MARKET BASKET ANALYSIS

Data Science

ABSTRACT

Apply data science techniques to customer shopping dataset and (1) predict possible items that a customer would likely reorder in future and (2) find interesting association among products that a customer would buy together using Apriori data mining algorithm.

Manikandan Eswaran, Vivek Aggarwal, Harminder Singh

CS5661 Spring 2018

Contents

Introduction

The object of this project is to apply the data science techniques and principles on the order-product details dataset and make the following 2 predictions:

1. Using standard scikit-learn algorithms and predict if a particular item (product) has the possibility to be re-ordered.
2. Apply Apriori association rule data mining techniques to find the combination of product items that has a possibility of being ordered together with a given confidence level.

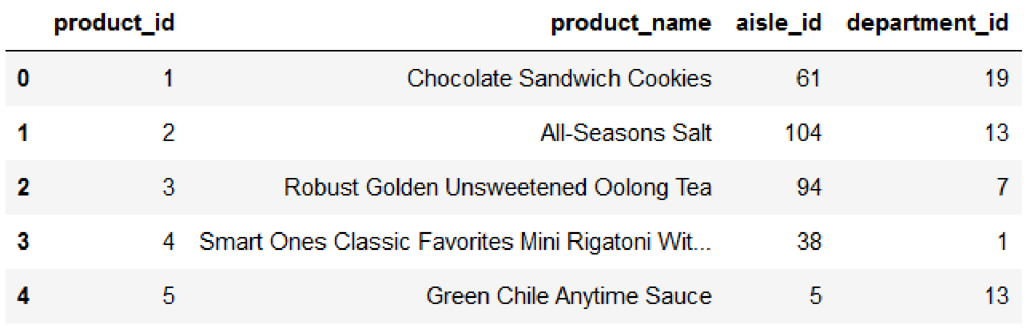
Dataset

The dataset for this project was taken from kaggle. The data set contains transactions data set that contains a set of orders and the list of products in each order with a flag specifying if a particular item in that order was a re-order or the first order.

Following are the data files used in this project:

products.csv

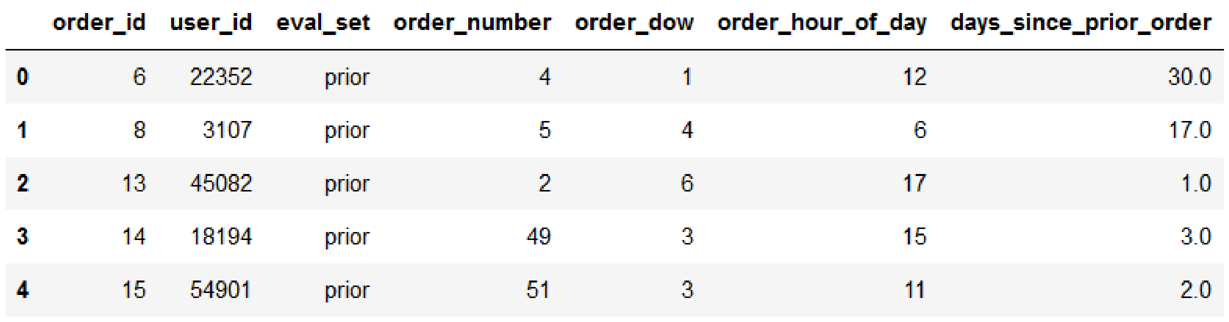
This file is the master product data sheet with product name for each product id.



	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

orders.csv

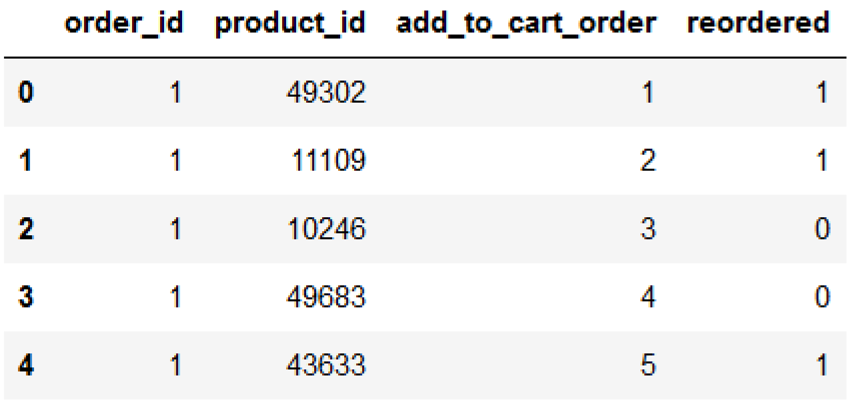
This file contains the order master data. The “eval_set” column contains one of the values prior, train or test. All rows with eval_set = “test” was extracted for test data.



	order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
0	6	22352	prior	4	1	12	30.0
1	8	3107	prior	5	4	6	17.0
2	13	45082	prior	2	6	17	1.0
3	14	18194	prior	49	3	15	3.0
4	15	54901	prior	51	3	11	2.0

products_orders__train.csv

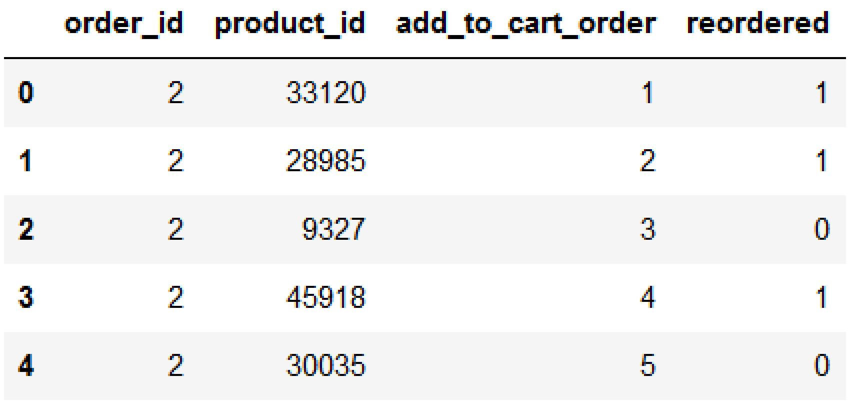
This dataset contains all order details to be used as training dataset.



	order_id	product_id	add_to_cart_order	reordered
0	1	49302	1	1
1	1	11109	2	1
2	1	10246	3	0
3	1	49683	4	0
4	1	43633	5	1

products_orders__prior.csv

For all orders that has reordered flag set to ‘1’ in products_orders__train.csv will have the prior order details in products_orders__prior.csv

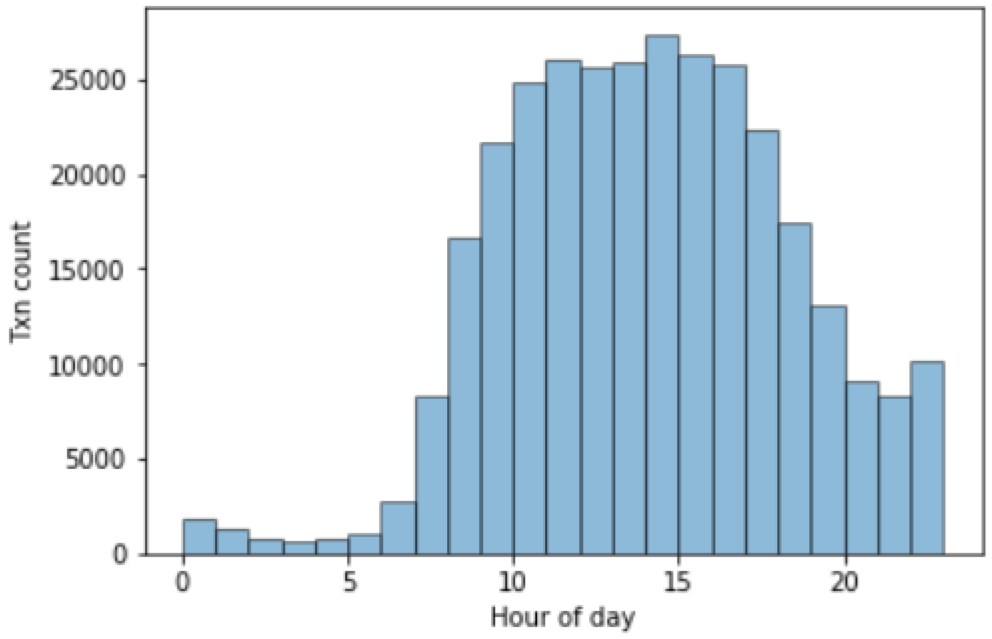


	order_id	product_id	add_to_cart_order	reordered
0	2	33120	1	1
1	2	28985	2	1
2	2	9327	3	0
3	2	45918	4	1
4	2	30035	5	0

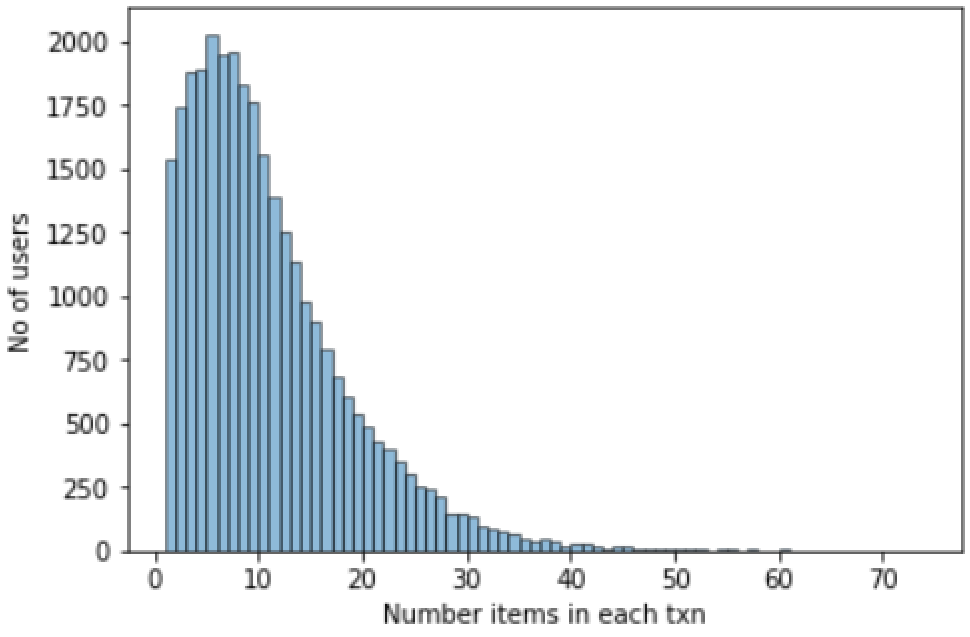
Analysis on the dataset:

To get an understanding of the distribution of the products within the transactions so that we can better correlate the results of applying Apriori algorithm on the data set, the following graphs were generated to highlight some insights into the dataset.

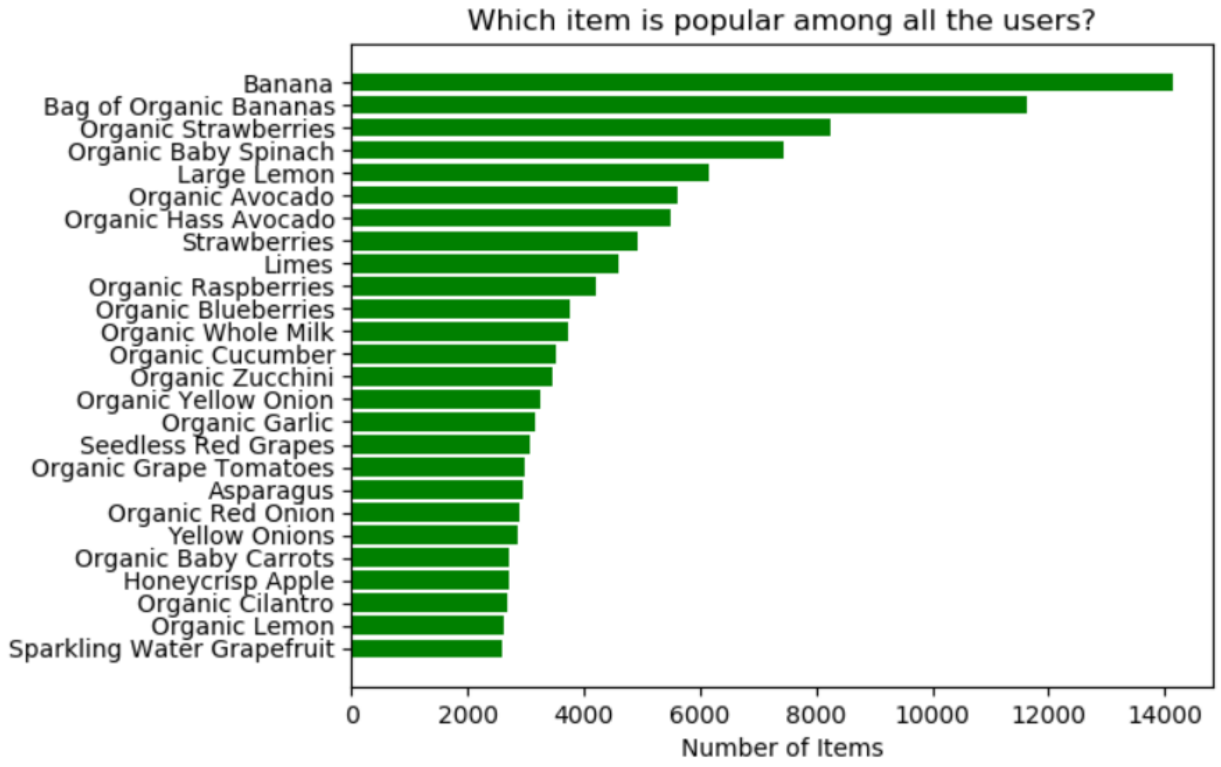
Distribution of transactions over the day:



Distribution of number of products purchased by users



Most purchased products by users



Part-1: Predicting reorder flag for orders in training dataset

Following are the results of applying standard Scikit-Learn algorithms on the market basket dataset. As a preprocessing step, the 'product_id' and 'order_id' values are one hot encoded and the PCA was applied for dimensionality reduction since number of features after applying one hot encoding was 26k+.

Algorithm	Parameters	Accuracy Score (%)
DecisionTreeClassifier	-	59.63
RandomForestClassifier	n_estimators=25	68.21

MLPClassifier	hidden_layer_sizes=(3,3)
	activation= 'logistic'
	solver='adam'
	alpha=1e-5,
	learning_rate_init = 0.01
		63.81

GridSearchCV/ MLPClassifier	`{'hidden_layer_sizes': (5, 5)}`	63.56
Deep NN	`Dense(10/relu)->Dense(5/relu)->Dense(1/softmax)`	60.45

Part-2: Apply Apriori algorithm to mine association rules among product groups

Introduction to Apriori algorithm

The second part of the project was to implement Apriori algorithm on the given dataset and find associations between various product groups within the given transactions to find interesting associations between products that were ordered together, with a confidence level above 60%.

Apriori is an association rule mining algorithm. There are various variations of this algorithm used in data mining to find association rules, our implementation uses the original Apriori algorithm. This algorithm is not part of the scikit-learn library and hence we have provided our own implementation of this algorithm and applied it on the dataset.

There are 2 basic parameters that control the output of this algorithm which are support and confidence.

Let us consider the following list of transactions to explain this algorithm.

Transaction Id	**Products**
1	{milk, egg, bread}
2	{milk, egg, coffee, bread}
3	{sugar, coffee, toothbrush}
4	{milk, bread, coffee}
5	{sugar, egg, vinegar}

Following are a subset of product combinations and their count of occurrences in the transactions

Product(s)	**No of txns containing product(s)**
{milk}	3
{bread}	3
{egg}	3
{sugar}	2
{toothbrush}	1
{milk, bread}	3
{egg, bread}	2
{sugar, toothbrush}	1

Support – Support for rule {X}->{Y} is the ratio of number of transactions where a product group {X U Y} is part of, to the total number of transactions.

$$support(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

Confidence – Confidence of a rule {X}->{Y} is the ratio of support of the given product group {X U Y} to the support of the product group {X}.

$$confidence(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

In the given example dataset, the following table shows the support and confidence values for 3 sample rules:

Rule	**Support**	**Confidence**
{milk}->{bread}	S({milk, bread})/N = 3/5 = 0.6	S({milk, bread})/S({milk}) = 3/3 = 1.0
{egg}->{bread}	S({egg, bread})/N = 2/5 = 0.4	S({egg, bread})/S({egg}) = 2/3 = 0.67
{sugar}->{toothbrush}	S({sugar, toothbrush})/N = 1/5 = 0.2	S({sugar, toothbrush})/S({sugar}) = 1/2 = 0.5

Given the rules derived from the example dataset, the support for a customer who buys milk to buy bread is 60% with a confidence value 100%. Whereas the support for toothbrush to be bought whenever someone buys sugar is just 20% and the confidence level is just 50%.

Applying Apriori algorithm in market basket dataset

The given dataset contains close to 1M order details. As a preprocessing step before applying the Apriori algorithm, the orders are grouped into unique order ids and a list of products in each order id. This preprocessed data is used as the list of transactions for the input to the algorithm. All order id and product id values are all numeric in the given data set making it easier to apply the algorithm on data as is.

Following are the various functions in our Apriori implementation and the description of the function:

Function apriori(txn_set)

The main function that implements Apriori that takes a dictionary of key=order_id and value=list of products as input. The functions calculates support, confidence for various product combinations and returns the support dictionary with {product group: support value} and a dictionary of confidence with {rule (X->Y): confidence}.

Function apriori_genrules(support_dict)

Function that calculates confidence for rules derived from combinations of products in support dictionary and returns only those rules whose confidence is above the threshold. This function prunes all child associations of a parent whose confidence is below threshold. For example if {a,b,c}->{d} has low confidence then it skips {a,b}->{c,d} and {a}->{b,c,d}.

Function get_all_single_item_count(txn_set)

Function that returns unique set of product ids and the count of occurrence of each product in all of the transactions.

Function get_candidate_itemsets(itemset_dict)

This function generate the candidate set of products and its count of occurrences in all transactions for each iteration based on the candidate set used in previous iteration. For example if we consider set of products {a,b,c} candidate set generation generates {ab,bc,ac}.

There are different approaches to generating candidate sets: brute force approach, Fk-1*Fk method and Fk-1*Fk-1 method. In this implementation Fk-1*Fk-1 method is used where each candidate set from previous iteration is combined it with itself to generate the new candidate set for the new iteration.

Function build_hash_tree(c_k)

This function generates the hash tree with the products in the candidate set of a given iteration. This hash tree data structure makes it efficient to check if each product group in candidate set is a subset of the list of products in any given transaction and increment its support count.

Function subset(c_k, item_list)

This function prunes the candidate list generated after calculating the support count for each product group in the candidate set based on list of products in each of the transaction and then removing all product groups whose support is less than the minimum support threshold. The collect_subsets function generates minimal required subsets from the list of products in a transaction and compares if the subset of products is part of the hash tree. If the subset if part of the hash tree then the product group in the candidate set is retained as part of the candidate set or else dropped from the candidate set.

Results from Apriori association analysis on market basket dataset

Following are results from Apriori with support threshold as 10 transactions and confidence threshold as 50%. Due to processing power limitations the algorithm was executed on 50k training records.

As we can see, the "likely buy" products "Banana" and "Large Lemon" in the predicted rules match with the most popular products that customers buy, as seen in the data analysis.

Rule	Confidence
{Limes,Bunched Cilantro} -> {Large Lemon}	0.50
{Organic Red Bell Pepper,Banana} -> {Organic Avocado}	0.58
{Broccoli Crown,Organic Strawberries} -> {Banana}	0.55
{Seedless Red Grapes,Organic Baby Spinach} -> {Banana}	0.50
{Limes,Asparagus} -> {Large Lemon}	0.54
{Seedless Red Grapes,Limes} -> {Large Lemon}	0.77

Conclusion:

This project helped us learn the application of few data science techniques to the product sales data to (1) predict the product items that might possibly be reordered in future, using standard algorithms in scikit-learn and keras and (2) mine association rules that help discover relation among product items that have the higher probability of being ordered together.

The original dataset included 318k training order dataset, but because of the computational complexity and hardware limitations, we had to limit tests to 51k dataset.

The Apriori implementation is our own implementation of the original Apriori algorithm and implements all algorithm optimizations in the reference whitepaper (like early pruning of traversal paths that won't yield results during candidate set generation using hash tree).

References:

1. https://www.kaggle.com/c/instacart-market-basket-analysis
2. https://www-users.cs.umn.edu/~kumar001/dmbook/ch6.pdf
3. https://en.wikipedia.org/wiki/Apriori_algorithm
4. https://machinelearningmastery.com