# Repeat Buyer Prediction with XGBoost

Mufu Han
Rutgers University
Piscataway, NJ, USA
mufu.han@rutgers.edu

Qianhan Chen
Rutgers University
Piscataway, NJ, USA
qianhan.chen@rutgers.edu

## ABSTRACT

In this paper, we focus on calculating probability of repeat buyers for different merchants based on their sales records and customers' activities. We used XGBoost to build our prediction model. Our paper consists of eight parts: Introduction, Data and Algorithm Overview, Data Exploring, Model Training, Performance, Future Improvement, Conclusion and Acknowledgements.

## 1 INTRODUCTION

Merchants release spectacular discounts and post advertisements on festivals, like Black Friday and Double 11(Nov 11), in order to promote sales and attract more potential customers. However, many of the attracted customers are one-time deal hunters. Investments of advertisements and sales promotions are actually in vain for those customers, and promotions may have little long-lasting impact on sales.

To alleviate this problem, merchants need to identify potential repeat buyers from all customers, which can help to improve the return of sales promotions and reduce general advertisements fee. But it's challenging to target repeat buyers, especially for those fresh customers.

In this paper, we are going to present our solution, which consists of comprehensive feature engineering and model training. We have extracted features from sales record and user behaviour record, and trained an eXtreme Gradient Boosting model. Then use our model to predict which fresh buyer will turn into a repeat buyers.

With our intelligent predictor, merchants are able to post precise advertisement and lay down precise strategy to attract customers who are more likely to be a repeat buyer in the future.

## 2 OVERVIEW

### 2.1 Data Overview

The data we picked is from a competition of Alibaba Tianchi. All kinds of objects have an unique id instead of name, which is a defect for visualization. But the merits of it overwhelm the demerits. First, using this data set spares us from data crawling, formatting and cleaning. The data correctness and credibility are insured. We can concentrating on processing and algorithm to generate an expected result. Second, we can reach some commercially confidential data. For example, in this case, merchants' sales records and customers' action records are not allowed and no way to be crawled. Third, we have a platform to evaluate our work by comparing score with others.

There are mainly three types of data:

- customer to merchant: repeat buyer/one-time buyer/not buyer
- customer action to item: click, add to Cart, purchase, add to favourite
- customer's information of age and gender

### 2.2 Algorithm Overview

Based on the feature of data, using single decision tree will cause over-fitting. We mainly use eXtreme Gradient Boosting(XGBoost), which derives from gradient boosting decision tree(GBDT).

XGBoost consists of Classification and Regression Trees and other basic classifier such as linear classifier, thus XGBoost generates decisions by multiple related decision trees. Further, XGBoost derives value of loss function by second-order Tailor expansion, using first-order and second-order derivative meantime, which relatively yield a more precise value.

Classification and Regression Trees have several advantages in terms of this data set. First, Normalization is not required because its high toleration of value range of input. Second, it can learn about the relations among features and is easy to expand.

## 3 DATA EXPLORING

We have used Spark[7] to explore and visualize our data. There are a lot of interesting facts we can observe in Ali's user behaviour log.

From Figure 1 and Figure 2, we explore that the data time stamps range in several months and end up in sales promotion. And the customers' activities drastically increase in sales promotion in comparison to other times.

Based on Figure 3 and Figure 5, we can see the majority of customers are female customers whose ages range in twenty-five to thirty-four. Throughout the period, female customers are more active and have stronger inclination to purchase. Thus, seemingly female customers are more likely repeat buyer. But it is just conjecture. We still need more solid evidence to draw the conclusion.

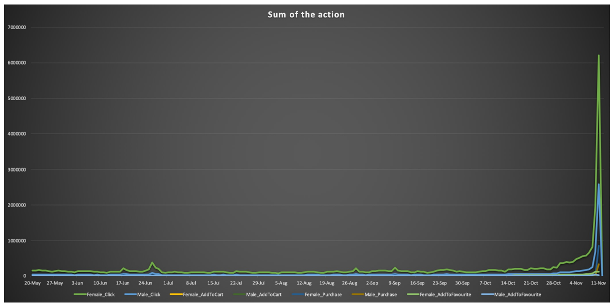
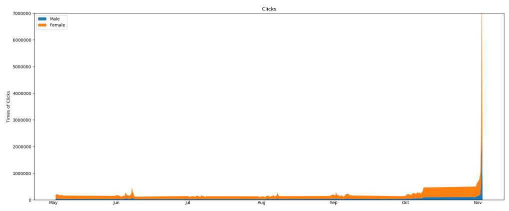


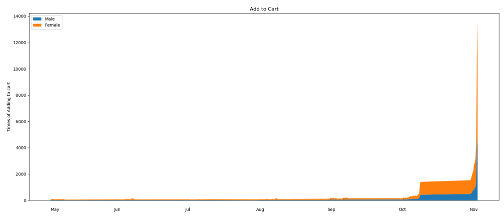
**Figure 1: Sum of Action**

Furthermore, it is not precise enough to identify repeat buyers from prototypes. Other than group prototype, customers need to be evaluated according to their personal actions, such as the their personal rate of purchases to clicks. In this way, personal behavior becomes an important factor for repeat buyer behavior, because the one who more likely to purchase is prone to be a repeat buyer.

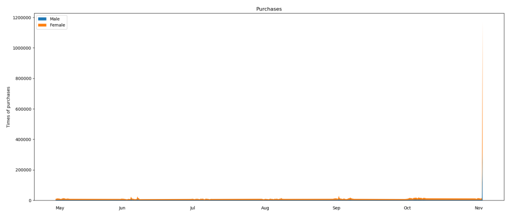
[Clicks]



[Add to cart]



[Purchases]



[Add to favourite]

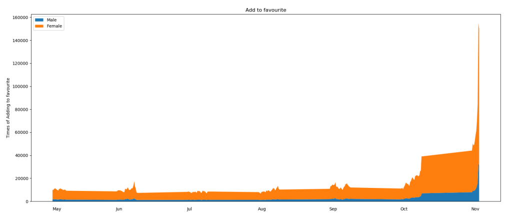


**Figure 2: Detail of each action**

As the figures show, data is imbalanced. Some of the actions are manipulated by merchants' requirements. Those extraordinary actions shouldn't be included, but for now we couldn't identify them from ordinary actions.

These statistic data is not merely used as features of model training, but also it can be complement of the conclusion. It provides a universal picture of customers, which helps merchants to make strategies target oriented. If necessary, the record of sales promotion may not be included in model training because of its anomaly.

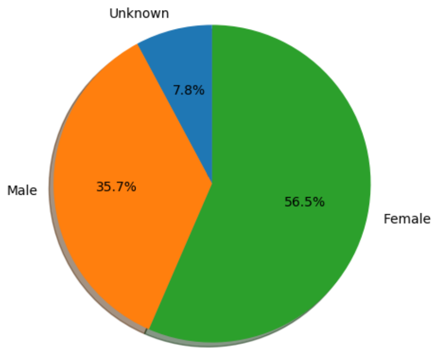


**Figure 3: Customer gender proportion**

Apparently, there is a relations of Clicks and Purchases. It is an essential feature that reveals customers' personal buying impulse for Clicks actions' inevitability before Purchases. However, add-to-cart and add-to-favourite are not. We can simply generate cart conversion or favourite conversion. But it could be misleading because many customers don't add commodities to cart or favourite before they buy, especially for male customers. Thus, those conversion rates couldn't be factors to predict a customer's willing to buy.

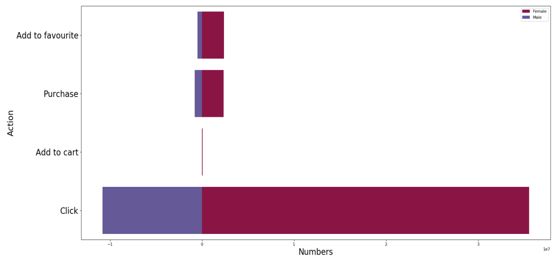


**Figure 4: Customer activity distribution based on gender**

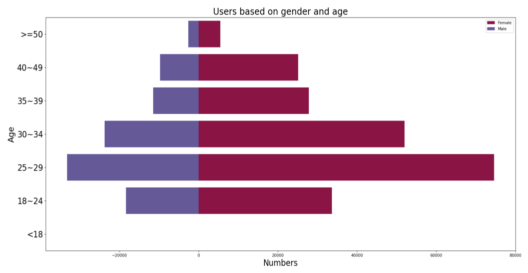


**Figure 5: Customer age distribution**

In Figure 4, it shows that purchases are far more than add-to-cart. The cart conversion rate is useful only when we need prediction of customers' willing to pay items in cart. On the contrary, it also misleads the model to evaluate customers according to customers' personal add-to-cart action.

Figure 5 shows the proportion of customers' age. One of the problem is customers are not totally represented by listed age range. In reality, for those customers who is under eighteen years old, their actions are included in their parents' accounts. Thus, some sorts of merchants which target customer are elders or kids, their target customers maybe turn out to be twenty to thirty-five years old. It will be unwise to post advertisement depends on statistic numbers without logical judgement. Therefore, data is straightforward but sometimes not the fact. It should be a reference but not a decision.

## 4 MODEL TRAINING

### 4.1 Data Cleaning

We download our data from Tianchi. This data set contains null values and missing values. Before feature engineering and training, we first prepare our data with data cleaning.

In user's age and gender, there are some null values for unknown age or gender. We fill these null with 2, which is a new label to denote that we do not know it. There are some null values in brand_id and item_id, we simply fill them with 0.

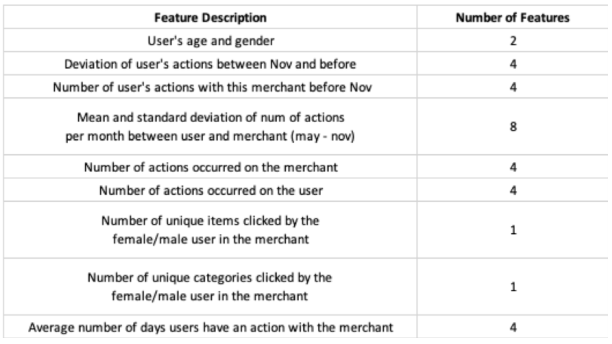
We did not normalize our numerical features because XGBoost is a tree-based model, it does not rely on assumptions of linear regression.

### 4.2 Feature Engineering

The data set contains two csv files, user profile and user activity log. The user activity log data contains five entities: users, merchants, brands, categories and items. It also contains users' actions with these five entities. The characteristics of these entities and their interactions can help us to predict which user-merchant pair will be positive. For example, merchants selling tissue are more likely attract repeated buyers. Merchants selling TVs are less likely, because few people will buy two TVs within six months. Based on user activity log data, we generate a large amount of features to describe users, merchants and their interactions.

The original training/testing data only contain user_id and merchant_id. We expanded them by merging new generated features into them. Features are basically divided into three parts: customers' statistic, merchants' statistic and general statistic. In each part, each column of data is counted. For example, in customer's statistic, for each customer, we count the number of items, categories, merchants and brands that customer had interactions with. Further, the age and gender are formatted as a Boolean value, namely, if one customer's age range in twenty to twenty-five, the column of age_2 is 1 while others are 0. Besides, rates of purchase to other actions are inserted as features based on different parts.

We have generated 38 features. The following table gives a summary of the types of features contained by our data.



| Feature Description | Number of Features |
|---|---|
| User's age and gender | 2 |
| Deviation of user's actions between Nov and before | 4 |
| Number of user's actions with this merchant before Nov | 4 |
| Mean and standard deviation of num of actions per month between user and merchant (may - nov) | 8 |
| Number of actions occurred on the merchant | 4 |
| Number of actions occurred on the user | 4 |
| Number of unique items clicked by the female/male user in the merchant | 1 |
| Number of unique categories clicked by the female/male user in the merchant | 1 |
| Average number of days users have an action with the merchant | 4 |

**Figure 6: feature description**

### 4.3 Dealing with Imbalanced Data

We observed that imbalanced data is distributing model. The model always trends to mark all users as negative, which is a non-repeated buyer. Such simple decision can guarantee a high accuracy, Because the positive samples only accounts for 6% in the data.

To eliminate the effects of unbalanced data, we used over-sampling and increased weight on positive label. For over-sampling, we used SMOTE - Synthetic Minority Over-sampling Technique[4] to over-sample our positive samples. By over-sampling, our model can be exposed to more positive samples, which can improve our model's resolution on positive label.

We also increase weight on positive samples to increase penalty when model incorrectly mark positive as negative. We want to stop our model from just marking all samples as negative.

### 4.4 Training

we have trained various classification models, including Linear Regression, Random Forest, GBM and XGBoost. XGBoost performed the best in them. So we will use XGBoost as our model. We used grid search and referenced a complete guide to parameter tuning in XGBoost[6] to set our parameters. The predictive model in the experiments is XGBoost with the following parameter setting: scale_pos_weight=10, eta=0.01, nrounds=2000, subsample=0.8, max_depth=7, min_child_weight=200. We used XGBoost and Scikit-learn[2] to split train and test data and fit our model.

## 5 PERFORMANCE

We have done concrete experiments to show our model's performance. Accuracy is not suitable for measuring our model, because our data is label-imbalanced. Thus, we used AUC score, ROC curve and confusion matrix to summarize our classification algorithm's performance. We got 0.67 on AUC, which is close to the score we got on Tianchi test data. We did not use SMOTE algorithm at last, because SMOTE will cause AUC dropping. This might because SMOTE over-sampled too many samples in our training data, which will lead to over-fitting. Increasing weight on positive label works fine. Before increasing weight, our model's AUC only achieves 0.62. We set scale_pos_weight=10, and it performs way better than model without increasing weight, got a 0.67 on AUC.

Feature engineering played a significant role in our project. When we only generated 8 features, the XGBoost model can only

14

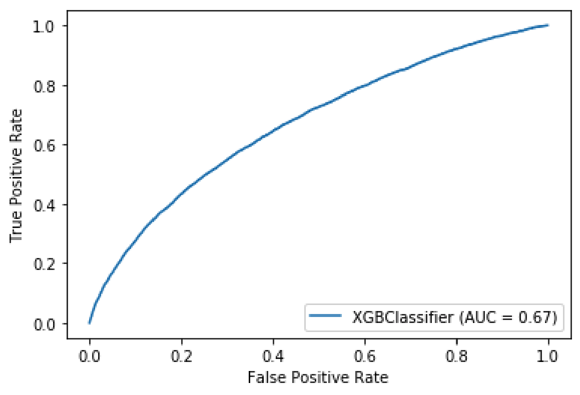


**Figure 7: ROC Curve**

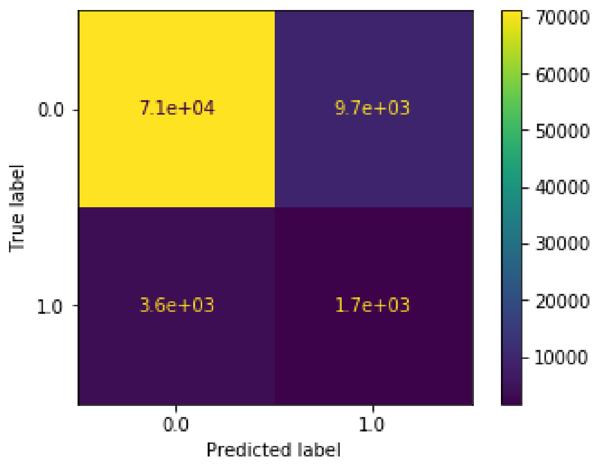


**Figure 8: Confusion Matrix**

got 0.52 on AUC. Fortunately, as we generate more features, the model's performance gradually improves. When we generate over 24 features, the performance of the model has stabilized.

Our model achieve a score at 0.6756 on Tianchi's online test data and rank 104 out of 2512. The number 1 on the leaderboard got a 0.7049.

## 6 FUTURE IMPROVEMENT

After comparing our solution with the competition champion's solution[1], we find that the main difference between us and them is features engineering. They generate more than 1300 features to describe users and merchants. And according to their paper, decreasing number of features will cause a significant decrease on score. So no such feature is a strong indicator of classification here. Also, if we have more time, we can blend multiple classifiers to gather a further improvement on performance.

## 7 CONCLUSION

16

In this paper, we presented our solution for the repeat buyer prediction competition hosted on Ali Tianchi. We generated a large number of features to capture users and merchants characteristics. And we found that for repeat buyers prediction, sophisticated feature engineering is more important than prediction methods. Increasing penalty on minor label works great for imbalanced data, but over-sampling does not work as our expected. At our next step, we hope that we can explore how to automate feature extraction and selection process on various data sets.

## 8 ACKNOWLEDGEMENTS

We would like to thank Professor Gerard de Melo for his support in this semester. We would like to thank Scikit-learn, Pandas, Numpy, xgboost, matplotlib, Spark and all developers in open source community, this project would have been impossible without them.

## REFERENCES

[1] Liu, G., Nguyen, T. T., Zhao, G., Zha, W., Yang, J., Cao, J., ... Chen, W. (2016, August). Repeat buyer prediction for e-commerce. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 155-164). ACM.
[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
[3] Lemaître, G., Nogueira, F., Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. The Journal of Machine Learning Research, 18(1), 559-563.
[4] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.
[5] Chen, T., Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). ACM.
[6] Aarshay Jain. (2016, March 1). Complete Guide to Parameter Tuning in XGBoost with codes in Python, from https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/
[7] Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., Stoica, I. (2010). Spark: Cluster computing with working sets. HotCloud, 10(10-10), 95.