Project report

Instacart Market Basket Analysis

Suhas Chowdary J (suhaschowdaryj@email.arizona.edu)

ELECTRICAL AND COMPUTER ENGINEERING

MATH 574M

Statistical Machine Learning

Fall-2017

Instructor: Prof. Hao Helen Zhang

Dec 09, 2017

College of Engineering

ABSTRACT

The objective of the project is to build a recommender system to predict the reordered items. It helps companies to manage its inventory effectively, improve the customer experience, analyze customer purchase patterns and to promote cross-selling. The instacart, an online grocery store has provided over 3.4 million anonymous orders of over 200,000 customers. To explore the data and to get proper insights of the data, Exploratory Data analysis has been performed. Feature engineering is used to explore various hidden features which helps in building robust models. We have used adaboost and Extreme gradient boosting (XgBoost) to build the predictive models in R language. A comparative study has been conducted on the performance of the two models. With adaboost we have achieved an F1 score of 0.349 whereas with XgBoost an F1 score of 0.416 is obtained.

1.INTRODUCTION

Online shopping seems to be entering a new era ruled by advances in technology. It is found that online shopping enterprises are receiving almost 43% of the revenue from repeat customers. Customer retention is one of the major income generating source for retail business. Analyzing the customer purchase patterns will help business owners to better predict the products he/she will purchase again. Also targeting the right set of customers for selling their products will help to improve profits, decrease advertisement costs and drive up the reachability of the products. Thus, building robust predictive models plays a significant role in improving the business.

This project focus on building predictive models for an online grocery store to predict what all products will a customer reorder in his next purchase. This problem face a cold start where we have to predict purchase of new items which we have never seen before, which is not the case with general recommender systems. For example, Netflix might recommend you a new movie based on the movie you previously watched where user needs and preferences are assumed fairly to be constant. Whereas, it's less clear that you'll want to reorder a fresh batch of almond butter or eggs if you bought them yesterday. Understanding

temporal behavior patterns makes it fairly a challenging problem when compared to standard predictive systems.

In the following sections, we have discussed about the learning models used followed by an overview of the data set and exploratory data analysis. In the later sections, we have briefed about the feature engineering, implementation details of XgBoost and adaboost models and a brief discussion on the results obtained.

2. LITERATURE REVIEW

2.1 ADABOOST

The AdaBoost algorithm proposed by Freund and Schapire [1] is a built on the boost-by-majority algorithm [2] and weighted majority algorithm [3]. The upper bound of the classification error of the boost-by-algorithm should be known ahead of time, which cannot take advantage of classifiers with small error rate. Adaboost using the adaptive approach sequentially apply the weak learning algorithm to keep the upper bound in limits and an ensemble model using a weighted majority vote to produce a strong model. During each round, it penalizes inaccurate classifiers as shown in fig 1 while bounding the error.

Algorithm 1 The AdaBoost algorithm.

Given: $(x_1, y_1), ..., (x_m, y_m)$; $x_i \in \mathcal{X}, y_i \in \{-1, 1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, ..., T$:

1. Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j$; $\epsilon_j = \sum_{i=1}^{m} D_t(i) I[y_i \neq h_j(x_i)]$
2. If $\epsilon_t \geq 1/2$ then stop
3. Set $\alpha_t = \frac{1}{2} \log(\frac{1+\epsilon_t}{\epsilon_t})$
4. Update

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Fig 1. The pseudo code for Adaboost algorithm.

2.2 XGBOOST

Extreme gradient boosting (XgBoost) is a scalable machine learning model developed using the penalized gradient tree boosting [4]. It uses sparsity-aware algorithm, block structures and continued training for parallel tree learning and to improve scalability. It performs additive optimization in functional space and incorporates a regularized model to prevent overfitting. Weighted quantile sketch which performs merge and prune operation simultaneously is used for tree learning [5].

3.DATA SET

The data is provided by instacart which is an online grocery shop. The data set can be found at: https://www.instacart.com/datasets/grocery-shopping-2017 . It consists of over 3.4 million anonymous grocery orders collected from over 200,000 users.

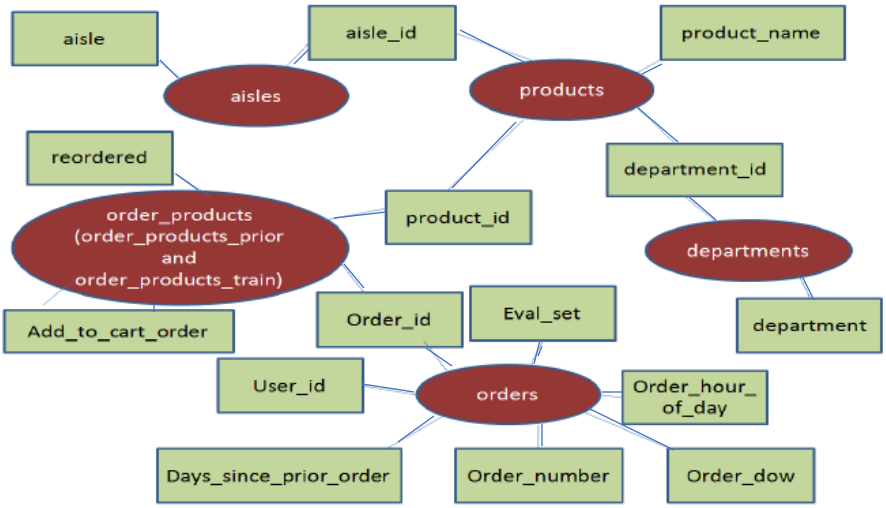

Fig 1. Overview of datasets and their dependencies.

The data is distributed over six datasets:

- Aisles: Consists of 134 aisles like butter, candy chocolate etc. mapped with aisle_id.

aisle_id	aisle
1	Prepared soups salads
2	Specialty cheeses
3	Energy granola bars

- Departments: Consists of 21 departments like bakery each mapped with department_id.

department_id	Department
1	Frozen
2	other
3	bakery

- Order_products_prior: It contains 32434489 order details with order_id, product_id, add_to_cart_order and reordered variables.

order_id	product_id	Add_to_cart_order	reordered
2	33120	1	1
2	28985	2	1
2	9327	3	0

- Order_products_train: It contains 1384617 training orders with order_id, product_id, add_to_cart_order and reordered variables.

order_id	product_id	Add_to_cart_order	reordered
1	49302	1	1
1	11109	2	1
1	10246	3	0

- Orders: It consists of user details and corresponding 3421083 orders with 7 features.

order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	Days_sinceprior_order
2539329	1	prior	1	2	8	NA
2398795	1	prior	2	3	7	15
473747	1	prior	3	3	12	21

- Products: It consists of over 49688 unique products mapped with the corresponding aisle and department ids.

product_id	product_name	Aisle_id	Department_id
1	Chocolate Sandwich Cookies	61	19
2	All-Seasons Salt	104	13
3	Robust Unsweetened Teaa	94	7

4.EXPLORATORY DATA ANALYSIS

We have used exploratory data analysis to uncover underlying structure of the data, check for missing data and to identify influential variables. Distribution of data over various aisles and departments are shown in figure 2.

It is found that majority of the people reorder again after one week or one month as shown in Figure 3. Perishable items like vegetables, curry leaves are reordered more after one week whereas non-perishable products such as soaps, paper towels are ordered for one month. People order the same items, if they order again on the same day whereas, if they order after one month, they are likely to try out new things as shown in figure 4.a. Also 59% of the ordered items are reorders figure 4.b.

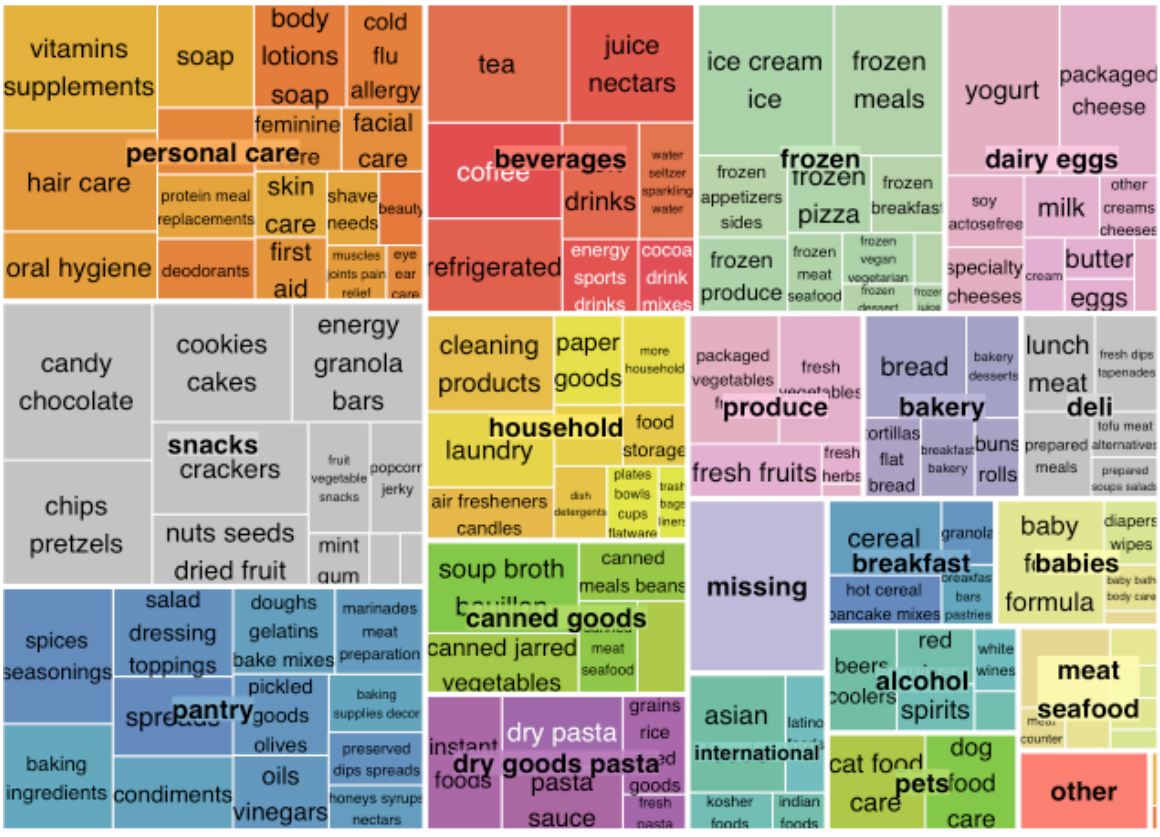

Fig 2. Distribution of Aisles over departments.

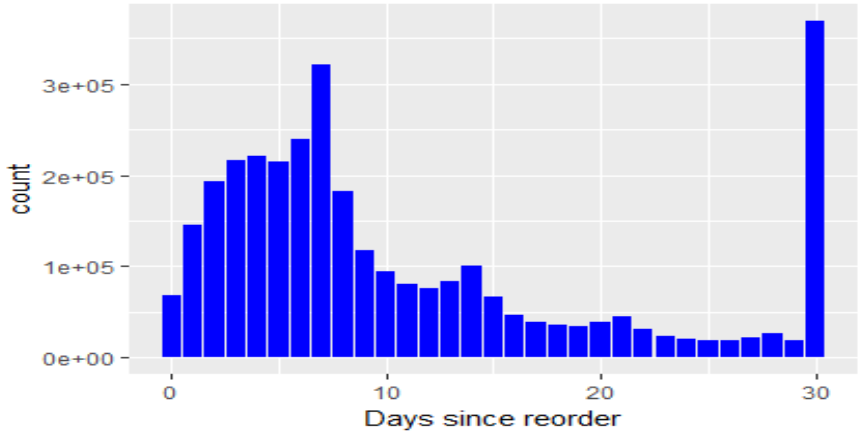

Fig 3. Days since prior order.

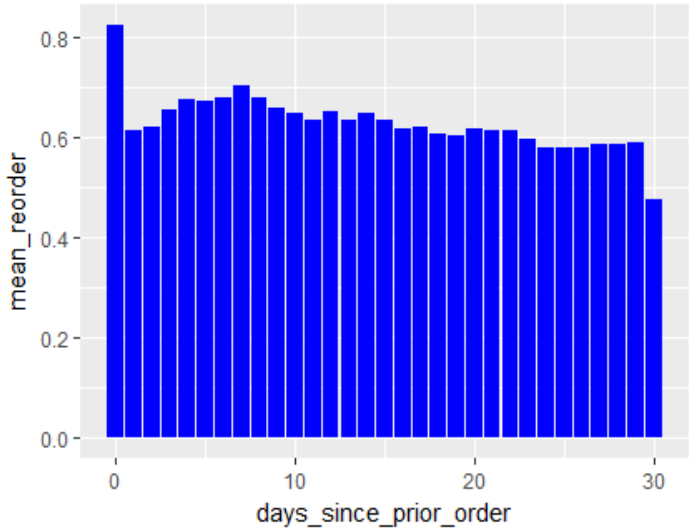

Fig 4.a. Time of last order and probability of last order.

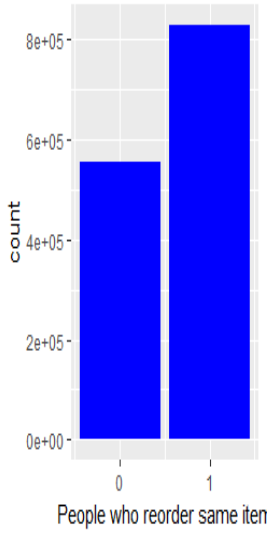

Fig 4.b. Reorder count

5.FEATURE ENGINEERING

Feature engineering, an integral part of machine learning, is often one of the most critical steps in developing models. It helps in to build a parsimonious and interpretable model. It helps to enhance the model prediction power and to improve the precision of the estimates. The instacart dataset consists of only 8 features which will not capture enough information to build an effective predictive model. We have generated 27 features or variables based on users, products and the interactions between user and products [table 1]. The following criteria is used to build the features.

1. Product features: Patterns based on the products
2. User features: Depends on the behavior of the customer.
3. Product x User features: How users are inclined or correlated to products.
4. User x Time features: How users have ordered products at various intervals of time.

Table 1. Additional features created.

	Product features	**User features**	**Product x User features**	**User x Time features**
1	Prod_reorder_prob	User_reorder_prob	User_prod_reorder_prob	Timegap_of_order
2	Prod_recent_orders	Repeated_orders_ratio	User_prod_timegap	Average_timegap
3	Prod_number_of_orders	User_mean_dow	Up_unique_products	
4	Prod_mean_dow	User_order_streak	Up_total_orders	
5	Prod_order_streak		Up_average_dow	
6			Up_inital_orders	

6.MODEL IMPLEMENTATION

6.1 MODEL SELECTION

We have used Xgboosting and adaboost for the prediction task. The predictive nature of task requires the model to effectively predict probabilities associated with multiple classes of the target feature. Some reasons for choosing ensemble models for prediction are:

• With the use of stronger objective functions, ensemble models are optimized to perform in logarithmic and sub-linear times.

• Ensembles are excellent at learning complex dependencies between features.

• The ensemble predictions are independent of the scale of the input in most of the cases.

• Identifying the best hypothesis is easy in search space with the help of ensembles.

6.2 PARAMETER TUNING

Parameter tuning is one of the critical steps in machine learning. It helps the model to improve performance, prevents overfitting, control computational overload and to achieve bias-variance tradeoff. Choosing the right set of parameters is frequently critical to realizing a model's full potential. We have used grid search to find the optimal parameters for Xgboost and adaboost. The grid search parameters for Xgboost and adaboost are provided in table 2(a) and table 2(b) respectively. Some of the parameters we considered during the parameter tunings are:

• Learning Rate: Small rates need more time to converge where as high learning rates often skips the optimal point. The learning rates (eta) fed into grid search are 0.01, 0.03, 0.1, and 0.3.

• Over-fitting prevention: To prevent overfitting, we have restricted the maximum tree depth to six in both the models. For Xgboosting, we have also used minimum child weight and gamma parameters to prevent overfitting. To make a partition of leaf node, gamma is used as minimum loss reduction. In adaboost, number of iterations and shrinking parameter (nu) are used to reduce overfitting.

• Computational overload: To prevent computational overload, we have constrained tree depth and maximum iterations in both Xgboost and adaboost models.

Table 2(a) Xgboost Grid parameters

Parameters	**Input values**
Learning rate (eta)	0.01,0.03,0.1,0.3
Maximum depth of tree	4,5,6
Minimum child weight	6,7,8
subsample	0.74,0.76,0.78
Colsample_bytree	0.94,0.95,0.96
gamma	0,0.1,0.5,1

Table 2(b) adaboost Grid parameters

Parameters	**Input values**
Shrinkage parameter (nu)	0.03,0.1,1,0.3
Bag.frac	0.4,0.5,0.6
Max.iter	30
cp	0.01,0.02,0.03
mfinal	60,70,80
maxdepth	4,5,6

6.3 MODEL TRAINING

We have trained the models with Xgboost and adaboost models. The best parameters obtained from grid search are shown in table 3(a). We have used logistic regression using logloss as evaluation metric to train the model in 80 iterations. For adaboost model, we have used exponential loss with complexity parameter (cp) as 0.02 for 60 rounds. The parameters used to train adaboost are shown in table 3(b). The performance of both the models and contrast between the models are discussed briefly in next section.

Table 3(a). Best Xgboost model parameters

Objective	reg:logistic
Eval_metric	logloss
Eta	0.1
Max_depth	6
Subsample	0.76
Min_child_width	7
Alpha	2e-05
Lamda	10
Gamma	0.5
Colsample_bytree	0.95

Table 3(b). Best adaboost model parameters

loss	Exponential
nu	0.1
maxdepth	4
Min_child_width	6
cp	0.02
iter	60
Bag.frac	0.5
Gamma	0.5
Kappa	0.3

7. RESULTS AND ANALYSIS

7.1 EVALUATION METRICS

We have used F1 metrics to evaluate the performance of the models. The F1 score for each order is calculated between the set of predicted re-ordered products and the set of true re-ordered products. Precision is defined as the percentage of correctly predicted reordered products among all predicted reordered products. The recall is given by the fraction of predicted re-ordered products that are indeed reordered among all true reordered products. F1 score is defined as:

$$F_1 = 2.\frac{1}{\frac{1}{recall}+\frac{1}{precision}} = 2.\frac{precision.recall}{precison+recall}$$

7.2 PERFORMANCE EVALUATION

With adaboost we have obtained a score of 0.349 whereas using Xgboosting, we have achieved an F1 score of 0.416. Both the algorithms try to boost the performance of a simple base learner by shifting the focus towards problematic observations that are difficult to predict. Adaboost makes changes to the subsequent models by up-weighting the misclassified observations. Whereas, Extreme gradient boosting identifies difficult observations by computing large residuals in the previous iterations. As the adaboost builds models by reweighting based on previous models, it is susceptible to outliers. If an outlier causes a previous model to provide a high error, the subsequent models adjust the parameters accordingly. The function margin by using exponential loss function depends on the number of iterations performed. Both these issues tend to overfit the data and leads to poor generalization.

Extreme Gradient Boosting takes a safe approach because of its cost function over different models. It monotonically (greedily) adjusts the parameters to reduce the cost function, thus performing predictably better than adaboost. Xgboost attempts to fit the base learner not to re-weighted observation as in adaboost, but to negative gradient vector of the logistic loss function evaluated at the previous iterations. It descends the empirical risk via steepest gradient descent in function space, where the function space is provided by the base-learners. This procedure effectively leads to data-driven variable selection during the model estimation. Also, the spare awareness, out-of-core computations and cache awareness makes Xgboost a robust model in building the predictive models. These made Xgboost to perform better than adaboost by huge margin.

8. CONCLUSION

In E-commerce industry, predicting the purchase patterns of users is key step to improve the business. It helps in better understanding of the consumer, increasing sales and revenue, cognitive purchasing and

assists consumers with proper items. For this we have built a recommender system to predict the reordered items using the data provided by an online grocery store, instacart which consists of over 3.4 million orders. An extensive exploration of the data has been done for feature extraction. We have used Adaboost and XgBoost to build the models. With Adaboost, we have achieved an F1 score of 0.349, whereas with Xgboost we have obtained F1 score of 0.416. Adaboost using exponential loss is overfitting the data and also computationally too expensive. XgBoost, by incorporating boosting with gradient loss for each iteration is performing better when compared to adaboost. Also owing to its sparse awareness and weighted quantile sketch, it is scalable and less susceptible to overfitting.

9. FUTURE WORK

Even though the Xgboost is performing well with limited resources available, there is still a lot of scope to build a better predictive model. Better feature extraction methods can be implemented to build robust models. Also, to process huge data the model can be implemented in distributed environment for better scalability and execution speed. We can stack different models using ensembles where each model can be used to capture dedicated patterns of the data.

REFERENCES

[1] Y. Freund and Schapire R.E. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1):119–139, August 1997.

[2] Yoav Freund. Boosting a weak learning algorithm by majority. Information and Computation, pages 121(2):256–285, September 1995.

[3] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. Information and Computation, pages 108:212–261, 1994.

[4] Chen, Tianqi, and Carlos Guestrin. "Xgboost: A scalable tree boosting system." Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016.

[5] M. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. In Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, pages 58–66, 2001.

[6] Mayr, Andreas, et al. "The evolution of boosting algorithms." Methods of information in medicine 53.6 (2014): 419-427.

[7]. “The Instacart Online Grocery Shopping Dataset 2017”, Accessed from https://www.instacart.com/datasets/grocery-shopping-2017