Solving RecSys Challenge 2015 by Linear Models, Gradient Boosted Trees and Metric Optimization

Róbert Pálovics[1,2] Péter Szalai[1,2] Levente Kocsis[1] Adrienn Szabó[1]
Erzsébet Frigó[1] Júlia Pap[1] Zsófia K. Nyikes[1,2] András A. Benczúr[1,3]
[1]Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI)
[2]Technical University Budapest [3]Eötvös University Budapest
{rpalovics, pszalai, kocsis, aszabo, fbobee, papjuli, nyzsofi, benczur}@ilab.sztaki.hu

ABSTRACT

The RecSys Challenge 2015 task requested prediction for items purchased in online web shop sessions. We describe our method that reached fifth place on the leaderboard by constructing a large number of item, session, and session-item features and using linear models and gradient boosted trees for learning. An important element of our method included optimization for the specific evaluation metric.

1. INTRODUCTION

The RecSys Challenge 2015 [1] considered online shop sessions of item views and purchases. The data contained information about clicks (item views): the IDs of the item and the session, a time stamp, and item category. For the training data, information about the buy events was also present: timestamp, session and item IDs, price of the item and the quantity bought. We constructed over 500 features out of this data, from which some depend only on the item, some on the session, and some on the item-session pair. From these we selected 145 for modeling.

By the lessons learned from RecSys Challenge 2014 where a large number of user, item and user-item pair features were available, we were primarily relying on linear models [2] and gradient boosted trees [5].

The challenge task actually combined two subtasks in one evaluation measure, one for predicting whether a session contains any buy events, and another for predicting the actual items bought. For both criteria, one had to give binary answers: a session had to be predicted either buy or not buy (by predicting at least one, or zero items of the session) and the predicted bought items also had to be listed. Our idea was to optimize for the specific evaluation function, in part by learning model combination weights, and in part by selecting different thresholds based on the number of items in the session and also considering whether or not we predict another high probability buy event in the session.

ISBN 978-1-4503-3665-9/15/09$15.00 DOI: http://dx.doi.org/10.1145/2813448.2813513

Table 1: Size of the training and test dataset.

	train click	train buy	test (click)
number of events	33,003,944	1,150,753	8,251,791
number of items	52,739	19,949	42,155
number of sessions	9,249,729	509,696	2,312,432

Some methods were not able to improve our final combination, probably only due to lack of time to find the optimal parameters. First, we computed matrix factorization over an item-item matrix where rows corresponded to items in a session and columns to items bought. In this sense, we constructed a recommender system that, given an item appearing in a session, predicts the likelihood of purchasing the other items. Second, we designed a stacked learning method that, given the predictions for the items in a session, constructs new session features by aggregating the predictions in the session. Given the new features, the model could then be retrained in a second iteration.

In our final best performing method that reached fifth place with a value of 59,845 on the leaderboard, we combined our linear models and Dato's[1] gradient boosted trees, finally we applied a threshold on the prediction that was based on the length of the session and the fact whether there was at least one other item in the session with high predicted probability of being bought.

2. THE RECSYS CHALLENGE 2015 DATA

The challenge dataset is split into *training* and *test* sessions sampled from the same time period between 01-04-2014 and 30-09-2014. The test data only consists of click events, and the task is to predict buy events for these sessions. Table 1 summarizes the size of the training and test data. In our internal experiments, we tuned parameters on a 10% random sample evaluation set.

2.1 Statistics

In order to generate features, we investigated various statistics of the sessions. Figure 1 shows the distribution of the number of clicks in sessions with buy events, and for all sessions. Both distributions are heavily tailed, however they slightly differ. In order to emphasize the difference between sessions with and without buy events, we investigated and plotted the buy-click ratio of the sessions as the function of various features. We plot the session length in Fig. 2, the days of the week in Fig. 3 (the ratio drops on Saturdays), and the session start time of the day in Fig. 4.

[1]https://dato.com

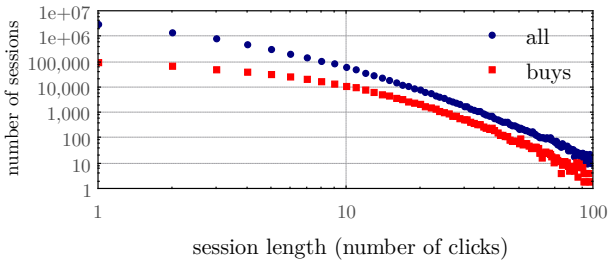

Figure 1: Session length distribution for all sessions, and for sessions with buy events.

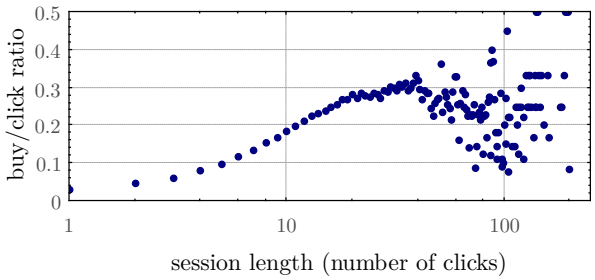

Figure 2: Buy ratio as the function of the session length.

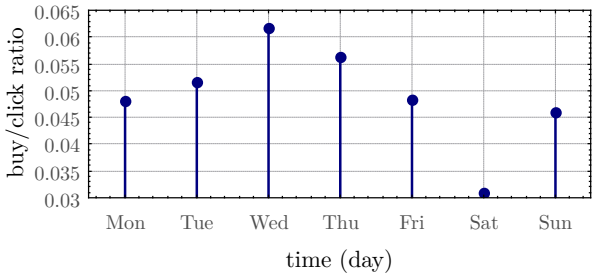

Figure 3: Buy ratio averaged for the days of the week.

For each item in each session, we computed the time passed since the last buy event of the item. Next we measure if this recency of the item correlates to higher buy ratio. In Figure 5 we can observe that the shorter the time elapsed since the last buy event, the higher the buy-click ratio. As the training and test data is sampled from the same time period, we could also compute this feature for clicks in the test data. Furthermore, as the structure of the dataset allows to use the future, we also computed the time difference till the next buy event in the future.

Figure 6 indicates that the more clicks a user made for an item in the session, the more likely she will buy the product. We observed the same for the session-item dwell time, i.e. the amount of time until the next click of the user during the session (see Fig. 7). Figure 8 shows that the position of the click in the session is also relevant. The peak of the curve is at the 15th click in the session.

Regarding the category labels of the items, there are 13 main categories indexed from 0 to 12. Category number 13 indicates if the item is on sale. Furthermore, there are special brand based categories that we merged to category 14. Figure 9 shows that the average buy-click ratio values for the 15 categories greatly differ.

2.2 Feature set

Based on our statistics, we have generated numerous features that belong to three main categories: session features (e.g. session length), item features (e.g. global item popularity), and session-item features that describe items within a given session (e.g. item-session dwell time). The number of our main features is 42 and their full list is found on our Website[2]. Besides these, we further generated several secondary features as functions of one of the main features. Where it was reasonable, we computed both the difference and absolute difference from the feature average, minimum, and maximum value. Furthermore, we could generate many session features by computing the average, minimum, and maximum values for item and session-item features within the given session. For each click, we also computed the difference of the feature and the session average, minimum,

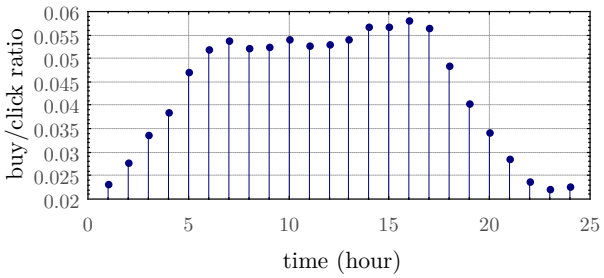

Figure 4: Buy ratio averaged for each hour in a day.

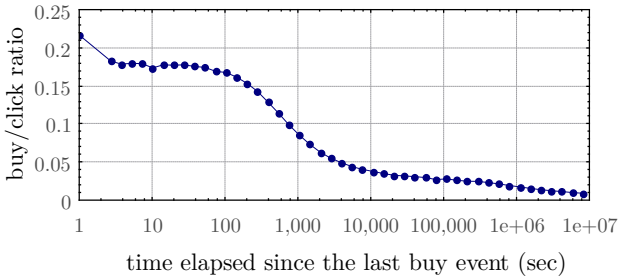

Figure 5: Buy ratio as the function of the time elapsed since the last buy event.

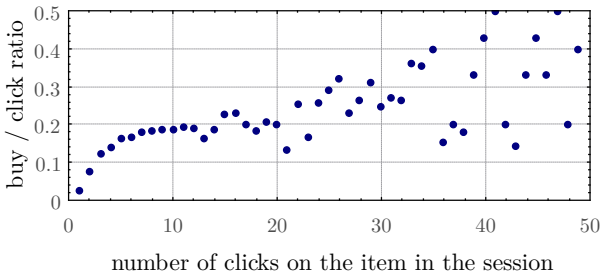

Figure 6: Buy ratio as the function of the number of clicks on the item in the given session.

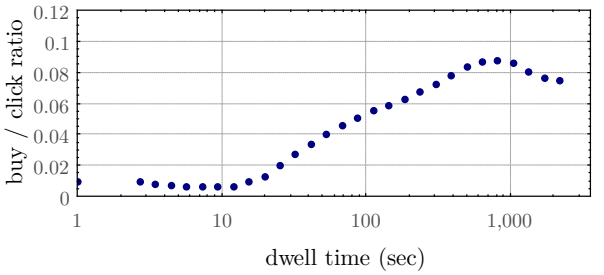

Figure 7: Buy ratio as the function of the time the user spent on the given item during the session.

[2] https://dms.sztaki.hu/en/download/recsys-challenge-2015

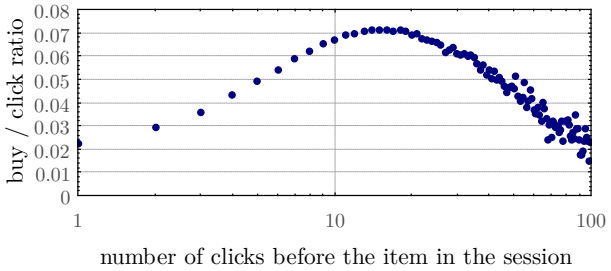

Figure 8: Buy ratio as the function of the position of the click in the sequence of clicks in the session.

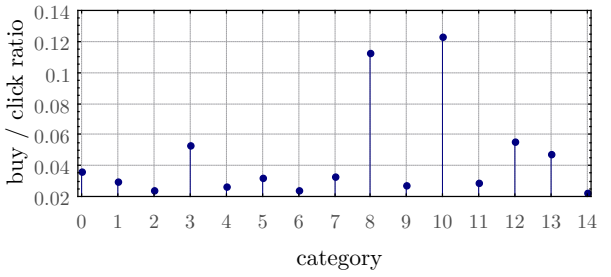

Figure 9: Buy ratio for each category. Sale ("S") is denoted by number 13. Special categories are merged to category number 14.

and maximum values. After discarding features that were irrelevant, our final dataset included 145 features.

3. EXPERIMENTS

3.1 The evaluation metric

A specific new measure was constructed for the competition based on the following session-related quantities:

- the set of sessions predicted to end with buy (Sol),
- the set of all the sessions in the test data (S),
- the set of test sessions that end with buy (S_b),
- the set of predicted bought items in session s (A_s), and
- the set of actual bought items in session s (B_s).

The score of the solution is

$$\sum_{\forall s \in Sol} \begin{cases} \frac{|S_b|}{|S|} + \frac{|A_s \cap B_s|}{|A_s \cup B_s|}, & \text{if } s \in S_b, \\ -\frac{|S_b|}{|S|}, & \text{otherwise.} \end{cases} \quad (1)$$

A solution gets non-zero score only for sessions with a predicted buy event. For each session that has at least one predicted buy event, it earns $\frac{|S_b|}{|S|}$, if the session has at least one real buy event. Otherwise, the metric penalizes with the same amount of score.

One can earn extra points by predicting the set of bought items correctly. The amount of the extra points is the Jaccard score between the predicted bought items and the actual ones, $\frac{A_s \cap B_s}{A_s \cup B_s}$. The minimum reachable score is $-|S_b| + \frac{|S_b|^2}{|S|}$, while the maximum is $|S_b| + \frac{|S_b|^2}{|S|}$. The maximum possible score on our evaluation set is around 54,000.

3.2 Item-to-item matrix factorization

We made several unsuccessful experiments to apply matrix factorization (MF) on the challenge dataset. First, we considered a click-to-buy matrix, where the item pair (i, j) is a positive instance, if i was clicked on in at least one session where j was bought. If j was not bought but clicked, we considered the pair as a negative instance. After training our MF model, we predicted for item j a score by summing up for all i items in the session the predicted (i, j) scores. Second, we intended to investigate items that are frequently bought together. We considered an item pair as a positive instance, if the items were bought together at least once. Our final goal was to combine item-to-item transitions extracted from the MF output with classifier results.

3.3 Gradient boosted trees

We labeled a click in session s on item i positive, if it had a buy event during s. Otherwise, we treated the click as a negative sample. We used the corresponding session, item, and session-item features to predict the label of the click. We trained Dato's boosted trees classifier model[3]. We made several measurements on smaller random training and evaluation datasets to fine tune the parameters of the classifier. In our final model we set `iter_num`= 42, `max_depth`= 8, and `step_size`= 0.4.

3.4 Linear model

The linear model included binarized version of some of the features discussed in Section 2.2, both simple and combined. Examples for simple features are item features, where with the attached weight binarization leads to item biases. A binarized version of the item and date feature combination leads to item biases specific for each day. The simple features used were: number of clicks on the item, the item, the day of the week, the hour of the day, dwell time on the item, and the maximum dwell time in the session after the item. The combined features included the item and one of the following: date, number of clicks on the item, dwell time, category, and subsequently clicked item. The model was trained by gradient descent minimizing the mean square error. The error was defined as the difference between the buy event and the predicted value. The error function included additional regularization terms. The gradients of positive instances were multiplied by 10.

3.5 RSPSA

After combining the model scores predicting whether an item will be bought or not, our next task is to set a threshold on the prediction score to predict a buy event in the session. In addition, we predict whether the session includes any buy events. If not, then we override the threshold and predict no items. The session based model relies on features such as the maximum item score, sum of item scores, sum of the squared item scores, number of clicks, and number of items above a threshold. The model was trained directly on the evaluation measure of the competition. Since the analytic expression of the gradient of this error is non-trivial, if it exists at all, we employed an algorithm that approximates the gradient using stochastic perturbation of the parameter vector, namely RSPSA [4].

Denoting the parameter vector of the model by θ, and the evaluation metric for sessions s by $Score_s(\theta)$, the ith component of the gradient is approximated by

$$\hat{g}_{si}(\theta_t) = \frac{Score_s(\theta_t + \Delta_{ts}) - Score_s(\theta_t - \Delta_{ts})}{2\Delta_{tsi}},$$

[3] `https://dato.com/products/create/docs/generated/graphlab.boosted_trees_classifier.create.html`

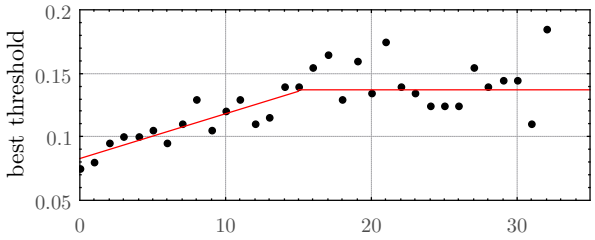

Figure 10: Optimal recommendation threshold as the function of the session length.

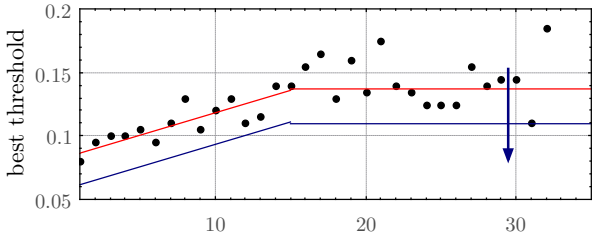

Figure 11: Lowering the threshold curve for items that are in sessions with at least one already recommended item.

where Δ_{ts} is the perturbation vector in tth iteration for session s. RSPSA uses the batch gradient update rule RPROP [3] that adapts the step sizes δ_{ti} depending on the sign of the gradient. The perturbation vector and the step sizes are coupled through a random vector c_t, whose components are typically ± 2: $\Delta_{tsi} = c_{tsi}\delta_{ti}$. Model parameters and step sizes change after every batch iteration, while separate perturbations are generated for every session in every iteration.

3.6 Session length based recommendation

After training the boosted trees classifier, first we recommended items that had classifier scores larger than a global threshold. Second, we introduced thresholds based on session length. On the evaluation set, we computed the optimal threshold to gain the highest score as the function of the session length in clicks, as shown in Figure 10. We approximated the curve with a linear function for length between 1 and 15. Beyond 15, the curve gets noisy and therefore we used a constant. As a result, session length based thresholds for the final recommendation improved our recommendation quality.

3.7 Optimizing for the final metric

The evaluation metric is the sum of two components (see Section 3.1). Based on (1), it is less risky to recommend a few more items for sessions that already have at least one recommended item. To recommend additional items, first we predicted session-item pairs based on Session 3.6. Then, we lowered the threshold curve by some constant (see Fig. 11). For each session that had at least one recommended item, we recommended a few more items based on the lowered threshold curve.

Table 2: Best results of the individual models and their combination on the evaluation set.

linear model	16,534.2
boosted tree, main features (42)	17,663.2
boosted tree, all features (145)	19,010.8
linear combination	19,676.5
combination + RSPSA	20,017.7
combination + score opt.	20,180.9

3.8 Final results

In table 2 we summarized the performance of our best models on the evaluation set. Note that the maximum possible score on our evaluation set is 54,030.8. The boosted trees classifier gained 17,663.2 with the main features, and 19,010.8 with the extended feature set. The linear model was weaker than the boosted trees model with score 16,534.2, but helped in the final combination. We gained 19,676.5 by the linear combination of the gradient boosted tree results and the linear model. Over this combination model the RSPSA could improve our score to 20,017.7. The optimization techniques explained in Sections 3.6-3.7 improved the combination result to 20,180.9, which yielded our final result scoring 59,845 on the leaderboard.

4. CONCLUSIONS

Our RecSys Challenge 2015 submission reached fifth place by relying on feature engineering, linear modeling, gradient boosted trees, and optimization for the evaluation metric. We expect additional ideas, most notably an item-item matrix factorization and a two-iteration stacked learning procedure, to be able to improve the quality of the prediction.

5. REFERENCES

[1] Ben-Shimon, D., Tsikinovsky, A., Friedman, M., Shapira, B., Rokach, L., Hoerle, J. RecSys Challenge 2015 and the YOOCHOOSE Dataset. In *Proceedings of the 9th ACM conference on Recommender systems*, ACM, 2015.

[2] F. Guillou, R. Gaudel, J. Mary, and P. Preux. User engagement as evaluation: a ranking or a regression problem? In *Proceedings of the 2014 Recommender Systems Challenge*, page 7. ACM, 2014.

[3] C. Igel and M. Hüsken. Improving the Rprop learning algorithm. In H. Bothe and R. Rojas, editors, *Proceedings of the Second International ICSC Symposium on Neural Computation (NC 2000)*, pages 115–121. ICSC Academic Press, 2000.

[4] L. Kocsis and C. Szepesvári. Universal parameter optimisation in games based on SPSA. *Machine learning*, 63(3):249–286, 2006.

[5] R. Pálovics, F. Ayala-Gómez, B. Csikota, B. Daróczy, L. Kocsis, D. Spadacene, and A. A. Benczúr. Recsys challenge 2014: an ensemble of binary classifiers and matrix factorization. In *Proceedings of the 2014 Recommender Systems Challenge*, page 13. ACM, 2014.