Description

The dataset used for this project purposes consists of 3 million open source online grocery store orders from more than 200 thousands of users. The dataset was available in one of the Kaggle's competitions named 'Instacart Market Basket Analysis'. This competition challenged data miners from all over the world to answer to the following question: "Which products will an Instacart consumer purchase in his next basket?".

Market basket analysis is an important component of every retail company. Simple, yet powerful - MBA is an inexpensive technique to identify cross-sell opportunities and engage customers. At the same time, personalized recommendation systems differentiate companies from the competition and they can lead to competitive advantages. Moreover, recommendation systems are proven to improve user experience, to increase user traffic and the number of purchases and to encourage user engagement and satisfaction. This competition was an opportunity for us to expand our knowledge and to gain hands on experience on models and techniques used in the fields of basket analysis and recommendation systems.

Mission

Aim of this project is to develop models that predict which products will be in the user's next basket. Firstly we analyzed and visualized our transactional data, we applied clustering analysis, association rules (a-priori) and developed three models (baseline model, markov chain modeling and SVD model /neural networks).

Data

The dataset used for this project purposes consists of 3 million open source online grocery store orders from more than 200 thousands of users. For each user, it contains between 4 and 100 of their orders, with the sequence of products purchased in each order. It also includes information concerning the week and hour of day that the order was placed, and a relative measure of time between orders.

The dataset was too large to be handled and in order to deal with the memory overload problem, we kept the last 6 orders of every user and dropped the rest. We then split the dataset into 2 sets: train set and test set. The test set contained the last order of every user and the train set contained the rest.

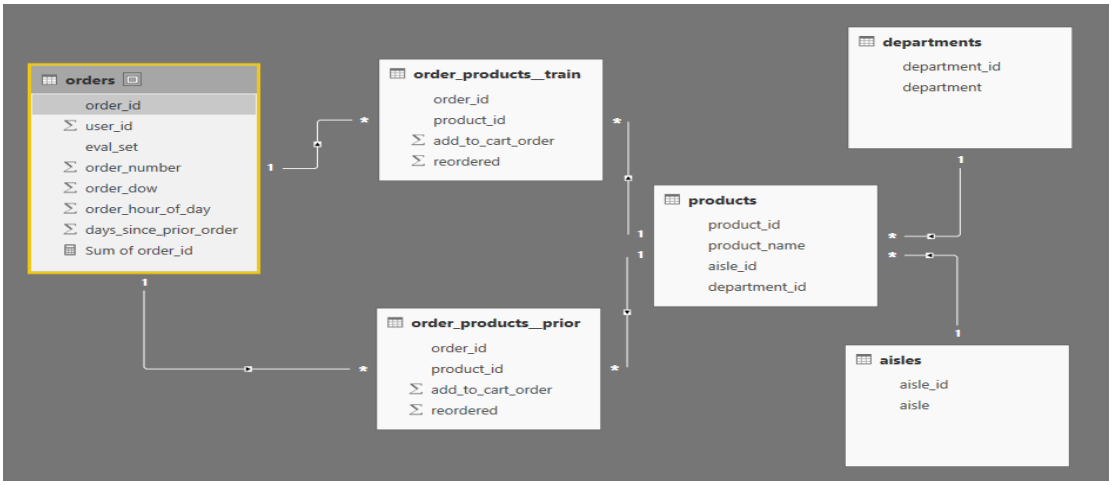

Figure 1 Dataset overview

The first file includes a list of all orders, one row per order. For example, we can see that user 1 has 11 orders, 1 of which is in the train set, and 10 of which are prior orders. This file also contains the order number, the day of the week and hour of the day when the order was made and finally the days since the user's prior order. The orders.csv doesn't not include which products were purchased in each order. This is contained in the order_products.csv

	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
0	2539329	1	prior	1	2	8	NaN
1	2398795	1	prior	2	3	7	15.0
2	473747	1	prior	3	3	12	21.0
3	2254736	1	prior	4	4	7	29.0
4	431534	1	prior	5	4	15	28.0

Figure 2 Orders.csv preview

The second and third file specifies which products were purchased in each prior and train order accordingly. Order_products__prior.csv contains previous order contents for all customers, while Order_products__train.csv contains the last order of every user which is the order that we want to predict. The attribute 'add_to_cart_order' describes the order in which the user bought one specific product while completing his order. The attribute 'reordered' indicates that the customer has a previous order.that contains the product.

	order_id	product_id	add_to_cart_order	reordered
0	2	33120	1	1
1	2	28985	2	1
2	2	9327	3	0
3	2	45918	4	1
4	2	30035	5	0

Figure 3 Order_products.csv preview

The fourth file contains the names of the products with their corresponding product_id. Furthermore the aisle and department are included.

	product_id	product_name	aisle_id	department_id
0	1	Chocolate Sandwich Cookies	61	19
1	2	All-Seasons Salt	104	13
2	3	Robust Golden Unsweetened Oolong Tea	94	7
3	4	Smart Ones Classic Favorites Mini Rigatoni Wit...	38	1
4	5	Green Chile Anytime Sauce	5	13

Figure 4 Products.csv preview

The fifth file contains the different aisles in which the products belong.

	aisle_id	aisle
0	1	prepared soups salads
1	2	specialty cheeses
2	3	energy granola bars
3	4	instant foods
4	5	marinades meat preparation

Figure 5 Aisles.csv preview

Finally, the fifth file contains the different departments in which the products belong.

	department_id	department
0	1	frozen
1	2	other
2	3	bakery
3	4	produce
4	5	alcohol

Figure 6 Departments.csv preview

Exploratory Data Analysis

The exploratory analysis (EDA) will help us to better understand the dataset. For the EDA part of this project, we used Python, R and Power BI. The dataset consists of information about 3.4 million grocery orders, distributed across 6 csv files as it was mentioned earlier. There are 206,209 customers in total. Out of which, the last purchase of 131,209 customers are given as train set and we need to predict for the rest 75,000 customers. The products belong to 134 aisles and the aisles belong to 21 departments.

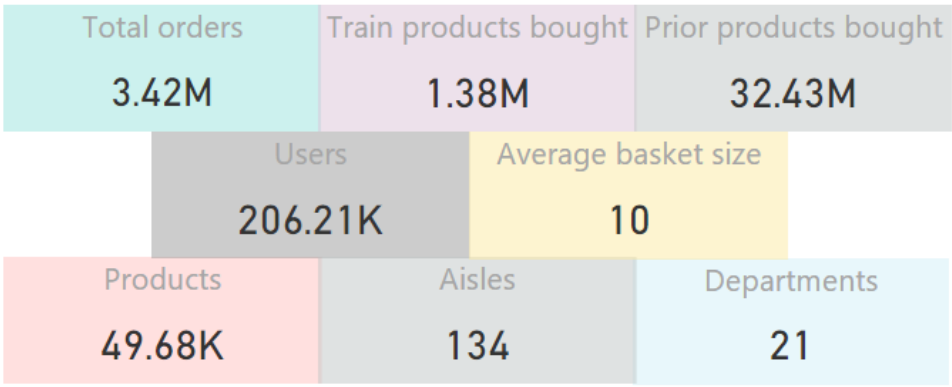

Figure 7 Dataset overview

Figure 8 represents the number of orders in the prior, train and test set.

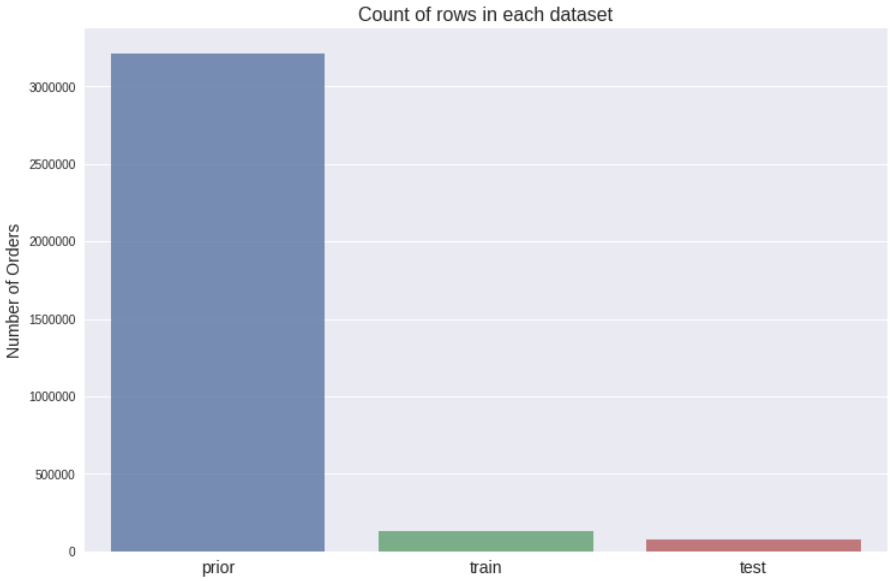

Figure 8 Prior, train and test set orders

Popular weekdays and hours

Figure 9 shows the number of orders over the days of the week. Due to the fact that the data was anonymized we cannot state with absolute certainty but we can safely assume that the high bars

represents weekends. Saturdays and Sundays are the most popular days for online shopping. People are not working during the weekend, which means more time to spend for shopping.

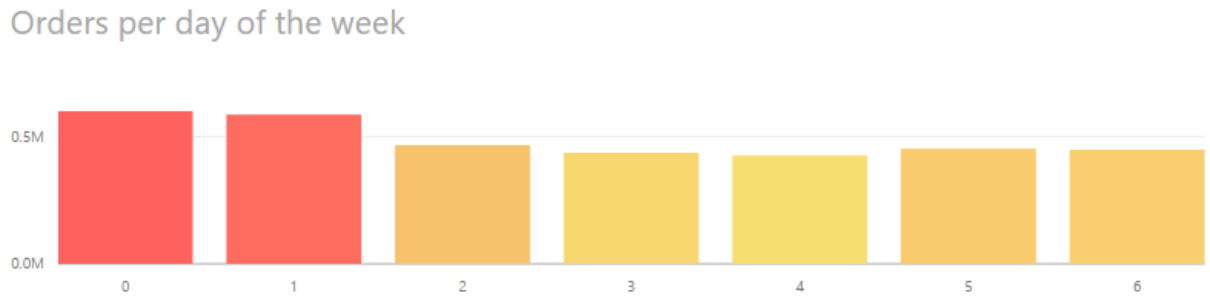

Figure 9 Day of week customers make the most orders

From figure 10 we understand that consumers are more likely to shop during the day. The most common shopping hours are 9:00 through 17:00. We must notice that peak hours are between 10:00-11:00 am and 15:00-16:00 am.

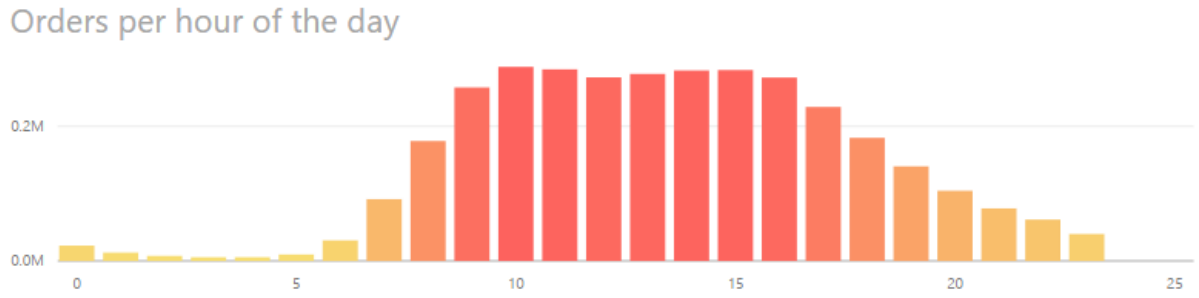

Figure 10 Orders during the day

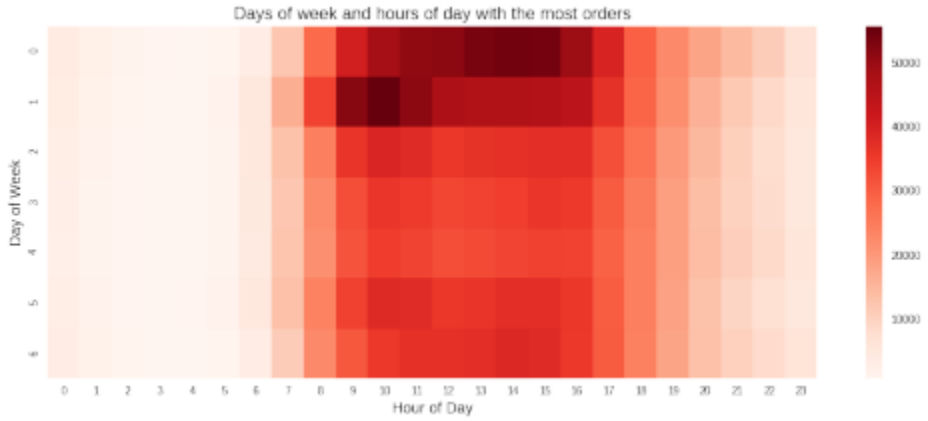

Figure 1 Das of the week and hours of the day with the most orders

Popular departments

There are 21 departments in total. The plot presented above visualizes the most popular departments. Produce with all the fresh fruits and vegetables is by far the department with the most sales. Fresh fruits and vegetables appear in the 33% of the total sales. Dairy eggs, snacks and beverages follow with much lower percents, 19%, 10% and 9% accordingly.

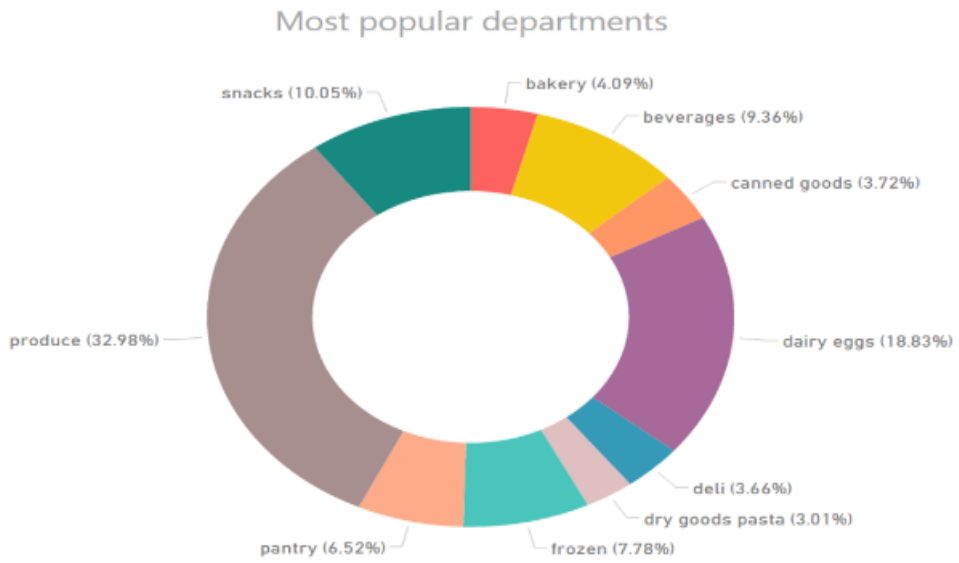

Figure 11 Departments with the highest sales

Figure 12 represents the number of products sold within each department. The size of the boxes signifies the number of products sold in each department. As it was observed before, produce department has the most sales even though it is a small department with fewer products than others.

Figure 12 Product sales within each department

Popular Aisles

In total there are 134 aisles. Figure 13 represents aisles with the most sales. Fresh fruits and fresh vegetables make the top of the list. The produce department is the most famous department.

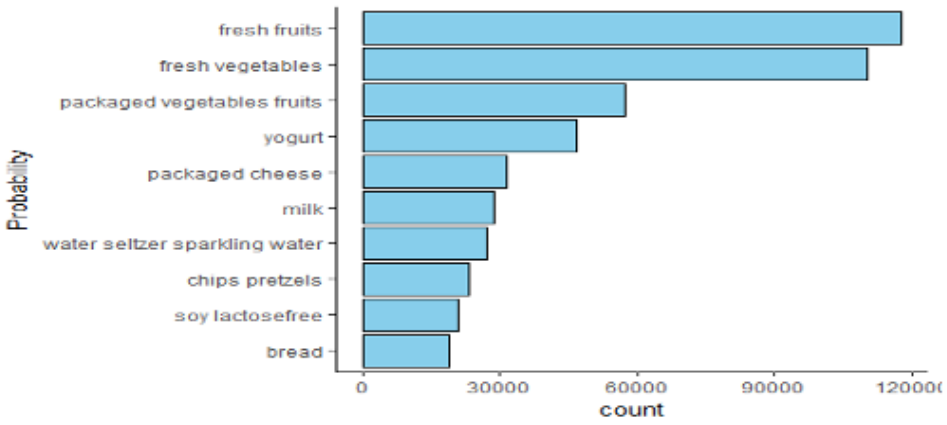

Figure 13 Aisles with most sales

Fresh fruits appear in 19 % of the total orders, fresh vegetables in 18% and packaged vegetable fruits in 9% of the total orders.

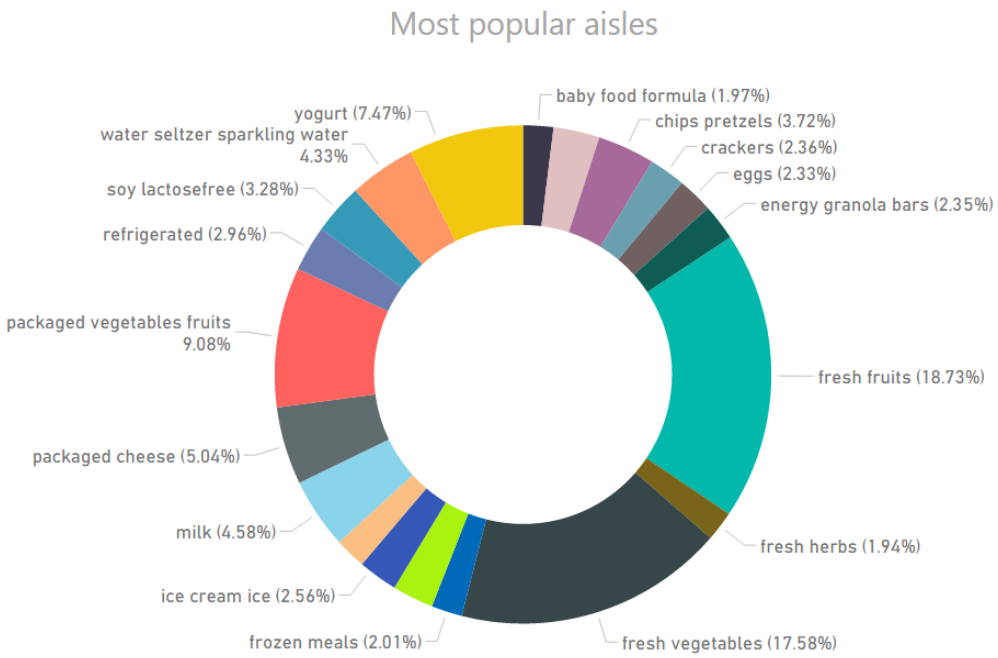

Figure 14 Aisles with the highest sales

Let's see how the aisle are distributed among the various departments (figure 14). It appears the personal care department controls majority of the aisle but sells less products when compared to the produce department.

Figure 15 Aisle distribution among departments

Popular products

There are approximately 50k products in the dataset. However, all of the top 10 selling products are fruits and vegetables and most of them are organic products. This is a really interesting finding. Figure 16 represents the most frequently bought products. As we see Bananas are the most ordered products. In addition Strawberries, Avocado and Limes also make it to the top ordered products.

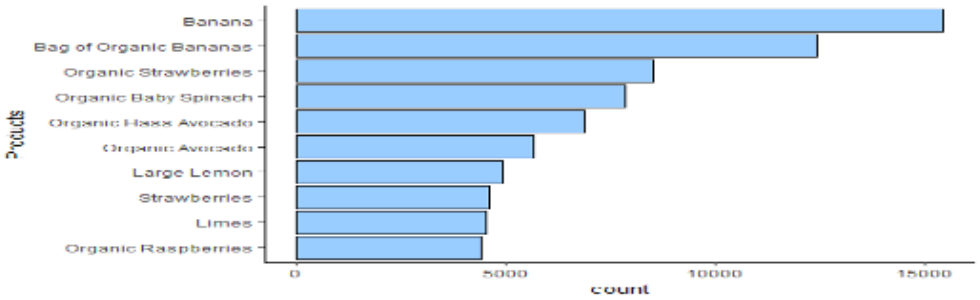

Figure 16 Products with highest orders

Bananas appear in 14% of all the order, organic baby spinach in 7% and organic hass avocado in 6%. Only organic whole milk appear in the top selling products and does not belong to the produce category.

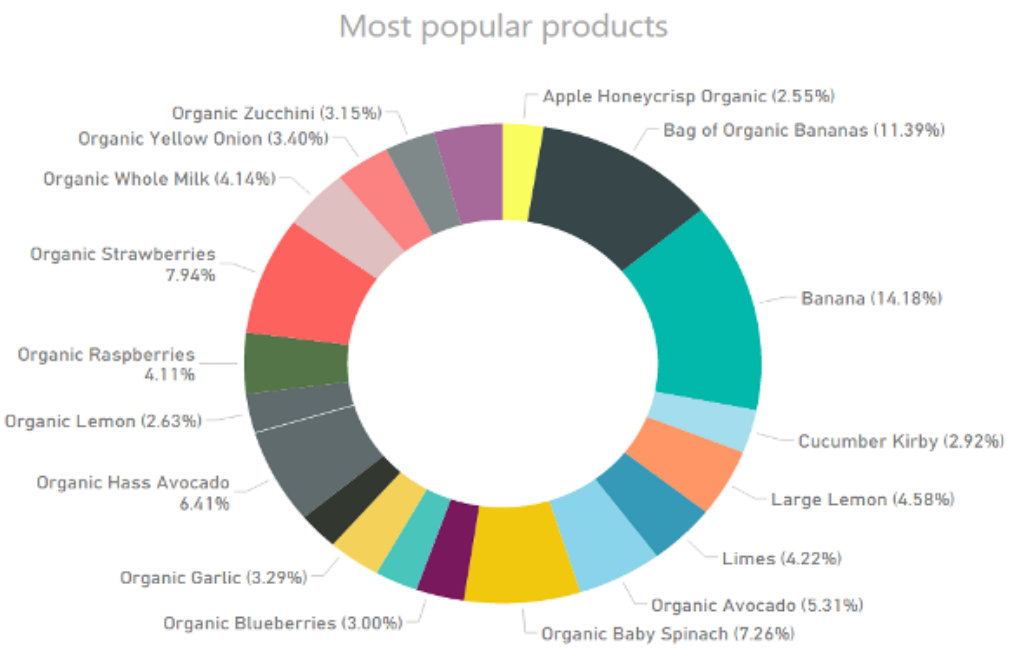

Figure 17 Products with the highest sales

Reordering

Since the aim of this project is to predict the products in the next baskets, we are interested to discover the parameters which affect the user's decision to reorder a product. The plot presented below highlights the proportion of products that were reordered. Values equal to 0 signify that the user in his order bought a product for the first time whereas 1 means the specified product was re ordered in the past. In figure 18 we observe that about 59% of all products in our database were reordered. This information is useful for our further analysis. That means consumers are satisfied with products and they wants to buy them again.

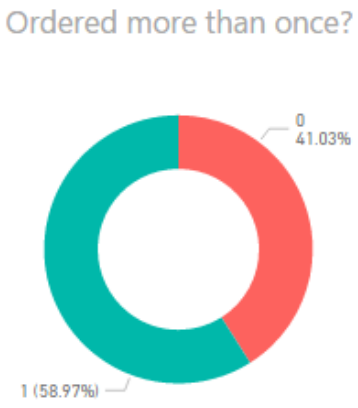

Figure 18 Proportion of re-ordered products

Figure 19 the reordering ration of every department. Personal care has lowest reorder ratio and dairy eggs have highest reorder ratio.

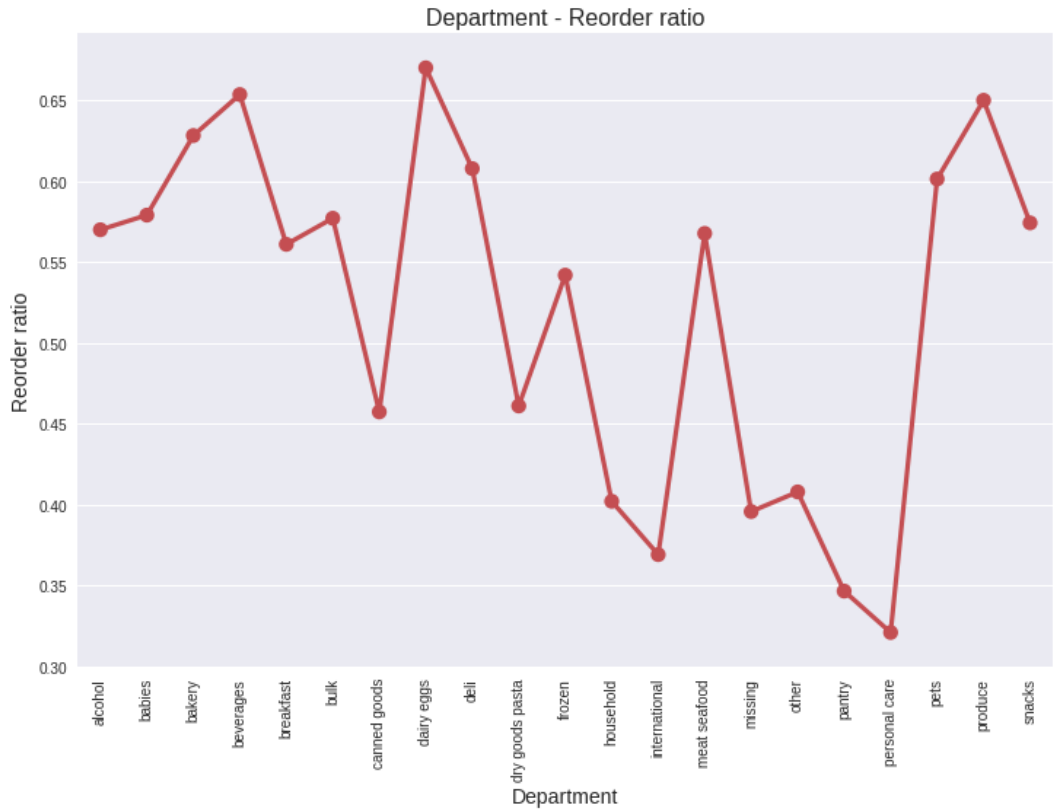

Figure 19 Department - Reorder ratio

Figure 20 represents the products which are most likely to be reordered. The 10 products in figure have the highest probability of being reordered. Even though Banana is the most ordered product (as we noticed previously), in the reordered products has the sixth rank. 2% Lactose Free Milk is the most reordered product. As expected, most of them are products of daily consumption, such as different types of milks and the sparkling water cans.

Product id	proportion re-ordered	n	product_name	aisle_id	department_id
1729	0.9347826	92	2% Lactose Free Milk	84	16
20940	0.9130435	368	Organic Low Fat Milk	84	16
12193	0.8983051	59	100% Florida Orange Juice	98	7
21038	0.8888889	81	Organic Spelt Tortillas	128	3
31764	0.8888889	45	Original Sparkling Seltzer Water Cans	115	7
24852	0.8841717	18726	Banana	24	4
117	0.8833333	120	Petit Suisse Fruit	2	16
39180	0.8819876	483	Organic Lowfat 1% Milk	84	16
12384	0.8810409	269	Organic Lactose Free 1% Lowfat Milk	91	16
24024	0.8785249	461	1% Lowfat Milk	84	16

Figure 20 More likely products to be re-ordered

Figure 21 represents the reordering ration based on the add to cart order. From this plot it is clear that users tend to reorder products that were placed first in the basket. People easily remember to by first their favorite products.

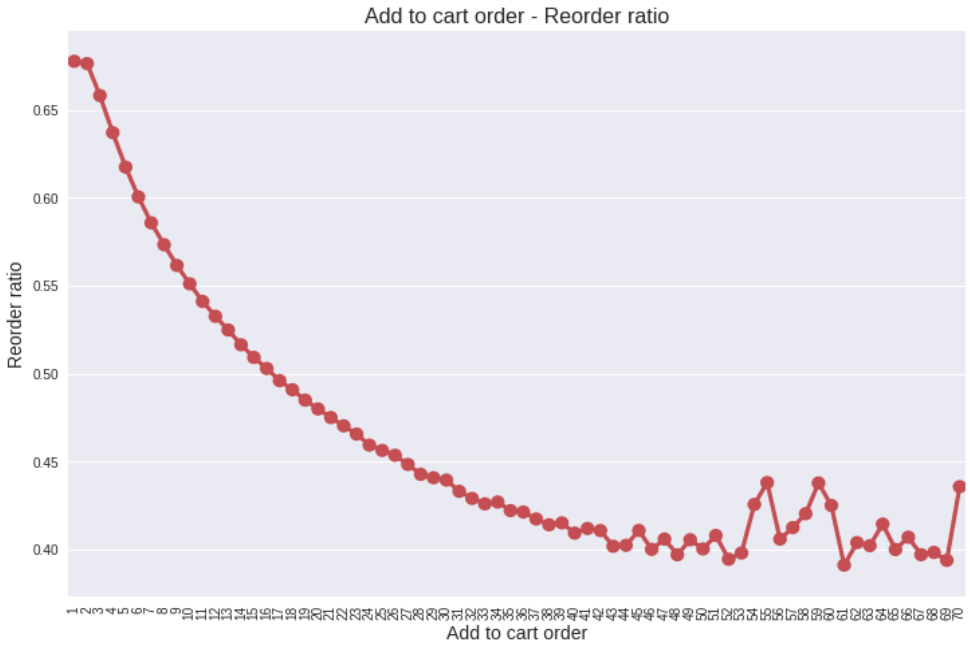

Figure 21 Add to cart order - reorder ratio

Figure 22 represents the correlation between the number of orders and probability of reordering. We observe that products with a high number of orders are naturally more likely to be reordered (for example Banana or vegetables). However, there seems to be a ceiling effect.

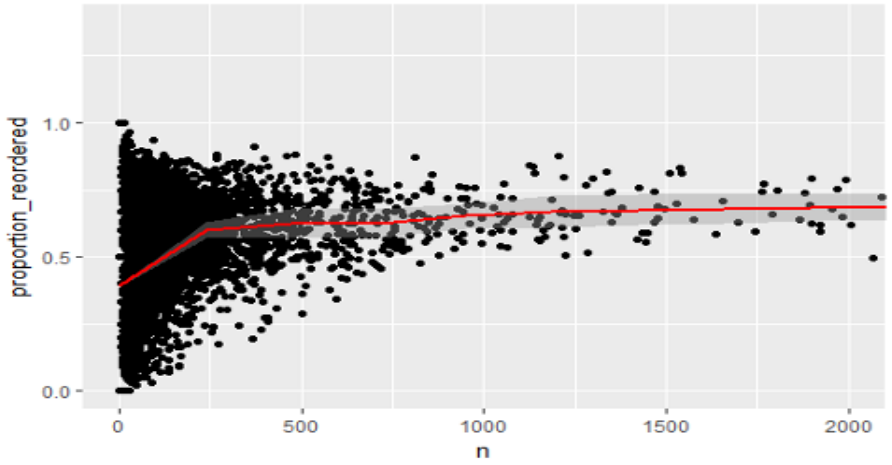

Figure 22 Correlation between number of orders and probability of reordering

Average basket size

Estimating the right basket size plays a significant role on the prediction of the next order. Let's have on look on how many items are in the orders found in the train and the prior set of orders. We can see that people most often order around 5 to 7 items. The distributions are similar between the train and prior order set.

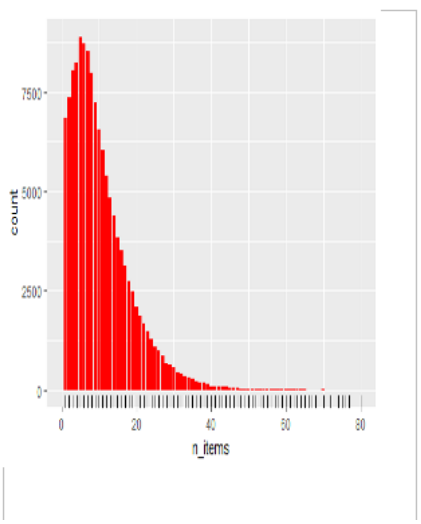

Figure 23 Train set - basket size

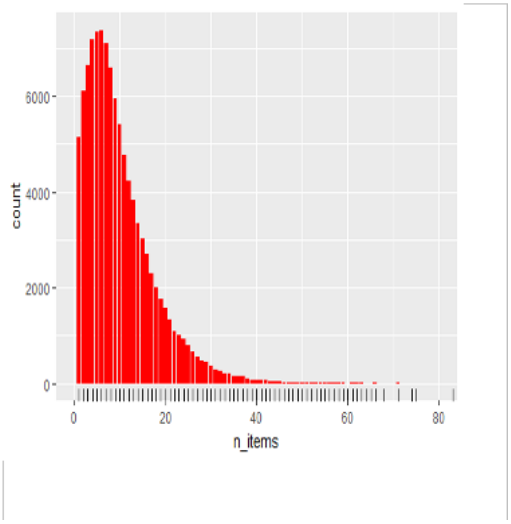

Figure 24 Prior orders - basket size

Singular Value Decomposition

The second model is less simple and consists of two steps. Once again, our aim is to predict the basket size as well as the products bought in the next order of each user. In the first step, we use Singular Value Decomposition (SVD) in order to estimate the size of the basket that we want to predict. Let's say for example that the estimated basket size equals to n. In the second step we will predict the n products which we believe that the user will buy in his next order. To do so we will use statistic metrics of users' consumer behavior. We will rank the importance of each product to each user taking under consideration user's past purchases and preferences. Finally, we will recommend to each user the n products with the highest ranking.

Step 1

Singular value decomposition (SVD), Tensorflow and neural networks were used during the first step of predicting the next basket size. SVD is a data dimensionality reduction technique but it can also be used in collaborative filtering. Factorization models such as SVD are very popular in recommendation systems because they can be used to discover latent features underlying the interactions between two different kinds of entities. Tensorflow is a general computation framework using data flow. It provides variant SGD learning algorithms, CPU/GPU acceleration, and distributed training in a computer cluster. Word2vec is a two-layer neural network that processes text. Its input is a text corpus and its output is a set of vectors. Word2vec's applications automatically learn relationships between two entities and therefore can extend beyond parsing sentences. It can be applied just as well to recommender systems, code, likes, playlists, social media graphs and other verbal or symbolic series in which patterns may be discerned. Since Tensorflow has several embedding modules for word2vc-like application, it is supposed to be a good platform for factorization such as SVD. Finally, neural networks were used to achieve higher accuracy. The model was retrained multiple times, and an error-correction learning rate was applied.

SVD requires a data matrix A of size nxm as input. In our case, the m rows of matrix A represent the user orders of the train set, the n columns represent the number of 'days since the previous order' and the matrix values represent the basket size of the specified order. What SVD does is to represent the matrix A as a product of different matrices U, Σ and V so that $\mathbf{A}_{[mxn]} = \mathbf{U}_{[mxr]}\mathbf{\Sigma}_{[rxr]}(\mathbf{V}_{[nxr]})^T$. According to theory, there is always a possible unique way to decompose a real matrix A into

three others, $U\Sigma V^T$, where U and V are column orthonormal (sum of the squared values in each column equals 1) and orthogonal (the inner product of their columns equals 0), while Σ is diagonal. Matrix U is called left singular value and has size mxr, Σ is a diagonal matrix of singular values, with size rxr and it has zeros everywhere except from the diagonal. The diagonal contains the positive singular values which are sorted in decreasing order. V stores the right singular vectors with size nxr. In our case, table U describes how possible is that the next basket of user mi is of size ri. Singular values of table Σ represent the strength of every concept, which in our case describe the certainty with which we relate every order with a possible basket size. Matrix V is a concept to 'day since prior order' matrix and therefore describes the certainty with which we relate the 'days since prior order' of every order with a possible basket size r.

$$\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$

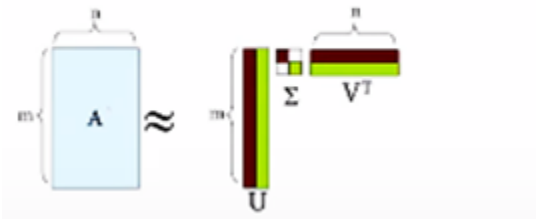

Figure 29 Singular value decomposition (SVD)

In neural network terminology, batch size is the number of training examples in one forward/backward pass of the training set and one epoch is one forward pass and one backward pass of all the training examples. Furthermore, learning rate is a technique of comparing the system output to the desired output value, and using that error to direct the training. In order to achieve higher accuracy, we split the training set into batches of size 100 and we repeated the training procedure for 100 epochs. A model was trained for every 100 row batch. The whole training process got repeated 100 times. We also defined an error-correction learning rate with value 0.001.

```
epoch train_error val_error elapsed_time
  0 11.435908 12.053858 0.063533(s)
  1 11.776064 11.979199 1.492668(s)
  2 11.843151 11.901427 1.520852(s)
  3 11.813773 11.815067 1.476690(s)
  4 12.047588 11.715937 1.569240(s)
  5 11.742437 11.602110 1.590145(s)
        ...      ...      ...
 95 3.449291 5.569919 1.637508(s)
 96 3.388213 5.575913 1.606004(s)
 97 3.484258 5.579345 1.582954(s)
 98 3.483150 5.586791 1.618102(s)
 99 3.525798 5.590288 1.573104(s)
100 3.538033 5.597464 1.606147(s)
```

Figure 30 Errors and elapsed time by epoch

Step 2

In this step we will predict the products in the next basket. For every user - product relationship we calculated a rating system based on the user's past consumer behavior. These statistics include the percentage of baskets in which user x purchased product y, the reorder rate of each product, the average purchase order in which user x purchases product y, the average purchase frequency of each product etc. Every possible user – product relationship was then ranked and sorted in decreasing order. Higher ranks mean that user x will probably buy product y in his next

order. Therefore, if in step 1 we had predicted that the size of the next basket of user x equals to 5, then in the second step we would recommend to user x the 5 products with the highest ranking.

```
user_id,product_id,prod_percent_baskets,percent_reordered,average_order_position,days_since_prior_order
1,196,1.0,1.0,1.8,16.6
1,10258,1.0,1.0,3.8,16.6
1,25133,1.0,1.0,4.0,16.6
1,12427,1.0,1.0,4.2,16.6
1,46149,0.6,0.666666666667,3.0,14.6666666667
1,49235,0.4,0.5,3.5,7.0
1,13032,0.4,1.0,6.5,25.0
1,39657,0.2,0.0,3.0,30.0
1,38928,0.2,0.0,4.0,30.0
1,35951,0.2,0.0,7.0,30.0
201,13712,0.8,1.0,1.5,3.5
201,329,0.8,0.75,3.25,10.75
```

Figure 31 User - product past purchases statistics

Future improvements

Step 2 doesn't know how to cope with missing values. For example, our model will only recommend to users, products that they have bought at least one time in the past. This fact, limits our model accuracy to a ceiling value. As a future improvement of this model we consider also implementing SVD in the second step of the methodology. More specifically we will develop a rating system for every user-product relationship based on the past consumer behaviour. We will then perform SVD to estimate the rating for the products that users have not purchased yet. In this way, when predicting the next basket, we will also be able to recommend to users products that they have not purchased in any time in the past.

Calculation

Software

1. Python 2.7 – Anaconda
2. Tensorflow 1.3 – CPU ONLY version

Hardware

1. Intel i3
2. 12GB DDR3
3. SSD