Expedia Hotel Recommendations

STA 208 Project
Bingxi Li, Qiwei Li, Xin Li, Qiaojuan Niu

June 2016

1 Introduction

Planning your dream vacation, or even a weekend escape, can be an overwhelming affair. Hundreds, even thousands, of hotels at every destination make it rather challenging in cutomizing hotel recommendations for users. Motivated by the robustness of machine learning algorithms, Expedia expected an intelligent recommendation system to users for hotel search. Historical records are given as references. A successful system should be able to predict users preferences according to their travel plan.

2 Data Overview

The training set has 37,670,293 observations and 24 variables.The test set has 2,528,243 observations and 22 variables. Each row is a user log (search) on the website. We can aggregate these variables into four basic categories.

- **Search Information**: search time, secondary website type, website continent, whether using mobile device or not, advertisement channel, whether a booking or just a look, number of similar events
- **User Information**: user location country, user location region, user location city, user id
- **Trip Information**: origin destination distance, whether booked with flight, check in date, check out date, adults count, children count, room count
- **Hotel Information**: destination id, destination type, hotel continent, hotel country, hotel market, hotel cluster

3 Challenges

3.1 Mining of Latent Variables

Most given variables are directly observed variables but cannot be directly utilized by machine learning model. A lot of mining and transformation is required to infer the latent variables from current variables. For example, date-time variables like check-in and -out date, should be translated into frequency in order to understand to the relative preferences of certain date. The latter will have important impact on the the demand / supply relation of certain date and people's final choice of hotels. Even thought the feature engineering work is huge here, it will significantly promote the quality of machine learning models.

3.2 Encrypted and Vague Data

Moreover, the data set has huge amount of encrypted records. Most of them are hashed to some IDs as categorical variables. It is therefore impossible to find additive information behind these data. For example, we have the 237 location ID for user city, 1,008 IDs for user region, 50,447 IDs for user country, and continent. Besides there are 59,455 IDs for hotel area (such as park or downtown), 2,118 IDs for hotel market (such as New York, San Francisco), 213 IDs for hotel country. The hashing isolates important information for predictions. Besides, single data is sometimes insufficient to make sense. For example hotel area id meaning downtown or suburb cannot well represent an exact location except when used with hotel market and country together.

3.3 Expensive Computation

In spite of only 24 variables, more than 37 million encrypted and vague observations will cost huge amount of computation to train machine learning model. Even thought with the help of high-performance data structure in Python, training a robust model for each hotels location is beyond our computational capabilities. But the model can be scaled up for each hotel location on high-performance computing platform since they are based on same paradigm. Here, we mainly present our idea for model design on subsets of the whole dataset.

4 Feature Engineering

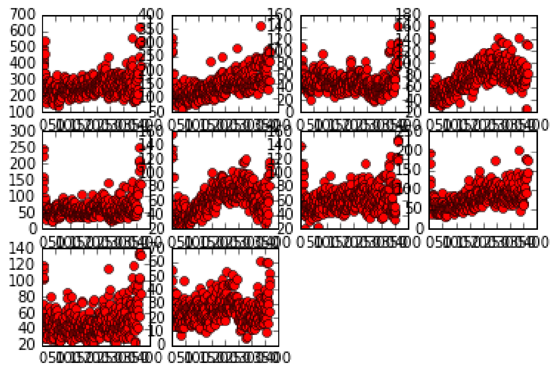
4.1 Brainstorming Features

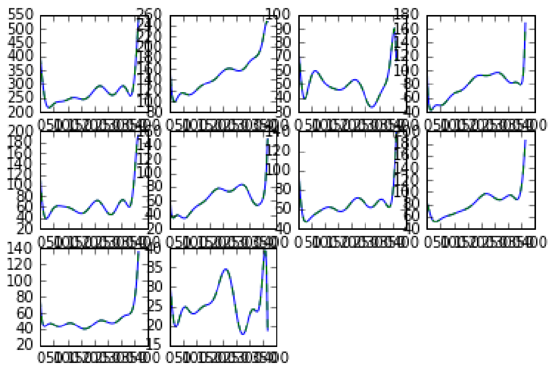
As mentioned in section 3.1, date-time related variables should be transformed to derive more important but latent variable. Here we list feature to create and transform and our motivation for these changes.

- **Time at Search**: Bookings for business travel are probably during day time while it might be night hours for vacations. Type of travel matters greatly to the choice of hotel
- **Weekday at Search**: Bookings for business travel during weekdays and while weekends are probably more favorable for vacation travel
- **Time before travel at search**: The free days left for checking in at search will affect greatly the availability of hotels and customers' final choice
- **Duration of travel**: People with short or long trips may tend to book certain clusters of hotels.

4.2 Transforming Features

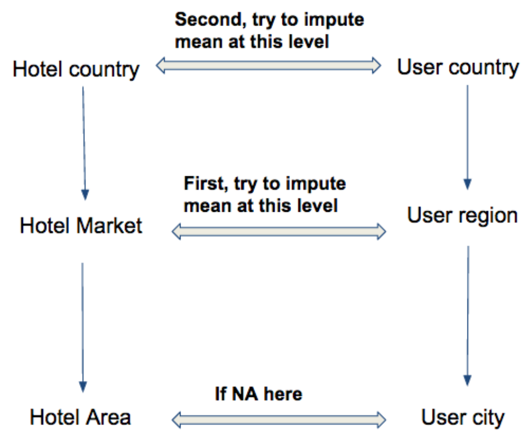
The hotel clusters are dynamic, meaning that it can change overtime for any particular hotel. Thus, check-in and check-out date contains information about the "current" state of the hotel at the time of trip. For example, when the destination location is very popular for a particular season, its cluster might change. Therefore, we first extract the most 7 popular destinations, then we transfer their date-time variables to hotels popularity by fitting functions to it check-in and check-out frequency for each destination. As you can see in the following figures, we first extract day of year of the check-in and check-out. Then we calculate the frequency of check-in and check-out against day of year. Finally, we fit a polynomial function to it and use the function value as a measure of the popularity of the hotel at the time of trip.


(a) Check In Frequency of Top 10 destinations


(b) Fitted function as hotel popularity

4.3 Checking and Improving Features

Through experimental models, we realized the variable 'orig_destination_distance' plays a crucial role in predicting hotels cluster. However, around 30% of them are NAs. We want to impute those NA values with mean values. The approach we used is to impute with mean distances between origin's higher level location ID and destination's higher level location ID. If the imputation fails at this level, we move one level higher.


Second, try to impute mean at this level
Hotel country
User country
First, try to impute mean at this level
Hotel Market
User region
If NA here
Hotel Area
User city

5 Modeling

5.1 Model Selection

Since the data is very large, we have limited machine learning that we can explore and tune, namely non-iterative model. We want our algorithm to have the following features.

1. Ability to handle both continuous and categorical variables
2. Ability to calculate the relative importance/significance of variables.
3. Ability to output predictions as probabilities.

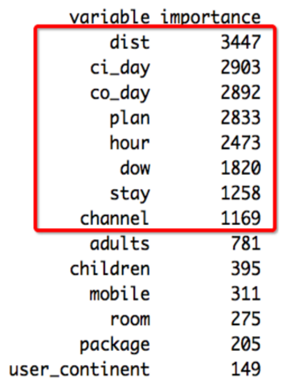
Since hotel cluster is categorical variable and predictors are both categorical and continuous variables, we need to find a model that can deal with both categorical and continuous variables. Actually, most regression model can be useful for that. However, we need to create many dummy variables for categorical variables. When using linear regression model, only the relative significance of each variable can be tested. Moreover, the top ranked 5 hotel_clusters are required for recommendation. which means we need a model to compute probability for each hotel_cluster. Even though logistic regression can do this job for us, it will create 99 models to compute the probability.

Random forest is very promising for this problem. It can handle categorical variables without creating dummy ones. Besides, random forest can help to compute variable importance for feature improvements. The voting procedure from aggregation of tree can serve as a pseudo probability output.

5.2 Variable Selection

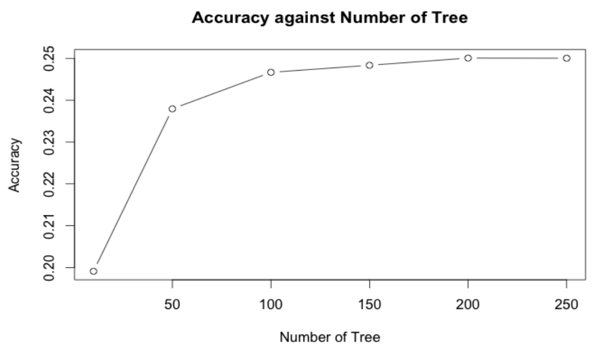
After all the feature engineering and the removal of ID variables, we get 14 variables. The random forest importance indicators will be used to determine the relative significance of each variable.

According to Table 2a, there exists a great difference in significance levels among variables. The random forest importance value 1,000 is set as the threshold to reduce variables for our model. Then we have following features left: dist (origin destination distance), ci_day (check-in date hotel popularity), co_day (check-out date hotel popularity), plan (how ahead the booking is), hour (search time hour), dow (search time day of week), stay (duration of trip), channel (Expedia advertisement channel).Notice that 6 out 8 of these important variables are features we generated or transformed.



```
    variable importance
        dist        3447
      ci_day        2903
      co_day        2892
        plan        2833
        hour        2473
         dow        1820
        stay        1258
     channel        1169
      adults         781
    children         395
      mobile         311
        room         275
     package         205
user_continent       149
```

(a) Variable Selection



(b) Parameter Tuning

Figure 2: Random Forest Model

5.3 Parameter Tuning

Random forest algorithm needs a certain number of trees for comparison. Therefore we tune the number of trees by balancing computation time and accuracy. When number of trees is set 100, the model has already converged in accuracy.

6 Result

Our tuned model is applied to 7 subsets of the whole data with the most popular destination locations. The accuracy is listed below. Here are some interesting conclusions. First, the model is as stable as we expect. The accuracy varies greatly between 0.05 to 0.4 with 0.2 being the average. Moreover, we can see that the accuracy has no correlation with number of unique level of responses or the train set size. This suggests user decisions on different destinations are affect by different variables. Our feature set is probably not good for all location.

Destination (Country-Region-City)	Records in Train	Levels of hotel cluster	RF Accuracy	Mean Average Precision
163-1503-11439	2742	50	0.4150	0.6121
50-1230-8279	7553	56	0.1045	0.1926
50-628-8250	30431	33	0.2484	0.4041
50-675-8267	12339	61	0.1811	0.3213
50-682-8268	5746	78	0.1257	0.2081
50-701-8260	4268	76	0.0521	0.1031
8-110-8791	5703	21	0.2940	0.4477
Total	-	-	0.2007	0.3320

7 Discussion

Our feature engineering work is very impressive on average. However when using with random forest model, it will be more robust to allow hotel-location dependent feature engineering work. It will be very promising to apply our current prediction paradigm for the whole data set. But the computation resources might be the bottleneck.