LETTER

Tree-Based Feature Transformation for Purchase Behavior Prediction

Chunyan HOU[†], *Nonmember*, Chen CHEN^{††a)}, *Member*, and Jinsong WANG[†], *Nonmember*

SUMMARY In the era of e-commerce, purchase behavior prediction is one of the most important issues to promote both online companies' sales and the consumers' experience. The previous researches usually use the feature engineering and ensemble machine learning algorithms for the prediction. The performance really depends on designed features and the scalability of algorithms because the large-scale data and a lot of categorical features lead to huge samples and the high-dimensional feature. In this study, we explore an alternative to use tree-based Feature Transformation (FT) and simple machine learning algorithms (e.g. Logistic Regression). Random Forest (RF) and Gradient Boosting decision tree (GB) are used for FT. Then, the simple algorithm, rather than ensemble algorithms, is used to predict purchase behavior based on transformed features. Tree-based FT regards the leaves of trees as transformed features, and can learn high-order interactions among original features. Compared with RF, if GB is used for FT, simple algorithms are enough to achieve better performance.

key words: feature transformation, purchase behavior prediction

1. Introduction

As online shopping becomes popular, the behavior prediction of e-commerce customers has become an increasingly important business tool for promoting sales. Both the consumers and online businesses benefit from this kind of prediction technique. However, purchase prediction becomes difficult when there is only a list of clicks in the http session. The prediction with the short-term history becomes an area of growing research and commercial interest in recent years. Especially, RecSys Challenge 2015 and CIKM Cup 2016 are associated with such problem. A history of user's click behavior during a browsing session at a website of online retailer is given, and the goal is to predict which items a user will purchase at the end of this session.

The previous researches usually use the feature engineering and ensemble machine learning algorithms for the prediction. There are a lot of categorical features which lead to high dimensionality of feature space. In addition, the large-scale data requires scalable ensemble machine learning algorithms. Inspired by the studies [1], [2], we explore an alternative to use tree-based Feature Transformation (FT) and simple machine learning algorithms (e.g. Logistic Re-

gression). A fraction of data is used by the tree-based TF to learn new features which can exhibit the high-order interaction among original features. Then, the another part of data is used to train simple machine learning algorithms. Therefore, when predicting a new sample, we convert its original features to transformed features by tree-based ensemble learning methods, and simple machine learning algorithms is used for the prediction based on the transformed features.

The rest of the paper is organized as follows. Section 2 introduces related works. Tree-based feature transformation is described in Sect. 3. Experimental setting and results are discussed in Sect. 4. In Sect. 5, we conclude our paper.

2. Related Work

Purchase prediction is usually based on the demographic information [3] and media advertisements [4]. Compared with those researches, our work is based on a list of clicks during a browsing session at an e-commerce website. From the perspective of the user behavior prediction using the click data, these studies [5], [6] focus on proposing new models to take advantage of click data. However, we explore tree-based FT.

FT is usually used as a pre-processing technique. Methods of FT can be divided into two groups: linear and nonlinear. Many linear FT methods [7], [8] have been proposed. The nonlinear FT methods include: (i) kernel-based methods [9], [10]; (ii) manifold learning models [11], [12]; (iii) deep neural networks [13], [14]. Compared with these methods, tree-based FT has two advantages. First, it is an efficient and simple dimension reduction method. It can transform features into the binary values. The kernel-based methods are not appropriate for the high-dimensional features. Tree-based FT does not need to model the nonlinear geometry of these manifolds as manifold learning methods do. Second, it is able to eliminate the non-linearity in the features and improve the linear classifier while the linear transformation cannot. The deep neural network model is time-consuming to learn from large-scale samples. Thus, our work focuses on the tree-based FT and its application to purchase behavior prediction.

3. Feature Transformation

3.1 Problem Definition

The dataset is divided into train and test dataset. Train and

Manuscript received September 26, 2017.

Manuscript revised January 6, 2018.

Manuscript publicized February 2, 2018.

[†]The authors are with the School of Computer and Communication Engineering, Tianjin University of Technology, Tianjin, China.

^{††}The author is with the College of Computer and Control Engineering, Nankai University, Tianjin, China.

a) E-mail: nkchenchen@nankai.edu.cn (Corresponding author)

DOI: 10.1587/transinf.2017EDL8210

test dataset consist of sets of sessions S_{train} and S_{test} respectively. Each session s is represented as a click stream

$$c(s) = (c_1(s), c_2(s), \dots, c_{n(s)}(s)) \quad (1)$$

$$c_j(s) = (i_j(s), t_j(s), y_j(s)), j \in \{1, \dots, n(s)\} \quad (2)$$

where $n(s)$ is the number of clicks in session s , $i_j(s)$ denotes the j -th clicked item in session s , $t_j(s)$ is the time when item $i_j(s)$ is clicked, $y_j(s)$ is one when item $i_j(s)$ is purchased at least once, and zero otherwise. A session s has the label $y(s)$ defined as

$$y(s) = \begin{cases} 1 & \exists j : y_j(s) = 1 \\ 0 & \forall j : y_j(s) = 0 \end{cases} \quad (3)$$

If $y(s) = 1$, session s is a buy session. We are given sets of purchased items for session $s \in S_{train}$, and are required to predict these sets for session $s \in S_{test}$. In our experiments, $y_j(s)$ is predicted, and then $y(s)$ can be inferred by $y_j(s)$.

3.2 Tree Ensemble Model

For a given dataset with N samples and M features $D = \{(x_i, y_i)_{i=1}^N | x_i \in \mathbb{R}^M, y_i \in \mathbb{R}\}$, a tree ensemble model uses K additive functions to predict the target.

$$\hat{y}_i(x_i) = \sum_{k=1}^K T_k(x_i) = \sum_{k=1}^K f_k(x_i) \quad (4)$$

where T_k is the k -th tree which maps a sample to the corresponding leaf with a weight. Each tree T_k corresponds to a function f_k which classify a sample x_i into the leaf and returns the weight of such leaf.

There are two kinds of tree-based ensemble methods: bagging and boosting. Bagging draws samples of the training data many times to fit the trees respectively while boosting makes trees evolve over time and fit them one by one. Trees generated in bagging are identically distributed, and this means the bias of trees is the same as that of the individual trees. In contrast, boosting makes the trees grow in an adaptive way to remove bias and are not identically distributed. Random Forests (RF) use the bagging technique to build a large collection of trees at training time for the prediction. Gradient Boosting decision tree (GB) is a highly effective and widely used machine learning method which is based on the boosting technique.

3.3 Feature Transformation

Formally, suppose K trees $\{T_k\}_{k=1}^K$ are grown, the function

Fig. 1 An example of feature transformation.

$FT(x)$ can transform x into another feature space.

$$FT(x) = (Leaf(T_1, x), Leaf(T_2, x), \dots, Leaf(T_K, x))$$

where $Leaf(T_i, x) = (w_1, w_2, \dots, w_{L_i})$

$$w_j = \begin{cases} 0 & x \notin T_i^j \\ 1 & x \in T_i^j \end{cases}, j \in \{1, \dots, L_i\} \quad (5)$$

where L_i is the number of leaves in the i -th tree T_i and T_i^j denotes the region which corresponds to the j -th leaf of T_i . In other words, if a sample x falls on the j -th leaf of T_i , then $x \in T_i^j$.

For example, there is a tree ensemble model which consists of two trees with the depth three in Fig. 1. The first tree has 3 leaves and the second tree has 4 leaves. If a sample falls on the second leaf in the first tree and the fourth leaf of the second tree, then $FT(x) = (0, 1, 0, 0, 0, 0, 1)$, where the first 3 elements correspond to the leaves of the first tree and last 4 to those of the second tree. In a simple way, if the feature is represented by $(index : value)$, $FT(x)$ can be expressed as $\{2 : 1, 7 : 1\}$.

Tree ensemble model may be RF or GB. Feature transformation based on the tree ensemble model can be regarded as a kind of supervised feature learning method which converts a real-valued vector into a compact binary-valued vector. A path from the root to the leaf represents a rule which is associated with a region in the feature space. This method only considers the leaf nodes of trees and ignore which tree the leaf node belongs to. Field-aware Factorization Machine (FFM) organizes features into fields. A field is associated with a class of features. The trees can be regarded as fields and the index of the leaf node in this tree can be treated as the feature. As shown in Fig. 1, if the feature is represented by $(field : index : value)$, $FT(x)$ can be expressed as $\{1 : 2 : 1, 2 : 4 : 1\}$.

4. Experiments

4.1 Dataset

We use two datasets in our experiments. Yoochoose provides a large number of sessions from an online e-commerce retailer for RecSys 2015 Challenge [15]. 5.51% of those session is buy sessions and the average number of clicks per session is 3.57. Diginetica releases user sessions for CIKM Cup 2016 extracted from an e-commerce search engine logs containing search and browsing logs, product data and transactions. 4.06% is buy sessions and the average

Table 1 Statistics of dataset. ALL denotes the original released dataset while EXP is the dataset used in the experiments. Y is the yoochoose dataset while D is the diginetica dataset.

name	data	#(events)	#(sessions)	#(items)
Y	ALL	33,003,944	9,249,729	52,739
	EXP	358,887	100,110	20,338
D	ALL	1,235,380	310,324	122,993
	EXP	413,347	103,539	78,671

Table 2 Features used in the prediction

type	feature	description
Session	Session Event	#(clicks or views in the session)
	Item	#(unique items clicked or viewed)
	Category	#(unique categories clicked or viewed)
	Time Span	session time span in milliseconds
Item	Item Event	#(clicks or views for the current item in the session)
	Relative Ratio	(Item Event) divided by (Session Event)
	Duration	duration between the event of current item and the next one
	Position	the position of current item in the event sequence of session
	Item ID	the current item's ID
	Category ID	the category ID of the current item
	Previous Item ID	the previous item's ID
	Next Item ID	the next item's ID
Time	Day of Year	day when the current item is clicked or viewed
	Day of Week	day of week when the current item is clicked or viewed
	Month	month when the current item is clicked or viewed

Table 3 Experimental results on the testing dataset

Dataset	Feature	Model	S-Prec	S-Recall	I-Prec	I-Recall	S-Score	I-Score	Score
Yoochoose	15	CF	0.1505	0.4817	0.1104	0.0966	-145.24	306.78	161.54
	15	GB	0.1838	0.5565	0.1420	0.1352	-124.30	396.89	272.59
	RF	LR	0.1395	0.6383	0.1045	0.1096	-214.11	472.76	258.65
	RF	FFM	0.1456	0.6452	0.1071	0.1120	-203.83	464.34	260.51
	GB	LR	0.1476	0.6557	0.1103	0.1179	-203.16	492.07	288.91
	GB	FFM	0.1398	0.6878	0.1046	0.1119	-230.08	511.80	281.72
Diginetica	15	CF	0.1033	0.2255	0.0729	0.0700	-55.71	118.09	62.38
	15	GB	0.1059	0.3408	0.0782	0.0755	-81.61	184.13	102.52
	RF	LR	0.1321	0.2429	0.0970	0.0887	-43.52	127.59	84.08
	RF	FFM	0.1290	0.2652	0.0989	0.0930	-49.06	149.37	100.31
	GB	LR	0.1148	0.3556	0.0865	0.0809	-76.75	190.80	114.05
	GB	FFM	0.1068	0.3916	0.0777	0.0744	-92.73	205.61	112.88

number of views per search session is 4,271. As shown in Table 1, we extract one hundred thousand session of all sessions (i.e. EXP dataset) randomly from both datasets respectively for experiments. The similar number of session is used to compare the effectiveness on both datasets. Then, we randomly select 60% of sessions as the training dataset, 20% as the validation dataset and 20% as the testing dataset from the EXP dataset. For all experiments, we use the training dataset to learn a model and tune all parameters by the validation dataset. The results are reported on the testing dataset. When FT is required, the half of the training dataset is used to transform features and another half is to learn a model based on the transformed features.

4.2 Experimental Settings

We list all features in Table 2. We divide features into three groups: Session, Item and Time. The categorical features (e.g. Item ID) use the one-hot encoding which leads to the high-dimensional and sparse feature representation.

Two tree ensemble methods (i.e., GB and RF) are used for FT. We use the XGBoost library[†] and concentrate on the binary classification with the logistic loss function and $L2$ regularization. We use an implementation of RF in the scikit-learn library^{††}. In addition, we use two algorithms

for the prediction. One is the Field-aware Factorization Machines (FFM)^{†††}. Another is the Logistic Regression (LR), which is also implemented in the scikit-learn library.

4.3 Evaluation Metric

The evaluation metric is the following score:

$$Score(S_p, S_b) = \underbrace{\frac{|S_b|}{|S|}(TP - FP)}_{\text{session score}} + \underbrace{\sum_{s \in S_b} \frac{|A(s) \cap B(s)|}{|A(s) \cup B(s)|}}_{\text{item score}} \quad (6)$$

where S is the set of all sessions in test dataset, S_p is the set of sessions which are predicted to be buy sessions, S_b is the set of sessions in test dataset which contain at least one buy event, $A(s)$ is predicted bought items in session s , $B(s)$ is actual bought items in session s , $TP = |S_p \cap S_b|$ and $FP = |S_p - S_b|$. It is easy to see that the score formula consists of two terms. The first term gives a reward for correctly predicted session and a penalty for each false one. The reward and penalty are equal to $|S_b|/|S|$. The second term computes the Jaccard similarity of predicted sets of bought items to the actual sets.

4.4 Experimental Results

Table 3 reports the results of machine learning algorithms

[†]<https://github.com/dmlc/xgboost/>

^{††}<http://scikit-learn.org/>

^{†††}<http://www.csie.ntu.edu.tw/~r01922136/libffm/>

with transformed features or original features on the testing dataset. CF in **model** column indicates collaborative filter based on matrix factorization. The **feature** column is the method of feature transformation and **15** means to use original features. While **S-P** and **S-R** columns represent the precision and recall of sessions, **I-P** and **I-R** columns are the average item precision and recall per session respectively. **S-Score**, **I-Score** and **Score** columns are the session score, the item score and score used in the challenge [15].

We can draw the conclusions: (i) GB is better than CF with the original 15 features for the both datasets. It reveals that the interactions of original features are able to improve the performance; (ii) When RF is used to transform features, the results are not consistent for two datasets. FFM is more stable model than LR when random forest is used for FT. The possible reason is that FFM can make better use of the interactions of transformed features than LR, even though the effectiveness of FT is not so good. In addition, GB with the original features is better than FFM and LR using random forest as FT. (iii) When GB is used for FT, the results are consistent for two datasets. In other words, the performance is stable when the GB is used to transform feature for different models. Especially, FFM or LR using GB for feature transformation is better than GB with the original features.

The participants in the RecSys Challenge 2015 present their solutions which are the state-of-the-art methods. Volkovs et al. [16] reveals that deep neural network model is worse than GB. Chen et al. [17] find that deep neural network model is worse than CF. Palovics et al. [18] shows that the linear model is worse than GB. Our proposed method can significantly outperform the GB and CF. Thus, our model can achieve the state-of-the-art performance.

5. Conclusions

In this paper, we explore to use tree-based feature transformation for purchase behavior prediction. Experimental results show that our method is effective. The tree-based features transformation is a promising way to improve the prediction. In the future, it is necessary to compare other methods of features transformation with the tree-based ones for purchase behavior prediction. It is interesting to extend our method to more applications of different domains.

Acknowledgments

Research was partially supported by National Natural Science Foundation of China (No. 61402333, 61402242, 61272450, 61702285) and Tianjin Municipal Science and Technology Commission (No. 15JCQNJC00400). Chen received partially financial support from the China

Scholarship Council (CSC).

References

- [1] X. He, S. Bowers, J.Q. Candela, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, and R. Herbrich, "Practical lessons from predicting clicks on ads at facebook," *Proceedings of ADKDD*, pp.1–9, 2014.
- [2] P. Yan, X. Zhou, and Y. Duan, "E-commerce item recommendation based on field-aware factorization machine," *Proceedings of RecSys*, pp.1–4, 2015.
- [3] F. Kooti, K. Lerman, L.M. Aiello, M. Grbovic, N. Djuric, and V. Radosavljevic, "Portrait of an online shopper: Understanding and predicting consumer behavior," *Proceedings of WSDM*, pp.205–214, 2016.
- [4] Y. Tanaka, T. Kurashima, Y. Fujiwara, T. Iwata, and H. Sawada, "Inferring latent triggers of purchases with consideration of social effects and media advertisements," *Proceedings of WSDM*, pp.543–552, 2016.
- [5] Ş. Gündüz and M.T. Özsu, "A web page prediction model based on click-stream tree representation of user behavior," *Proceedings of the SIGKDD*, pp.535–540, 2003.
- [6] Y. Zhang, W. Chen, D. Wang, and Q. Yang, "User-click modeling for understanding and predicting search-behavior," *Proceedings of the SIGKDD*, pp.1388–1396, 2011.
- [7] J. Goldberger, G.E. Hinton, S.T. Roweis, and R.R. Salakhutdinov, "Neighbourhood components analysis," *Proceedings of NIPS*, pp.513–520, 2005.
- [8] W. Liu, N. Zheng, and Q. You, "Nonnegative matrix factorization and its applications in pattern recognition," *Chinese Science Bulletin*, vol.51, no.1, pp.7–18, 2006.
- [9] J. Yang, A.F. Frangi, J.-Y. Yang, D. Zhang, and Z. Jin, "Kpca plus lda: A complete kernel fisher discriminant framework for feature extraction and recognition," *IEEE Trans. PAMI*, vol.27, no.2, pp.230–244, 2005.
- [10] J.T. Kwok and I.W. Tsang, "Learning with idealized kernels," *Proceedings of ICML*, pp.400–407, 2003.
- [11] S.T. Roweis and L.K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol.290, no.5500, pp.2323–2326, 2000.
- [12] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," *Proceedings of NIPS*, pp.585–591, 2001.
- [13] G.E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol.313, no.5786, pp.504–507, 2006.
- [14] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol.23, no.4, pp.596–608, 2012.
- [15] e.a. Ben-Shimon, "Recsys challenge 2015 and the yoochoose dataset," *Proceedings of RecSys*, pp.357–358, 2015.
- [16] M. Volkovs, "Two-stage approach to item recommendation from user sessions," *Proceedings of RecSys*, pp.1–4, 2015.
- [17] W. Chen, Z. Li, and M. Zhang, "Linear and non-linear models for purchase prediction," *Proceedings of RecSys*, pp.9:1–9:4, 2015.
- [18] R. Pálóvics, P. Szalai, L. Kocsis, A. Szabó, E. Frigó, J. Pap, Z.K. Nyikes, and A.A. Benczúr, "Solving recsys challenge 2015 by linear models, gradient boosted trees and metric optimization," *Proceedings of RecSys*, pp.4:1–4:4, 2015.