Data mining assignment 2

Qinghe Gao 2659024, Wenlin Chen 2664925, and Xiaoqing Han 2686611

Group 126

1 Introduction

This paper writes about our team's data mining project of Expedia hotel recommendation in detail, participated in 2020 VU Data Mining Techniques Cup. The content includes Exploratory data analysis, data preprocessing, feature engineering, model establishment (lambdaMART LightGBM) and fitting result generally. Firstly, 54 attributes are grouped into 5 categories: search criteria, hotel characteristics, customer information, competitor information and other information. Then we filled the missing values based on its own distribution and scale. Next, in feature engineering, by combination, extraction and normalization, 52 features are selected as parameters for models. Then, lambdaMART model is tested but it is time-consuming, instead, LightGBM is more suitable for ranking and performs better. Finally, the result of NDCG shows best score are training's NDCG: 0.472645 and validation ndcg: 0.421116. Final grade in Kaggle is 0.40119 and ranking 12 of 232 teams(till 4pm 22/05).

2 Business understanding

Tracing back to the "Personalize Expedia Hotel Searches - ICDM 2013" competition hold by Expedia in kaggle platform [4], which is the origin of our assignment 2. The top 4 winner teams mainly use **Gradient Boosting Machines (GBM)** and **LambdaMART** as the predictors. Except these two instances using fast learning. (2) To extract features, all features are divided into 4 kinds: numerical features, composite features, category features converted into numerical features and estimated position of hotels based on historical data. (3) Since GBM model is not sufficient for big data. this team divided the dataset into 26 instances. Test model with EXP features and without it AND test model with problems or without problems after fixing it. (4) The NDCG scores of GBM are the sum of weighted scores from various model results mentioned in step 3.

The No.2 team with lambdaMART model: (1) For missing data of hotel description, the team fill missing value with worse case scenario; for users' historical data, the team highlight the matching and (or) mismatching between users' historical data and the given hotel data; For competitor descriptions, missing values are all set to zero. (2) In feature extraction, the team generate two kinds of features by statistical ways: Hotel quality estimation and Non-Monotonicity of Feature Utility. And they did features normalization to remove the effect of scales. (3) The team test two Linear models: regression or SVM-Rank and a

nonlinear model: LambdaMART. (4) The NDCG result shows that the LambdaMART is the best.

Among all teams, the general steps are : Data preprocessing ⇒ Feature Engineering ⇒ Model ⇒ Evaluation.

3 Data understanding

There are 4958347 (49.9958%) results are training set and 4959183 (50.0042%) results are testing set. And we combine them together to analyze. In this part, The exploratory data analysis is implemented to the whole dataset (train+test).

Missing Value : In general, there are 54 attributes in this dataset. However, 4 attributes "position", "click_bool", "booking_bool" and "gross_booking_usd" are only provided for training data. As the result, the missing rate of these four attributes are effected obviously. The missing rates of factors, whose missing rate is higher than zero, are sorted in descending order in the figure 1.

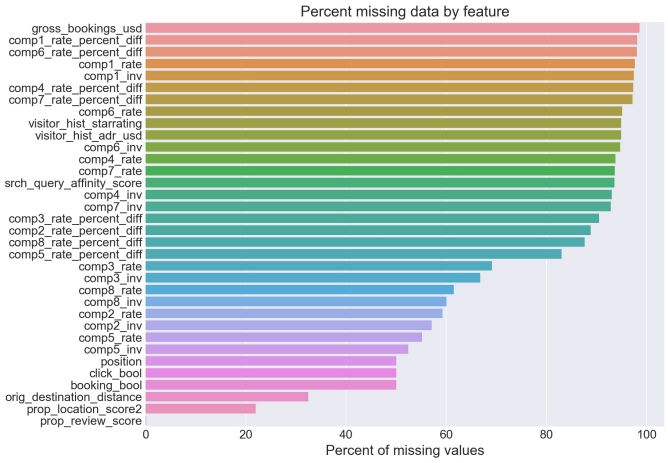

Fig. 1. Missing values of whole dataset(train and test).

Look into the characteristics of 54 attributes, they are divided into 5 groups: search criteria (srch_length_of_stay, srch_booking_window", etc.), hotel characteristics (prop_starring, prop_review_score, etc.), visitor information (visitor_his_starrating, etc.), competitor information (com1_rate to com8_rate) and other information (date_id, site_id). Due the the large scale of attributes, the data understanding focuses on each group in the following part.

Search Criteria : From the distributions in the figure 3, several findings are that (1) most customers prefer to book hotel fewer days before departure. (2) Usually, no more than 4 children will go outside together. (3) The length of stay is densely distributed within 10 days. (4) The mode of booking rooms is 2 rooms. Looking at the correlation between attributes, (5) the days between booking and checking in is relatively correlated with the length of stay at hotel.

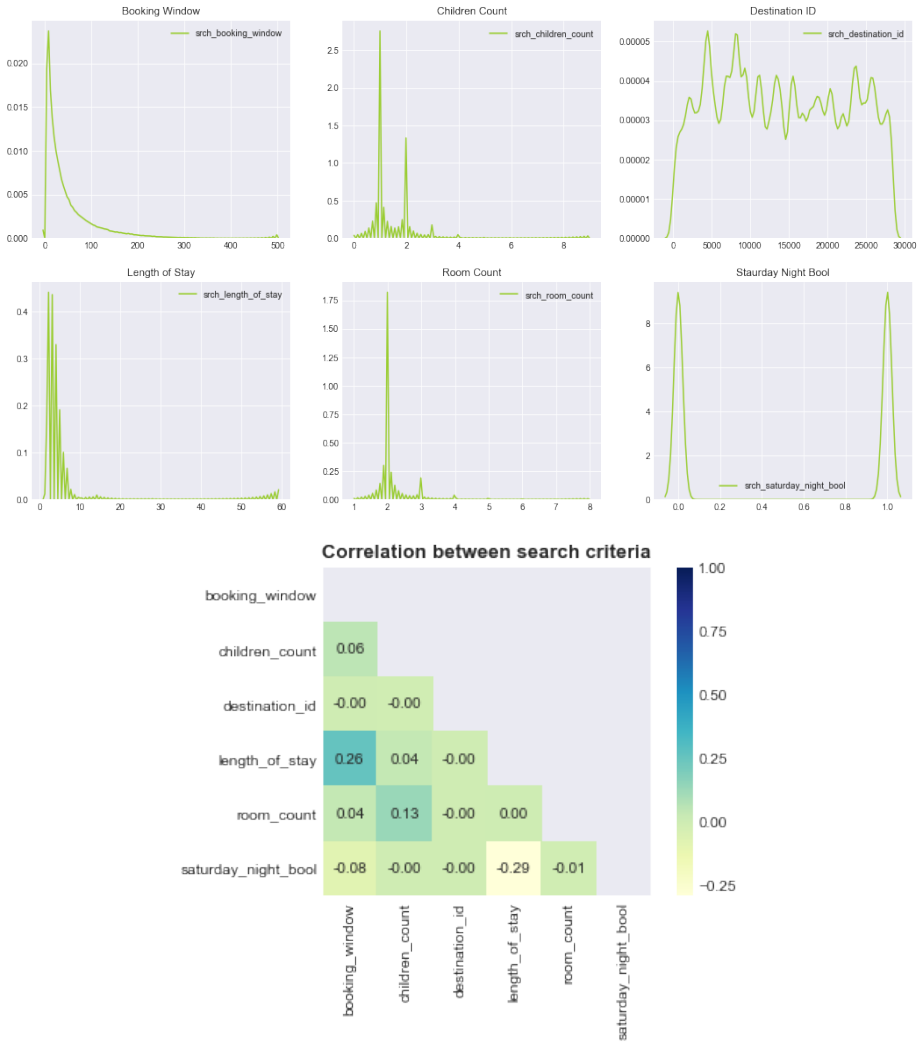

Fig. 2. Distribution of and Correlation between search criteria attributes.

Hotel Characteristics:

The distribution and correlation of hotel attributes give us some reasonable rules that (1) the score of location and review score are both correlated with the star rating of hotel. (2) The score of location 1 and score of location2 are

correlated, while the location 1 seems to have higher score than location 2. (3) The price of per night or whole nights is left skewed distributed and mainly located around 100,000.

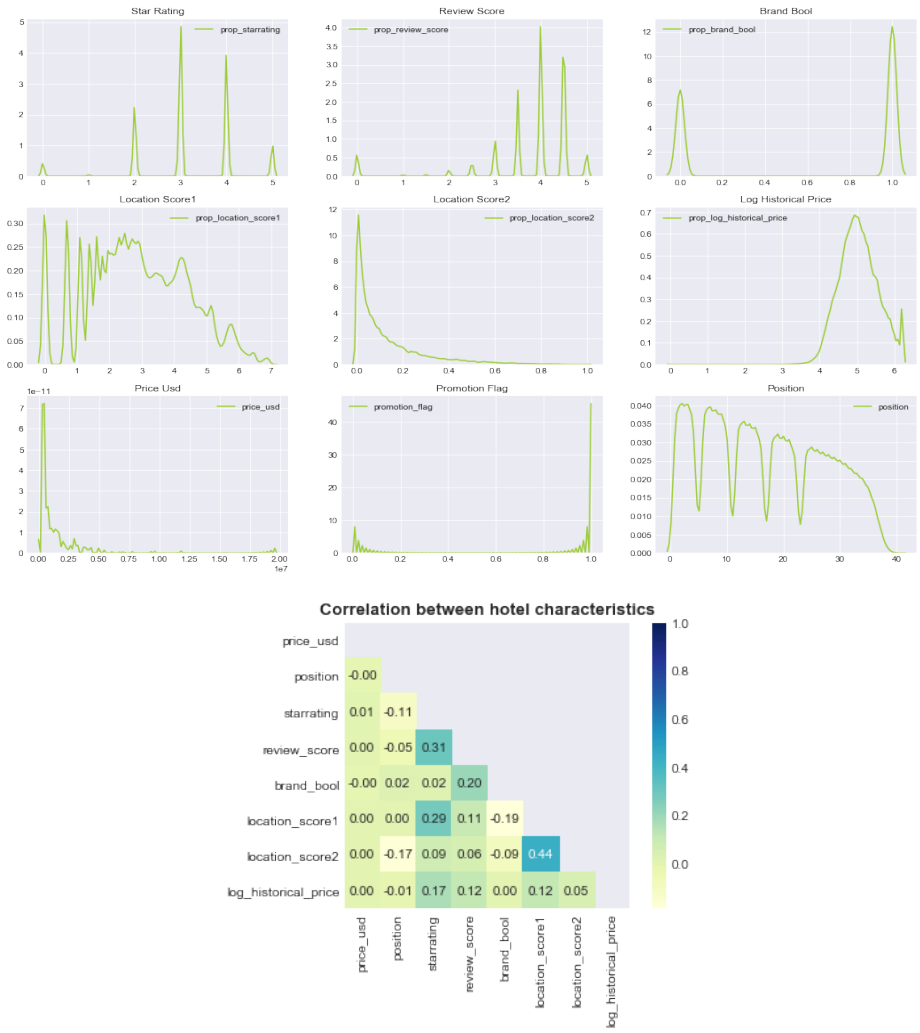

Fig. 3. Distribution of and Correlation between hotel characteristics attributes.

Customer Information: In this part, the preference of majority customers is studied by data analysis. (1) A large part of customers are from a same country, or this location seize a vast proportion of market. (2) The affordable price for visitors are left skewed distributed around 200 USD/per night. (3) Besides the price, 2, 3 and 4 stars rating are accepted by more people, 5 and 1 stars rating are not popular. (4) obviously, the historical mean price is correlated with the historical mean star rating of visitors.

Fig. 4. Distribution of and Correlation between customer information attributes.

For visitors, the range of price and the range of star rating of clicking hotels are higher than the those of finally booking hotels.

Competitor Information: The price and room availability of 8 different competitors are provided in this dataset. In the figure 3, For each competitor (in each subplot), we compare the booking rate with clicking rate of Expedia platform under three cases : when the price of Expedia > price of competitor; price of Expedia = price of competitor; price of Expedia < price of competitor.

There are some common rules in each subplot that when the price of hotel in Expedia is lower than that in competitors, the booking rate and clicking rate

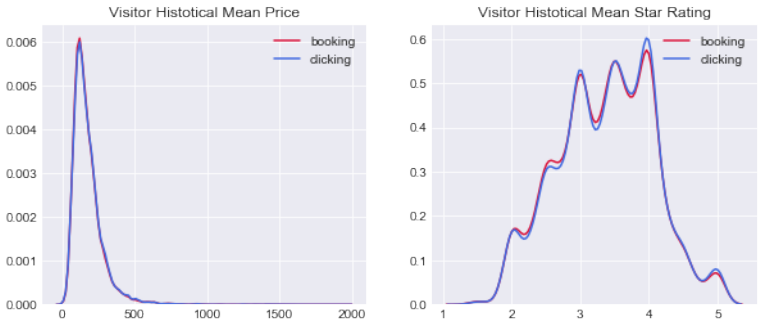

Fig. 5. Difference between clicking and booking hotel property in historical mean price and star rating.

are the highest; and in all cases clicking rate is than booking rate obviously. In addition, the competitor 8 seems to be most competitive.

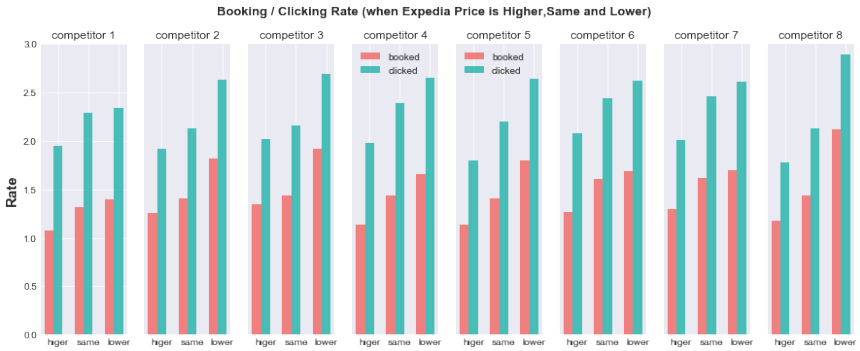

Fig. 6. Clicking rate and booking rate of Expedia when the its price compared with competitors.

Other Information: Random bool attributes is aimed to test the algorithm of training data. The X-axis is the displayed position of hotels in the result page. In the top picture in figure 3, although the clicking rate increases with the position, which means people are likely to clicking the hotels from top to bottom, the booking rate seems to have little connection with the position when random sorted. In contrast, when the positions are sorted with the recommend algorithm, the booking rate is higher for top positions, which is exactly our purpose to recommend best suitable hotels at high position to attract customers.

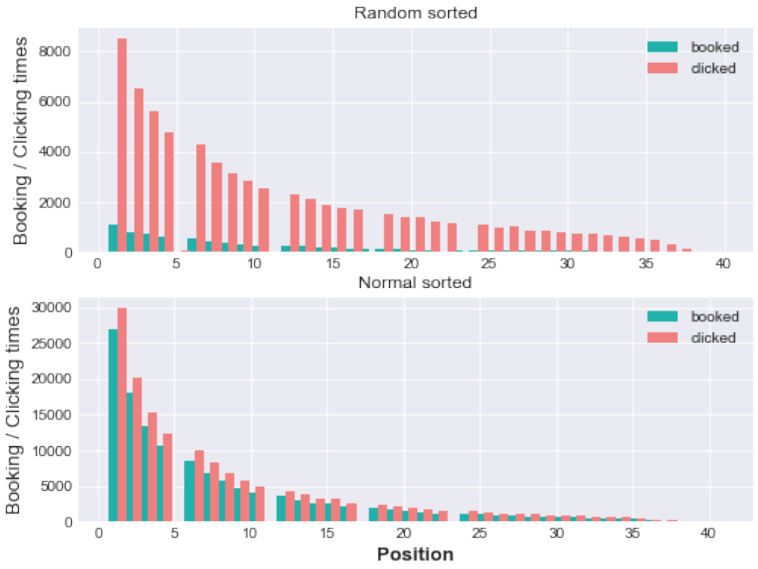

Fig. 7. The clicking rate and booking rate grouped by position for both randomly sorted order and normally sorted order hotels.

4 Data Preparation

4.1 Missing Value processing

As figure 8 shows, there are 31 columns have missing values. And we found the missing values can be categorized into *competitors data, hotel data, customer data.* And we directly delete *gross bookings used* attribute. Because it only appears in training data and has largest missing values.

Firstly, we found the largest part of missing value is from competitors' data. Firstly, we delete all the columns about *comp rate percent diff and comp inv.* Because there are large missing values in these two attributes and there is no useful and interesting finding based on these two attributes. Then We fill columns about *comp rate* all the missing values with 0 because as figure 8a (comp3 rate as an example) shows we find out there is no big difference of percentage of hotel being booked between 0 and NA.

Secondly, we deal with customers' missing values: *visitor hist starrating and visitor hist adr usd.* We both use median to fill the missing values. Because we find that the medians of the hotel being successfully or unsuccessfully booked for each attribute are quite close.

Finally, we deal with hotels' missing values. Firstly, we used several methods to fill the missing values of *prop location score2 and prop review score.* But we found out when we didn't fill the missing values the result was the best. The possible explanation is that this two attributes are vital for the next step of feature combination. Filling missing values will bring extra errors.

Furthermore, for *srch query affinity score* we fill it with the minimum value. For *orig destination distance*, we first log the values because the scale of value is too large. Then, we divide all the values into seven group(step is 1) and calculate

the book percentage for each group. As figure 8b shows, we can see for after 4 the percentage rate have small change. Thus, we fill the missing values with 75% of data which is 7 after log.

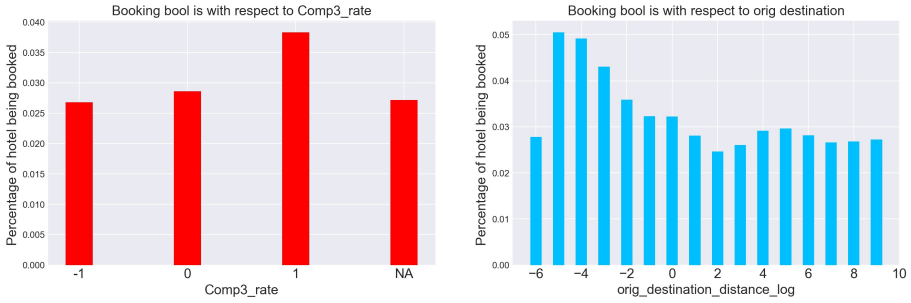

Fig. 8. (a): Successfully booking percentage is with respect to comp3 rate. We can see the percentage of 0 is closest to NA. (b): Successfully booking percentage is with respect to orig destination distance. We can see after 4 percentage become stable.

4.2 Feature normalization

In order to build a robust model for hotel ranking, the values in each features should be normalized. Because normalization can reduce the scale of data to increase divergence when training the machine learning model. The main idea in this processing is using *srch id and prop id* to normalize each feature. Firstly, for price usd attribute we need use log function to decrease the scale otherwise the range of data will be too large. Then, we use srch id or prop id to aggregate each mean and standard deviation of feature. Then, we use original data to minus mean value and divide by standard deviation. For example, we calculate corresponding mean and standard deviation of price usd in srch id 1. And using original data to minus mean value and divide by standard deviation to get the normalized data.

Table 1 shows the normalized features corresponding method. And we can see the normalized features are all about hotel information because these features are vital for hotel ranking and it is intuitive to determine the rank by hotel quality. And we actually did all five features for both srch id and prop id. But in the end most normalized feature of prop id based had been removed because of either the standard deviation is 0 or too many missing values(prop location score2 and prop review score).

Table 1. Feature normalization

Normalized method	Normalized Features	Normalized method	Normalized Features
srch id	price usd	srch id	prop starrating
prop id	price usd	srch id	prop location score2
srch id	prop location score1	srch id	prop review score

4.3 Feature extraction

In this part we extract feature from original dataset to combine or generate new feature for prediction. Firstly, we process the date time attribute. We extract features from data time to generate three new features *month, week, hour*. In this way we reduce the scale of data time which is effective for modelling. Then we extract new features from hotel description. The main idea is using *srch id, prop id and srch des id* to aggregate mean or median of hotel features. Because as we stated before, hotel description is the most important features. And then we use original hotel feature to minus the mean or median and then we get the new features. The whole extraction shows as table 3. At first, we did both mean and median method. At the end we drop all the median feature because the features are not ideal. The possible explanation is that the medians of some features are relatively bigger or smaller than mean, which will cause huge deviation from original data scale.

Table 2. Feature extraction

Aggregated method	Aggregated Features	Aggregated method	Aggregated Features
srch id	price usd	srch id	prop starrating
prop id	price usd	srch id	prop location score2
srch id	prop location score1	srch id	prop review score
srch id	promotion flag	srch destination id	price usd

4.4 Final step of data processing

In this part we process the attribute position, booking bool and clicking bool. Booking bool and clicking bool are the prediction attribute and we need to combine them into one attribute: successfully booked items are 5, only clicked items are 1 and else is 0. Furthermore, we deal with position attribute. Position attribute only occurs in training set and we need to find a method to incorporate it by exist attributes in both training and testing dataset. Again we use method like normalization and extraction step to process position. Both srch id and prop id are used to aggregate the mean of position. And we save this new feature and use corresponding srch id and prop id to macth the item in test dataset. Then, the new feature can be used in test dataset.

Finally, all the features are ready and there are total 52 features in model.

5 Model information

As we discussed before, basically all the wining algorithms use GBM or lambdaMART model. We first tried lambdaMART because it is one of the most effective model for ranking. But after building the model we found it is extremely time-consuming. Because when using lambdaMART model train and test dataset are needed to transform into corresponding format. And it cost 8 hours to fit the model, which is extremely inefficient. Thus, we decided to use GBM model - LightGBM, which is fast, efficient, distributed, accurate and effective model for ranking(LightGBM-lambdarank). The reason of efficiency is LightGBM is

based on the decision tree algorithm. It uses the optimal leaf-wise strategy to split the leaf nodes. However, other lifting algorithms generally use depth-wise or level-wise instead of leaf-wise (show as figure 9). Therefore, in the LightGBM algorithm, when growing to the same leaf node, the leaf-wise algorithm reduces more loss than the level-wise algorithm. This results in higher accuracy, which cannot be achieved by any other existing lifting algorithms. At the same time, its speed is also shocking.[2]

Furthermore, we need to determine the parameter of LightGBM-lambdarank. n estimator is number of base learners. It directly determines efficiency and accuracy of model and we choose 1500 to ensure comparatively high accuracy and efficiency, which usually took about 20-30 minutes to fit the model. Learning rate determines whether the objective function can converge to the local minimum and when it converges to the minimum.[3]Thus, we choose 0.12, which is comparatively larger than default value. Next, for the metric Normalized discounted cumulative gain(ndcg) it is evaluation method for ranking. And max position determine the max position of ndcg. Additionally, for *dart* boosting this method can drop trees while fitting in order to solve the over-fitting[1]. The final parameter is categorical feature. Some feature in dataset can be described as categorical feature such as *month, site id, prop country id and visitor location country id.*[5]

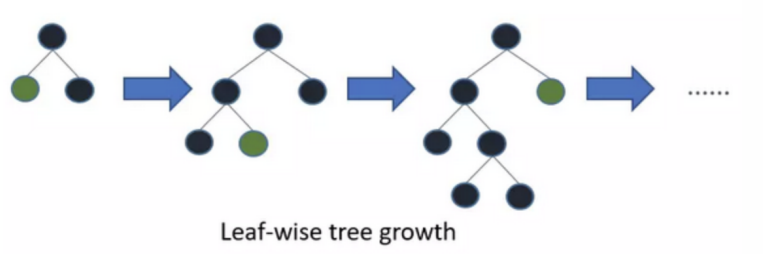

Fig. 9. Theory of leaf-wise algorithm of LightGBM

Table 3. Parameters of LightGBM-lambdarank

Parameter	Value	Function
n estimators	1500	Number of base learners
learning rate	0.12	Boosting learning rate
metric	ndcg	Evaluation method
max position	5	Optimizes NDCG at this position
boosting	dart	Boosting strategy
categorical feature	4	Specify the feature category for model training

6 Fitting, prediction and result

After setting all the parameters we fitted the model. We spilt the training dataset into train and validation dataset(100000 items). Then we fit LightGBM-

lambdarank model. The result of ndcg shows best score are training's ndcg: 0.472645 and validation ndcg: 0.421116. Then we use same data processing for test dataset and make prediction. The final result shows the accuracy is 0.40119 and ranks 12 out of 232(till 4pm 22/05).

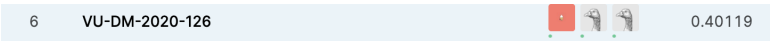

Fig. 10. Final result

7 Gain

We gained various knowledge and skills from this project. This is the first time we process large dataset like this task. And in the part of data understanding, we trained our skills in doing EDA especially combine relevant features in the same plots. And we also gain a sea of knowledge on how to process features. For example, the way to fill a certain missing value based on its own distribution or scale, the way to explore inner relation of features, the way to combine and extract new feature based on inner relation of features. And we also learned a lot from fitting model. The choice of model, selection of parameters, and validation size have vital influence on final model and prediction. During this four weeks difficulties were unavoidable. The main difficulty is that this dataset is super large and it will cost lots of time on running code. And also for selecting attributes and digging useful information, giving a clearer and right understanding of the dataset. Another difficulty is model part. Finding a perfect machine learning model and tuning parameter cost lots of time. And we feel frustrated when we tried so hard to manipulate features and fit model but the results were disappointing. And we feel excited and happy when we had huge improvement on Kaggle result. In short, this is long and though jounary but it's worth.

8 Conclusion

In this report, we did the data mining project Expedia hotel recommendation, including business understanding, data understanding, data preparation, modelling and evaluation. What we done for this project has concluded in the above contents of the report. In total, after doing data processing, we applied LightGBM-lambdarank model to fit data and got the best prediction score of Kaggle: 0.40119.

References

1. dart. https://www.cnblogs.com/wzdLY/p/9867719.html
2. lightgbmshort. https://www.aboutyun.com/thread-24339-1-1.html
3. Minimum. https://www.cnblogs.com/lliuye/p/9471231.html
4. Personalize expedia hotel searches - icdm 2013. https://www.kaggle.com/c/expedia-personalized-sort/overview
5. Result. https://blog.csdn.net/qq$_2$4852439/*article*/*details*/88693144