Collaborative Embedding Features and Diversified Ensemble for E-Commerce Repeat Buyer Prediction

Zhanpeng Fang*, **Zhilin Yang*** **and Yutao Zhang**
Department of Computer Science and Technology, Tsinghua University
{fzp13, yzl11, yt-zhang13}@mails.tsinghua.edu.cn

Abstract

In online advertisement, after sales promotion, it is important to predict which buyers will return and become loyal repeat buyers. Given the action logs of users, brand and category information of items, and user profiles, we study the problem of repeat buyer prediction on E-commerce data, which aims to predict whether a new buyer of a merchant is a one-time deal hunter or will become a loyal repeat customer.

We develop a set of useful features to capture the underlying structure of the data, including features regarding users, merchants, categories, brands, items and their interaction. We also propose to learn collaborative features with embeddings, which represent users and merchants in a shared feature space.

We use logistic regression, gradient boosting decision trees, and factorization machines as individual classification models. We develop a diversified ensemble model to combine different feature sets and models.

Our solution obtained AUCs of 70.4762% in stage 1 and 71.0163% in stage 2, ranked 2nd and 3rd places respectively in IJCAI 2015 Repeat Buyer Prediction Competition.

1 Introduction

Online sales promotions usually attract a large number of buyers on E-commerce sites. However, only a small portion of the users will return and become loyal repeat buyers. In IJCAI 2015 Repeat Buyer Prediction Competition, we study the problem of predicting repeat buyers on E-commerce data.

In this task, we are given user profiles and action logs on `Tmall.com`. Tmall runs large-scale sales promotions on November 11th each year, which is usually called the "Double 11" day. The dataset of the challenge consists of two tables. In the first table, we have user profiles, including age ranges, gender and user IDs. The second table contains user action logs, in which each log represents a click/add-to-cart/add-to-favorite/purchase action between a user u and a merchant m on an item i at time t. Each item i is associated with a brand b_i and a category c_i. These action logs are collected in a span of six months prior to the "Double 11" day. Our task is to predict whether a new buyer of a merchant on the "Double 11" day will return and purchase in the following six months.

The competition has two stages. In stage 1, the data set contains $260,865$ training instances, $261,478$ test instances and action logs of 7 million distinct user-merchant pairs. We ran our algorithms offline and submitted the prediction results for evaluation. In stage 2, we submitted our code to a distributed computing platform provided by the organizer, where the platform distributes the records of different users to different nodes. The data set in stage 2 contains $3,441,313$ training instances, $3,443,594$ test instances, and action logs of 192 million distinct user-merchant pairs. The IDs of merchants, items, categories and brands are shared in training and test sets, while user IDs of the two sets do not overlap.

One key challenge of the problem is how to leverage the collaborative information between users and merchants. Conventional methods like collaborative filtering [Sarwar *et al.*, 2001; Breese *et al.*, 1998], matrix factorization [Koren *et al.*, 2009], and factorization machines [Rendle, 2010] address the issue by modeling similar users, modeling similar merchants, or factorizing the low-rank interaction between users and merchants. However, these methods represent users and merchants in separate spaces, and thus simply applying dot products across spaces will not be expressive enough. Also, with such methods, similar users do not directly interact with each other to learn better features. In this work, we propose to model users and merchants as continuous embeddings in a shared latent space based on random walks. By modeling users and merchants in a shared space, applying dot products can have clear explanations of similarities. Also, through random walks, similar users and similar merchants are at the same time incorporated into the optimization framework.

Another challenge is how to leverage the rich action logs to improve the prediction. Extra information including categories, brands and items is very important in this task. We develop a set of collaborative features by examining repeat purchase statistics and the frequent categories, brands and items for a given pair of user and merchant.

$*$ indicates equal contribution.

Based on logistic regression [Cox, 1958], gradient boosting trees [Friedman, 2001], and factorization machines [Rendle, 2010], we develop a diversified ensemble model for prediction. We combine different feature sets and different models, and use ridge regression [Hoerl and Kennard, 1970] to blend the prediction on the test data.

In stage 1, we took the 2nd place by achieving an AUC of 70.4762%. In stage 2, because we have to deploy our algorithms on the distributed computing platform, we did not implement all of our algorithms in stage 1. As a result, we took the 3rd place by achieving an AUC of 71.0163%. The score in stage 2 is better than that in stage 1 because of larger data amount.

2 Framework

In this section, we illustrate the proposed prediction framework.

Our solution can be divided into three components: feature engineering, individual models, and model ensemble. Figure 1 illustrates the architecture of our solution.

We design a bunch of feature engineering techniques to extract useful features from the data. Besides basic general features for recommender systems, we design two novel feature sets to better capture the underlying structure of the data. We develop embedding techniques to represent users and merchants in a shared latent space, and use the cosine similarities as features. We also focus on repeat action statistics to utilize the statistical power of category/brand/item information.

We propose a diversified ensemble model for prediction. Let $\mathcal{F}$ be a set of feature sets, where each element is a set of features. Let $\mathcal{M}$ be a set of models, which are logistic regression, gradient boosting decision trees, and factorization machines in our solution. We apply a Cartesian product on the two sets $\mathcal{F} \otimes \mathcal{M}$ to obtain combinations of models and features. We use ridge regression to ensemble different combinations to obtain the final prediction.

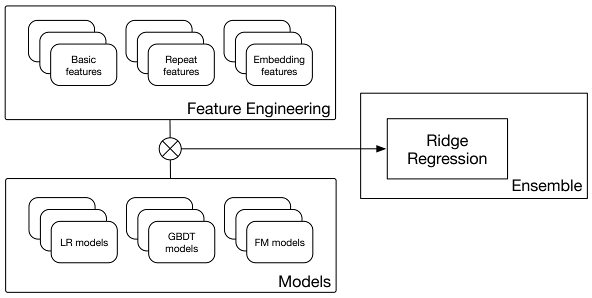

Figure 1: Architecture of our solution.

3 Feature Engineering

In this section, we introduce the three sets of features we derive - basic features, repeat features, and embedding features. Basic features are general features for recommender systems. Repeat features are designed for this specific problem, focusing on repeat statistics regarding users and the interactions with categories, brands and items. Embedding features are extracted with advanced learning techniques, which jointly represent users and merchants in a shared latent space.

3.1 Basic Features

Generally, the basic features can be categorized into three groups: the user-related features that describe users' characteristics; the merchant-related features that are only related to merchants' information; and the user-merchant features that capture the interactions and the characteristics of user-merchant pairs.

User-Related Features

The dataset provides information on users' age and gender, and we take those fields as categorical features directly. We count the total number of click, purchase, and add-to-favourite actions of a user to measure user activity. We omit add-to-cart actions in all our features because we found that the information provided by add-to-cart actions is almost identical to that of purchase actions, and in both stages of the contest the performances of our method improved after removing the features related to add-to-cart actions. We also use the number of distinct items/merchants/categories that a user clicked/purchased/favored for characterizing a user's activity. We utilize the time information by using the number of active days, the number of recent active days, and the oldest active day for a user as numerical features.

Merchant-Related Features

Merchant ID is a very strong indicator for a merchant's likelihood of having repeat buyers. However, we also derive some other merchant-related features, because the training data of some merchants is sparse, and the feature weights of these merchants' IDs can not be estimated accurately. Besides merchant IDs, we count the number of actions and distinct users that clicked/purchased/favored a merchant as numerical features. We use the ratio of purchased items in summer to model a merchant's seasonality.

Notice that we only use this category of features in stage 1 of the contest, since the distributed environment of stage 2 does not guarantee that all data records of one merchant are sent to the same node.

User-Merchant Features

The user-merchant features are similar to the user-related features. We compute the number of click/purchase/favorite actions and the number of total/recent/oldest active days of a user-merchant pair as numerical features. In addition, we use the category IDs and the brand IDs of the items that a user purchased from a merchant as categorical features to model the context of a user-merchant pair.

Post Processing For Feature Vectors

In order to leverage the nonlinear effect of features, in stage 1 we discretize all the numerical features into bins of equal frequencies before feeding the features into linear models like logistic regression. However, we do not adopt this practice in stage 2, since the split points can not be computed in the provided distributed environment. Thus, we convert the feature value x of each numerical features to $\log(1+x)$ in stage 2.

Experiments show that both processing techniques perform similarly and are much better than the original one.

3.2 Repeat Features

In this section, we introduce a set of repeat features. Because the problem is to predict repeat buyers, we focus on the repeat buying statistics to extract features. We leverage extra information like categories and brands to develop the new feature set.

User Repeat Features

In the given datasets, unlike merchants, the sets of users do not overlap between training and test. Therefore, we cannot use user IDs as features, though they may be informative. Instead, it is important to examine the repeat buying statistics of users.

For each user, we extract the following features to indicate the user's repeat buying pattern.

- Average span between any two actions.
- Average span between two purchases.
- How many days since the last purchase.

The features above characterize the activity and repeat buying pattern of each user, which are important statistics of user behaviors.

User-Merchant/Category/Brand/Item Repeat Features

We also leverage merchant/category/brand/item information to better capture the repeat buying patterns of users. We extract the following features.

- Average active days for one merchant, category, brand or item.
- Maximum active days for one merchant, category, brand or item.
- Average span between any two actions for one merchant, category, brand or item.
- Ratio of merchants, categories, brands or items with repeated actions.

Category/Brand/Item Repeat Features

Besides repeat buying patterns of users, it is also important to utilize the repeat buying patterns of categories, brands and items. To this end, we extract the following features for each category/brand/item.

- Average active days on the given category, brand or item of all users.
- Ratio of repeated active users on the given category, brand or item.
- Maximum active days on the given category, brand or item of all users.
- Average days of purchasing the given category, brand or item of all users.
- Ratio of users who purchase the given category, brand or item more than once.
- Maximum days of purchasing the given category, brand or item of all users.
- Average span between two actions of purchasing the given category, brand or item of all users.

Given a pair of user u and merchant m, we compute the top-k frequent categories, brands and items in the interaction log between u and m, and then use the according features above as the features of $< u, m >$.

3.3 Collaborative Embedding Features

To leverage the collaborative information between users and merchants, we develop a set of collaborative features to characterize how *well* a user' interest aligns with a merchant. We model users and merchants in a shared latent space to uncover the underlying structure of the user-merchant interaction data. We compute the similarities between users and merchants as features based on the learned embeddings.

Algorithm 1: Learning collaborative features with user-merchant embeddings

```
Input: Action logs $\mathcal{L}$, user set $U$, merchant set $M$, walks per vertex $\gamma$
Output: Embeddings $\mathbf{f}$
// construct the user-merchant interaction graph
1  initialize vertex set $V \leftarrow U \cup M$
2  initialize edge set $E \leftarrow \emptyset$
3  foreach $< u, m > \in \mathcal{L}$ do
4      $E \leftarrow E + < u, m >$
5  construct a graph $G = (V, E)$
// generate random walks
6  initialize random walks $\mathcal{W} \leftarrow \emptyset$
7  for $i \leftarrow 1$ to $\gamma$ do
8      $V' \leftarrow$ random_shuffle$(V)$
9      foreach $v \in V'$ do
10         $W \leftarrow$ random_walk$(v, G)$
11         $\mathcal{W} \leftarrow \mathcal{W} + W$
// learn embeddings
12 initialize $\mathbf{f}'$ and $\mathbf{f}$
13 SGD$(\mathcal{W}, \mathbf{f}, \mathbf{f}')$
14 return $\mathbf{f}$
```

Algorithm 1 illustrates the procedure of learning user-merchant embeddings. Let U and M be the set of users and merchants. The input of the algorithm is a set of action logs, where each element is a user-merchant pair $< u, m >$, where $u \in U$ and $m \in M$, indicating that u interacted with m. We consider all types of actions equally. If u has multiple actions with m, the pair $< u, m >$ will appear in $\mathcal{L}$ multiple times.

$\mathbf{f}$ is a $(|U| + |W|) \times d$ matrix, where d is the dimension of the embeddings. Each row of $\mathbf{f}$ represents the embedding of a user or merchant. $\mathbf{f}'$ is an auxiliary matrix.

Following [Perozzi *et al.*, 2014] and [Mikolov *et al.*, 2013], we generate sequences of random walks on the user-merchant interaction graph and learn the user-merchant embeddings based on the Skipgram model.

We construct a user-merchant interaction graph. The vertex set of the graph is $U \cup M$. For each action log $< u, m >$ in $\mathcal{L}$, we add an edge between vertex u and vertex m.

After constructing the user-merchant interaction graph, we generate a series of random walks starting from each user and merchant.

Given random walks $\mathcal{W}$, the loss function of the Skipgram model is defined as

$$L = -\sum_{W \in \mathcal{W}} \frac{1}{T} \sum_{t=1}^{T} \sum_{-c \le j \le c, j \ne 0} ({f'_{W_{t+j}}}^{\intercal} f_{W_t} - \sum_{w \in V} {f'_w}^{\intercal} f_{W_t})$$

where $\mathbf{f}'$ and $\mathbf{f}$ are input and output embeddings. T is the length of random walk W and c is the window size of context. We perform stochastic gradient descent (SGD) [Bottou, 2010] to update $\mathbf{f}$ and $\mathbf{f}'$. We use the implementation of Gensim [Řehůřek and Sojka, 2011] for learning the Skipgram model.

Diversified Features at Different Iterations
Given a pair of user u and merchant m, we compute the cosine similarity between f_u and f_m and apply it as a feature.

We develop a method to extend the dimensionality and diversity of the embedding features. As is shown in Algorithm 1, we use SGD to iteratively update the embeddings $\mathbf{f}$. To further diversify our features, we read out the embeddings $\mathbf{f}$ every k iterations. For example, if we set $k = 10$ and run SGD for 100 iterations, then we read the embeddings $\mathbf{f}$ at iteration $10, 20, \cdots, 100$ and obtain a 10-dimensional feature vector. This idea is similar to ensembling classifiers with different regularization parameters, which is effective at diversifying the results and improving the prediction, as demonstrated in our experiments.

4 Models

In this section, we describe our ensemble method based on logistic regression, ridge regression, gradient boosting decision trees and factorization machines.

4.1 Individual Models

Formally, our task is a binary classification problem. Thus, we leverage three popular classification models, including logistic regression (LR), gradient boosting decision trees (GBDT), and factorization machines (FM) in our solution.

Logistic regression is a widely-use linear classifier. It is intrinsically simple and so is less prone to over-fitting. In fact, our experimental result shows that LR achieves the best performance among all the individual models. We use L2 regularizer in stage 1 and L1 in stage 2. We use the implementation of LibLinear [Fan *et al.*, 2008].

Gradient Boosting Decision Tree is a tree-based additive model. GBDT learns multiple decision trees iteratively, where the learning target of the current tree is defined as the loss gradient of the previous trees. GBDT outputs the additive predicitions of all trees as the final prediction. It has a strong predictive power and naturally handles data with heterogeneous features. We use the implementation of XGBoost [Chen and He, 2015].

Factorization Machines combine the advantages of support vector machines (SVM) and matrix factorization models. In contrast to linear models like LR and SVMs, FMs model all interactions between variables using factorization method. Thus, FMs are able to overcome the sparsity problem in many tasks. We use the implementation of LibFM [Rendle, 2012].

4.2 Diversified Ensemble

We describe the ensemble techniques we develop in stage 1.

We initially extract a baseline feature set $\mathcal{F}_0$. Iteratively, we design new features and merge them with original features to obtain a new feature set. In this way, we obtain a sequence of feature sets $\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_n$, where $\mathcal{F}_i \subseteq \mathcal{F}_{i+1}, \forall i < n$. This method makes a trade-off between stability and diversity. For one thing, by introducing new feature sets, we introduce diversity into different models. For another, by incorporating the previous feature sets into the new feature sets, we maintain the stability of our models. This diversified feature ensemble method can yield better results, especially when the appended features cannot give reasonably good prediction alone but will improve the prediction together with the original features.

Let $\mathcal{F} = \cup_i \mathcal{F}_i$, and $\mathcal{M}$ be the set of three individual models. We apply Cartesian product to obtrain $\mathcal{F} \otimes \mathcal{M}$. Then we train all models in $\mathcal{F} \otimes \mathcal{M}$ separately to generate $|\mathcal{F} \otimes \mathcal{M}|$ predictions, denoted as $\hat{\mathbf{y}}$.

In order to diversify the ensemble result, we perform nonlinear extension to obtain new predictions $\sqrt{\hat{\mathbf{y}}}$, $\hat{\mathbf{y}}^2$, $\exp(2\hat{\mathbf{y}})$. We then train a ridge regression model on the extended predictions, and output the final predictions. We tune the hyperparameter of the ridge regression model to have relatively strong regularization.

5 Experiments

Stage 1 Performance. Table 1 shows the best performances on stage 1 test set for each type of individual models we use and that of the diversified ensemble method. The table also presents the performance of a simple ensemble method that ensembles the three best individual models for comparison. It can be seen that logistic regression obtains the best performance among the three individual models, which suggests that the derived features may not have strong interaction patterns that can be captured by factorization machines and GBDT. For ensembling, the diversified ensemble method achieves an improvement of $+0.7\%$ in terms of AUC compared to the best single model, and it performs significantly better than the simple ensemble method, which demonstrates the effectiveness of the diversified ensemble method.

Table 1: Performances for each type of individual models and ensemble methods on stage 1 test set.

Model	AUC (%)
Logistic regression	69.782
Factorization machines	69.509
GBDT	69.196
Simple ensemble	70.329
Diversified ensemble	70.476

Factor Contribution Analysis. We provide an analysis on the contribution of the two sets of features: embedding features and repeat features. Table 2 shows the performance gains after adding each set of features. As shown in the table, we can observe a clear performance increase when adding each set of features. This indicates that both embedding features and repeat features contribute to performance improvement, and our method works well by combining different sets of features.

Table 2: Performance gains for each set of features on stage 1 test set (with logistic regression).

No.	Method	AUC (%)	Gain
1	Basic features	69.369	-
2	1 + Embedding features	69.495	0.126
3	2 + Repeat features	69.782	0.287

Stage 2 Performance. Table 3 gives a brief introduction of several important performance gains of our method on stage 2 test set. It can be seen that repeat features still play an important role in the large data set, which brings an improvement of $+0.243\%$ compared to basic features and performs consistently in both stages. We do not use embedding features in this stage since we cannot extract those features in the given distributed environment. Moreover, data cleaning achieves a significant improvement $+0.309\%$ in stage 2, because the data set of stage 2 contains duplicated/inconsistent records that the data set of stage 1 does not have.

Table 3: Performance gains on stage 2 test set.

No.	Method	AUC (%)	Gain
1	Basic features	70.346	-
2	1 + Repeat features	70.589	0.243
3	2 + Data cleaning & more features	70.898	0.309
4	3 + Fine-tuning parameters	71.016	0.118

6 Conclusion

In this paper, we introduce our solution at IJCAI 2015 Repeat Buyer Prediction Competition.

We develop sets of novel features to better solve the problem of repeat buyer prediction on E-commerce data. We propose to jointly learn user and merchant embeddings in a shared latent space, and use the cosine similarities as features. We design a set of repeat features to further mine the users' behavior patterns and leverage category and brand information.

We propose a diversified ensemble model based on three individual models. We apply Cartesian product to obtain combinations of different models and features, and train a ridge regression model to generate the final predictions.

Our solution took the 2nd place in stage 1 and 3rd place in stage 2. Experiments show that embedding features, repeat features and diversified ensemble contribute significantly to the result.

References

[Bottou, 2010] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[Breese *et al.*, 1998] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[Chen and He, 2015] Tianqi Chen and Tong He. xgboost: extreme gradient boosting. 2015.

[Cox, 1958] David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

[Fan *et al.*, 2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[Friedman, 2001] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[Hoerl and Kennard, 1970] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[Řehůřek and Sojka, 2011] R Řehůřek and P Sojka. Gensim–python framework for vector space mo delling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 2011.

[Rendle, 2010] Steffen Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.

[Rendle, 2012] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[Sarwar *et al.*, 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.