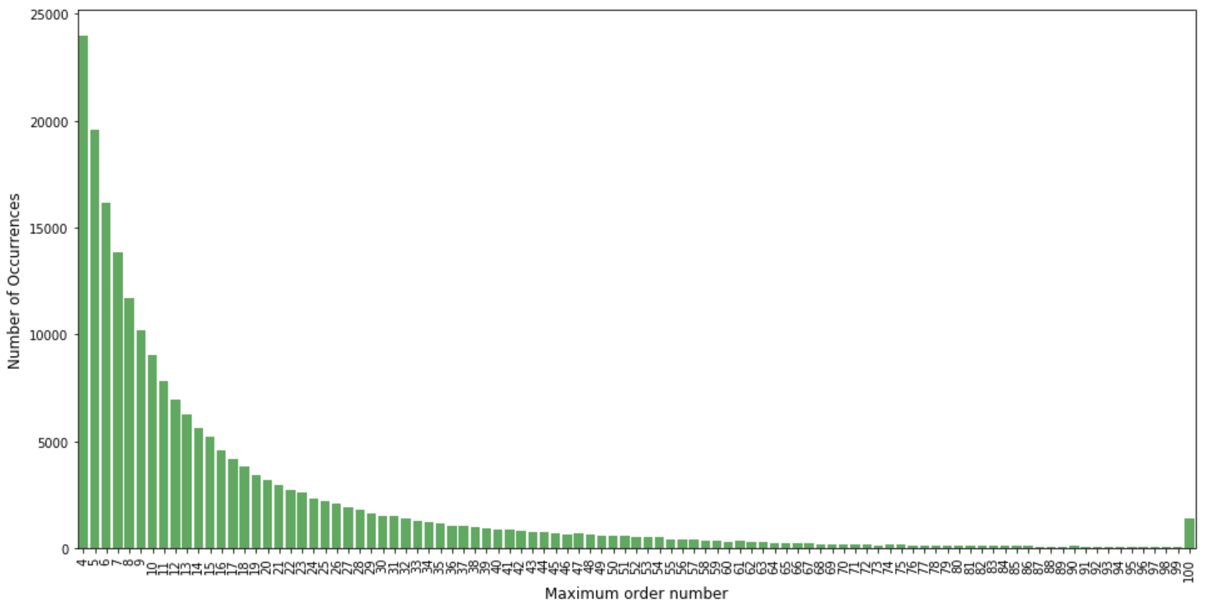
The main focus for Data Visualization and Exploratory Data Analysis is for the files

- Orders
- Order_products__*

After individually analyzing both the files, all the files are merged on order_id, user_id, product_id, and aisle_id for the sake of further analysis.

2.2.1 No. of orders by each customer:

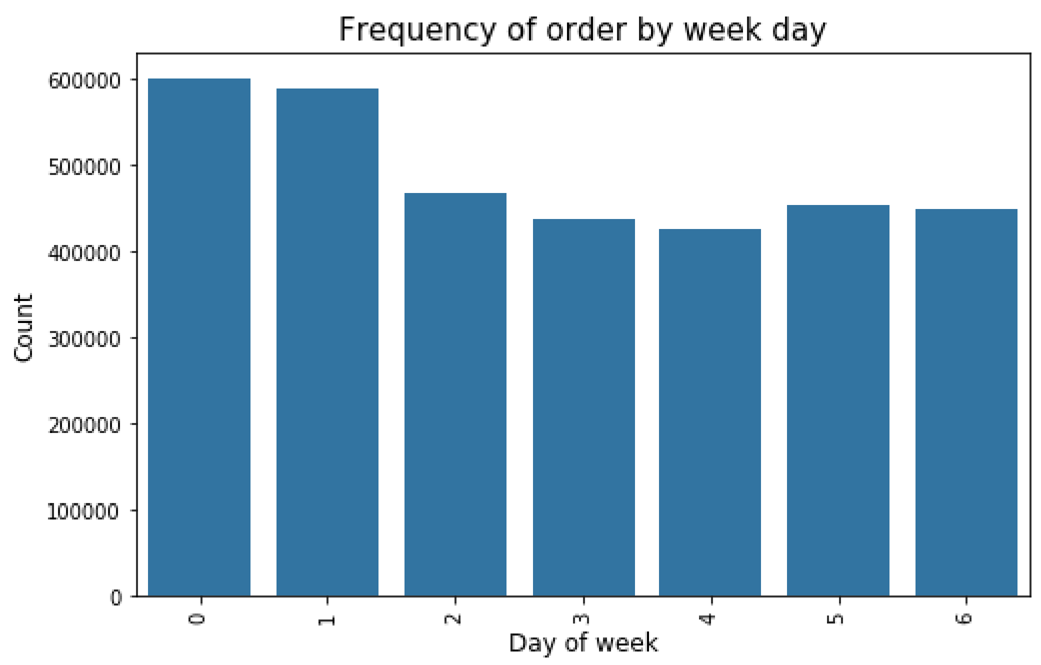
In the introduction of the dataset, it was mentioned earlier that order information of each customer was provided and that each customer made between 4 to 100 orders.



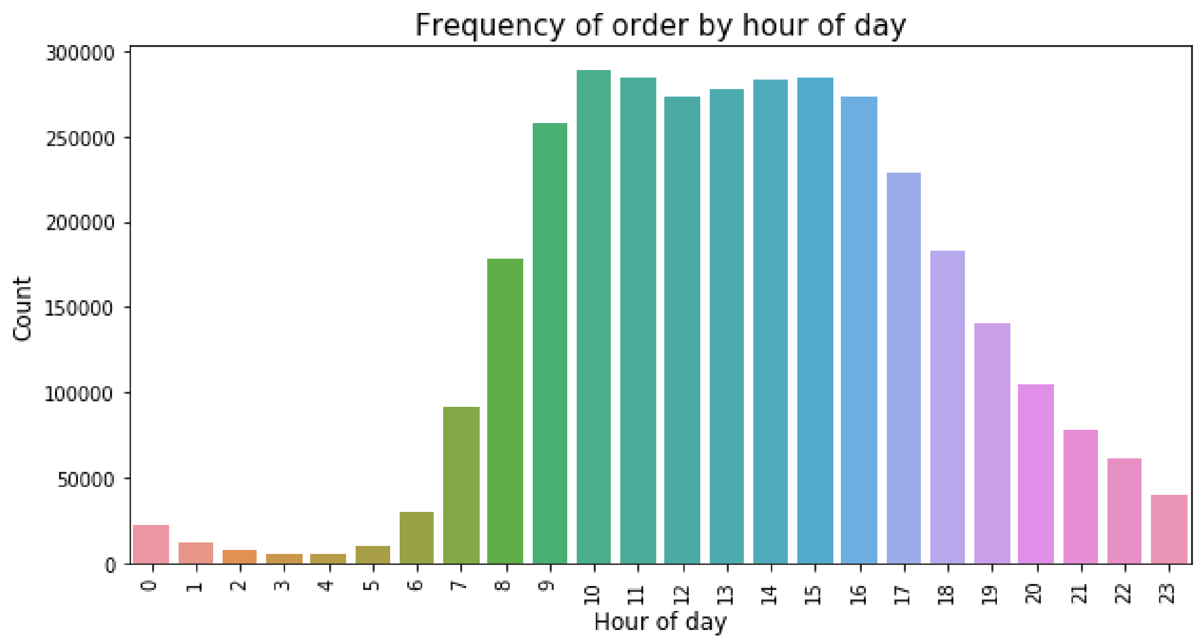
So there are no orders less than 4 and the maximum is capped at 100 as given in the introduction.

2.2.2 Ordering habit changes with day of the week:

The basic countplot of the orders vs their respective day of the week is given below. The orders are usually higher on Saturday and Sunday. And the lowest on Wednesday.

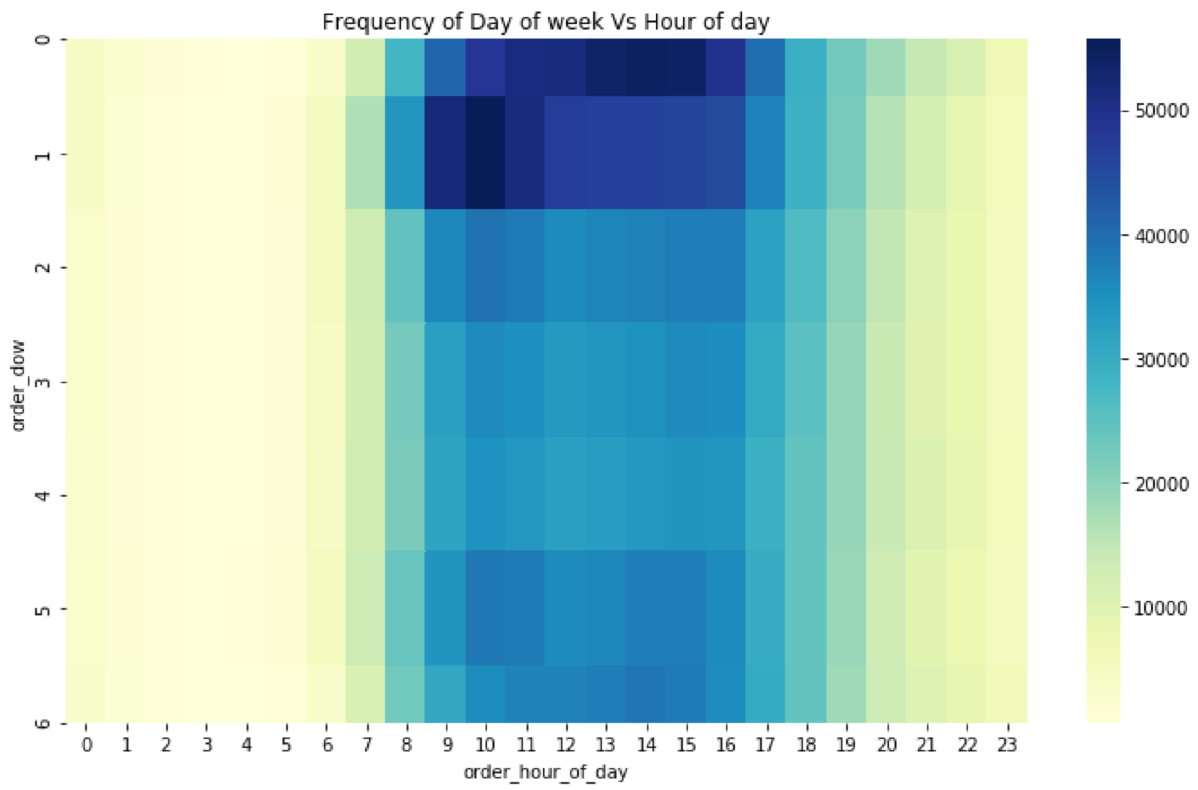


2.2.3 Order Distribution w.r.t time of day:



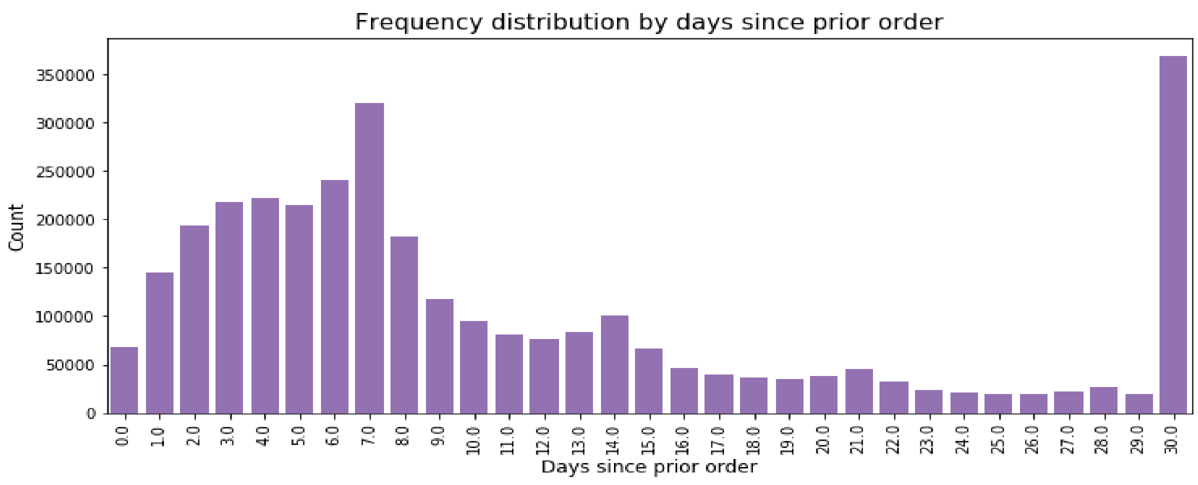
It can be inferred from the above barplot that most of the orders happen during the day time.

2.2.4 Order Distribution - Time of day vs. day of the week:



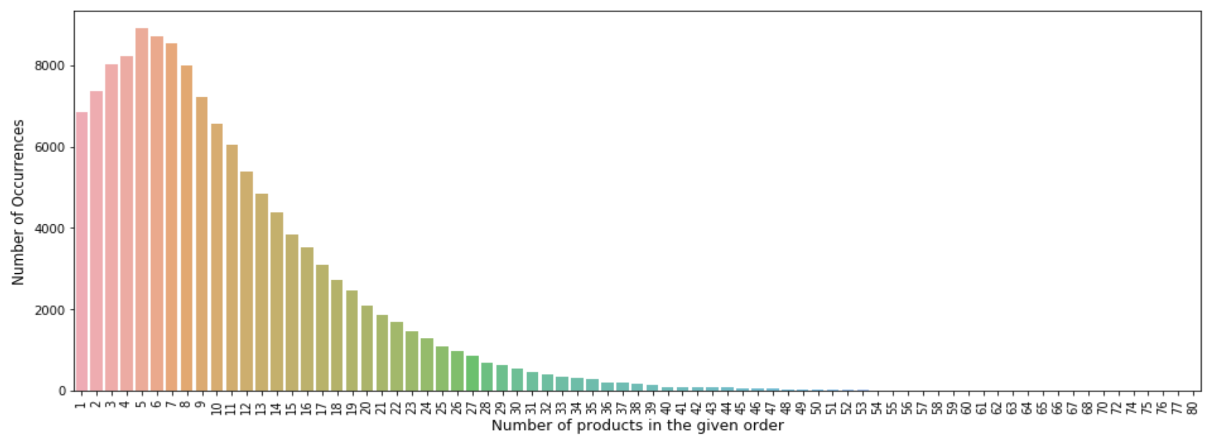
The inference we got from the two plots we plotted individually with Time of day and day of the week are combined to make a heatmap. From the heatmap, it can clearly be understood that Saturday evenings and Sunday mornings are prime time for orders.

2.2.5 The time interval between the orders:



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

2.2.6 Number of products bought in each order:



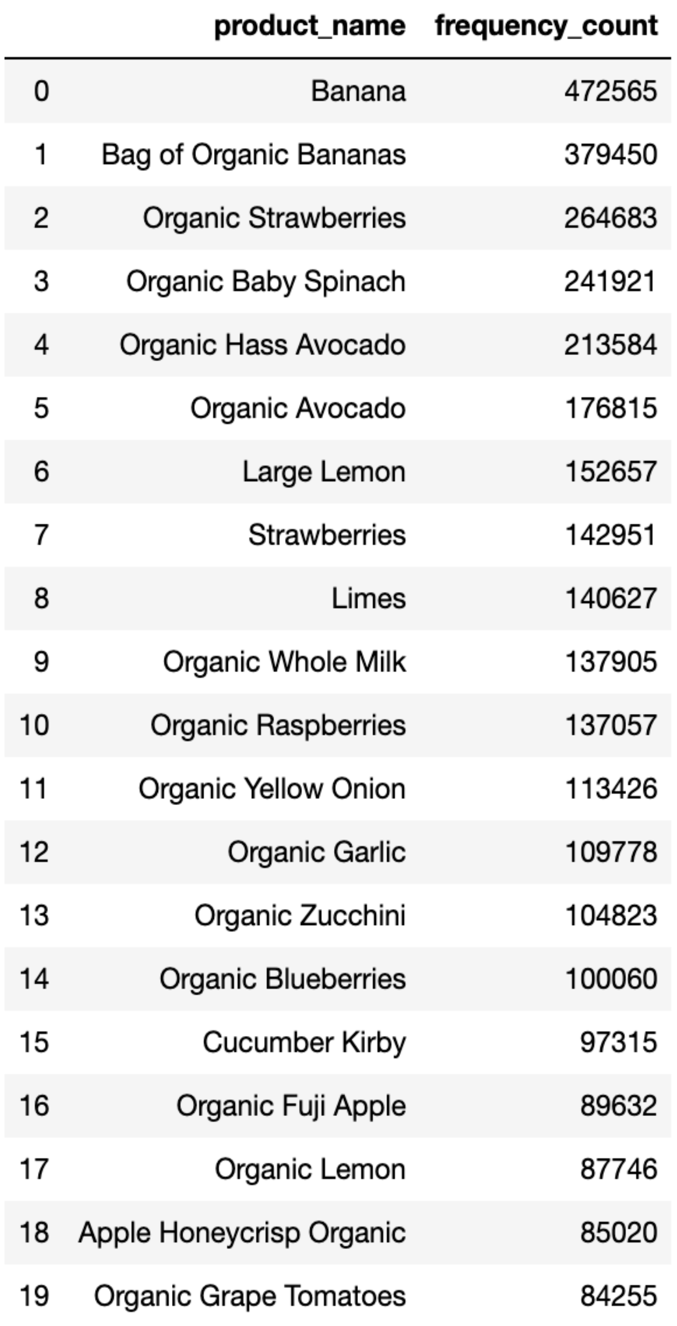
There is a peak of 5. But most customers ordered anywhere between 1 and 15 products per order. It is interesting to note that the max number of products per order is 80. Although small, there is a sizable no of customers for whom the number of products per order is greater than 40.

2.2.7 Merging files:

The order_products__prior is merged with the files- products, aisles, and departments. This is done with the help of the left merge by aligning product_id, aisle_id, and department_id.

	order_id	product_id	add_to_cart_order	reordered	product_name	aisle_id	department_id	aisle	department
0	2	33120	1	1	Organic Egg Whites	86	16	eggs	dairy eggs
1	2	28985	2	1	Michigan Organic Kale	83	4	fresh vegetables	produce
2	2	9327	3	0	Garlic Powder	104	13	spices seasonings	pantry
3	2	45918	4	1	Coconut Butter	19	13	oils vinegars	pantry
4	2	30035	5	0	Natural Sweetener	17	13	baking ingredients	pantry

The top 20 products that were bought after repeatedly is then extracted from the merged file with the help of value_counts().

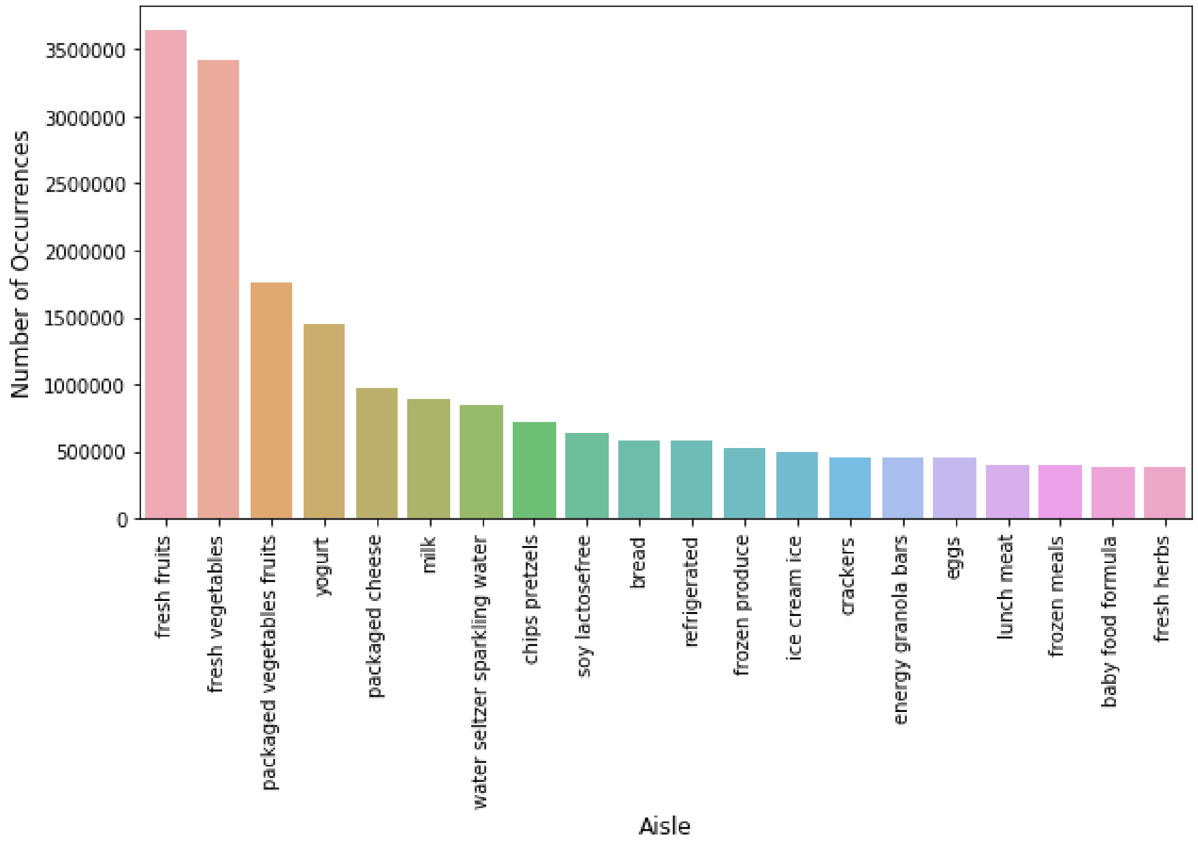


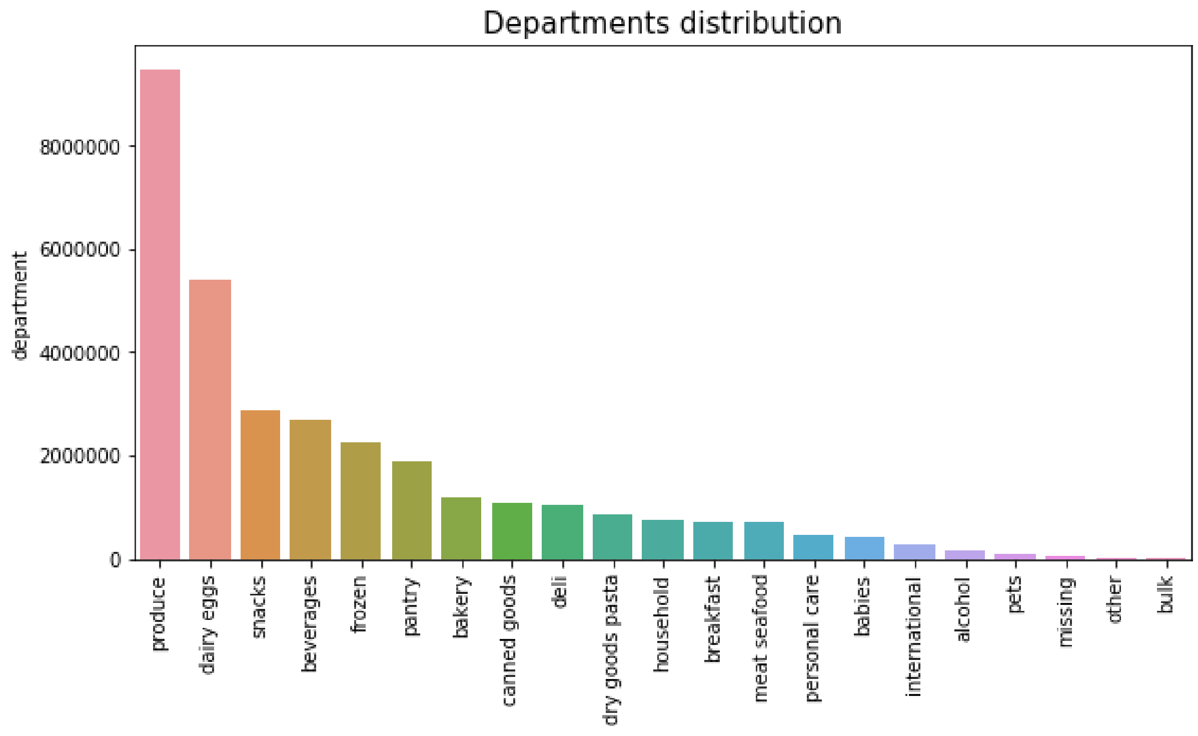
	product_name	frequency_count
0	Banana	472565
1	Bag of Organic Bananas	379450
2	Organic Strawberries	264683
3	Organic Baby Spinach	241921
4	Organic Hass Avocado	213584
5	Organic Avocado	176815
6	Large Lemon	152657
7	Strawberries	142951
8	Limes	140627
9	Organic Whole Milk	137905
10	Organic Raspberries	137057
11	Organic Yellow Onion	113426
12	Organic Garlic	109778
13	Organic Zucchini	104823
14	Organic Blueberries	100060
15	Cucumber Kirby	97315
16	Organic Fuji Apple	89632
17	Organic Lemon	87746
18	Apple Honeycrisp Organic	85020
19	Organic Grape Tomatoes	84255

It is interesting to note that most of the products are organic and the majority of them are fruits.

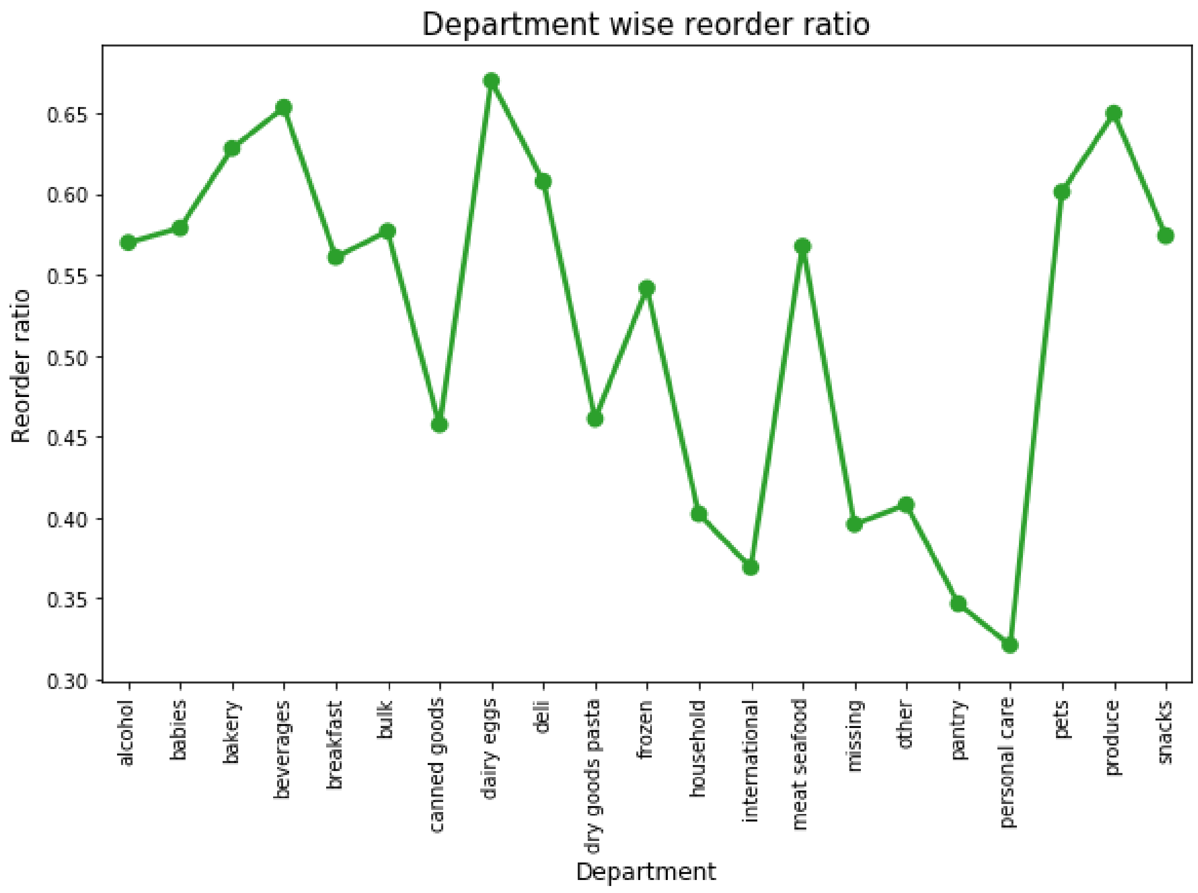
2.2.8 Important Aisles and Departments:

The most important aisles are the aisles that contain the produce (the top department)- fresh fruits, fresh vegetables, packaged vegetable fruits. And then comes dairy products. The plot describing the top 20 aisles and departments:





2.2.9 Re-order ratio of each department:



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

2.3 Conclusion from EDA

The prime time for the orders is Saturday night and Sunday morning with the highest number of orders observed during Saturday night and the lowest is during the nights. The lowest number of orders during the day time is also observed on a Wednesday.

The customers usually order once a week. But it can also be noted that that there is a weekly(also bi-weekly, tri-weekly) and Monthly trend observed from the data.

The usual order for most customers is fruits and vegetables, dairy products. And the re-order ratio follows the same trend. Another important find is that customers prefer organic items.

3. K-Means Clustering and PCA

This section deals with data that is available in two tables: 1. Order products in the prior and 2. Order products in the training data set. The goal is to split the data into an appropriate number of clusters that fit the data and understand which cluster each user belongs to. Based on the knowledge of which cluster each user belongs to, the data is further analyzed to recommend a product to a given customer in the test set.

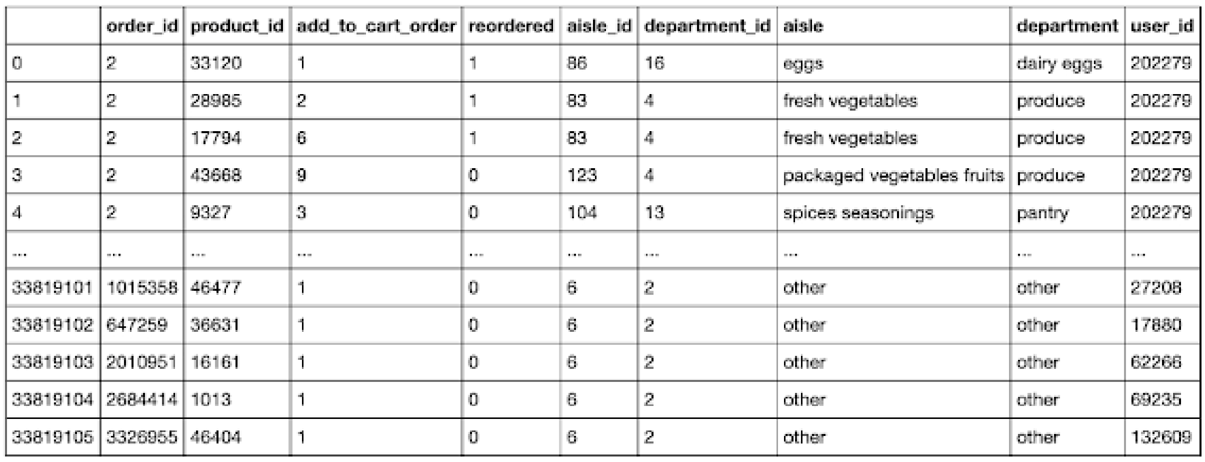
This section primarily deals with-

- Merging all the tables
- Preparing Data for further analysis
- PCA analysis to perform feature reduction
- Heatmap for aisle and department distribution for each cluster
- Top aisle distributions for each cluster

3.1 Merging Tables

The tables prior and train have a similar structure. Each row represents a product ordered. Hence it has an order_id representing the order number, product_id, add_to_cart_order talks about the order in which products were added in a particular order_id. Finally, it has a column called 'reordered' which says if the product was reordered by a user again. These two tables prior and train are concatenated such that these two share the same columns.

The resulting table is further merged with the products table using the product_id. The products table has two more columns called aisle_id and department_id. Using these two ids, the resulting table is merged with the tables "aisles" and "department" to obtain the aisle name and department name.



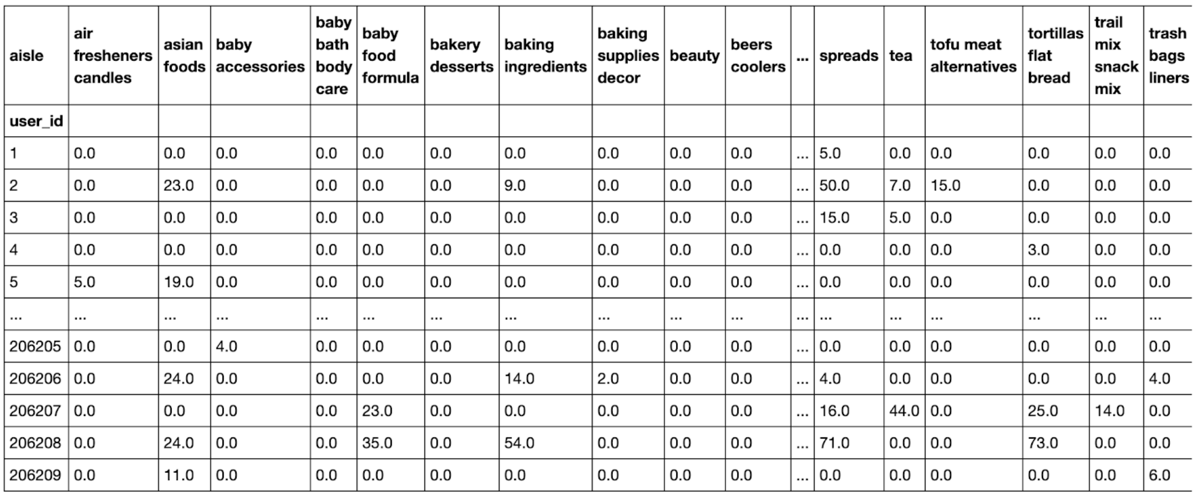
	order_id	product_id	add_to_cart_order	reordered	aisle_id	department_id	aisle	department	user_id
0	2	33120	1	1	86	16	eggs	dairy eggs	202279
1	2	28985	2	1	83	4	fresh vegetables	produce	202279
2	2	17794	6	1	83	4	fresh vegetables	produce	202279
3	2	43668	9	0	123	4	packaged vegetables fruits	produce	202279
4	2	9327	3	0	104	13	spices seasonings	pantry	202279
...	...	...	...	...	...	...	...	...	...
33819101	1015358	46477	1	0	6	2	other	other	27208
33819102	647259	36631	1	0	6	2	other	other	17880
33819103	2010951	16161	1	0	6	2	other	other	62266
33819104	2684414	1013	1	0	6	2	other	other	69235
33819105	3326955	46404	1	0	6	2	other	other	132609

33819106 rows × 9 columns

Now, this table is merged with the columns “order_id” and “user_id” from the orders table. The resulting table does not contain other columns such as order day of the week, order time of day, days since prior order, and order number. These columns were already analyzed during the exploratory data analysis. But these columns are not required for K-means clustering.

3.2 Preparing Data for Further Analysis:

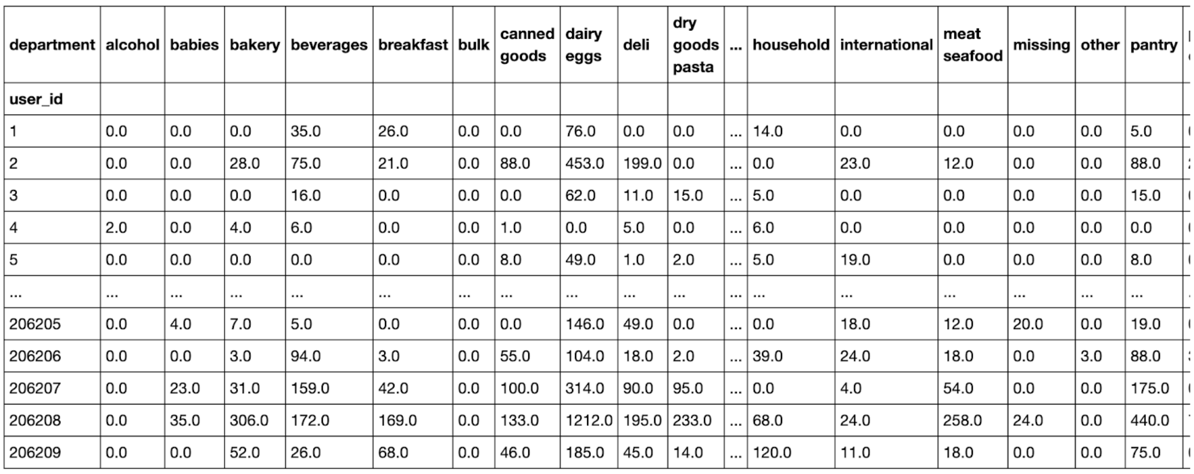
For the sake of PCA and K-Means clustering, the data has to be in the form of a pivot table. The goal of K-Means clustering is to determine which cluster a particular user falls into based on their previous purchasing history. Each user has a certain set of aisles that they generally prefer to buy from. Based on their purchasing history from different aisles, they are clustered. Hence the data that is given as an input to the PCA is the pivot table with users in each row, different aisles in each column, and the total number of products bought i.e. add to cart orders as the values.



aisle	air fresheners candles	asian foods	baby accessories	baby bath body care	baby food formula	bakery desserts	baking ingredients	baking supplies decor	beauty	beers coolers	...	spreads	tea	tofu meat alternatives	tortillas flat bread	trail mix snack mix	trash bags liners
user_id																	
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	5.0	0.0	0.0	0.0	0.0	0.0
2	0.0	23.0	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0	...	50.0	7.0	15.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	15.0	5.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	3.0	0.0	0.0
5	5.0	19.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
206205	0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0
206206	0.0	24.0	0.0	0.0	0.0	0.0	14.0	2.0	0.0	0.0	...	4.0	0.0	0.0	0.0	0.0	4.0
206207	0.0	0.0	0.0	0.0	23.0	0.0	0.0	0.0	0.0	0.0	...	16.0	44.0	0.0	25.0	14.0	0.0
206208	0.0	24.0	0.0	0.0	35.0	0.0	54.0	0.0	0.0	0.0	...	71.0	0.0	0.0	73.0	0.0	0.0
206209	0.0	11.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	6.0

206209 rows × 134 columns

The same process is repeated on the department’s data to understand the data.

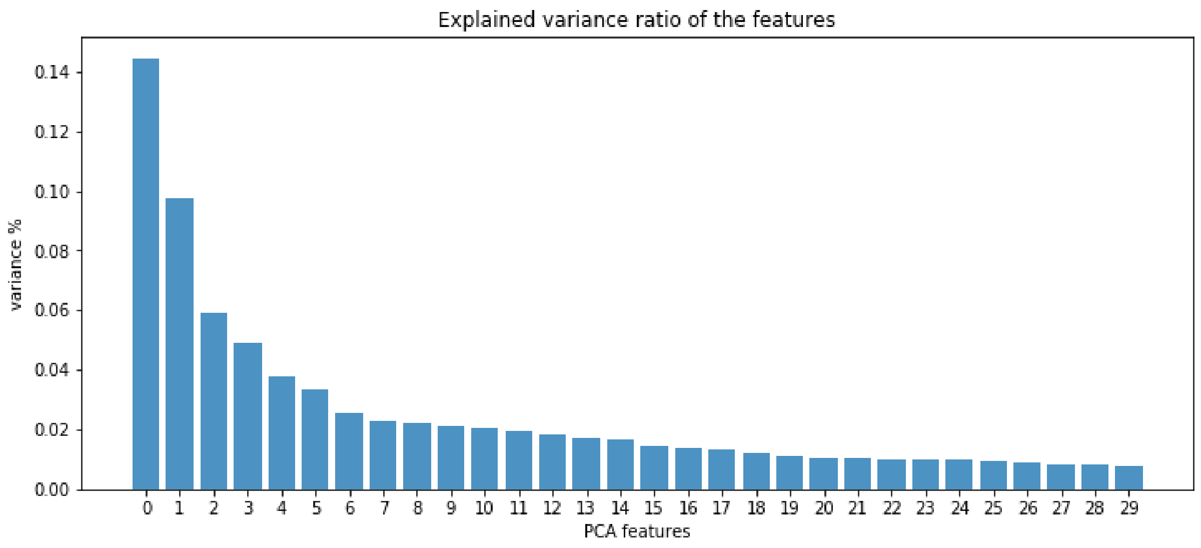


department	alcohol	babies	bakery	beverages	breakfast	bulk	canned goods	dairy eggs	deli	dry goods pasta	...	household	international	meat seafood	missing	other	pantry
user_id																	
1	0.0	0.0	0.0	35.0	26.0	0.0	0.0	76.0	0.0	0.0	...	14.0	0.0	0.0	0.0	0.0	5.0
2	0.0	0.0	28.0	75.0	21.0	0.0	88.0	453.0	199.0	0.0	...	0.0	23.0	12.0	0.0	0.0	88.0
3	0.0	0.0	0.0	16.0	0.0	0.0	0.0	62.0	11.0	15.0	...	5.0	0.0	0.0	0.0	0.0	15.0
4	2.0	0.0	4.0	6.0	0.0	0.0	1.0	0.0	5.0	0.0	...	6.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	0.0	8.0	49.0	1.0	2.0	...	5.0	19.0	0.0	0.0	0.0	8.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
206205	0.0	4.0	7.0	5.0	0.0	0.0	0.0	146.0	49.0	0.0	...	0.0	18.0	12.0	20.0	0.0	19.0
206206	0.0	0.0	3.0	94.0	3.0	0.0	55.0	104.0	18.0	2.0	...	39.0	24.0	18.0	0.0	3.0	88.0
206207	0.0	23.0	31.0	159.0	42.0	0.0	100.0	314.0	90.0	95.0	...	0.0	4.0	54.0	0.0	0.0	175.0
206208	0.0	35.0	306.0	172.0	169.0	0.0	133.0	1212.0	195.0	233.0	...	68.0	24.0	258.0	24.0	0.0	440.0
206209	0.0	0.0	52.0	26.0	68.0	0.0	46.0	185.0	45.0	14.0	...	120.0	11.0	18.0	0.0	0.0	75.0

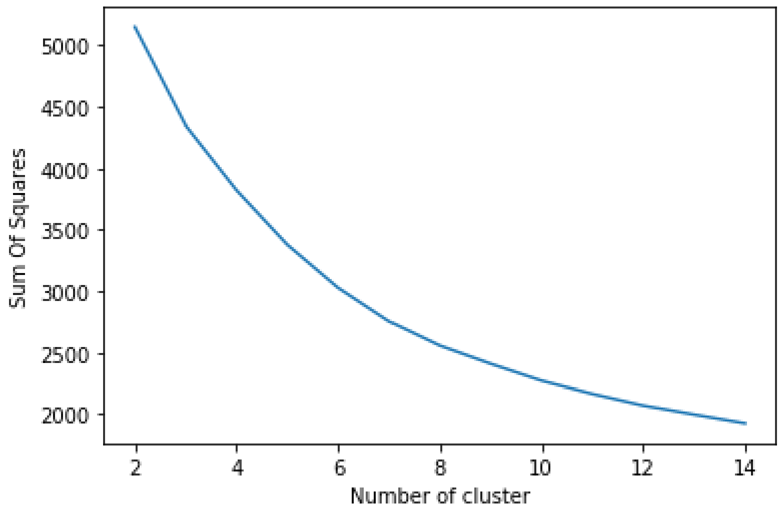
206209 rows × 21 columns

3.3 Principal Component Analysis for Feature Reduction:

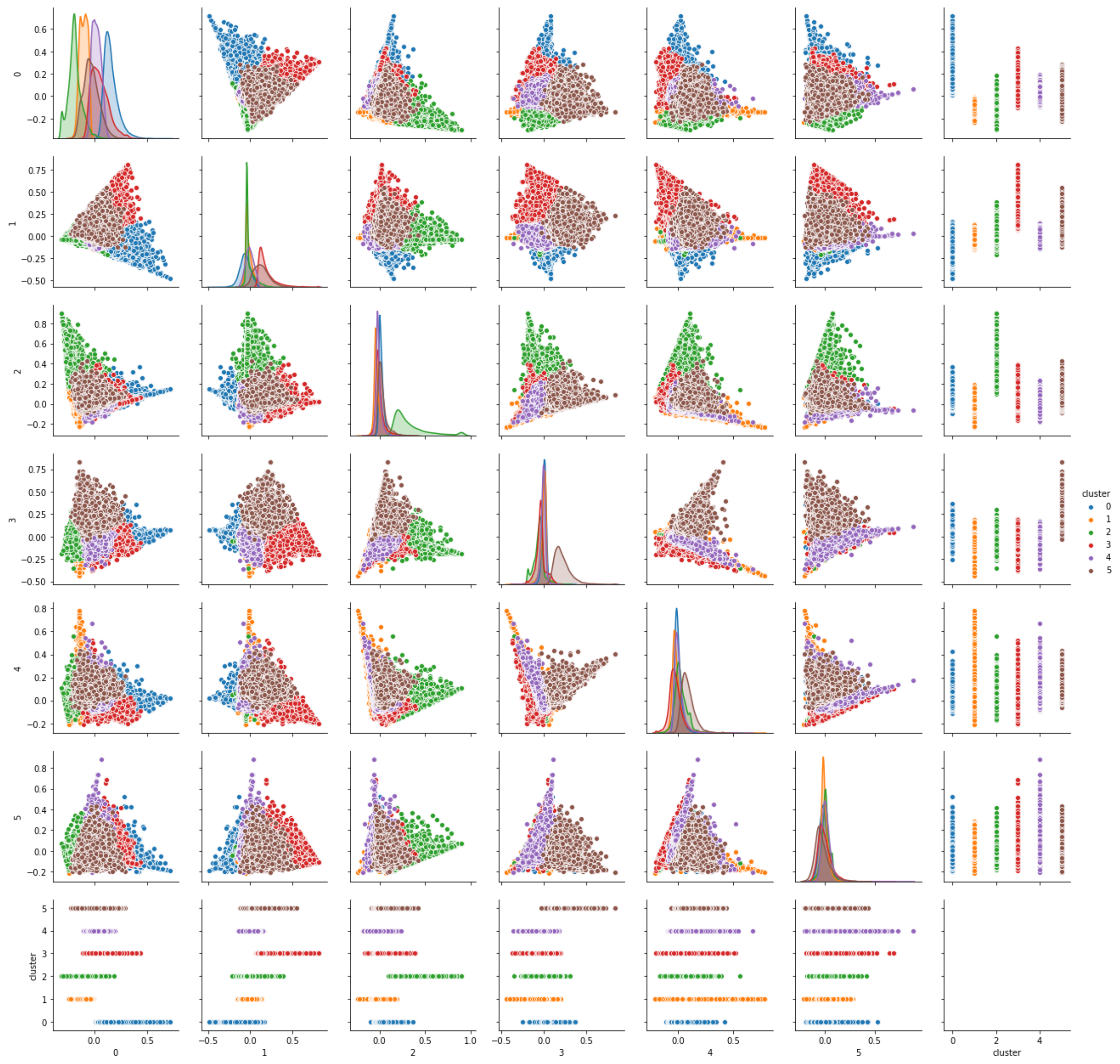
Principal Component Analysis (PCA) is one of the most efficient ways of feature reduction by understanding the explained variance ratio of each feature. The number of features, data frames, in this case, would be 6 ie., 0, 1, 2, 3, 4, 5. After feature no. 5, the explained variance ratio is very low.



The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. It seems that the curve starts to flatten at cluster 5 so I moved forward with 6 clusters.



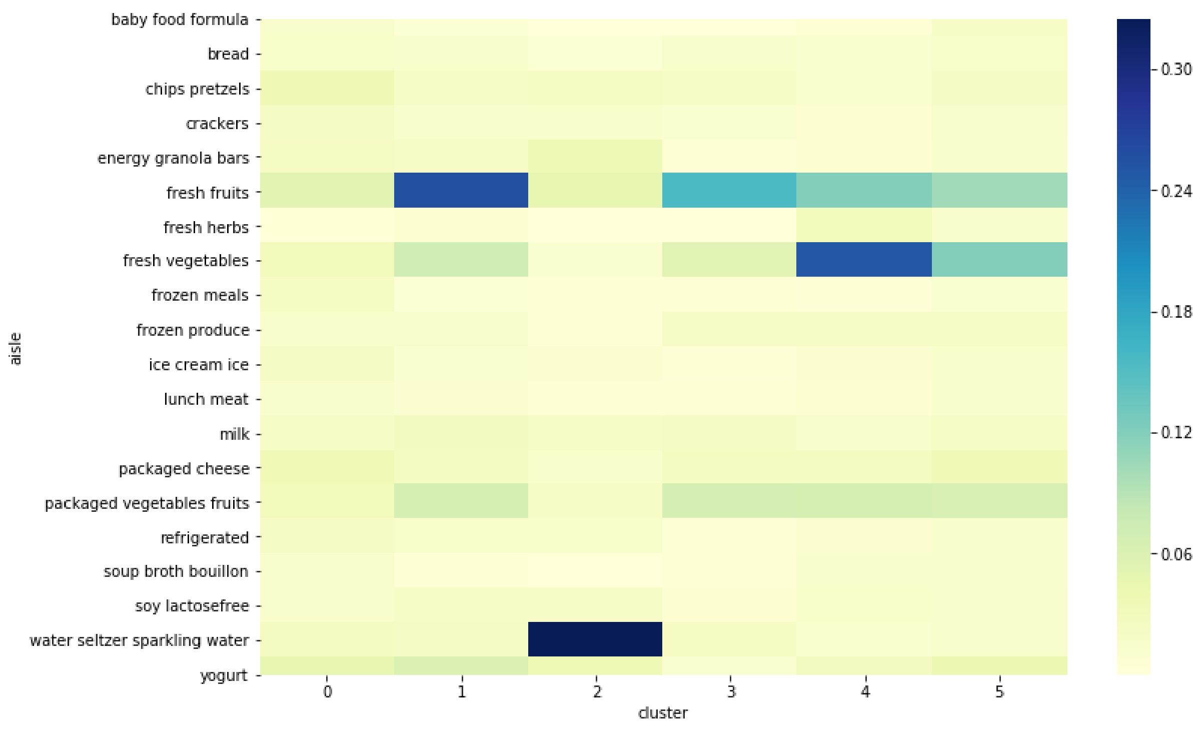
Here are those clusters plotted on a scatterplot matrix of the 6 PCA components:



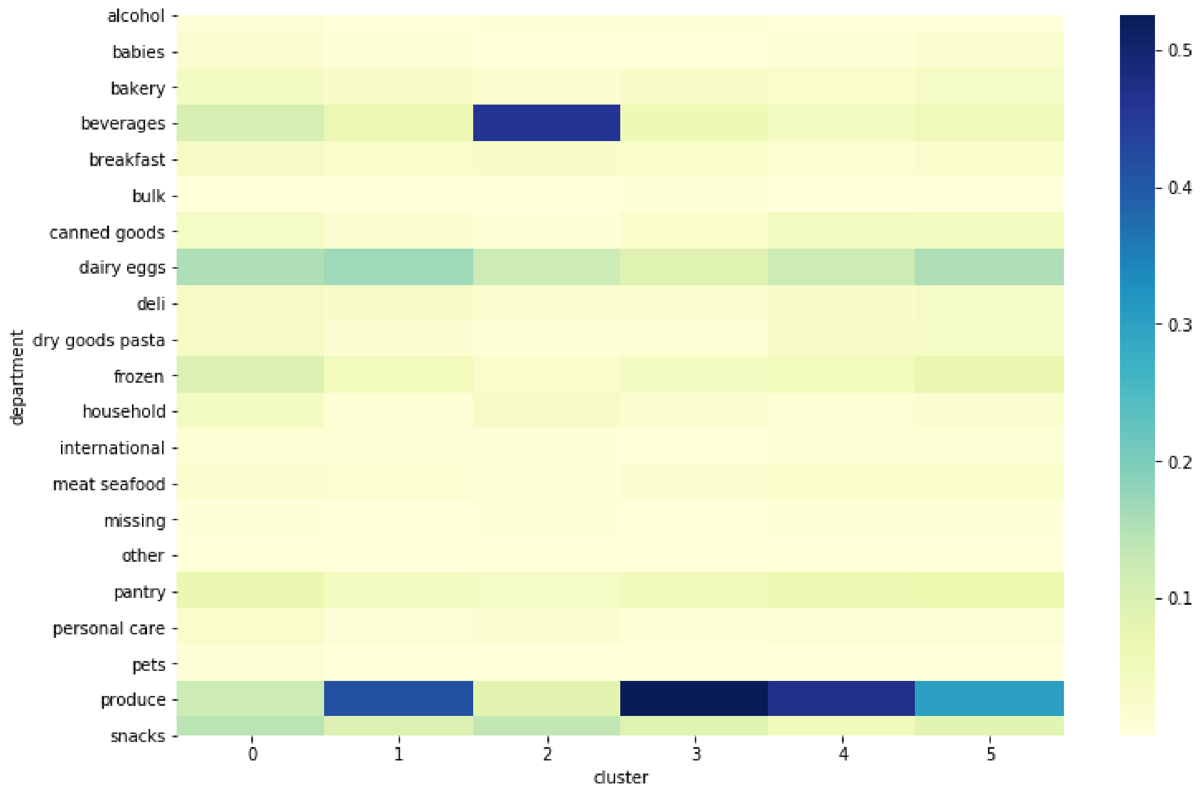
3.4 Heatmap for aisle and department distribution

Earlier, while preparing the data for PCA and K-Means clustering, two different data frames were extracted from the merged data frame, one for the aisles and one for department distribution. The aisle share of each user for an aisle is calculated by getting the sum of add to cart orders for each aisle of a particular user and dividing it by total add to cart orders.

Though not perfect, it does seem that I have identified 6 distinct groups that should result in differences in the aisle purchase history. The best way to check this of course is to look at each cluster's aisle share for unit purchases. Below is a heatmap for a share of purchases by aisle for the top 20 Instacart aisles:

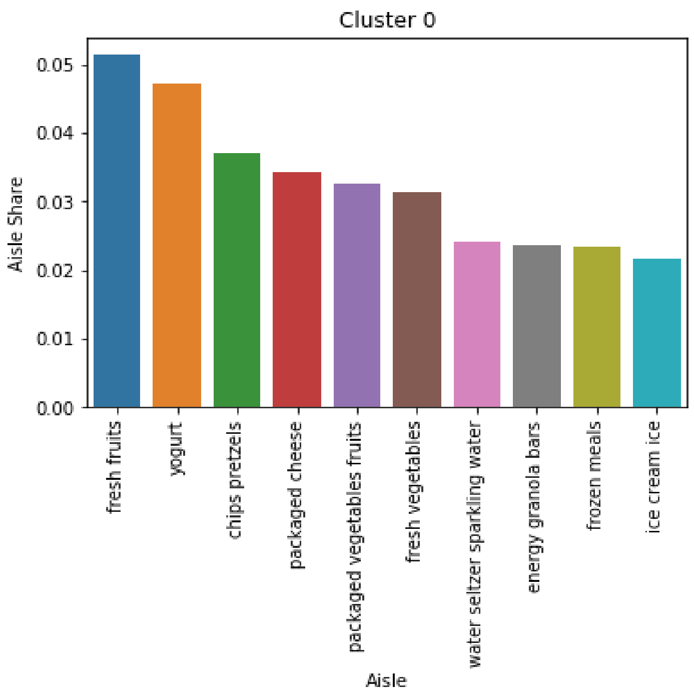


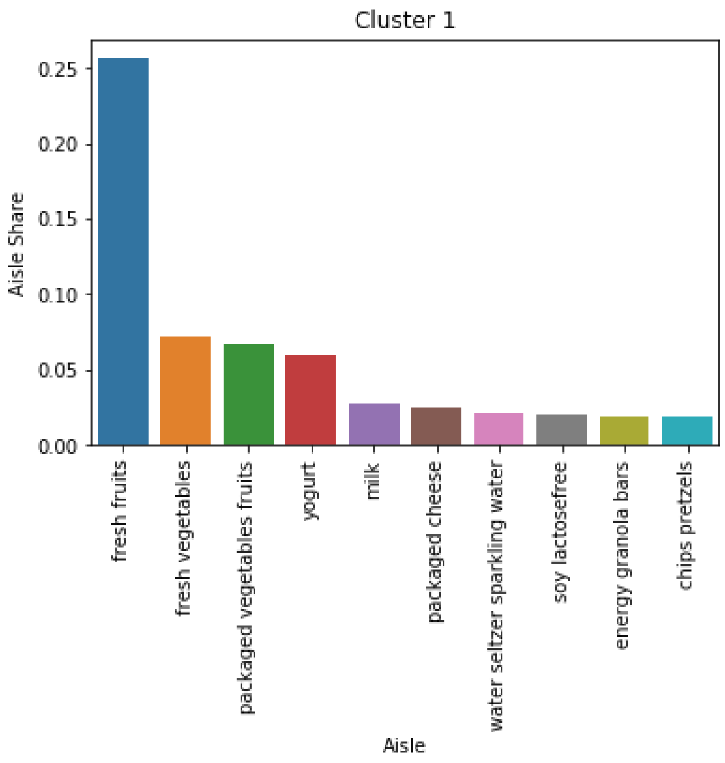
This process is repeated to do the same for departments.

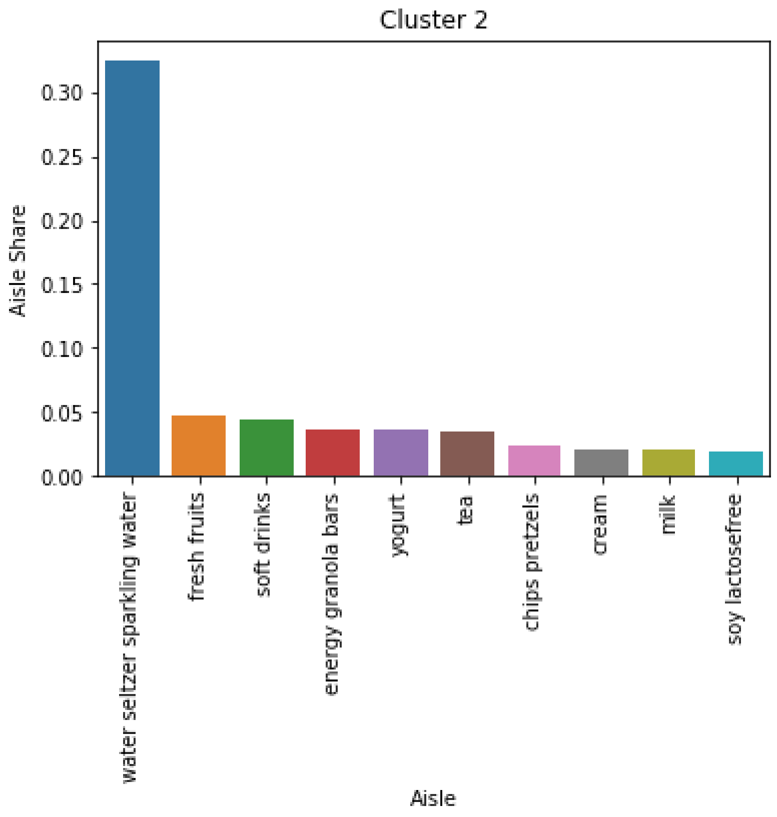


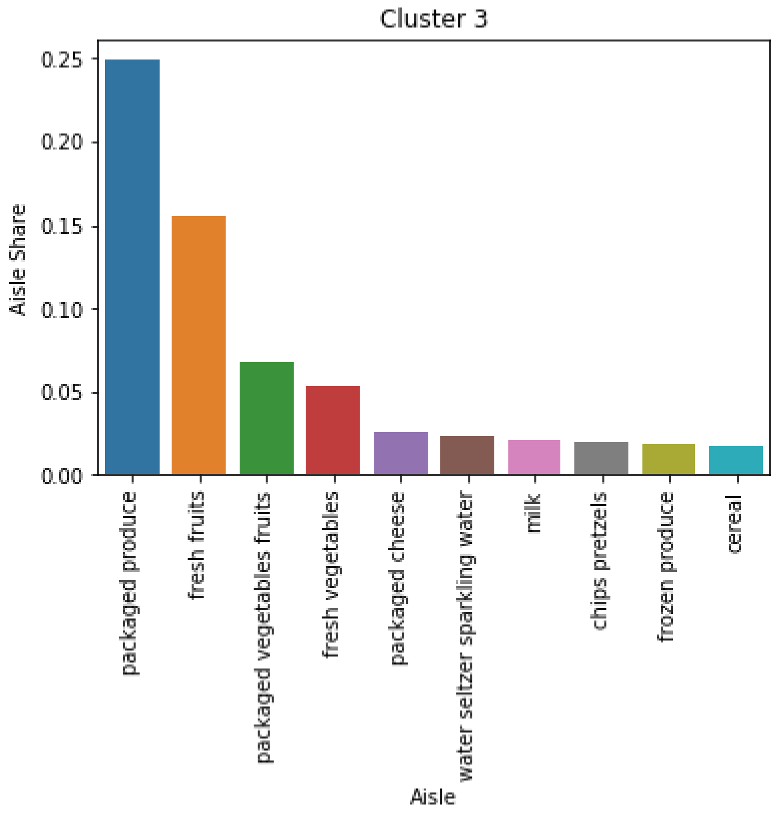
3.5 Top aisle distributions for each cluster

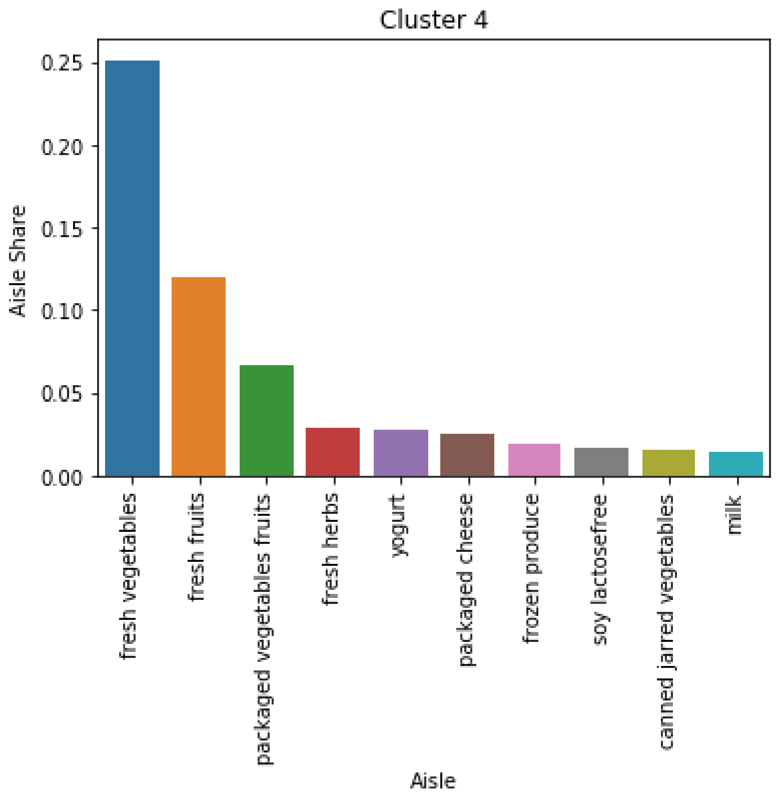
There are clear differences between the 6 clusters, with the most obvious being the relative amounts they buy fresh fruits and fresh vegetables. This makes sense given that produce makes up over 25% of Instacart unit sales. The differences for each cluster may be better brought out by looking at them each individually.

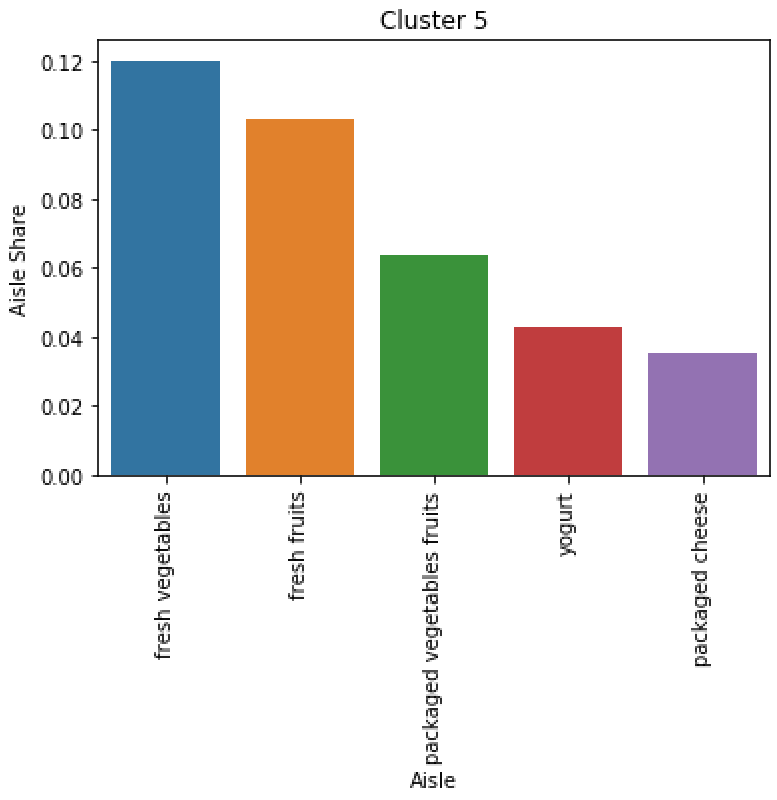





Cluster 2
Aisle Share
0.30
0.25
0.20
0.15
0.10
0.05
0.00
water seltzer sparkling water
fresh fruits
soft drinks
energy granola bars
yogurt
tea
chips pretzels
cream
milk
soy lactosefree
Aisle


Cluster 3
Aisle Share
0.25
0.20
0.15
0.10
0.05
0.00
packaged produce
fresh fruits
packaged vegetables fruits
fresh vegetables
packaged cheese
water seltzer sparkling water
milk
chips pretzels
frozen produce
cereal
Aisle


Cluster 4
Aisle Share
0.25
0.20
0.15
0.10
0.05
0.00
fresh vegetables
fresh fruits
packaged vegetables fruits
fresh herbs
yogurt
packaged cheese
frozen produce
soy lactosefree
canned jarred vegetables
milk
Aisle


Cluster 5
Aisle Share
0.12
0.10
0.08
0.06
0.04
0.02
0.00
fresh vegetables
fresh fruits
packaged vegetables fruits
yogurt
packaged cheese
Aisle

3.6 Conclusion

6 distinct clusters can be seen from the above bar plots. There is one cluster of users that heavily buys sparkling water alone. There is one cluster of users that buys products from various aisles. There is one cluster of buyers who rely on fresh vegetables etc.

The merged data set is converted to CSV for further use. The cluster information of each user is also saved as CSV for further use.

4. Association Rule Mining

This section of the project deals with building item pairs that are a good fit by using association rule mining. The method used in identifying the most important product that must be recommended with a given product is the “Apriori Algorithm”.

This section primarily deals with:

- Association rule mining using Apriori algorithm
- Key metrics to consider when evaluating association rules.
- Product recommender

4.1 Apriori Algorithm

Apriori is an algorithm for frequent itemset mining and association rule learning over relational databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent itemsets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as “Market Basket Analysis”.

- It is used to find frequent item pairs using the “Bottom’s up” approach. It identifies the individual items that satisfy the minimum occurrence threshold.
- Then, it extends the item set, adding one item at a time and checking if the resulting item set still satisfies the specified threshold.
- Algorithms stop when there are no more items to add that meet the minimum occurrence requirement.

Instead of allocating the memory and creating various item pairs, the code uses the help of Python Generator like itertools library.

4.2 Key metrics when evaluating association rules:

There are three metrics used for evaluating association rules between the item pairs. They are support, confidence, and lift.

4.2.1 Support

Support can be calculated as the fraction of orders that contain the item set. For example, if customers both strawberries in 10 orders and the total number of products in all the orders is 100, the support for strawberries will be (10/100) ie., 0.1.

For instacart data, the support of 0.0001 is used for eliminating the items or item pairs that are lower than the given support. Since the data is large and the total number of items is very low compared to the total number of items in all the orders, the support of 0.0001 is good.

4.2.2 Confidence

Given two items, A and B, confidence measures the percentage of times that item B is purchased, given that item A was purchased. This is expressed as:

```
confidence{A->B} = support{A,B} / support{A}
```

Confidence values range from 0 to 1, where 0 indicates that B is never purchased when A is purchased, and 1 indicates that B is always purchased whenever A is purchased. The confidence measure is directional. This means that we can also compute the percentage of times that item A is purchased, given that item B was purchased:

```
confidence{B->A} = support{A,B} / support{B}
```

4.2.3 Lift

Given two items, A and B, lift indicates whether there is a relationship between A and B, or whether the two items are occurring together in the same orders simply by chance (ie: at random). Unlike the confidence metric whose value may vary depending on direction (eg: confidence{A->B} may be different from confidence{B->A}), lift has no direction. This means that the lift{A, B} is always equal to the lift{B, A}:

```
lift{A,B} = lift{B,A} =support{A,B}/(support{A} * support{B})
```

In summary, a lift can take on the following values:

- lift = 1 implies no relationship between A and B.

  (ie: A and B occur together only by chance)

- lift > 1 implies that there is a positive relationship between A and B.

  (ie: A and B occur together more often than random)

- lift < 1 implies that there is a negative relationship between A and B.

  (ie: A and B occur together less often than random)

4.3 Association Rule

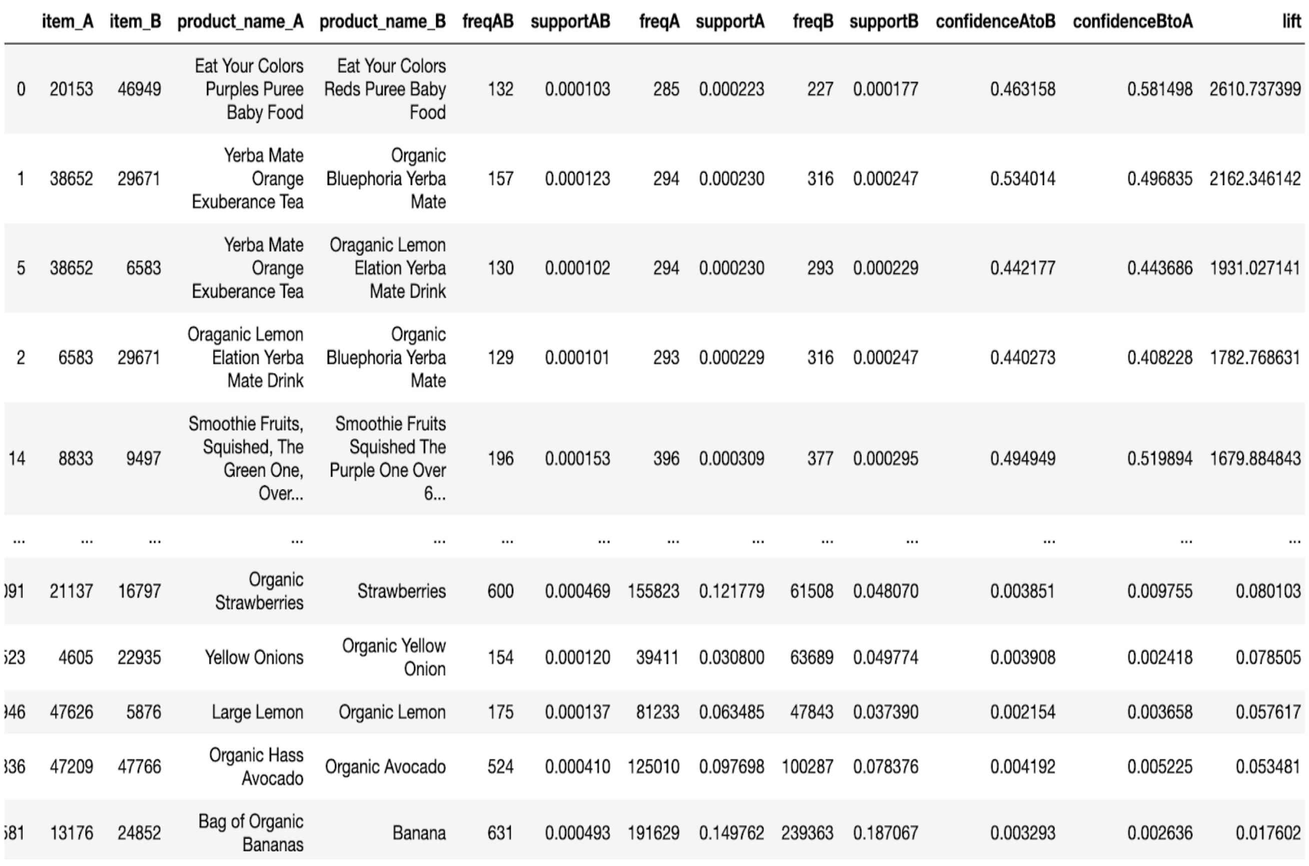
The main outline of the Association rule function is to first calculate support for every item. Based on the chosen item support of 0.0001, the items that have support lower than that will have to be removed. The goal is to identify item pairs that are to be recommended together. Hence, if there are any orders with less than 2 items, They cant give us an idea of what to recommend. After filtering those rows, item support is recalculated to filter any items that have support lower than the given support.

Now, item pairs are generated using the groupby function from the itertools library. All possible item pairs are generated from the filtered out items. The item pairs are returned to a counter function. This way all the item pairs are discarded from the memory. We are only left with the count of all the item pairs.

The support is calculated again for all the item pairs. The item pairs are filtered again based on the above criteria.

The confidence of A to B, Confidence of B to A, and lift of the item pairs are calculated.

By following the above mentioned steps, a table is generated that contains association rules of item pairs. The table consists of item id and item name for both the products in the item pair. And also, confidence measures between item A and item B and also its associated lift.



	item_A	item_B	product_name_A	product_name_B	freqAB	supportAB	freqA	supportA	freqB	supportB	confidenceAtoB	confidenceBtoA	lift
0	20153	46949	Eat Your Colors Purples Puree Baby Food	Eat Your Colors Reds Puree Baby Food	132	0.000103	285	0.000223	227	0.000177	0.463158	0.581498	2610.737399
1	38652	29671	Yerba Mate Orange Exuberance Tea	Organic Bluephoria Yerba Mate	157	0.000123	294	0.000230	316	0.000247	0.534014	0.496835	2162.346142
5	38652	6583	Yerba Mate Orange Exuberance Tea	Oraganic Lemon Elation Yerba Mate Drink	130	0.000102	294	0.000230	293	0.000229	0.442177	0.443686	1931.027141
2	6583	29671	Oraganic Lemon Elation Yerba Mate Drink	Organic Bluephoria Yerba Mate	129	0.000101	293	0.000229	316	0.000247	0.440273	0.408228	1782.768631
14	8833	9497	Smoothie Fruits, Squished, The Green One, Over...	Smoothie Fruits Squished The Purple One Over 6...	196	0.000153	396	0.000309	377	0.000295	0.494949	0.519894	1679.884843
...	...	...	...	...	...	...	...	...	...	...	...	...	...
91	21137	16797	Organic Strawberries	Strawberries	600	0.000469	155823	0.121779	61508	0.048070	0.003851	0.009755	0.080103
23	4605	22935	Yellow Onions	Organic Yellow Onion	154	0.000120	39411	0.030800	63689	0.049774	0.003908	0.002418	0.078505
46	47626	5876	Large Lemon	Organic Lemon	175	0.000137	81233	0.063485	47843	0.037390	0.002154	0.003658	0.057617
36	47209	47766	Organic Hass Avocado	Organic Avocado	524	0.000410	125010	0.097698	100287	0.078376	0.004192	0.005225	0.053481
81	13176	24852	Bag of Organic Bananas	Banana	631	0.000493	191629	0.149762	239363	0.187067	0.003293	0.002636	0.017602

4.4 Product Recommender

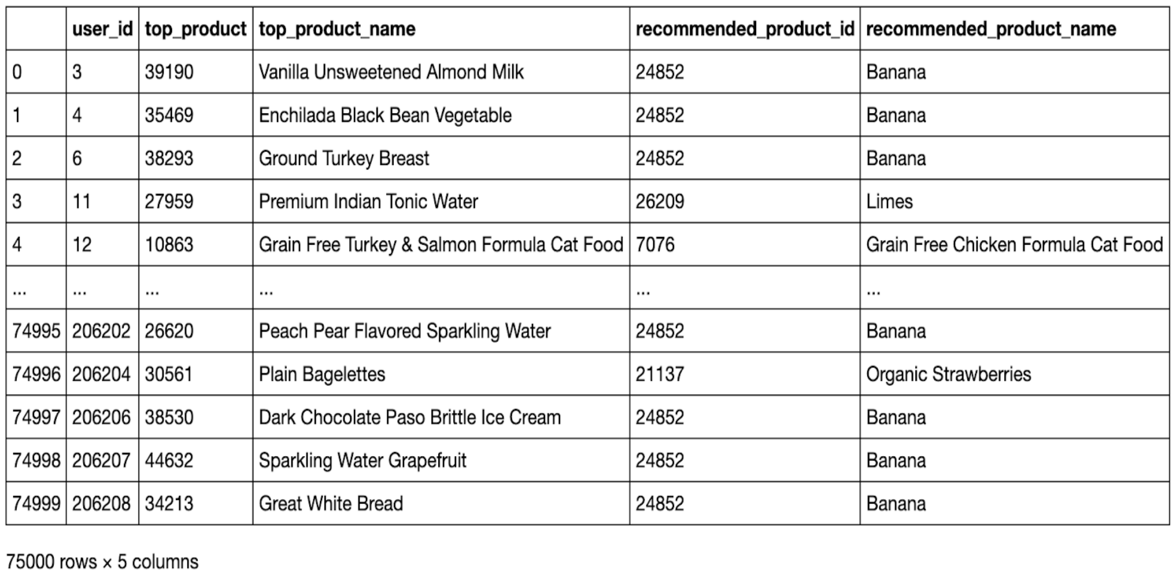
The items are organized in the descending order of their lift. And items with lift lower than or equal to 1 are discarded since they are considered to be occurring by chance.
For a given product, the product recommender checks whether a given item is in item A or item B of the item pairs. The item pairs are rearranged according to the confidence measure required and it returns the product that has the highest confidence. In case if the recommender has no products that have a lift greater than 1, the top product purchased by all users is returned.

The dataset “orders” contain the information of about 75,000 customers with user IDs that are marked as a “test” evaluation set. These users are isolated and the user IDs are converted to a list to serve as an input to the product recommender.

5. Conclusion

The product recommender recommends a product given an input of what is already in the cart. But the available data does not have an item of the test set that is already in the cart. For that reason, a product is recommended based on the top purchase history of each user.

The top product of each user is computed and it is fed as an input to the product recommender. The association rules data frame contains item pairs and their associated evaluation metrics. If the top product of the customer is in item A or item B of the data frame, then the list is filtered to have a lift of greater than 1. Then, the item pair with the most confidence is returned as the most probable recommendation. And the resulting data frame of both the top purchase of the user and the recommended product is converted to CSV.



	user_id	top_product	top_product_name	recommended_product_id	recommended_product_name
0	3	39190	Vanilla Unsweetened Almond Milk	24852	Banana
1	4	35469	Enchilada Black Bean Vegetable	24852	Banana
2	6	38293	Ground Turkey Breast	24852	Banana
3	11	27959	Premium Indian Tonic Water	26209	Limes
4	12	10863	Grain Free Turkey & Salmon Formula Cat Food	7076	Grain Free Chicken Formula Cat Food
...	...	...	...	...	...
74995	206202	26620	Peach Pear Flavored Sparkling Water	24852	Banana
74996	206204	30561	Plain Bagelettes	21137	Organic Strawberries
74997	206206	38530	Dark Chocolate Paso Brittle Ice Cream	24852	Banana
74998	206207	44632	Sparkling Water Grapefruit	24852	Banana
74999	206208	34213	Great White Bread	24852	Banana

75000 rows × 5 columns

This project can also be utilized in returning the top 5 or 10 recommendations instead of 1 recommendation too. The item pairs that are arranged in descending order of their confidence can be used in returning the required number of product recommendations.

One more important thing to note is that during Saturday nights and Sunday mornings, customers shop for their groceries compared to the other days. And based on the re-order ratio, it is would still be a good idea to recommend products based on their purchase history on top of recommendation through association rule mining.