**Recommending Hotels Based On Expedia Historic Data**

*Summer Wang*

*Yawen Ye*

*Matt Staton*

*Wenduo Wang*

1. **Project goals**

In this project, we investigated a hotel booking dataset randomly selected by Expedia Inc. from the company's database. The stated objective is to recommend to users hotels that they are most likely to book based on historical data. Here we apply different predictive methods and compare their performance in terms of speed and accuracy.

1. **Prospect of the underlying solution**

A common theme of applying machine learning and predictive technology in business is to present the most relevant services to the user without explicitly request. For Expedia, an international online booking conglomerate whose customers are mostly individuals with diverse demands, knowing their users' preferences could create a win-win situation by saving the consumer effort comparing hotels and encouraging recurring orders. This benefit transcends all of e-commerce.

This predictive technology drives enhanced efficiency and lower cost. As pointed out in Expedia Inc.'s 2014 annual report, data center related costs in 2013 and 2014 totaled 199 and 229 million dollars respectively, accounting for 4% of total revenue. Additionally, data center costs accounted for 61% and 92% of the annual net income from 2013 to 2014. The potential profit improvement through leaner data analytics provides extra motivation for the business to employ novel technologies.

1. **A peek into the data**

The provided data consists of 3 files: a training set consisting of approximately 37 million rows, a test set of nearly 2.5 million rows, and a destinations file containing 149 features for all destinations.

The goal is to predict the probabilities of a user booking different hotel groups, coded as *hotel\_cluster*.

The training set spans from 2013 to 2014, and contains 24 columns recording the users' information and browsing history, including the hotel clusters the user viewed (*is\_booking* = 0) and booked (*is\_booking* = 1).

The records in the test set date from 2015, only containing successful transactions (*is\_booking* = 1), and therein *hotel\_cluster* is left out for prediction. The *user\_id’s* in the test set are a subset of the training *user\_id’s*.

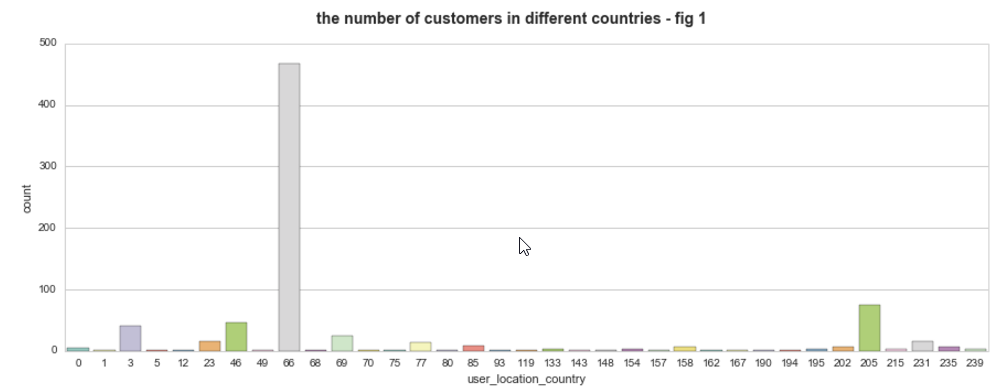
The separate destinations file is intended to provide more insight into the *srch\_destination\_id* feature in the training and test sets, as the *srch\_destination\_id*field contains more than 60 thousand distinct values.

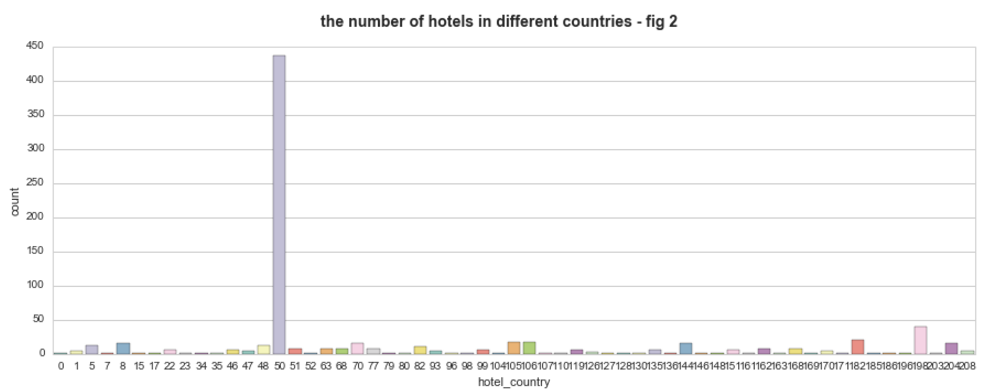
1. **Exploratory modeling and intuitions**

Pandas often reads data in incorrectly. For example, time data were recorded as strings, and accordingly we converted these columns to date formats.

We first tried logistic regression to understand which variables are important. However, the result showed little statistical significance for most variables. This makes sense because many variables are randomly coded by the company and are therefore categorical values, which are troublesome for linear regression, however the insight helped us to focus on more important variables when fitting a complex model.

Another way to gain an intuition of the data is to graph plots and identify patterns relevant to the recommendation. For instance, let’s have a look at hotel booking by *user\_country* and *hotel\_country*.





It should be noted that countries are coded randomly by Expedia. The plot shows a strong pairwise pattern, with which we can infer that domestic booking dominates the dataset. This implies domestic booking records are more abundant, and presumably domestic hotel recommendation results are more reliable.

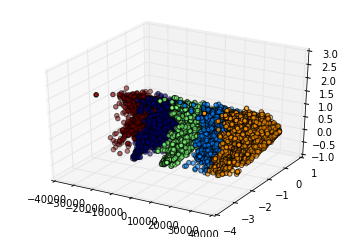
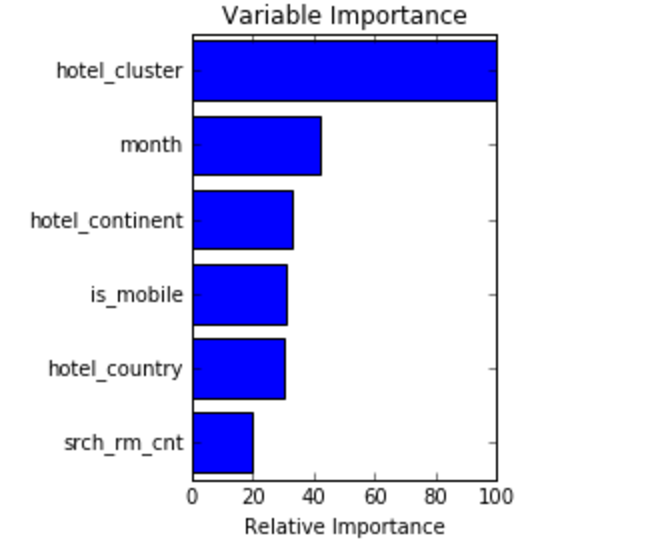
1. **Prediction**

The original training set is 4GB in size, which is unwieldy to work with in-memory. We connected to a virtual server for extra computing power, yet the standalone server struggled to produce results quickly, especially under high iterations. To work around the issue, we randomly picked 10,000 user\_id’s and retrieved their corresponding records, thereby relaxing the memory requirement. Distributed computing is another alternative, but the goal is the same, that is to increase speed without inducing considerable data selection bias.

* 1. **Feature Engineering**

The first task before applying any machine learning algorithm is to generate features. A primary feature in this case is *destination* which consists of each destination’s id, along with 149 latent attributes. To reduce the dataset’s dimension, we utilized Principal Component Analysis to reduce *destination t*o 3 dimensions.

To visualize the effect of PCA, we applied K-means clustering to divide our destination set into 5 groups (left figure.)

**5.2 Applying Machine learning algorithm for prediction**

Among parametric models random forest is an efficient first choice. Random forest has high tolerance of data types and is resilient to over-fitting. As demonstrated during lectures, we imported the random forest model from scikit learn package to fit the model and applied the 3-fold cross-validation across the training data set. Ultimately, we had an average score of 0.06 for RF.

The figure on the right are the six most important features in our random forest model. The variable importance feature in the scikit learn package shows how often each feature assisted in the classification process on average.

It is counter-intuitive and even confusing that *hotel\_cluster* is the most important variable to predict, well, *hotel\_cluster*. However, it actually makes sense, because in real life people tend to stay in the hotel they had experience with or in a popular and well rated hotel.

**5.3 Prediction based solely on destination and hotel\_cluster**

Since the number of *hotel\_cluster* is large, and boundaries between these clusters are fairly fuzzy, random forest may not be a good way to make predictions. Another problem with RF is its speed of training process, which is not scalable in real world, since as more data are available, the model needs to be retrofitted. Nonetheless, random forest has given us insight into the most important variables, with which we can further narrow down the variable scope to destination related variables and *hotel\_cluster*.

First, for each destination, we picked out the most booked and most clicked *hotel\_clusters*, which are assigned higher scores, and thereby we essentially created a list of dictionaries keeping record the most popular hotels at each destination. Second, since all the customers in the test data could be found in the train data, we could find matches among users in training dataset, and use match user’s booking hotels as preferable booking hotels for users in testing dataset. Thereby our accuracy score is improved to 0.23.

**6 Conclusion**

In this study, we investigated the hotel booking dataset, and created two methods to recommend hotels that users are more likely to book. The result shows that complex machine learning algorithms did not produce the best results in this case, with low accuracy and fitting speed. Several factors have contributed to this difficulty, including high dimensions of variables which resulted in lower information concentration. For example, there are 100 different hotel clusters, and the boundaries of all these clusters overlap frequently. Generally, as the number of clusters increase, classifiers decrease in accuracy.

To overcome the difficulty, we applied an alternative method by creating a list of dictionaries recording popular hotels at each destination. This method does not involve complex machine learning algorithms, and the system only needs to maintain and update a “score book”. When a user searches for hotels in a certain destination, the model will return the top 5 *hotel\_cluster* with highest scores. And unlike complex models such as RF, the simpler approach scales more easily, with lower requirements on computing power, and the data storage could be confined to the enumerable variables rather than every detail. With the new method, the cross-validation prediction accuracy increased from 0.06 to 0.23, and running time is shortened from 15mins to several seconds.

1. **Acknowledgement**

Thanks to James Coombes for his assistance on setting up a server for us.

1. **Reference**
2. Expedia Inc. annual reports, 2013 and 2014.
3. Kaggler tutorials
4. Prof. Chakrabarti’s lecture materials