Repeat Buyer Prediction for E-Commerce

Guimei Liu*, Tam T. Nguyen*, Gang Zhao#, Wei Zha*, Jianbo Yang§
Jianneng Cao*, Min Wu*, Peilin Zhao*, Wei Chen#

*Data Analytics Department, Institute for Infocomm Research, Singapore 138632,
{liug,nguyentt,zhaw,caojn,wumin,zhaop}@i2r.a-star.edu.sg

#Development Bank of Singapore, {george.g.zhao, nus.waltchan}@gmail.com

§General Electric, jianbo.yang@ge.com

ABSTRACT

A large number of new buyers are often acquired by merchants during promotions. However, many of the attracted buyers are one-time deal hunters, and the promotions may have little long-lasting impact on sales. It is important for merchants to identify who can be converted to regular loyal buyers and then target them to reduce promotion cost and increase the return on investment (ROI). At International Joint Conferences on Artificial Intelligence (IJCAI) 2015, Alibaba hosted an international competition for repeat buyer prediction based on the sales data of the "Double 11" shopping event in 2014 at Tmall.com. *We won the first place at stage 1 of the competition out of 753 teams.* In this paper, we present our winning solution, which consists of comprehensive feature engineering and model training. We created profiles for users, merchants, brands, categories, items and their interactions via extensive feature engineering. These profiles are not only useful for this particular prediction task, but can also be used for other important tasks in e-commerce, such as customer segmentation, product recommendation, and customer base augmentation for brands. Feature engineering is often the most important factor for the success of a prediction task, but not much work can be found in the literature on feature engineering for prediction tasks in e-commerce. Our work provides some useful hints and insights for data science practitioners in e-commerce.

Keywords

Repeat Buyer Prediction; Feature Engineering; E-commerce

1. INTRODUCTION

Large business-to-consumer (B2C) e-commerce websites, such as Amazon and Alibaba, often run nationwide sales promotions on special days like Black Friday and Double 11 (Singles' Day). Merchants acquire new customers during these events. However, most new customers are one-time deal hunters, and promotions to them usually do not generate *return on investment* (ROI) as expected by merchants. Therefore, merchants need to identify potential loyal ones from these new customers, so as to conduct targeted advertisements (and promotions) towards them to lower the promotion cost. It is difficult for any individual merchant to identify its potential loyal customers as it has little information on its new customers. B2C e-commerce websites instead have the click stream data and purchase history of all the customers at all the merchants on their platforms. Thus, they can learn the preferences and habits of the new customers from their historical data, and then predict how likely a new customer will buy again from a same merchant.

At IJCAI 2015, Alibaba hosted an international competition[1] for repeat buyer prediction based on the sales data of the "Double 11" day of 2014 at Tmall.com—the largest B2C platform in China. Double 11 is the biggest online shopping event in China with sales (in Tmall and Taobao) at US$5.8 billion in 2013, US$9.3 billion in 2014, and over US$14.3 billion in 2015[2]. Data provided to the competition include a number of merchants and their new buyers acquired during the event, and six months of user activity log data before the event. The task is to predict which new customers of a given merchant would buy items from the same merchant again within six months. These new buyers are called *repeat buyers* of the respective merchants.

We won the first place at stage 1 of the competition. Our winning solution consists of comprehensive feature engineering and model training. In particular, we generated various types of features to describe users, merchants, brands, categories, items and their interactions from different aspects. We have trained various classification models, including Factorization Machine [14, 11], Logistic Regression [1, 2], Random Forest [5], GBM [10], and XGBoost [6]. We have also used ensemble techniques to blend multiple classifiers together to further improve the performance.

The repeat buyer prediction problem can be formulated as a typical classification problem, as most of the competition participants did. Model training of this task is not much different from that of other classification tasks. Instead, feature engineering is the main component that distinguishes this task from others. Feature engineering, an integral part of data science, is often the key to the success of a machine learning project. It can be more difficult than learning

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

KDD '16, August 13-17, 2016, San Francisco, CA, USA

© 2016 ACM. ISBN 978-1-4503-4232-2/16/08...$15.00

DOI: http://dx.doi.org/10.1145/2939672.2939674

[1]http://ijcai-15.org/index.php/repeat-buyers-prediction-competition

[2]https://en.wikipedia.org/wiki/Singles_Day

Table 1: Statistics of training and testing data

data	#users	#merchants	#pairs	#positive pairs	positive%
train	212,062	1,993	260,864	15,952	6.12%
test	212,108	1,993	261,477	16,037	6.13%

Table 2: Statistics of log activity data

#rows	#users	#merchants	#items	#categories	#brands
54,925,330	424,170	4,995	1,090,390	1,658	8,444

Table 3: Statistics of action_types

click	add-to-cart	purchase	add-to-favourite
48,550,713 (88.39%)	76,750 (0.14%)	3,292,144 (5.99%)	3,005,723 (5.47%)

because it is domain-specific, while machine learning algorithms are largely general-purpose. Much trial and error can go into feature design, and it is typically where most of the effort in a machine learning project goes [8]. While thousands of classification algorithms have been proposed and studied in the research community, not much work has been reported on feature engineering for prediction tasks in e-commerce. Therefore, in this paper we focus on feature engineering. We will describe how to generate various types of features from user activity log data and study the importance of these features via extensive experiments. The features we generated can be used in all kinds of e-commerce applications, such as customer segmentation, product recommendation, and customer base augmentation for brands. We hope that our work can be valuable for data science practitioners, who need to develop solutions for prediction tasks in e-commerce.

The rest of the paper is organized as follows. Section 2 gives the problem description. Section 3 describes the features we have generated. Model ensemble is briefly described in Section 4. In Section 5, the importance of features is studied and top features are listed. Finally, Section 6 concludes the paper.

2. PROBLEM DESCRIPTION

For the repeat buyer prediction competition, the following data are provided as shown on the top of Figure 1: demographic information of users, six months of user activity log data prior to the "Double 11" promotion, and training and testing ⟨new buyer, merchant⟩ pairs, where the first purchase of the new buyer from the merchant is on the "Double 11" promotion. User demographic data contains the age and gender of users. The age values are divided into seven ranges. The class label of a training ⟨new buyer, merchant⟩ pair is known, and it indicates whether the new buyer bought items from the merchant again within six months after the "Double 11" promotion. The class labels of testing ⟨new buyer, merchant⟩ pairs are hidden. The task is to predict the class labels of the testing pairs. The competition was carried out in two stages. In stage 1, all the data were released to the contestants except for class labels of testing pairs, which were released after stage 1. Stage 2 ran on the cloud platform of Alibaba for bigger data, and the data were not released. Therefore, in this paper, we focus on the data of stage 1.

Table 1 shows the statistics of the training and testing data. The set of merchants in training data and that in testing data are the same except for a single merchant. Users in the training and testing data have no overlap. The second last column is the number of positive ⟨new buyer, merchant⟩ pairs such that the new buyer bought items from the merchant again within six months. The last column is the percentage of such positive pairs. The percentage of positive pairs is around 6%, which indicates that most of the new buyers are indeed one-time deal hunters.

The user activity log data contains the following fields: user_id, merchant_id, item_id, cat_id, brand_id, action_type and time_stamp. Action_type takes four values: 0 for click, 1 for add-to-cart, 2 for purchase and 3 for add-to-favourite. Products sold in different merchants are assigned different item_ids even if the products are exactly the same. Table 2 shows the statistics of the user activity log data. Many merchants in the log data do not have new buyers in the training or testing data. They are included in the log data because some new buyers visited them. The activities of the new buyers at these merchants are valuable information for inferring the preferences and habits of the new buyers.

Table 3 shows the number of the four types of actions. The majority of actions are clicks. The number of add-to-cart actions is very small, so we merge the add-to-cart actions with click actions.

The user activity log data provided in this competition are very typical in e-commerce prediction tasks. However, the log data are not in a form that is amenable to learning. We need to construct new features from them and then join the new features with the training and testing data. In the next section, we describe how we do this.

3. FEATURE ENGINEERING

The user activity log data contain five entities: users, merchants, brands, categories and items. The characteristics of these entities and their interactions can be predictive of the class labels. For example, users are more likely to buy again from a merchant selling snacks than from a merchant selling electronic products within six months, since snacks are cheaper and are consumed much faster than electronic products. We generated a large number of features to describe the characteristics of the five types of entities and their pairwise interactions. In the rest of this section, we first give an overview of all the generated features, and then describe the features in details.

3.1 Overview of features and profiles

The features we generated range from basic counts to complex features like similarity scores, trends, PCA (Principal Component Analysis) and LDA (Latent Dirichlet allocation) features. All the features of an entity form the profile of the entity. We have five entity profiles and five interaction profiles as shown at the bottom of Figure 1. Table 4 gives a summary of the types of features contained in these profiles. User-merchant interaction is the most important interaction among the five pairwise interaction profiles as the task is to predict whether a user will return to a merchant to buy again. Therefore, user-merchant profile contains more features than the other interaction profiles.

The original training/testing data contain only user_ids and merchant_ids as shown on the top of Figure 1. We expanded the training/testing data by adding age_range and gender of users, item_id, brand_id, and category_id as shown in the middle of Figure 1, where item_id is the id of the item bought by the user from the merchant on the Double 11 day,

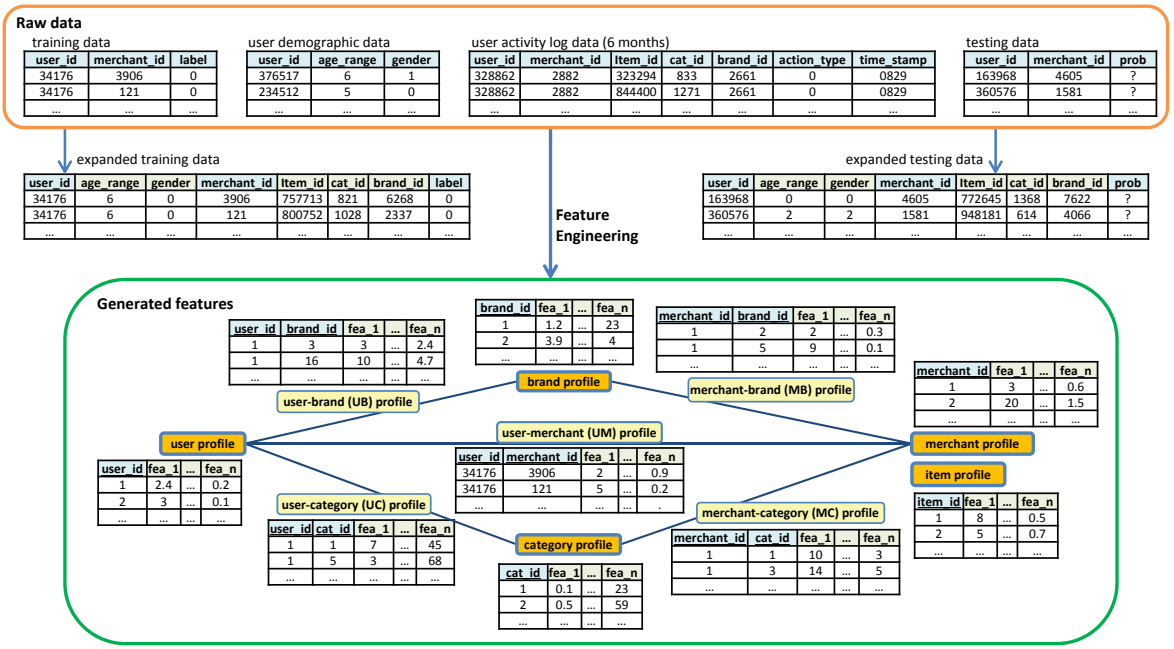


Figure 1: Feature engineering on raw data

Table 4: Summary of features and profiles.

	feature types	user profile	merchant profile	category profile	brand profile	item profile	user-merchant (UM) profile	user-brand (UB) profile	user-category (UC) profile	merchant-brand (MB) profile	merchant-category (MC) profile
count/ratio	overall action count/ratio	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	overall day count	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	monthly action count/ratio	✓	✓	✓	✓	✓	✓				
	product diversity	✓	✓				✓				
	penetration		✓	✓	✓	✓					
aggregation	monthly aggregation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	merchant aggregation	✓									
	user aggregation		✓								
recent activity	Double 11 features	✓	✓	✓	✓	✓	✓				
	latest one-week	✓	✓	✓	✓	✓	✓				
	latest one-month						✓				
complex features	trend	✓	✓				✓				
	repeat buyer features		✓	✓	✓	✓				✓	✓
	market share									✓	✓
	similarity						✓				
	LDA features	✓	✓								
	PCA features		✓								
age/gender related	age related features		✓	✓	✓	✓					
	gender related features		✓	✓	✓	✓					

and brand_id and category_id are the brand and category of the item. If a user bought multiple items from a merchant on the Double 11 day, then the most frequent one is used. The features are joined with the expanded training/testing data as follows: entity features are joined based on their respective ids; age related features are joined by the respective entity id and age_range; gender related features are joined by the respective entity id and gender; interaction features are joined by the two entity ids involved.

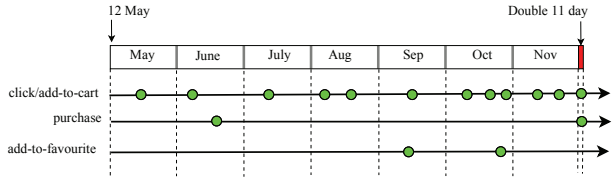


Figure 2: Action history of an example entity

3.2 Count/ratio features

Each entity has three types of actions—click, purchase and add-to-favourites—over the six-month period from 12 May, 2014 to 11 Nov, 2014. Figure 2 shows the action history of an example entity. The three types of actions are represented by green circles.

Action count, action ratio and day count. *Action counts* are number of clicks, purchases, add-to-favourite actions in each month (monthly counts) or over the whole data period (overall count). Count features are the basis for generating more complex features. For the entity shown in Figure 2, the monthly click counts are (1, 1, 1, 2, 1, 3, 3), the monthly purchase counts are (0, 1, 0, 0, 0, 0, 1), and the monthly add-to-favourite counts are (0, 0, 0, 0, 1, 1, 0). The overall counts of click, purchase, and add-to-favorite are 12, 2, and 2, respectively. *Action ratio* is the proportion of a particular action type over all action types, and it can be calculated in each month or over the whole data period. For the entity in Figure 2, the overall click ratio is $12/(12+2+2) = 0.75$, and June click ratio is $1/(1+1+0)=0.5$.

Day counts are the number of days with a particular action type in each month or over the whole data period. Day count features are mainly used to differentiate regular buyers from occasional buyers. For example, a user with 10 purchase actions in one single day is different from another user with the same number of purchase actions that spread over 10 different days. The latter user is considered a more regular buyer than the former.

Action count, action ratio, and day count features can be generated for pairs of entities as well. For example, for each ⟨user, merchant⟩ pair, monthly click counts are the number of times the user clicked some items of the merchant in each month. For a ⟨merchant, brand⟩ pair, overall purchase counts are the number of times products of the brand were purchased from the merchant over the whole data period.

Not all the monthly features are directly used, otherwise there will be too many features. Instead, we use these features to generate more complex features, like monthly aggregation features and trend features as described later.

Product diversity features. For a user, *product diversity* features are the number of unique items, brands and categories that the user clicked, purchased or added to favourites in each month or over the whole data period. For a merchant, product diversity features are defined in a similar way. For a ⟨user, merchant⟩ pair, product diversity features are the number of unique items, brands and categories of the merchant that were clicked, purchased or added to favourites by the user in each month or over the whole data period. The intuition behind product diversity features is that if a user is interested in more items of a merchant, then the user is more likely to buy again from the merchant (see Figures 6(a), 7(d) and 7(i)).

Penetration features. *Penetration* feature of an item is defined as the number of users, who have purchased the item in a given time interval. We have also computed penetration features for merchants, brands and categories. A large customer base usually indicates that the entity has a good reputation, so users are more likely to come back.

3.3 Aggregation features

Monthly aggregation features are mean, standard deviation, max and median of monthly action counts, monthly day counts, monthly product diversity counts and monthly penetration counts.

User aggregation features are calculated for merchants, brands, categories, ⟨merchant, brand⟩ pairs and ⟨merchant, category⟩ pairs. The *user-purchase-day-aggregation features* of a merchant are calculated by first counting the number of days that each individual user bought items from the merchant, and then calculating the mean, standard deviation, max and median over all the users of the merchant. *User-purchase-item-aggregation features* of a merchant are defined in a similar way over the number of unique items that each user purchased from the merchant. For click and add-to-favourite actions, user-action-day-aggregation and user-action-item-aggregation features are calculated in the same way for merchants. For other entities, only the purchase action is considered. User aggregation features are also generated by considering only users of a specific gender or age range.

The intuition behind user aggregation features is — given a merchant, if users visited it or bought items from it more than once on average, then new buyers of the merchant on "Double 11" day are more likely to come back as well (see Figures 5(c), 5(d), 6(e), 6(f), 7(c), 7(e), and 7(f)).

Merchant aggregation features are generated for users. Given a user, his/her *merchant-purchase-day-aggregation* features are calculated by first counting the number of days that the user bought items from each individual merchant, and then calculating the mean, standard deviation, max and median over all the merchants, from which the user had made at least one purchase. *Merchant-purchase-item-aggregation* features are calculated over the number of unique items that the user purchased from each merchant in a similar way. For click and add-to-favourite actions, merchant-action-day-aggregation and merchant-action-item-aggregation features are calculated in the same way.

Merchant aggregation features reflect users' habit. If a user tends to buy from or visit merchants multiple times on average, then he/she is likely to buy from new merchants again (see Figures 5(a), 5(b) and 7(l))

3.4 Recent activity features

Double 11 features are counts of clicks, purchases, add-to-favourites on the Double 11 day. The ratio of the double 11 counts to the overall counts are also calculated. For the entity in Figure 2, its Double 11 click count is 1, its Double 11 click ratio is 1/12=0.083 and its Double 11 buy ratio is 1/2=0.5. If a user has a high Double 11 buy ratio, then the user is more likely to be a one-time deal hunter.

Latest one-week features and **latest one-month features** are counts/ratio of clicks, add-to-favourites, and purchases in the last one week and in the last one month before Double 11, respectively.

3.5 Complex features

Trend features are calculated based on monthly features. Given monthly counts or monthly ratios $y=(y_1, y_2, \cdots, y_7)$ over seven months from May to November, the slope of the trend line is calculated as $\alpha = \frac{n\sum_{i=1}^{n}(x_i y_i) - \sum_{i=1}^{n}(x_i)\sum_{i=1}^{n}(y_i)}{\sum_{i=1}^{n}(x_i^2) - (\sum_{i=1}^{n}(x_i))^2}$, where n=7 and $x_i = i$.

We also calculated the deviation of the latest month from the previous months and normalized it using either mean or standard deviation as follows: $d_1 = \frac{y_7 - \mu}{\mu}$ and $d_2 = \frac{y_7 - \mu}{\sigma}$, where y_7 is the feature value in November, μ and σ are the mean and standard deviation of the feature values over the previous six months.

Repeat buyer features. *Repeat buyer number* of a merchant is defined as the number of users, who bought on at least two different days from the merchant. For items, brands and categories, *repeat buyer number* is defined as the number of users, who bought the item/brand/category on at least two different days. *Repeat buyer ratio* of a merchant/item/brand/category is defined as the ratio of repeat buyers to all buyers (including non-repeating buyers) of the merchant/item/brand/category.

A *repeat buy day* of a user at a merchant is defined as a day, such that the user bought items from the merchant both before and on the day. *Repeat day number* of a merchant is the sum of the repeat buy day of all its users. *Repeat buy day ratio* is the ratio of the repeat day number of a merchant to the sum of the buy days of all the users of the merchants.

Repeat buyer features are also calculated for pairs of entities. For example, for a ⟨merchant, brand⟩ pair, repeat buyer number is the number of users, who bought items of the brand on at least two different days from the merchant.

A high number or a high proportion of repeat buyers indicates that the entity is widely liked, so the customers are more likely to come back again. Our experiment results confirmed this (see Figures 7(a), 7(g) and 7(j)).

Market share features measure how important a brand/ category is to a merchant, or how important a merchant is to a brand/category. Take a ⟨merchant, brand⟩ pair as an example. Let N_{MB} be the number of purchases of the brand from the merchant, N_M be the total number of purchases from the merchant, and N_B be the number of purchases of the brand from all the merchants. Similarly, we define U_{MB} as the number of users buying the brand from the merchant, U_M the total number of buyers of the merchant, and U_B the number of buyers of the brand from all the merchants. The following four features are then generated:

1) *merchant's market share on the brand* $= N_{MB}/N_B$
2) *merchant's user share on the brand* $= U_{MB}/U_B$
3) *brand's market share within the merchant* $= N_{MB}/N_M$
4) *brand's user share within the merchant* $= U_{MB}/U_M$

The first two features measure how important a merchant is to a brand, and the last two features measure how important a brand is to a merchant. Similarly, we can calculate market share features for ⟨merchant, category⟩ pairs.

User-merchant similarity features measure how similar a user and a merchant are based on brands or categories. They are calculated based on the four market share features defined above and the preferences of users on brands/ categories. The preferences are measured by the times or the number of days the user clicked, purchased, or added to favorites the brand/category. Suppose that a merchant has five brands with respective market shares (0.1, 0.2, 0.05, 0.3, 0.01), and that the number of times of a user buying the five brands are (0, 1, 2, 0, 2). We can compute the inner product of the two vectors, and take it as the similarity score between the user and the merchant, that is, $0.1 \times 0 + 0.2 \times 1 + 0.05 \times 2 + 0.3 \times 0 + 0.01 \times 2 = 0.32$. We can also take the max, instead of sum, over all brands, then the similarity score is $0.2 \times 1 = 0.2$.

Intuitively, the more similar a user and a merchant are, the more likely the user will buy from the merchant again (see Figure 7(h)).

PCA features are generated based on the similarity between merchants. Give a pair of merchants, we use the number of users who bought items from both of them as their similarity score. The total number of merchants is 4,995. Therefore, a matrix of 4,995×4,995 is built. This matrix is highly sparse with most elements equal to 0. Simply adding it into the feature list does not obviously improve the accuracy of classification models, but dramatically increases the model training time. As such, we have applied PCA (principal component analysis) [3, 12] on the similarity matrix. Then for each merchant, the top-10 principal coordinates are used as merchant features.

LDA features. Latent Dirichlet Allocation (LDA) [4] is often used in text mining to retrieve topics from a corpus of documents. It views each document as a mixture of various topics, where each topic is characterized by a distribution over words. The retrieved distribution of the topics can be taken as a feature of the document. We first model users as documents and merchants as words. Given a user, we extract from log activity data all the merchants, from which the user purchased items, and treat these merchants as the words of the user's document. By applying LDA on these created documents, we generate features (i.e., distributions of topics over merchants) for users. Similarly, we model merchants as documents and users as words, and generate features (i.e., distributions of topics over users) for merchants. We set the number of topics to 40 based on the performance of predictive models.

3.6 Age/gender related features

Different user groups may favor different types of products. For example, clothes and cosmetics are more attractive to women while electric products are more appealing to men. As such, we generated features to describe the popularity of merchants, brands, categories, and items within different user groups, where users are grouped based on their gender or age range. These features include overall buy counts, monthly aggregation on monthly buy counts, penetration features and repeat buyer features. Only users of a particular age range or a particular gender are used to calculate these features.

3.7 Feature ranking

We have generated 1364 features in total. It is crucial to identify important ones and remove those that are of little use to reduce the training cost [13, 7]. During the competition, we tested both the wrapper method described in [15, 16] and the feature ranking function provided by XGBoost, and we found that all the methods yield very similar feature rankings. In this paper, we report the results by the feature ranking function of XGBoost. Besides ranking all the features together, we also group features based on their types or the profiles they belong to. We rank the features within each group separately, and output the top features. We have also evaluated the importance of each feature group by leaving it out.

4. MODEL TRAINING

In the competition, we have trained various classification models, including Factorization Machine [14], Logistic Regression [1], Random Forest, GBM [10], and XGBoost [6], where grid search was used to select the optimal parameters. XGBoost performed the best.

To further improve the performance, we used ensemble techniques to blend together the predictions made by the above single classifiers. The blending model is basically a weighted sum as defined below.

$$p(u, m) = \sum_{i=1}^{k} w_i \times p_i(u, m), \tag{1}$$

where $p(u, m)$ is the final probability that a user u will make a repeated purchase from a merchant m, $p_i(u, m)$ is the probability predicted by the i-th single model, w_i is the weight assigned to the i-th single model, and k is the number of single models.

We tested two methods to assign the weight w_i to the i-th single model. For the first method, we manually assign weights to single models, such that single models with higher AUC (the area under the ROC curve) scores receive bigger weights. For the second method, we built a classifier to learn the weights. In particular, we generated a k-dimensional feature vector for each user-merchant pair (u, m), where the i-th dimension is the probability $p_i(u, m)$ by the i-th

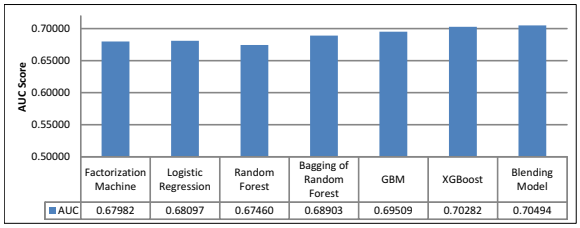


Figure 3: AUC of single models and the blending model

single model. We trained a linear model on these feature vectors to learn the weights.

Our experiments showed that manually assigned weights are often as good as and sometimes even better than the learned weights in this application. Therefore, in the competition, we mainly manually assigned weights to blend the predictions of different models together. We did it in an incremental manner – in each round we kept the best prediction thus far, and blended it with the prediction of a single model. If the resultant AUC score did not improve, we discarded both the blended model and the single model, otherwise, we updated the best prediction using the blended model.

Figure 3 shows the AUC scores of the single models on the testing data. The two linear models, Factorization Machine and Logistic Regression, performed closely. Although their scores are not really high, they contribute to the overall AUC score in the blending model. In ensemble algorithm family, Random Forest has the worst AUC score. However, we found that bagging of Random Forest models can improve the score significantly. XGBoost has the best AUC score of 0.70282. Comparing with the runner up Gradient Boosting Machine, its improvement is more than 0.7%. We have blended around 20 single models with various parameter settings and feature settings, and achieved an AUC score of 0.70494, which is an improvement of 0.21% over the best single model (i.e., XGBoost).

5. A PERFORMANCE STUDY

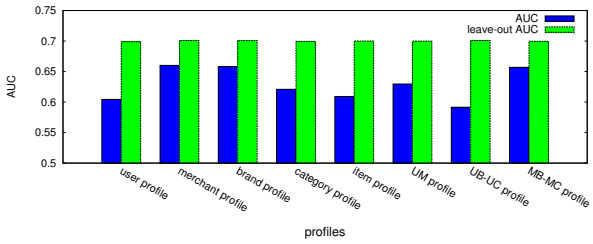
In this section we evaluate the importance of features in groups. We first conduct the experiments on training data by five-fold cross validation to measure the importance of each group of features (Section 5.1), and to find the top features locally in each group as well as globally in the full feature set (Section 5.2). Extensive experimental results show that some features are less important – removing them has only marginal effect on the performance of the predictive models. Such a finding can help a user to determine the subset of features to be applied in real applications for a good balance between model accuracy and training time. Then, we carry out the experiments on testing data (Section 5.3). We study how the prediction accuracy increases as we incrementally use more features to train models.

5.1 Importance of feature groups

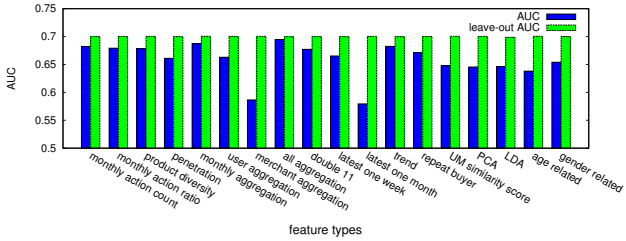
We have generated 1364 features in total. They are organized in groups either by type or by profile as summarized in Table 4 and discussed in Section 3. Features in a same group are generated based on the same hypothesis, and their importance are evaluated together. If one group

Table 5: Feature groups and their AUC scores

	Feature groups	#features	AUC	leave-out AUC
	all features	1364	0.70036	-
entity profiles	user profile	201	0.60442	0.69886
	merchant profile	221	0.6601	0.70096
	brand profile	90	0.65818	0.70071
	category profile	90	0.62087	0.69944
	item profile	70	0.60915	0.70002
interaction profiles	user-merchant profile	107	0.62952	0.69991
	UB and UC profile	26	0.59148	0.70088
	MB and MC profile	58	0.6569	0.69954
count/ratio	monthly action count	161	0.68227	0.70034
	monthly action ratio	163	0.6791	0.70026
	product diversity	61	0.67829	0.70029
	penetration	43	0.66095	0.70001
aggregation features	monthly aggregation	164	0.68729	0.70058
	user aggregation	88	0.66289	0.70031
	merchant aggregation	24	0.58637	0.70039
	all aggregation	276	0.6946	0.69973
recent activity	double 11	79	0.67713	0.69996
	latest one week	24	0.66516	0.69988
	latest one month	10	0.57935	0.70037
complex features	trend	109	0.68234	0.69989
	repeat buyer	72	0.67099	0.70014
	UM similarity score	192	0.64762	0.70043
	PCA	10	0.64546	0.70023
	LDA	80	0.64639	0.69877
age/gender related	age related	30	0.63794	0.70044
	gender related	40	0.65397	0.70017



(a) entity and interaction profiles

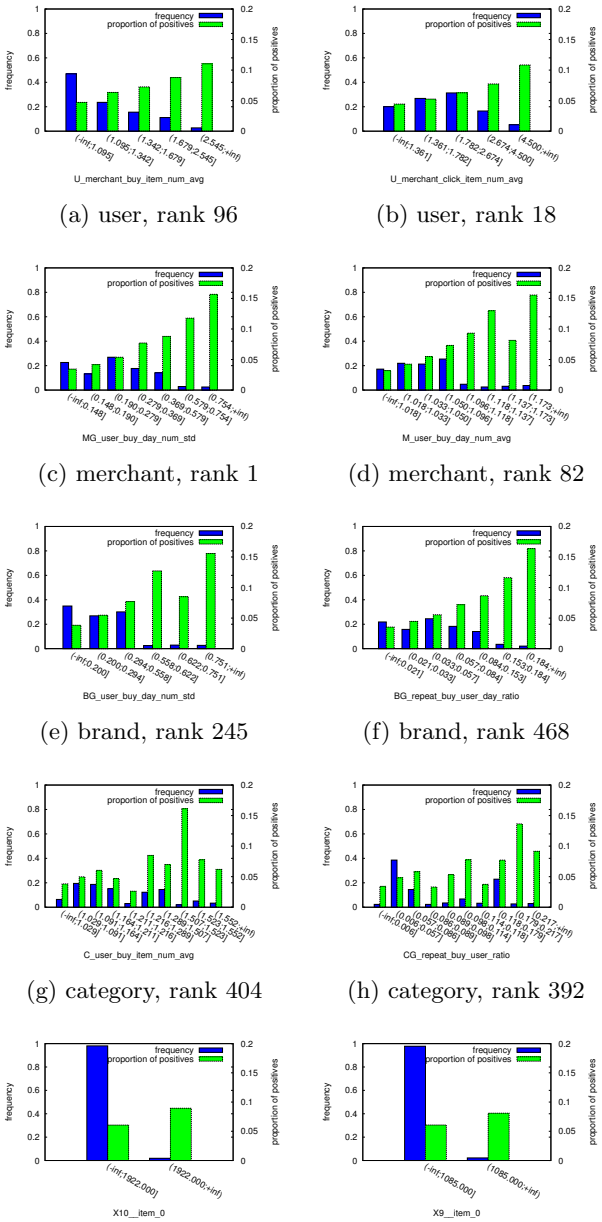


(b) feature types

Figure 4: AUC scores of feature groups

of features turns out to be not important, we can remove the whole group to save the cost of both feature engineering and model training. Table 5 lists the feature groups together with their sizes. The second row shows the AUC score of the full feature set. We study merchant-brand (MB) profile and merchant-category (MC) profile together, since they are of small sizes. Likewise, user-brand (UB) profile and user-category (UC) profile are investigated together.

We use five-fold cross validation to evaluate the importance of each group of features. The predictive model in the experiments is XGBoost with the parameter setting: eta=0.04, nrounds=400, max.depth=7, min_child_weight=200, and subsample=0.8. The five folds are the same for all the experiments, and the reported results are the averages over the five folds. We first train XGBoost by each group of features alone. The "AUC" column in Table 5 and the "AUC" bars in Figure 4 are the averaged AUC scores of XGBoost over the five folds. Clearly, the higher the score, the more important the group of features is. We also evaluate



(a) user, rank 96 (b) user, rank 18

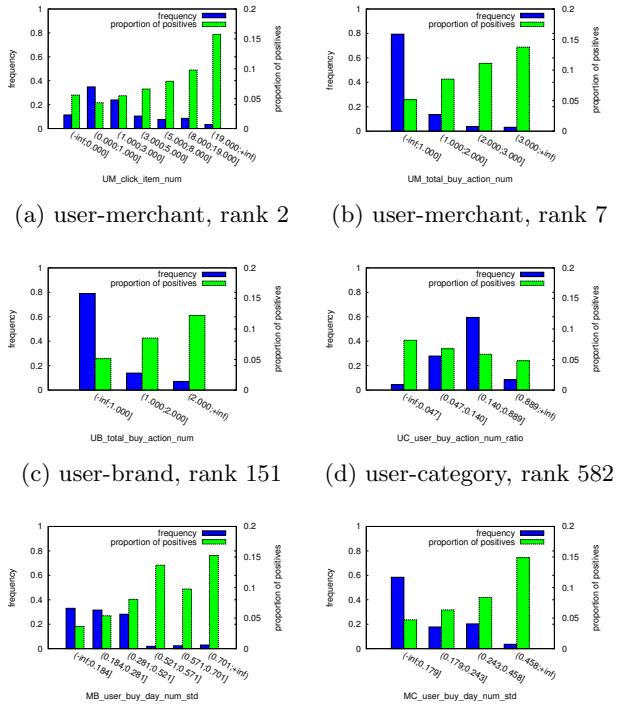
(c) merchant, rank 1 (d) merchant, rank 82

(e) brand, rank 245 (f) brand, rank 468

(g) category, rank 404 (h) category, rank 392

(i) item, rank 74 (j) item, rank 178

Figure 5: Top features in entity profiles



(a) user-merchant, rank 2 (b) user-merchant, rank 7

(c) user-brand, rank 151 (d) user-category, rank 582

(e) merchant-brand, rank 3 (f) merchant-category, rank 12

Figure 6: Top features in interaction profiles

the importance of a group of features by considering the full feature set but excluding the group. In this way, if the AUC score drops more, then the group is more important. The "*leave-out* AUC" column in Table 5 and the "*leave-out* AUC" bars in Figure 4 give the results.

Among the 5 entity profiles in Table 5, merchant profile and brand profile have the highest AUC scores: 0.6601 and 0.65818, respectively. However, if one of them is removed, the leave-out AUC score is even a bit higher than 0.70036—the AUC score of the full feature set. This indicates the redundancy of the two profiles. User profile has the lowest AUC score, but the AUC score drops the most if it is removed. This implies that user profile has important information that does not exist in other profiles. For interaction profiles, MB and MC profiles together have the highest AUC score, and the AUC score drops the most if they are excluded.

Among all the feature types, monthly aggregation has the highest AUC score—0.68729. When all the 276 aggregation features (164 monthly aggregation features, 88 user aggregation features and 24 merchant aggregation features) are used to build XGBoost, the AUC score is 0.6945, which is just 0.82% lower than the AUC score when all the features are used. LDA has the lowest leave-out AUC score (i.e., AUC score drops most if LDA features are excluded), which is still just 0.23% lower than the AUC score of the full feature set. The results above suggest that any feature group can be removed without decreasing the AUC score much.

The "leave-out AUC" column in Table 5 indicates some feature groups may be redundant given all other features, including merchant profile, brand profile, UB and UC profiles, monthly aggregation features, merchant aggregation features, latest-one-month features, user-merchant similarity features and age-related features. The leave-out AUC scores of these feature groups are even slightly higher than that of the full feature set. For example, if we exclude merchant profile from the full feature set, then the leave-out AUC score is 0.70096, which is slightly higher than 0.70036. If we remove all the above seemingly redundant feature groups, the number of remaining features is 691 (50.7% of total features) and the AUC score becomes 0.69936, which is lower than the AUC score of the full feature set.

5.2 Top features

XGBoost calculates a gain score for each feature, which measures how important a feature is to the model. We rank

Table 6: Features with high profile ranking

profile	global rank	type	Feature name	Description
user	96	merchant aggregation	U_merchant_buy_item_num_avg	average number of unique items bought from merchants by the user
	18	merchant aggregation	U_merchant_click_item_num_avg	average number of unique items clicked in merchants by the user
merchant	1	user aggregation & gender related	MG_user_buy_day_num_std	standard deviation of the number of days that users made a purchase from the merchant, only users of a particular gender are considered.
	82	user aggregation	M_user_buy_day_num_avg	average number of days that users made a purchase from the merchant
user-merchant	2	product diversity	UM_click_item_num	number of unique items clicked by the user in the merchant
	7	overall action count	UM_total_buy_action_num	total number of purchases made by the user from the merchant
brand	245	user aggregation & gender related	BG_user_buy_day_num_std	standard deviation of the number of days that users purchased the brand, only users of a particular gender are considered
	468	repeat buyer & gender related	BG_repeat_buy_user_day_ratio	proportion of repeat buy days of the brand, only users of a particular gender are considered.
category	404	user aggregation	C_user_buy_item_num_avg	average number of items in the category that were bought by users
	392	repeat buyer & gender related	CG_repeat_buy_user_ratio	proportion of repeat buyers of the category, only users of a particular gender are considered.
item	74	monthly action count	X10_item_0	times that the item was clicked in October
	178	monthly action count	X9_item_0	times that the item was clicked in September
user-brand	151	overall action count	UB_total_buy_action_num	total times that the user bought the brand
user-category	582	overall action ratio	UC_user_buy_action_num_ratio	ratio of the times that the user purchased the category to the total actions taken by the user on the category
merchant-brand	3	user aggregation	MB_user_buy_day_num_std	standard deviation of the number of days that users bought the brand from the merchant, only users of a particular gender are considered.
merchant-category	12	user aggregation	MC_user_buy_day_num_std	standard deviation of the number of days that users bought the category from the merchant, only users of a particular gender are considered.

Table 7: Features with high global ranking

global rank	profile	type	Feature name	Description
4	merchant-brand	repeat buyer	MB_repeat_buy_day_ratio	proportion of repeat buy days of the brand in the merchant
5	user-merchant	trend	user_seller_store_visit_day_count_MDP	deviation of the number of times the user clicked the merchant in the latest month from the mean of the previous months normalized using mean
6	merchant-brand	user aggregation	MB_user_buy_day_num_avg	average number of days users bought the brand from the merchant
8	user-merchant	product diversity	UM_click_cat_num	number of unique categories clicked by the user in the merchant
9	merchant	user aggregation & gender related	MG_user_buy_day_num_avg	average number of days users bought some item from the merchant, only users of a particular gender are considered
10	merchant-category	user aggregation	MC_user_buy_day_num_avg	average number of days that users bought the category from the merchant.
11	merchant	repeat buyer & age related	MA_repeat_buy_user_day_ratio	proportion of repeat buy days of the merchant, only users of a particular age group are considered.
13	user	monthly aggregation	U_monthly_click_merchant_num_std	standard deviation of the number of merchants clicked by the user every month
14	user-merchant	similarity score	UM_buy_action_num_brand_merchant_user_share_simscore_sum	similarity score between the user and the merchant, and the score is obtained by first calculating the product of the times that the user bought a brand and the brand's user share within the merchant, and then taking sum over all brands in the merchant.
15	user-merchant	product diversity	UM_buy_item_num	number of unique items purchased by the user in the merchant
16	merchant-category	repeat buyer	MC_repeat_buy_day_ratio	proportion of repeat buy days of the category in the merchant
17	category	user aggregation	C_user_buy_day_num_avg	average number of days that users bought the category.
19	user	merchant aggregation	U_merchant_click_day_num_std	standard deviation of the number of days that the user clicked merchants.
20	user	product diversity	U_buy_merchant_ratio	ratio of the number of merchants that the user made a purchase from to the total number of merchants that the user took some actions

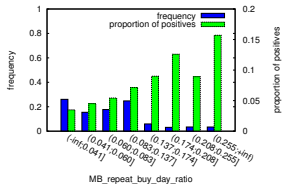
features based on their average gain scores over the five folds. Each feature has two rankings: 1) *profile ranking* is the ranking of a feature when only features in the corresponding profile are used to build XGBoost models, and 2) *global ranking* is the ranking of a feature when all of the 1364 features are used to build XGBoost models. Table 6 shows the top one or two features in each profile based on profile ranking. The global rankings of these features are in column "global rank".

All the features we generated are numeric. To visualize their correlations with the class labels, we discretize their values using the method in [9]. To avoid generating too many small bins, we set the minimum number of instances in a bin to 5000. Figure 5 reports the relative frequency of each bin (blue bar with the frequency given by the left y-axis) and the proportion of positive instances therein (green bar with proportion value given by the right y-axis) for top features in entity profiles. Figure 6 reports the top features in interaction profiles. The x-axis in Figures 5 and 6 gives the value ranges of the discretized bins. The proportion of positive instances increases with the feature values in most cases. None of the features is a strong indicator of class labels. The maximal information gain of all features is only 0.00868 after discretization.

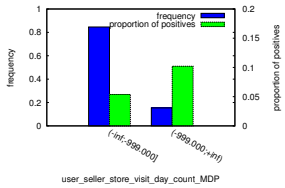
Among features in user profile, the average number of items clicked in or purchased from merchants are the top-2 features. For merchant profile, the average and standard deviation of the number of days that users made a purchase in the merchant are the top-2 features in the profile, with the latter being the top feature globally. In profiles of user-merchant, merchant-brand, and merchant-category, top features locally in the profiles also have high global rankings.

Table 7 lists the top-20 features based on global ranking. Those already reported in Table 6 are not repeated in Table 7. Top features are mainly from user aggregation (7 features), repeat buyer (3), and product diversity (3), which account for almost 2/3 of the top-20 features. Figure 7 shows the statistics of the top features, including discretized bins, the frequency, and proportion of positive instances in the bins. Features *U_monthly_click_merchant_num_std* (rank 13) and *U_buy_merchant_ratio* (rank 20) are not shown in the figure, because they have only one bin after discretization. Feature *user_seller_store_visit_day_count_MDP* is set to -999, if a user did not visit the merchant from May to October. This feature is discretized into two bins: (-inf, -999] and (-999, +inf), and the second bin has a higher proportion of positive classes. It indicates that if a user visited a merchant before November, then the user is more likely to buy from the merchant again after Double 11.

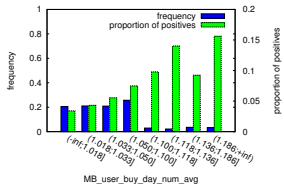
Some user aggregation features and repeat buyer features capture like characteristics of entities or interactions from different aspects, and they are highly correlated. For example, feature *MB_repeat_buy_day_ratio* (Figure 7(a)) and feature *MB_user_buy_day_num_avg* (Figure 7(c)) are merchant-brand features, and they show similar patterns. The former



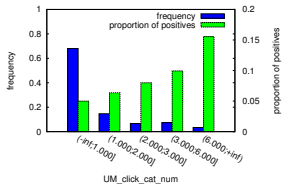
(a) repeat buyer, rank 4



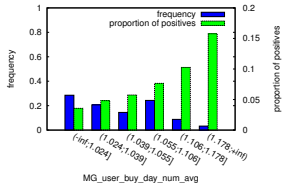
(b) trend, rank 5



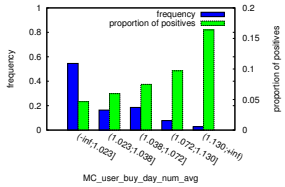
(c) user aggregation, rank 6



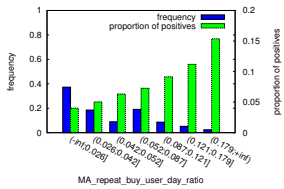
(d) product diversity, rank 8



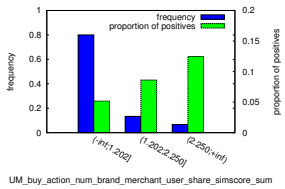
(e) user aggregation, rank 9



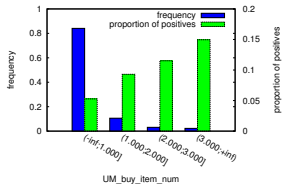
(f) user aggregation, rank 10



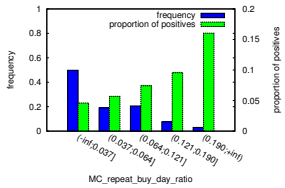
(g) repeat buyer, rank 11



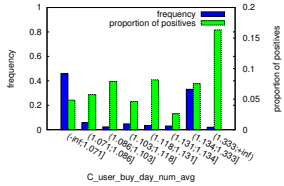
(h) UM similarity score, rank 14



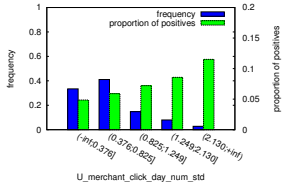
(i) product diversity, rank 15



(j) repeat buyer, rank 16



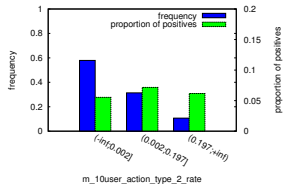
(k) user aggregation, rank 17



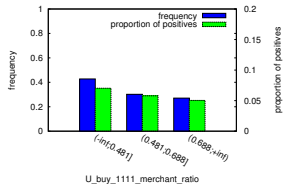
(l) merchant aggregation, rank 19

Figure 7: Features with high global ranking

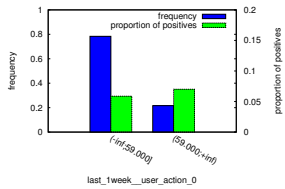
is the proportion of buyers, who bought the brand from the merchant on at least two different days, among the users who bought the brand from the merchant at least once. The latter is the average number of days that users bought the brand from the merchant. A larger *MB_repeat_buy_day_ratio* value often implies a larger *MB_user_buy_day_num_avg* value



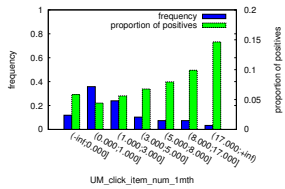
(a) action ratio, rank 25



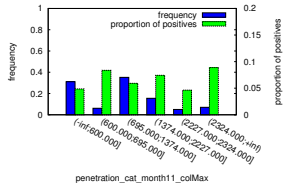
(b) double 11, rank 40



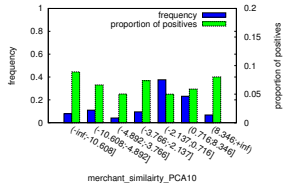
(c) latest one week, rank 38



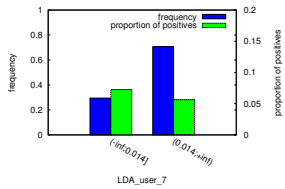
(d) latest one month, rank 50



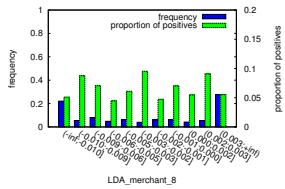
(e) penetration, rank 65



(f) PCA, rank 68



(g) LDA user, rank 42



(h) LDA merchant, rank 122

Figure 8: Top features in other feature type groups

and vice versa. The same correlation is observed for feature *MC_user_buy_day_num_avg* (Figure 7(f)) and feature *MC_repeat_buy_day_ratio* (Figure 7(j)). Both features are merchant-category features.

Several feature types, including monthly action ratio, double 11, latest-one-week, latest-one-month, penetration, PCA and LDA, do not occur in the global top-20 feature list (Table 7). Table 8 reports the top features in these feature groups based on global ranking. Figure 8 shows the statistics of these features.

5.3 Performance on testing data

Section 5.1 and 5.2 evaluate the feature importance on training data by five-fold cross validation. In this subsection, we evaluate the importance of features on the testing data. We set the parameters of XGBoost as follows: eta=0.01, nrounds=2000, max.depth=7, min_child_weight= 200, and subsample=0.8. When all the features are used, the testing AUC score is 0.702508 [3] as shown in the second row of Table 9. The last column of the table is the percentage of AUC

[3]In the competition, we used smaller learning rate and more rounds to achieve the slightly better AUC score of 0.70282.

Table 8: Features with the highest global ranking among features in the remaining feature groups

global rank	profile	type	Feature name	Description
25	user	monthly action ratio	m_10user_action_type_2_rate	ratio of the number of purchases made by the user in October to the total number of actions taken by the user in October
40	user	Double 11	U_buy_1111_merchant_ratio	ratio of the number of merchants that the user made a purchase from on Double 11 to the total number of merchants that the user took some actions on Double 11
38	user	latest one week	last_1week_user_action_0	the number of clicks made by the user in the last week before Double 11
50	user-merchant	latest one month	UM_click_item_num_1mth	the number of unique items clicked by the user in the merchant in the latest one month
65	category	penetration	penetration_cat_month11_colMax	number of users who purchased some item of the category in November
68	merchant	PCA	merchant_similairty_PCA10	the 10-th component of PCA
42	user	LDA user	LDA_user_7	the value of the 7-th topic when users are regarded as documents
122	merchant	LDA merchant	LDA_merchant_8	the value of the 8-th topic when merchants are regarded as documents

Table 9: AUC on testing data

	feature types	profiles	#features	AUC	% of drop
1	all feature types	all profiles	1364	0.702508	-
2	overall action counts/ratio, overall day counts, product diversity, monthly aggregation, user aggregation, merchant aggregation, repeat buyer, double 11, latest one month	user profile, merchant profile, user-merchant profile	354	0.694927	1.08%
3	feature set 2 plus LDA features	same as feature set 2	434	0.696812	0.81%
4	same as feature set 3	feature set 2 plus MB and MC profiles	492	0.699226	0.47%
5	same as feature set 3	feature set 4 plus brand profiles and category profiles	616	0.701392	0.16%
6	feature set 3 plus user-merchant similarity features	all profiles	866	0.701913	0.08%
7	feature set 6 plus monthly action counts, penetration features and PCA features	all profiles	1053	0.702250	0.04%

score drop, when only subsets of features (as specified in the second and third columns of the table) are used.

Feature set 2 contains 354 features from nine feature types and three profiles. Its AUC score is only 1.08% lower than the AUC score when all the 1364 features (i.e., feature set 1) are used. When more feature types and/or profiles are added (top-down in Table 9), the AUC score increases marginally. The results again imply that we can use a smaller number of features to train predictive models without decreasing the AUC score significantly.

6. CONCLUSION

In this paper, we presented our winning solution for the repeat buyer prediction competition hosted at IJCAI 2015 conference. We generated a large number of features to capture the preferences and behaviors of users, characteristics of merchants, brands, categories and items and the interactions among them. Our study shows that none of the features generated is a strong indicator of class labels, so we need hundreds of features to achieve a relatively high AUC score. We hope our winning solution, along with concrete analysis on feature engineering, would serve as a solid stepping stone for practitioners to solving future e-commerce problems. It is a tedious task to generate and manage a large number of features. As our next step, we will explore how to automate the feature generation and selection process for e-commerce prediction tasks.

7. REFERENCES

[1] Fitting generalized linear models. Available on https://stat.ethz.ch/R-manual/R-devel/library/stats/html/glm.html.

[2] Generalized linear models. Available on http://scikit-learn.org/stable/modules/linear_model.html.

[3] H. Abdi and L. J. Williams. Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459, 2010.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, 2003.

[5] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[6] T. Chen and T. He. Xgboost: extreme gradient boosting. Available on https://github.com/dmlc/xgboost.

[7] M. Dash and H. Liu. Feature selection for classification. *Intelligent data analysis*, 1(1):131–156, 1997.

[8] P. Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[9] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. of the International Joint Conference on Uncertainty in AI*, pages 1022–1027, 1993.

[10] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[11] Y.-C. Juan, W.-S. Chin, and Y. Zhuang. Field-aware factorization machines. Available on https://github.com/guestwalk/libffm.

[12] S. Lê and F. H. Julie Josse. FactoMineR: an R package for multivariate analysis. *Journal of statistical software*, 25(1):1–18, 2008.

[13] L. C. Molina, L. Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *ICDM*, pages 306–313, 2002.

[14] S. Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology*, 3(3), 2012.

[15] K.-Q. Shen, C.-J. Ong, X.-P. Li, and E. Wilder-Smith. Feature selection via sensitivity analysis of svm probabilistic outputs. *Machine Learning*, 70(1):1–20, 2008.

[16] J.-B. Yang and C.-J. Ong. An effective feature selection method via mutual information estimation. *IEEE Transactions on Systems, Man and Cybernetics (Part B)*, 42(6):1550 – 1559, 2012.