Abstract:

In this project, we used the customer's searching behavior data provided by Expedia to build a classifier for predicting a particular customer's preference on a specific hotel cluster.

The goal of this project is to build a good classifier for the given dataset. However, getting good performance is not our sole purpose. Interpreting data and justifying our decision choice for the problem is also an important part of our goal.

After explored the datasets, we first did a simple comparison among different models and selected random forest for further development. Then we tested different random forest model configurations on downsized training data. Finally, we trained the random forest model on the entire training set and generated additional predictions (five predictions) for each testing record.

After all the operations, we upload our predicted result on Kaggle, and get the evaluation score from it. We obtained MAP@5 Score evaluation score of 0.16924, which improved a lot compared to the beginning.

1. Introduction

1.1. Problem Statement

In this project, we are going to use the customer's searching behavior data provided by Expedia to build a classifier for predicting a particular customer's preference on a specific hotel cluster.

Improving the recommendation system is of great importance to Expedia because that with a better recommendation in place, the company can better serve its customer by recommending places that cater to each customer's preference and need. And this improvement may also lead to revenue growth of the company.

To tackle this problem, we selected a representative sample from the whole training set, trying various algorithms on the sample set, with basic adjustment of parameters in each algorithm. The models we constructed in this step are KNN, Neural Network, Naive Bayes, Random Forest, Decision Tree, and SVM. We then compared the result from different models, and select the best performing model for the purpose of further development. After comparison, we selected Random Forest to be our prospected model, and we then focused on detailed tweaking of model parameters to improve performance, and spent effort on interpreting the result.

1.2. Literature Review

Recommender system with data mining techniques has been applied to various situations. Generally speaking, the main goal of a recommender system is to provide personalized recommendation to users[1]. For example, recommender systems have been used in financial market to assist in stock portfolio selection [2]. Another typical example is the utilization of data mining techniques in target marketing, that recommend to customers the likely commodities that they may willing to buy [3].

There are different types of recommender systems based on the methods used to build the systems. Some of the popular techniques include association rules, decision trees, artificial neural networks, fuzzy set techniques etc[4]. Still, the topic of Improving the performance of recommender system is an open research area.

1.3. Report Outline

In section 2, we describe the Expedia dataset, and discussed the exploratory data analysis and data preprocessing approaches we choose.

In section 3, we provide details of our methodology, and described in depth our decision choice in the model refinement stage.

In section 4, we discuss our method of evaluation, and provide the performance result of our refined model.

In section 5, we referenced the articles and web page that we referred to during this project.
In the last section, we include the individual sections for the team members.

2. Dataset

2.1. Description

The training set contains 37,670,293 objects, and each of them has 24 attributes (see the attached Attributes Details List). It's the log of customer behavior collected by Expedia in 2013 and 2014.

The testing set contains 2,528,243 objects, and each of them has 22 attributes (cnt, is_booking, and hotel_cluster are excluded). It's the log of booked customer behavior (where the training set also contains clicked but unbooked) collected by Expedia in 2015.

The dataset contains rich information about the customer's past behaviors and other associated information that we believe to have predictive power on its behavior patterns.

Attributes Details List (training set)

Attribute	Type	Level	Description
date_time	string	Interval	Timestamp
site_name	int	Nominal	ID of the Expedia point of sale (i.e. Expedia.com, Expedia.co.uk, Expedia.co.jp, ...)
posa_continent	int	Nominal	ID of continent associated with site_name
user_location_country	int	Nominal	The ID of the country the customer is located
user_location_region	int	Nominal	The ID of the region the customer is located
user_location_city	int	Nominal	The ID of the city the customer is located
orig_destination_distance	doubl	Ratio	Physical distance between a hotel

	e		and a customer at the time of search. A null means the distance could not be calculated
user_id	int	Nominal	ID of user
is_mobile	bool (int)	Nominal	1 when a user connected from a mobile device, 0 otherwise
is_package	bool (int)	Nominal	1 if the click/booking was generated as a part of a package (i.e. combined with a flight), 0 otherwise
channel	int	Nominal	ID of a marketing channel
srch_ci	string	Interval	Check-in date
srch_co	string	Interval	Check-out date
srch_adults_cnt	int	Ratio	The number of adults specified in the hotel room
srch_children_cnt	int	Ratio	The number of (extra occupancy) children specified in the hotel room
srch_rm_cnt	int	Ratio	The number of hotel rooms specified in the search
srch_destination_id	int	Nominal	ID of the destination where the hotel search was performed
srch_destination_type_id	int	Nominal	Type of destination
hotel_continent	int	Nominal	Hotel continent
hotel_country	int	Nominal	Hotel country
hotel_market	int	Nominal	Hotel market
is_booking	bool (int)	Nominal	1 if a booking, 0 if a click
cnt	int	Ratio	Number of similar events in the context of the same user session
hotel_cluster	int	Nominal	ID of a hotel cluster

For the testset, we do not have attributes “is_booking”, “cnt” and “hotel_cluster”. And other attributes are the same as the training set.

2.2. Preprocessing

For our purpose of data analysis, we removed the attributes “user_id”, “date_time”, “srch_ci”, “srch_co”, because from our knowledge we believed that such attribute don’t have much predictive power on predicting our target variable, the hotel_cluster. We also removed “is_booking”, “cnt” when building the models, because such attribute are missing in the test set, and for the models we are building, we need to train the model with the same attributes in the test set.

We didn’t normalized the dataset at the beginning for few reasons: Most of the attributes are categorical. Also,In building models, we have some specific data preprocessing steps for each model according to the characteristic of the model itself.

2.3. Exploratory data analysis

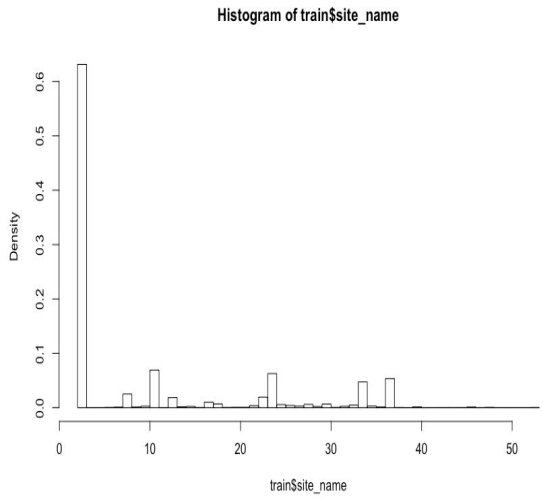
First, we perform an exploratory data analysis on the training set.

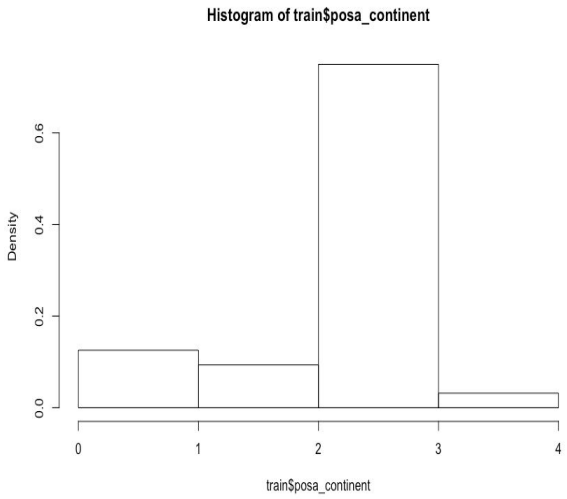
```
             date_time          site_name      posa_continent user_location_country user_location_region user_location_city
 2014-10-17 21:45:34:      15   Min.   : 2.000   Min.   :0.00   Min.   :  0.00       Min.   :   0.0       Min.   :    0
 2014-11-06 17:34:48:      15   1st Qu.: 2.000   1st Qu.:3.00   1st Qu.: 66.00       1st Qu.: 174.0       1st Qu.:13009
 2014-12-18 07:55:29:      12   Median : 2.000   Median :3.00   Median : 66.00       Median : 314.0       Median :27655
 2014-06-19 19:35:22:      11   Mean   : 9.795   Mean   :2.68   Mean   : 86.11       Mean   : 308.4       Mean   :27753
 2014-07-16 11:19:34:      11   3rd Qu.:14.000   3rd Qu.:3.00   3rd Qu.: 70.00       3rd Qu.: 385.0       3rd Qu.:42413
 2014-11-18 19:25:11:      11   Max.   :53.000   Max.   :4.00   Max.   :239.00       Max.   :1027.0       Max.   :56508
 (Other)            :37670218

orig_destination_distance    user_id          is_mobile        is_package        channel            srch_ci
Min.   :    0             Min.   :      0   Min.   :0.0000   Min.   :0.0000   Min.   : 0.000   2014-12-26:  223153
1st Qu.:  313             1st Qu.: 298910   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 2.000   2014-12-27:  200654
Median : 1140             Median : 603914   Median :0.0000   Median :0.0000   Median : 9.000   2014-12-31:  175212
Mean   : 1970             Mean   : 604452   Mean   :0.1349   Mean   :0.2489   Mean   : 5.871   2014-12-30:  170182
3rd Qu.: 2553             3rd Qu.: 910168   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.: 9.000   2014-12-28:  165339
Max.   :12408             Max.   :1198785   Max.   :1.0000   Max.   :1.0000   Max.   :10.000   2014-12-25:  165089
NA's   :13525001                                                                               (Other)   :36570664

     srch_co        srch_adults_cnt srch_children_cnt  srch_rm_cnt    srch_destination_id srch_destination_type_id  is_booking
 2015-01-02:  218447  Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :    0      Min.   :0.000          Min.   :0.00000
 2015-01-01:  213499  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:1.000   1st Qu.: 8267      1st Qu.:1.000          1st Qu.:0.00000
 2014-12-28:  195159  Median :2.000   Median :0.0000   Median :1.000   Median : 9147      Median :1.000          Median :0.00000
 2015-01-03:  189827  Mean   :2.024   Mean   :0.3321   Mean   :1.113   Mean   :14441      Mean   :2.582          Mean   :0.07966
 2014-12-30:  172938  3rd Qu.:2.000   3rd Qu.:0.0000   3rd Qu.:1.000   3rd Qu.:18790      3rd Qu.:5.000          3rd Qu.:0.00000
 2014-11-30:  167439  Max.   :9.000   Max.   :9.0000   Max.   :8.000   Max.   :65107      Max.   :9.000          Max.   :1.00000
 (Other)   :36512984
```

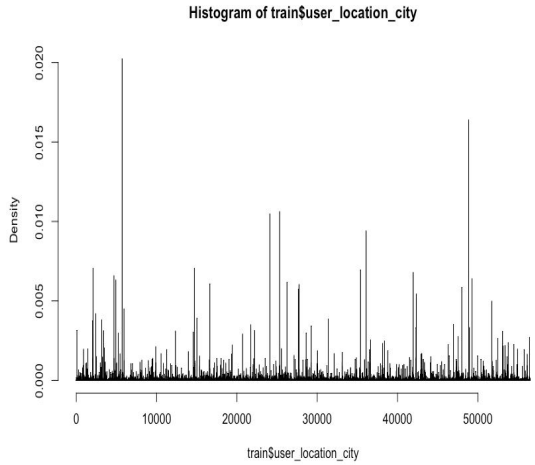
cnt	hotel_continent	hotel_country	hotel_market	hotel_cluster
Min. : 1.000	Min. :0.000	Min. : 0.0	Min. : 0.0	Min. : 0.00
1st Qu.: 1.000	1st Qu.:2.000	1st Qu.: 50.0	1st Qu.: 160.0	1st Qu.:25.00
Median : 1.000	Median :2.000	Median : 50.0	Median : 593.0	Median :49.00
Mean : 1.483	Mean :3.156	Mean : 81.3	Mean : 600.5	Mean :49.81
3rd Qu.: 2.000	3rd Qu.:4.000	3rd Qu.:106.0	3rd Qu.: 701.0	3rd Qu.:73.00
Max. :269.000	Max. :6.000	Max. :212.0	Max. :2117.0	Max. :99.00

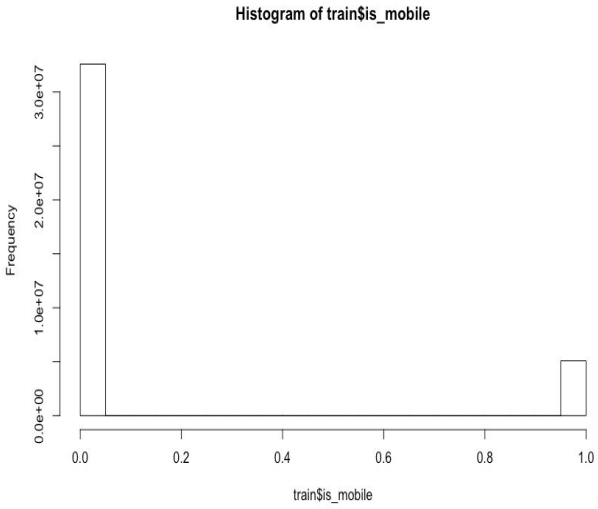
Distribution of single variables:

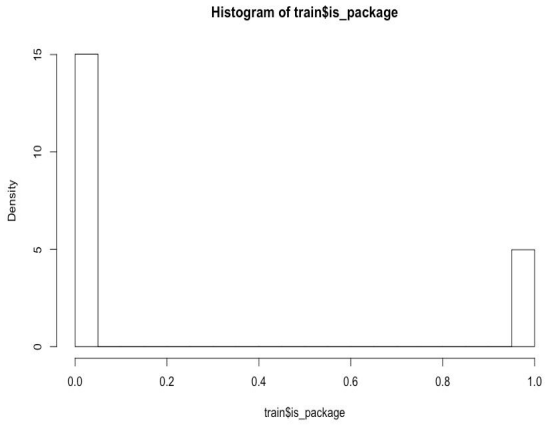
For the nominal variable, we draw the histogram to see the distribution of each variable. For ratio variable, we use boxplot to visually see the distribution and detect possible outliers.

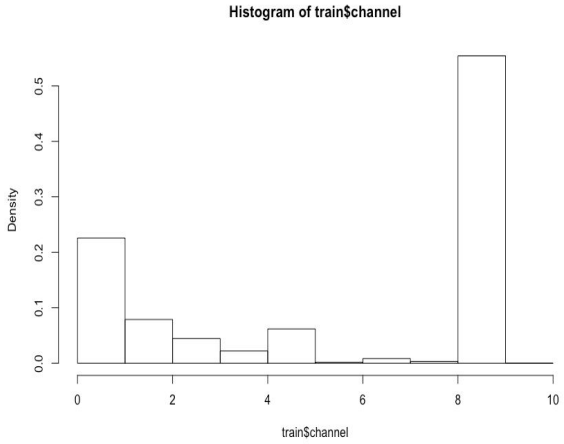


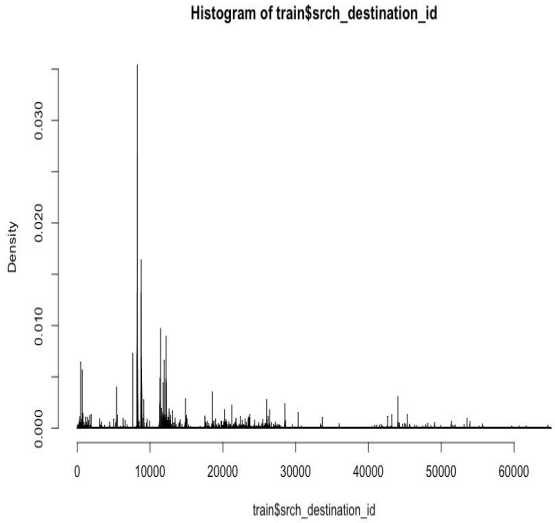


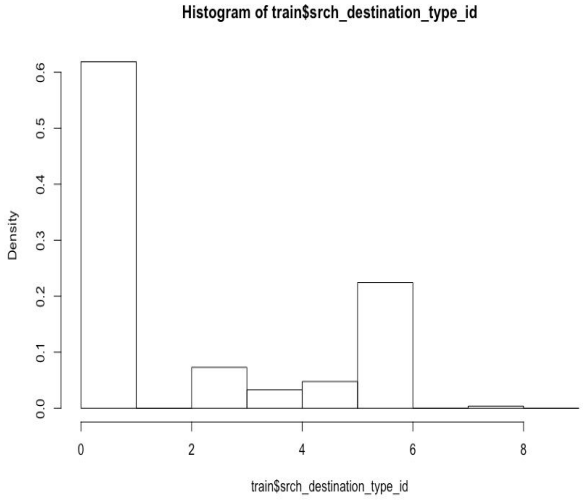


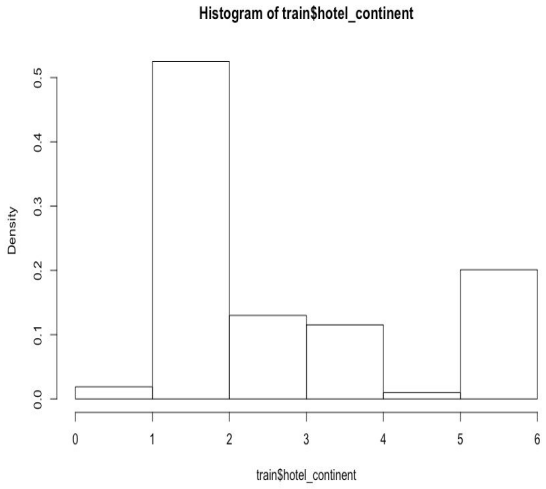


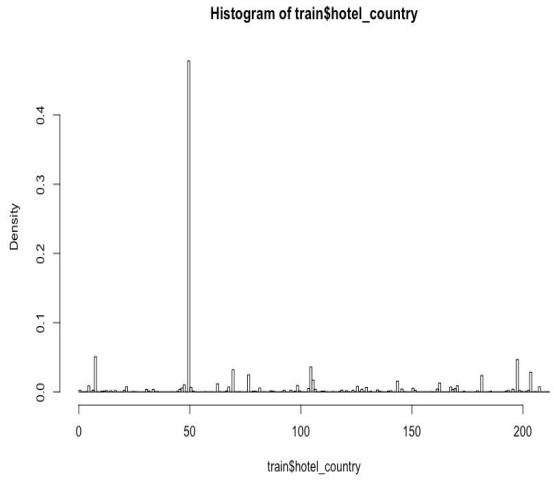

Histogram of train$is_package
Density
train$is_package

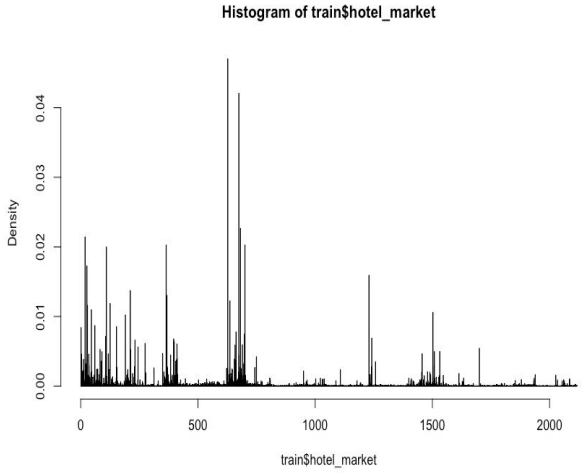

Histogram of train$channel
Density
train$channel

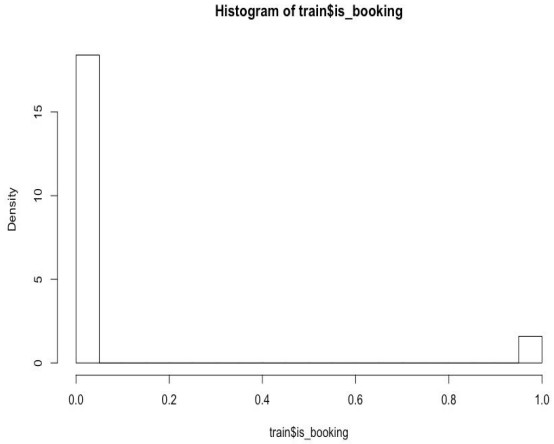

Histogram of train$srch_destination_id
Density
train$srch_destination_id

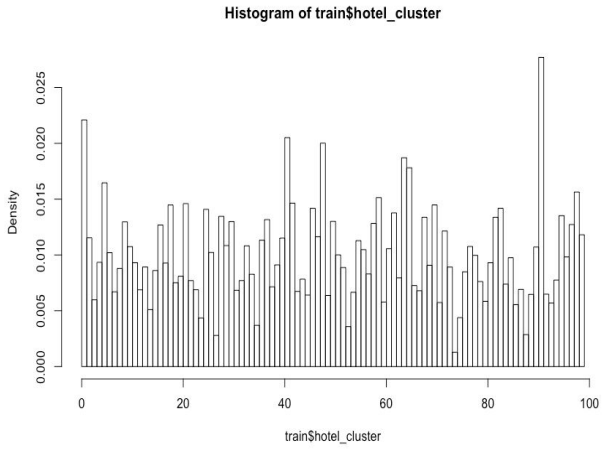

Histogram of train$srch_destination_type_id
Density
train$srch_destination_type_id

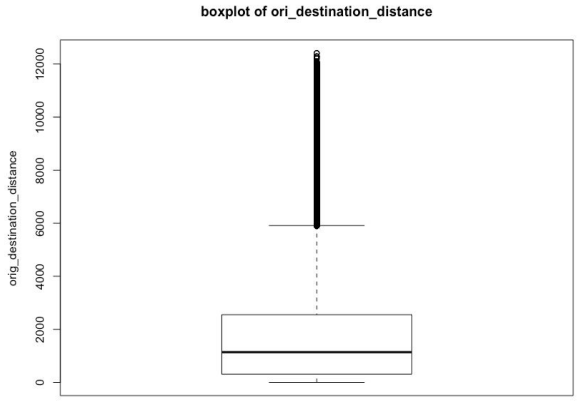

Histogram of train$hotel_continent
Density
train$hotel_continent

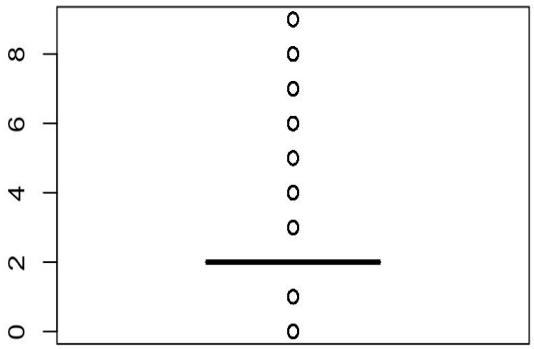

Histogram of train$hotel_country
Density
train$hotel_country

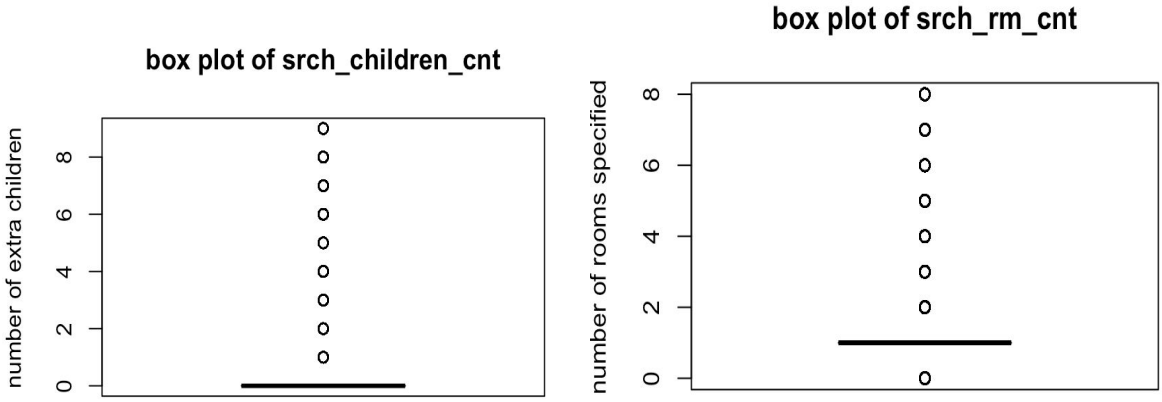

Histogram of train$hotel_market
Density
0.00
0.01
0.02
0.03
0.04
0
500
1000
1500
2000
train$hotel_market


Histogram of train$is_booking
Density
0
5
10
15
0.0
0.2
0.4
0.6
0.8
1.0
train$is_booking


Histogram of train$hotel_cluster
Density
0.000
0.005
0.010
0.015
0.020
0.025
0
20
40
60
80
100
train$hotel_cluster


boxplot of ori_destination_distance
orig_destination_distance
0
2000
4000
6000
8000
10000
12000


0
2
4
6
8



For the possible outliers indicated by the box plot, we decided to include it in the dataset, since these “extreme”data point may indicate certain type of customer behavior that has a strong predictive power on predicting which hotel_cluster a certain type of customer most likely to book.

Checking missing value:

date_time	site_name	posa_continent	user_location_country	user_location_region
0	0	0	0	0
user_location_city	orig_destination_distance	user_id	is_mobile	is_package
0	13525001	0	0	0
channel	srch_ci	srch_co	srch_adults_cnt	srch_children_cnt
0	0	0	0	0
srch_rm_cnt	srch_destination_id	srch_destination_type_id	is_booking	cnt
0	0	0	0	0
hotel_continent	hotel_country	hotel_market	hotel_cluster	
0	0	0	0	

We see that the attribute “orig_destination_distance” has 13525001 missing values, which approximately ⅓ are missing. So we decided to remove this attribute in our models.

Exploratory data analysis of the sample set:

Since we are building models from the sample set, we need to as possible as we can to ensure that our selected sample are a good representation of the whole training set, otherwise, the sample set may not truly capture the characteristics our training set.

To do so, we also perform an exploratory data analysis, to check visually that if the distribution of any attribute of our downsized sample is significantly different from the original set. After the comparison, we confirmed that the EDA give consistent results on the downsized sample and the whole training set.

3. Methodology

3.1. Techniques

3.1.1. Model Comparison

Each model has its own virtues, and may be suitable for different tasks and different datasets. In order to get some insight on which algorithm may work the best in our project, we used a 0.01% downsized dataset (for model selection), and constructed 6 models in R: KNN, Neural Network, Naive Bayes, Random Forest, Decision Tree, and SVM, with preprocessing of data for each model and some adjustment of model parameters.

We used 5-fold cross-validation to evaluate the performance of the models. The main metric we used is misclassification error, while also taking into account to computation cost.

From the result, by mainly considering misclassification error rate, we selected RandomForest to be our model for further development and analysis, since it gives the lowest misclassification error rate among the models.

3.1.2. Dataset Downsizing

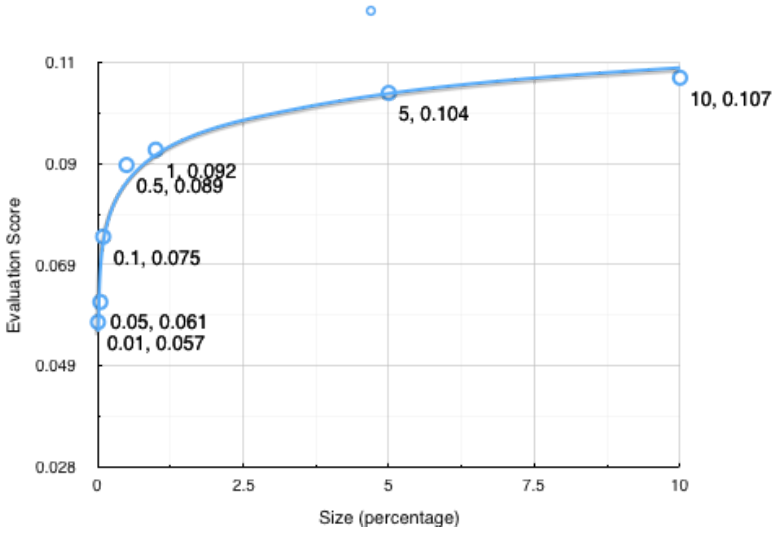
After selecting random forest as the model to accomplish the task, the next step should be adjusting the parameters of the random forest model. However, build model on the entire training set (4GB, and 37 million records) usually takes a 4-core/8-thread CPU several hours to process. Accordingly, in order to balance the costs (time cost and computational cost) and precision of the parameter adjustment process, we decide to downsize our training set to a reasonable size.

First of all, we write a python program (downSize.py) that uses random permutation to downsize the training set. Then, under the same configuration, we build our model by using different size of training sets. The test results (learning curve) are shown below / in next page.

Configuration: Default random forest classifier

Evaluation: 5-folds cross validation

Size	Percentage (%)	Evaluation Score
3767	0.01	0.057343
18835	0.05	0.061428
37670	0.1	0.074701
188351	0.5	0.089248
376702	1	0.092349
1883514	5	0.103869
3767029	10	0.106951



Based on the results, we decide to use the 1-percent (376,702 records) downsized training data for the adjustment of model parameters.

3.1.3. Adjustment of Parameters

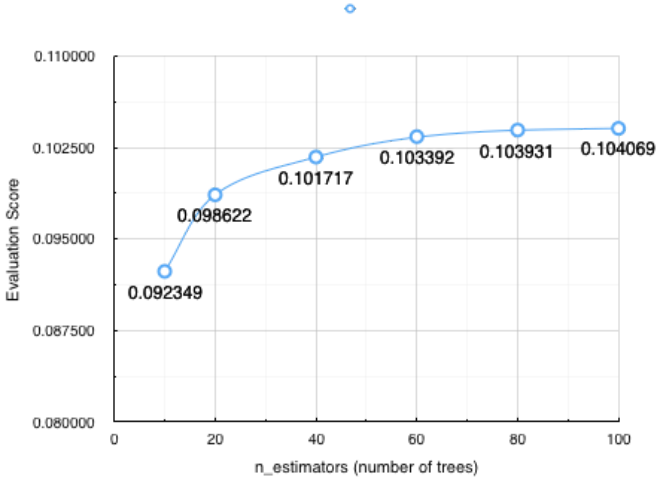
3.1.3.1. n_estimators

n_estimators represents the number of trees in the forest.

Configuration: Default random forest classifier (except changing n_estimators)

Evaluation: 5-folds cross validation

n_estimators	Evaluation Score
10	0.092349
20	0.098622
40	0.101717
60	0.103392
80	0.103931
100	0.104069



Generally, more trees yield better result, but the improvement decreases as the number of trees increase. According to our testing results, we decide to set the number of trees as 50.

3.1.3.2. criterion

Criterion determines the quality of a split. The sklearn random forest library supports Gini and Entropy as the measure function. Our tests indicate that Gini (evaluation score: ~0.103) is slightly better than Entropy (evaluation score: ~0.101), so Gini is chosen as the criterion.

3.1.3.3. max_depth and min_samples_split

In sklearn, the default value of max_depth is None and the default value of min_samples_split is 2, which means the trees would be expanded as much as possible. Fully expanded trees give better estimations, so we decide to leave these two parameters as default.

3.1.4. More Predictions

The final evaluation method of this project (MAP@5, specified in section 4.1) allows up to 5 predictions. Hence, we tried to generate more reasonable predictions for each single record. When using single prediction for each record, our MAP@5 score is 0.09123.

Sklearn random forest library provides a method called predict_proba which can predict the probability of each record's class belonging. We sort the probability map and select the five most likely classes of each record as the predictions. This process increased the MAP@5 score to 0.16924.

3.1.5 Using is_booking?

The training set contains both clicking records and booking records while the testing set only comes from the booking records (This is reasonable since we want to recommend user hotel that they would like to book). Based on this fact we think that booking records in the training set may be better for generating the predictions. However, our experiments indicate that simply use the booking results make little difference.

3.2. Software

3.2.1. R Studio

Rstudio is used for Exploratory data analysis and constructing 6 classification models including Naive Bayes, SVM, Decision Tree, Random Forest, Neural Network and KNN.

Library Used: e1071, nnet, rminer, caret, caret, RCurl, mlbench, randomForest, Metrics, party

3.2.2. Python

Python 2.7: Python 2.7 is used for dataset manipulations (such as downsizing, splitting, etc.)

Python 3.5: Python 3.5 is used for fit model and make predictions.

Library Used: sklearn (random forest and cross validation), pandas(read & write data), numpy(math).

4. Evaluation

4.1. Strategy

The results will be evaluated by Mean Average Precision @ 5 (as provided by the kaggle). The evaluation formula is given below:

$$MAP@5 = \frac{1}{|U|}\sum_{u=1}^{|U|}\sum_{k=1}^{min(5,n)} P(k)$$

where: - |U|: the number of user events
- P(k): the precision at cutoff k
- n: the number of predicted hotel clusters

For each testing record, it is allowed to give *up to* five predictions.

Other evaluations for developing the model are discussed in the techniques section.

4.2. Results

The performance of our random forest model at different states are shown below. Most improvements are made during the parameter adjustment stage and more predictions generation stage. We include the predictions generated at each stage in the folder named "predictions".

Stage (when finishing)	MAP@5 Score
Model Comparison	0.04552
Parameter Adjustment	0.09123
More Predictions	0.15438
Training on entire dataset	0.16924

We include the predictions generated at each stage in the folder named "predictions". You can submit it directly to kaggle for MAP@5 evaluation.

@ https://www.kaggle.com/c/expedia-hotel-recommendations/submissions/attach

Our model also indicates that the following features are informative for predicting the hotel_cluster.

Rank	Feature
1	hotel_market
2	hotel_country
3	hotel_continent
4	srch_destination_type_id
5	srch_destination_id
6	channel
7	is_package
8	user_location_city
9	user_location_region
10	user_location_country
11	posa_continent
12	site_name
13	srch_children_cnt
14	is_mobile
15	srch_rm_cnt
16	srch_adults_cnt

4.3. Discussion

The model development process works well and smooth. Consistent improvements has been made. However, the relationship of user related data versus hotel cluster and hotel related data versus hotel cluster are not well explored respectively. The following improvements can be made in the future:

- Enhance the method for generating multiple predictions (Such as class voting by different model)
- Explore more combinations of features as training set.
- Using date_time of the training set's record to match the range of the testing set's record.

5. Bibliography

1. Moran Beladev, Lior Rokach, and Bracha Shapira. 2016. Recommender systems for product bundling. Knowledge-Based Systems 111 (2016), 193–206.

2. Preeti Paranjape-Voditel and Umesh Deshpande. 2011. An Association Rule Mining Based Stock Market Recommender System. 2011 Second International Conference on Emerging Applications of Information Technology (2011).

3. Lee, S.-L. (2010). Commodity recommendations of retail business based on decisiontree induction. *Expert Systems with Applications*, *37*(5), 3685–3694.

Dataset: https://www.kaggle.com/c/expedia-hotel-recommendations

Webpage reference:

1. Anon. 3.2.4.3.1. sklearn.ensemble.RandomForestClassifier. Retrieved December 15, 2016 from http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.ht
2. 3.1. Cross-validation: Evaluating estimator performance — scikit-learn 0.18.1 documentation. (2010). Retrieved December 15, 2016, from http://scikit-learn.org/stable/modules/cross_validation.html#k-fold
3. Zhao, Y. (2016). R and data mining: Examples and case studies - RDataMining.Com: R and data mining. Retrieved December 15, 2016, from http://www.rdatamining.com/docs/r-and-data-mining-examples-and-case-studies
4. quant signals. (2012, September 25). Learning kernels SVM. Retrieved December 15, 2016, from https://www.r-bloggers.com/learning-kernels-svm/
5. Team, T. D. Machine learning in R for beginners. Retrieved December 15, 2016, from https://www.datacamp.com/community/tutorials/machine-learning-in-r#gs.hZyiVCY

Individual Sections - Zhuoer Wang

1. Contribution

I'm responsible for the further development of the selected model and evaluation parts of this project. After Zhe Huang has finished the initial model selection (random forest selected), I made the further model development. First of all, I tested and evaluated the model's performance on different size of training set based on the same configuration. Based on the learning curve, an appropriately downsized training set is selected for further testing. Then, I tested the model by using different parameter settings and evaluated the associated results. What I spent a lot of effort on was to interpret the result, and making sense of the result. We believe attaching meaning to the data is more important than merely getting a high performance. Next, I modified our program for the support of predicting more classes.

As for evaluation, I maintained the 5-folds cross validation for the results evaluation during the model development (using training set) phase. The predicting results are evaluated by the MAP@5 evaluation provided by Kaggle.

I wrote all the python code and mainly contributed to the Methodology and Evaluation Section of this report. Also, I assisted Zhe Huang during the entire data exploration and preprocessing phase.

2. Setbacks

The largest setback is to handle the big dataset. Our dataset is 4GB in size and has 37 millions of records with 24 attributes. Due to the size of the dataset, it is critical to well consider the time efficiency and space efficiency of the code. Parallel processing is used for our random forest model for speeding up the computation. However, parallel processing can easily lead to memory leaks when growing the trees and using the model for class prediction. Accordingly, I have to spend additional time to split the workload of CPU and memory properly.

Also, in consideration of the limited time we have for developing our model, we need to select a reasonable downsized training data to test different configurations. One setback we encountered during this phase is that we found the accuracy of the model decreased as we increase the training set's size. After spending a lot of time determining the reason as well as try python's libraries, we realized that our error rate calculation in R is wrong.

3. Lessons Learned

Through the project,

I got a better understanding of

- the Random Forest Classifier
- feature selection
- using categorical features for prediction

I learned

- how to handle the analysis of large dataset
- different ways of giving multiple predictions
- parallel computing
- basics of Spark and MapReduce

Individual Sections - Zhe Huang

1. Contribution

I perform the exploratory data analysis on the whole training set, checked the missing value, draw the histogram of nominal variables and boxplot for the ratio variable. After exploring the data, we discussed how to handle missing values, and possible outliers indicated by the boxplot, and how our operations related to our following steps, and I preprocess the data for latter model constructions.

I select a 0.01% downsized sample to use when doing model selection. Since we are building models from the sample set, we need to as possible as we can to ensure that our selected sample are a good representation of the whole training set, otherwise, the sample set may not truly capture the characteristics our training set. To do so, I also perform an exploratory data analysis, to check visually that if the distribution of any attribute of our downsized sample is significantly different from the original set. After the comparison, I confirmed that the EDA give consistent results on the downsized sample and the whole training set.

After that, I constructed 6 models in R: KNN, Neural Network, Naive Bayes, Random Forest, Decision Tree, and SVM. Adjusting the input variable according to the specific requirement of each model, and tune the parameters based on accuracy result. There are many decision choices on how to build the models. For some decision choices, I can leverage the internal package in R to help me, for example using “tune” function to test some parameters, and select the best performing ones (basic adjustments,deep exploration are left till the next stage). For the other, I have to make the discretion choice in a case-by-case manner. For example, many of the attributes are displayed as “int” but actually a categorical variable, for the Naive Bayes model, I can either input them as numeric or factor, which will lead to very different models, and the result may also be different a lot. Similarly, for Neural network model, it can be used either as regression or classification, and the input attributes can be either numerical or categorical.

To select the models that may suit our dataset the best, I perform 5-fold cross validation on the the 6 models, and using misclassification errors as metric. By doing this, we selected RandomForest as our target models for further development.

2. Setbacks

The most setbacks I encounter are during model building in R. For each model, there is always something to be careful, some trivial mistakes like inputting a wrong type of variable into the model may lead to strange errors, and takes a long time to debug. For example, the “hotel_cluster” in the dataset are giving as int, but in R, actually I need to transfer it into factor in order for the models to work properly.

Beside technical problems when writing R code,how to make reasonable decision choices were a big challenge I faced. Since I am building 6 models in R, some of which we didn’t deeply discuss in the lecture, I have to self-learned the materials, develop a fair understanding of how the algorithms work, so that I can make reasonable decision choices, like setting different hyperplane in SVM, setting a reasonable, whether to normalized the categorical attributes for KNN etc.

3. Lessons Learned

As a result of this project, I developed deeper understanding the classification algorithms, how each algorithm work, what each algorithm suit for, what are some of the cost and benefits etc. I also implemented these algorithms to find meaningful results, to attach meaning to the numbers for solving practical problems. This is a process of trial and error, I made a lot of mistakes and then corrected a lot of mistakes, and in such way, I learned.