Datamining Techniques 2018 Project 2

Group 28 Scientific report

Changxin Miao 11853018 Cmo440, Michael Mo 10770518 Mmo740, and
Féliciën Veldema 10739335 Fva350

University of Amsterdam

Abstract. We analyse our group process of constructing a predictor on whether users book or view a hotel on Expedia data. We prepare the data, analyse and transform the data in order to optimize our predictor. Predictions are evaluated with NDCG on a test set.

Keywords: Data mining techniques Expedia hotel prediction

1 Introduction

This report explains the second Datamining techniques course project of 2018. For this project we need to construct a model which takes searching data on hotels as input and predicts whether the user reserves the option. In section 2 we go more in depth into the problem while examining work from peers. Next, we analyse the data ourselves and prepare the data for our selected model explained in sections 3-5. We end our report by presenting and evaluating the obtained results followed by a conclusion.

2 Business Understanding

The data originates from a competition held by Kaggle[1] by using the dataset from Expedia to create a ranking algorithm that maximizes the hotel purchases. 337 teams participated and were scored on the Normalized Discounted Cumulative Gain on the output of the 38 best options per search query (NDCG@38). As the competition concluded in 2013, presentations on the dataset and the approaches of the top 3 competitors are published. To not deter from our own research we will review the keypoints most important to our approach.

The Expedia presentation[2] shows a strong feature correlation in number of adults and rooms, and length of stay and booking window. With a strong anti-correlation between a short night stay and length of stay. The amount of stars a hotel has received is also highly correlated to most other dynamic features.

[1] https://www.kaggle.com/c/expedia-personalized-sort

[2] https://www.dropbox.com/sh/5kedakjizgrog0y/_LE_DFCA7J/ICDM_2013
2_data_description.pdf - provided by Adam the competition host

There are some problems within the dataset such as the position bias of the data. First presented options are more likely to be clicked, which also factors in the results, as clicked hotels increase the final score. Likewise, more than the half of the features have 50% or more missing inputs. These will be modified in section 4 to ensure data completeness.

Significant insights could be derived from wining teams. [3]. Owen has down sampled negative instances in order to facilitate faster learning and composited new features such as price order within search id. Furthermore they have adjusted categorical features into numerical onces by using a weighted average. For modeling they have used an ensemble of Gradient Boosting Machines (GBM) and concluded that position, price and location desirability are the most important features.

Team Jun assumes that relevance score is based on rational behavior. Which means that the user only books the best option, therefore irrational decisions are treated as random noise. The team modeled several linear and nonlinear models, but the nonlinear Lambda Multiple Additive Regression Tree (LambdaMART) outperformed the others. For feature preparation they opted with adjusting missing values with the worst-case scenario and using discriminative features for the classifier. As most of the historical data is missing, they highlight the matching data. Features are then normalized with respect to different indicators such as search id and month.

Binghsu preprocessed the data using 10% and form a balanced data selection. They found that presenting the prices of the hotels as a list works well. GBM and LambdaMART were the highest scoring models on the validation set. Their conclusion is that GBM or LambdaMART is the most robust in ranking. A Linear Model is efficient in ranking but requires more data preparation. A deep learning model that ranks the hotels pointwise is not efficient.

From these presentations we are able to focus our attention on important aspects. Our model should be either a GBM or LambdaMART. Furthermore we should prepare the data by composting features and making new features by ordering on price for example. As all the presentations indicate that giving an ordered feature and normalizing on different conditions boosts the NDCG prediction score.

3 Data Analysis

3.1 The data

The dataset we use is a 1.2GB csv file containing information about search queries for hotels of multiple users. Different hotels are shown to the user together

[3] Referred presentation links are found in footnote 2

with their properties, user information and the resulting user activities (i.e. user clicks and bookings) are recorded. To understand the whole dataset better we first look at a global level what data we have and compute descriptive statistics. In total our dataset covers 199.795 unique queries which together make a total of 4.958.347 data rows. All the data comes from an eight month period (2012-11-01 to 2013-06-30), with roughly 600.000 data points per month.

3.2 Correlation analysis

In addition to all insights we have gained so far, it will also be interesting to find the correlation of the internal variables and their influence on the booking and clicking behaviors. Therefore, we divide variables into five different groups: search criteria, visitor information, hotel information (static), hotel information (dynamic) and competitor information. At first the correlation analysis is applied to all variables, then more detailed insights could be gained by performing the analysis on grouped variables.

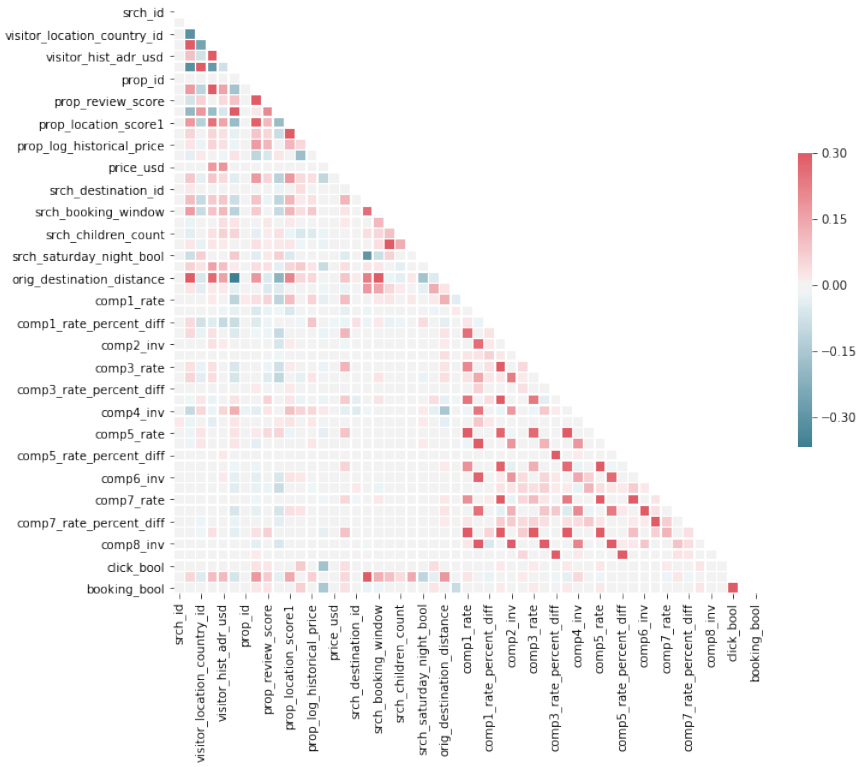


Fig. 1: Variable correlation matrix

According to the above correlation matrix, figure 1, and detailed visualization for grouped variables, several significant variables could already be identified. They are 'price usd', 'prop starrating', 'prop review score', 'prop location score1', 'prop location score2', 'prop log historical price', 'position', 'promotion flag', 'srch query affinity score', 'random bool', 'click bool'[4]. All of the above variables will be included as static exogenous variables in all models.

3.3 Position and price analysis

Next, we analyze whether the position of the hotel shown to the user influences the click rate. We first check the number of times some hotel was shown at each position and notice that hotels shown in positions 5,11,17 and 23 all have a very low count. We assume that in most cases something like an advertisement was shown there, meaning no hotel was at that position. The two cases (random / normal sort ordering) are then compared in Figure 2. In both cases the higher the hotel is shown, the higher the click rate is. However, the click rate for random ordering is less steep. This implies that the clickrate will not only depend on ordering, but also on properties of the hotel itself.

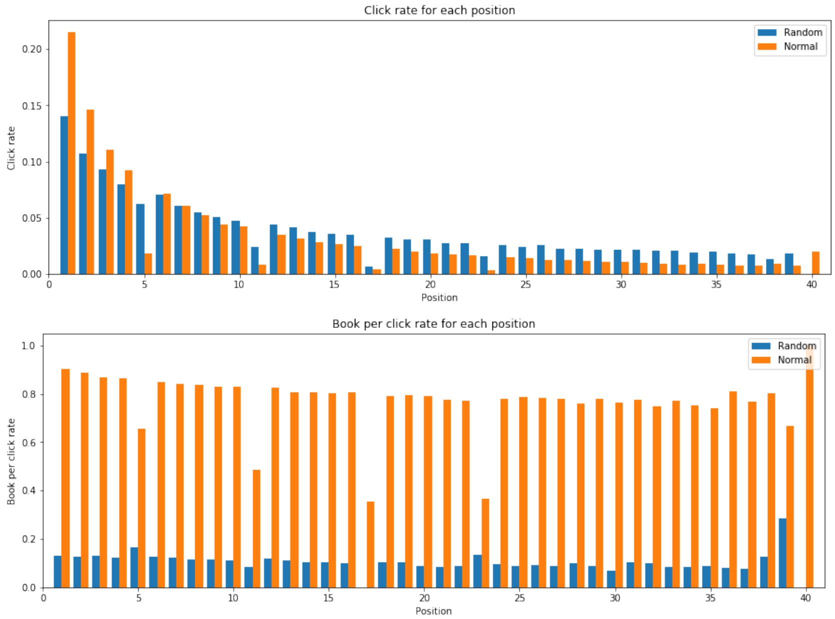


Fig. 2: Click and book-per-click rates per position for normal and random ordering.

[4] Not all variable names are displayed in the figure due to size

Looking at the book per click rate for the two orderings we also deduce that in random ordering most of clicks were made as result of the position of the hotel rather than hotel properties since the booking rate is lower.

At last we investigate how much the booking depends on the cost of the hotel. For each booking the rank of the price of the hotel relative to all hotels shown in the same query is calculated, with the results shown in Figure 3 where we see that users mostly book the cheapest hotel available.

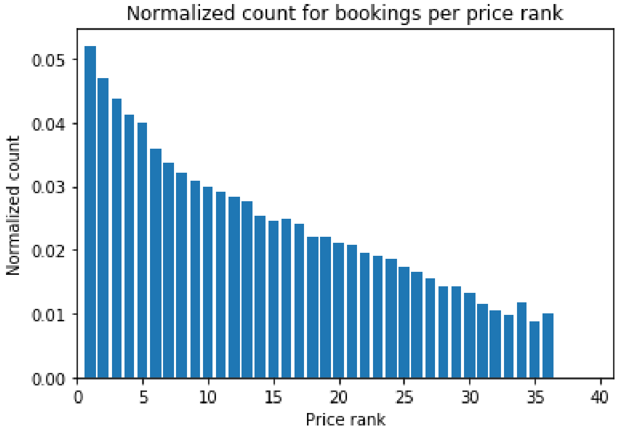


Fig. 3: Overview of number of bookings per hotel sorted on prices available

Prices of the hotels shown also depends on the country of the hotel. How much the price differs for each country is computed, and we can conclude that hotel prices differ hugely per country, as well as within the same country.

4 Data Preparation

4.1 Feature engineering

After we gained sufficient insights into the descriptive statistics for each variable and paired the correlation, we decide to perform feature engineering for key attributes. Referring back to the correlation analysis in section 3, there are five groups of variables within the current dataset. A strong internal correlation relationship is indicated by the 'competitor info' group. Nevertheless, the whole group of variables is not significantly correlated with external variables. This implies that there exists a large amount of noise within this data group and it results in our decision to perform PCA to extract representative features for this variable group. In this way, the whole group of variables are replaced by the representative variables 'cr1' and 'cr2'.

For all significant variables we have identified[5], we decide to standardize it based on the search visitor country id and property location id. The reason is that the visitors' original country could significant influence their perception on expensiveness. If we treat them equally, the model could not identify the different economic backgrounds and fair price level, leading to incorrect predicting results. All significant numeric variables are standardized based on the key 'visitor location country id' and 'srch destination id'.

4.2 Handling the missing data

Boolean variables such as 'book bool', 'promotion flag', 'random bool', and 'click bool' are essential on constructing the normalized data set. Hence, data points with missing values from these variable groups were dropped.

For numerical variables, at first we group data points with unique 'visitor location country id' and 'property country id' into groups. Then the missing values from each column is replaced by the mean of each group. When we need to normalize unseen data and encounter an unseen group (an unseen 'visitor location country id','property country id' pair), it is done by using the average mean and standard deviation of all encountered groups of the training data set. As we standardized the data beforehand, it is expected that to be filled in values will be around 0. Furthermore, if a the whole group is not available, we will replace all numerical values of that group with 0.

4.3 Splitting the test and training set

The whole training set would be split into training (80%), and testing (20%) sets. 20% of data within the training data set is separated as the test set but with labels, for the purpose of training evaluation. K-fold cross validation method is also applied to construct models.

5 Model

5.1 Random forest

Random forest classifier implements the bootstrap sampling (sample with replacement) strategy to construct and ensembles different decision trees. Hence, it decreases the variance of different trees and avoids the over-fitting problem. The advantage of random forest is that further randomness is introduced by identifying the best split feature from a random subset of available features. In this way, we could implement this model to identify most significant features which contribute to the click and booking variables.

[5] see section 3

After several times of trail and error, it is identified that the model performs best with a random forest model 1 to predict 'click bool' variable at first, then random forest model 2 uses this additional variable to predict 'booking bool'. All variables in the training dataset as well as PCA features except 'position' are input into the model. 10-folds cross validation technique is also implemented within this model. Furthermore, we notice that a significant amount of '0' exists in the current data frame. The 'class weight' of the model will be set to be 'balanced' in order to include more examples with '1' in the model.

5.2 Gradient Boosting Machines

Gradient Boosting Machines are a family of learning techniques which are usable in a wide range of applications. This is due to their customization aspect such as being applicable with different loss functions. In our case we build a gradient boosting machine classifier as separating booked hotels from not booked hotels suffices for the problem.

GBM learn by consecutively fitting new models on the training set, with each next model attempting to reduce the loss. Next to being highly customizable GBMs are simple to implement compared to other learning algorithms. However, GBMs require a lot of trial and error on selecting the most optimal hyperparameters for the problem.

5.3 LambdaMART

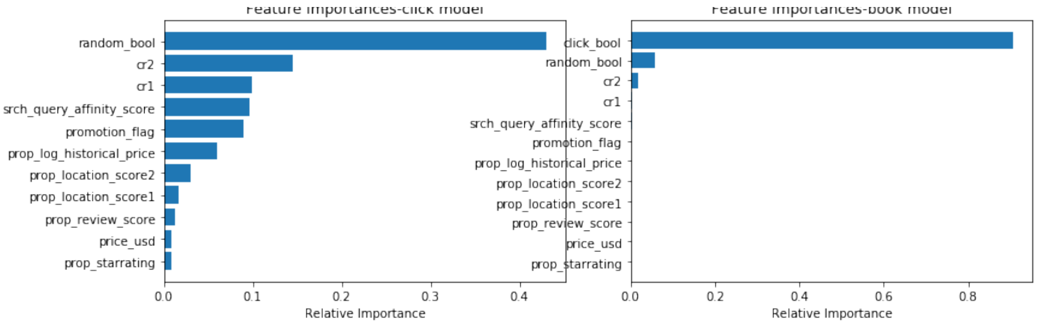
The third model we consider to use is LambdaMART. Instead of using the traditional machine learning approach which is implemented by the previous two models (classifier based with input-target pairs), LambdaMart is a model designed to perform ranking. It is a listwise approach where a relative ranking is calculated by comparing pairs [1].

For this task, the model is a more direct approach for the problem to compute rankings. LambdaMART has been proven to be giving good results in many real-world ranking problems, therefore we expect it to be able to outperform the other models in this research. We created the LambdaMART model via RankLib [2] implementation.

6 Results

6.1 Random forest

Some interesting insights that we could gain from the random forest model are feature importance of each variables.



As we could observe from the above graph, 'random bool', 'cr2', 'cr1' contribute most to the prediction of click while the main determinant for booking is 'click bool'.

7 Evaluation

7.1 Model accuracy

	Random forest model	GBM model
Accuracy rate	0.6809(rf1) 0.9938(rf2)	0.95

Table 1: Accuracy rate for classifier model

Although the accuracy for second random forest model is impressive, but it requires 'click bool' as the input. The accuracy rate of predicting 'click bool' is only 68%. It decreases the performance of classifier significantly. For the GBM model, although the prediction accuracy is relatively high, almost all predicted values are '0's, .

7.2 Model performance

To get an estimate how good our models will perform on "test_set_VU_DM_2014.csv", we compute for each search ID in our created test set the nDCG of the ranked list according to each model. Hereby we also create a baseline model, which outputs a random ranking of the hotels for each search ID.

The number of search IDs in the test set we evaluated each model on is 40012. The mean nDCG and standard deviation of the models is visible in Table 2. We see that both the random forest and LambdaMART give a higher nDCG than the baseline model. LambdaMART also performs a bit better than the random forest model. Note that the nDCG can also vary quite a bit per search ID which can be seen from the standard deviation. We also perform a t-test between each pair of models and the results are given in Table 3. As expected (because of the large sample size) we get p-values almost equal to 0, and conclude that on the test set our models perform significantly better than the baseline and that LambdaMART performs siginificantly better than the random forest model.

	Baseline	Random forest	LambdaMART
Mean nDCG's	0.352	0.422	0.463
SD nDCG's	0.192	0.234	0.259

Table 2: Mean and standard deviation of nDCG's for all search IDs in test set for the models.

	t-statistic, p-value
Baseline vs Random forest	46.3, 0.00
Baseline vs LambdaMART	69.2, 0.00
Random forest vs LambdaMART	23.7, $5.26 \cdot 10^{-124}$

Table 3: t-tests for average mean per search ID for different models.

The performance for the GBM is not displayed as the model performed poorly. GBM prefers balanced data to work with and balancing the provided dataset proved to be a delicate task. The output performance has an accuracy of 95 by only predicting not booked or clicked. Due to an abundance of negative samples in the dataset the GBM underperformed. Optimizing a balanced dataset would require more time.

7.3 Final model choice

We would expect all models to give similar results for the output file because of relative small divergence in the nDCG value. The LambdaMART model gave the best results as expected since it is specially designed for the ranking algorithm. Our final comparison also emphasizes this point, therefore we are going to select this model to predict results.

8 Conclusion

We implemented the random forest, gradient boosting machine and LambdaMart models to accurately predict whether a property is likely to be clicked or booked based on search criteria, property information and competitor information.

In the explanatory data analysis part, we identify significant variables for outcomes, correlated variables for feature engineering and exterminate the potential biases. Specifically, the price bias helps us to make the decision to normalize the dataset based on country and search id group.

For random forest and GBM, it is obvious that the variables 'click cool' contributes most on the final decision of being booked. Additionally, in order to predict the 'click bool',results show that the 'random bool', 'cr1' and 'cr2' are important features.

A classifier model, such as GBM, could achieve a high accuracy rate overall, but

doing so by only predicting negative outcomes due to the content of the dataset. The final evaluation after performing student t-test entails us to make decision on selecting LambdaMART as the most suitable model to predict the outcomes. This decision is likewise based on the fact that LambdaMART's ranking objective takes list-wise comparison into account.
Some trivial business insights we could gain from this project is that the users price perception differs per countryand showing a relative price ratio of the property will help searchers make decisions. Furthermore, competitor information and hotel promotion flag influences the users' decision more than solely the presented position of the hotel.
A limitation of the project is not optimally tuning the balanced dataset for GBM as it seems ideal but is not possible within the time constrains. Further improvements could be incorporating the on-line feedback features of users into the current model, in order to adjust the predicted values more accurately.

9 What we learned

This course enables us to go over the necessary steps of analyzing a dataset, incorporate different techniques to explore the data and gain profound insights from it. Necessary steps for analyzing the dataset is explanatory data analysis, data pre-processing, model construction and model evaluation. Different models (SVM, neural network and random forest) are experimented to achieve various goals (classification, prediction or association rules.) Practical projects entail us to apply theories into practice.

Project 1 required us to hone in on feature selection and model comparison. We gained a deeper understanding about time-series analysis from that experience. For project 2, we applied classification and ranking models in order to build a model with high accuracy rate. It is also possible for us to identify key drivers for each model. It is exiting for us to combine information retrieval and machine learning techniques to finish this project

In addition to practical skills, we also come across some interesting topics, such as ethics and moral issues, text mining and the experience in project team. This topics capture our attention to the integrity and regulations appeared in the big data industry and enrich our comprehension with the data scientist career.[6]

References

1. Burges, Christopher JC. "From ranknet to lambdarank to lambdamart: An overview." Learning 11.23-581 (2010): 81.
2. Dang, V. "The Lemur Project-Wiki-RankLib." Lemur Project,[Online]. Available: http://sourceforge.net/p/lemur/wiki/RankLib.

[6] Code of the project can be found on: `https://github.com/felicienveldema/DataMT`