# **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.

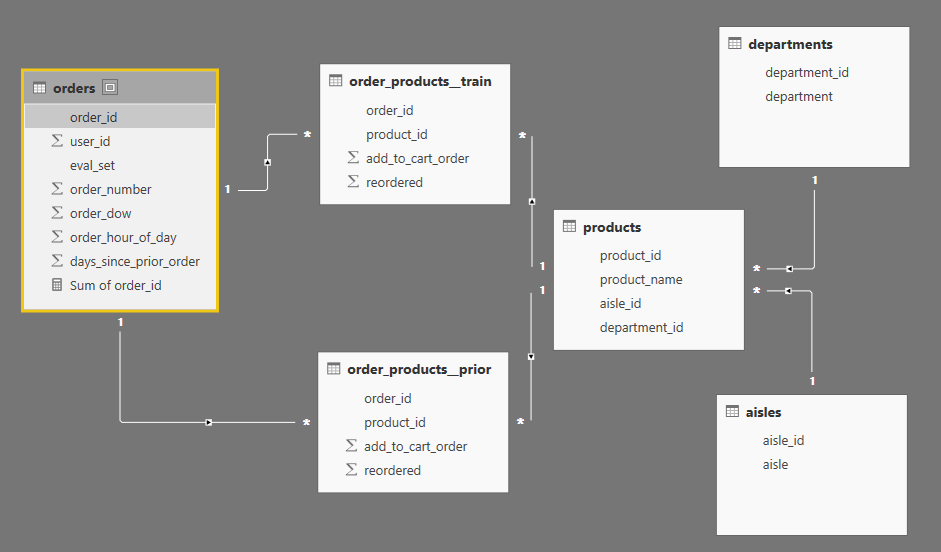
# **Mission**

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**

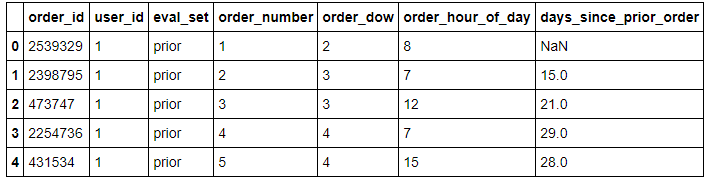
The dataset used for this project purposes consists of 3 million open source online grocery store orders from more than 200 thousands of users. For each user, it contains between 4 and 100 of their orders, with the sequence of products purchased in each order. It also includes information concerning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



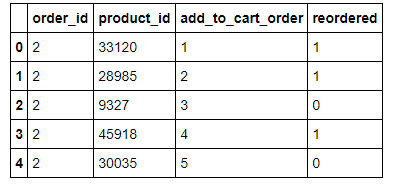
*Figure 1 Dataset overview*

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

**

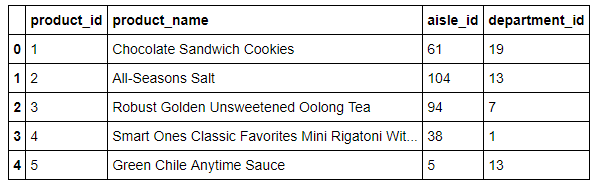
*Figure 2 Orders.csv preview*

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.

**

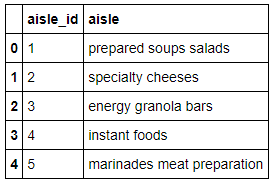
*Figure 3 Order\_products.csv preview*

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

**

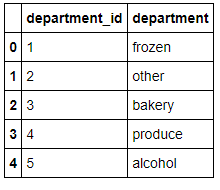
*Figure 4 Products.csv preview*

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



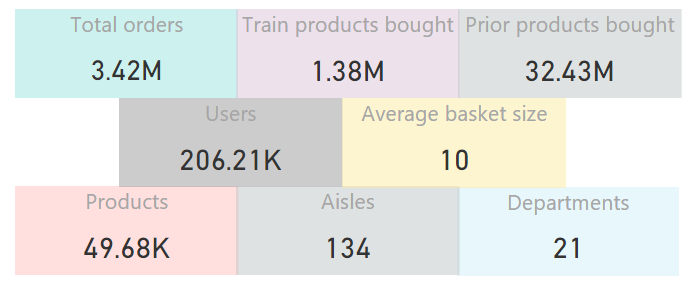
*Figure 5 Aisles.csv preview*

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



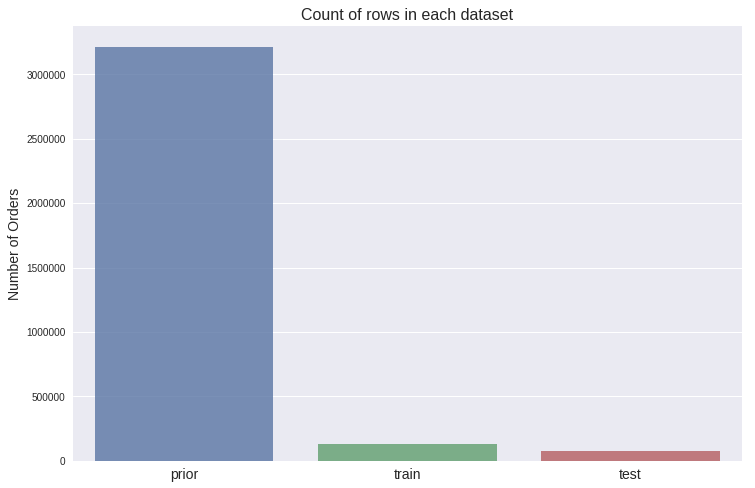
*Figure 6 Departments.csv preview*

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*

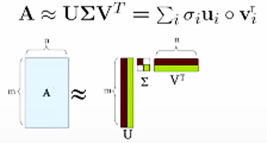
## **Singular Value Decomposition**

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**

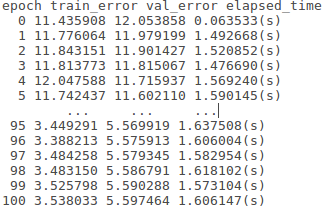
Singular value decomposition (SVD), Tensorflow and neural networks were used during the first step of predicting the next basket size. SVD is a data dimensionality reduction technique but it can also be used in collaborative filtering. Factorization models such as SVD are very popular in recommendation 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 **A[mxn] = U[mxr]Σ[rxr](V[nxr])T**. According to theory, there is always a possible unique way to decompose a real matrix A into three others, UΣVT, 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.



*Figure 29 Singular value decomposition (SVD)*

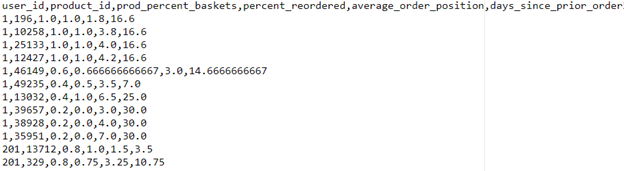
In neural network terminology, batch size is the number of training examples in one forward/backward pass of the training set and one epoch is one forward pass and one backward pass of all the training examples. Furthermore, learning rate is a technique of comparing the system output to the desired output 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.



*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.



*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