Amazon Purchase Prediction

Li-An Yang A53242336
CSE 258 fa18

***Abstract*—In this report, I will illustrate my approach in a Purchase Prediction competition on Kaggle, of which the rank is based on the accuracy. With a focus on similarity-based inference, my work is ranked 7th with an accuracy of 72.96%. In addition to the evaluation results, several potential manipulations to achieve further improvement are discussed.**

I. Introduction

Kaggle is an online platform for entities to host data mining competitions, and provide an interface for people to participate and contribute with possible rewards. Our task on Kaggle is to predict whether or not a given user (represented by IDs) will buy this particular item based on 2,00,000 entries of Amazon review data.

Table 1 contains an example of some of the important given information, including categories, rating, reviews, and so on. With this real-life dataset, I move on to explain my methods to tackle the problem of purchase prediction.

II. Methods

There are a handful of methods to try them out for this particular binary classification task. Ranging from collaborative, filtering, logistic regression, to latent factor models, the following illustrates my developed works to make predictions on observed users and items.

A. Item-based Collaborive Filtering

By exploiting category similarities between items, I could have more (or less) confidence about whether a user will buy a certain item. The reasoning in behind is that a user is more likely to buy an item if there exist some products already bought by the user which are similar to the target item. For collaborative filtering I have considered two strategies: examining the most similar item and the average similarities of top N items. If the returned similarity is above a self-specified threshold, the results of this user-item pair will be 1, otherwise 0. The second strategy can be referred as K-nearest neighbor [1] as well.

B. Popularity Index and Consumer Buying Power

Another useful information we can incorporate is the number of times this item has been bought (*buyCnt*). Therefore, we can construct "Popularity Index", specified as follows,

$$Popularity\ Index = \frac{buyCnt - minBuyCnt}{maxBuyCnt - minBuyCnt}$$

where *maxBuyCnt* is the largest buyCnt discovered, and minBuyCnt is simply 1 since every discovered item has at least been bought once.

Likewise, we can extend this operation to users, recording a *Consumer Buying Power* for each user. These two features are then integrated into the similarity-based method to reach the highest accuracy of all my trials.

Feature	Value
User/Item ID	U490934656/I402344648
Categories	Clothing, Shoes & Jewelry
Rating	4.0
Reviews	Good quality. It fits really well!

Table 1. Review data example

C. XGBoost – Gradient Boosting

XGBoost[2] is an optimized distributed gradient boosting library that has brought success in many Kaggle competitions. One evident advantage of using XGBoost is that it is capable of discovering useful features for us and assign correspondent weights to them along the way of boosting trees enumeration.

For the sake of time, the designed feature vector simply contains the aforementioned features, as well as the average ratings that a user/item has given/received. With this library, I am able to achieve an accuracy of 69% on the first few submissions. However, it is relatively challenging to tune the various parameters XGBoost has to offer in limited time. Results and performance will be indicated in the next section.

D. Bayesian Personalized Ranking (BPR)

Mentioned by the professor in class, it is also possible to formulate this problem as maximizing the probability of correctly predicting pairwise preferences using BPR [3]. The objective function is specified as follows,

$$max \ln \sigma (\gamma_u \cdot \gamma_i - \gamma_u \cdot \gamma_j)$$

The goal is to provide rankings of items for each user, so that we can devise a threshold on the ranking to make purchase prediction.

After investing a massive amount of effort on this method, I would suggest that BPR does not really work well under our scenario due to severe sparseness in a user-item matrix of up to 40,000 users and 20,000 items, which is a cold-start problem.

E. Genrate Negative Set

Another factor that we need to take into consideration is how to generate negative data for training. Because we can only obtain positive results from Amazon reviews (entries indicating that the user has bought this item), we have to figure out ways to sample user-item pairs with negative results from observed users and items. We can directly sample unobserved pairs and assign them to be 0, but this way we will have no grounds to believe they truly indicate negative result.

We can also exploit some constraints on the generation. For example, it is reasonable to a include an unobserved pair with high Popularity Index and Consumer Buying Power in our negative set since a popular item might have already been bought by a productive buyer if it is his type and hence should be observed in the dataset. Metrics such as the aforestated average similarities also look feasible to serve as conditions to select negative pairs.

F. Unobserved Users and Items

For the test dataset, there are hundreds of users and items that we have not seen in the training set. To cope with this issue and make predictions on unseen user/item, predictions are made based on average rating of the corresponding item/user. If the average rating is higher than, say 3, then we are more confident that an arbitrary user will purchase it. If neither user and item is observed before, I would simply assign a negative result under the assumptions that there are far more negative pairs than positive ones.

III. RESULT

Some of the most promising results are shown in Table 2. During the competition, we can only test our methods on 50% of test dataset (public). After the competition ends, the accuracy of another 50% will be shown as the private score. It can be observed that there is a uniform growth of accuracy from public to private scores, pumping my rank from 11 to 7 eventually.

For the naming, hw3 is implemented following the instructions from hw3 of CSE 258. simPop is an incorporated version of II(A) and II(B). The hybrid method combines xgboost and simPop to make more confident decisions.

IV. DISCUSSION

It turns out a heuristic inference such as simPop outperforms other sophisticated models, unfortunately. Hence, there are definitely a few points worth mentioning to come up with a far better accuracy.

A. Ratio between positive and negative pairs

Whether we should generate fewer or more negative pairs is untrivial. Generating fewer negative pairs mean potential imbalanced classification while generating too many negative pairs cannot promise its truthfulness (because they aren't guaranteed to be true negative pair). In the case with XGBoost, it is observed that higher negative size would result to lower accuracy. For instance, Training on dataset with 2 times more negative pairs lead to a 63.6% accuracy.

Method	Public Score	Private Score
hw3	0.62849	0.63278
xgb 3 features	0.6945	0. 69585
xgb 5 features	0.67	0.67285
simPop	0.7 [illegible]	0.7 [illegible]
hybrid	0.71528	0.72442

Table 2. Evaluation results in terms of accuracy.

B. Feature Selection with XGBoost

One important characteristic of XGBoost is that it provides comparable feature importance score for each element of its feature vector. In addition to using it to adjust on XGBoost, we can examine if these features are truly helpful in predicting purchases or not and extract useful features out of it for subsequent model training.

C. More heuristic inference

A lot more rules can be inferred from our review data and possibly improve prediction accuracy. We can make an attempt to predict a user's sex by examining the categories of items he/she has bought. If the user has bought a lot more women products, then we could make reasonable inference that it might be a female and therefore assign less confidence to men product for this user accordingly.

V. CONCLUSION

In this report, I have presented numerous methods to make reliable user-item purchase predictions. With all the libraries and algorithms being utilized, I also try to dissect the reasons behind some unsuccessful results and discuss about some future works with potential aspects to benefit from. Standing at the 7th place on the Kaggle leaderboard, I have to say that I am a little overwhelmed, partly because I didn't really succeed on any sophistic settings or make a big leap on the basis of a strong foundation. To wrap up, I am glad to have this experience to compete with many people and inspire one another on the run.

REFERENCES

[1] Cover, Thomas, and Peter Hart. "Nearest neighbor pattern classification." *IEEE transactions on information theory* 13.1 (1967): 21-27.

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

[3] Rendle, Steffen, et al. "BPR: Bayesian personalized ranking from implicit feedback." *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 2009.