Expedia Hotel Recommendation

DSO 593 Independent Research Report

Anyu Li

May 8, 2020

Table of Contents

Executive Summary

The purpose of this research is to study the methods for recommendation engine. The project carried out in the research is the Expedia hotel recommendation project which can be accessed via Kaggle. As there is a target labeled in the data, which is hotel_cluster, supervised models were built to predict the correct hotel cluster. Seven Supervised models were built, including Naïve Bayes, K-Nearest Neighbors, Logistic, Random Forest, Boosted Tree, Support Vector Machine, and Neural Network. The performance of each model can be seen in the Machine Learning Algorithm and Result section. Among all the models, the best model is the random forest, which has over 50% accuracy and exceeds most performance as shared on Medium and Kaggle.

For recommendation engine, generally, there are 3 ways to generate recommendations: content-based filtering, collaborative filtering, and the mixed of the previous two.

Content-based filtering methods are based on the description of the target items, which in this case, should be the hotels. However, since in this dataset, hotels were already grouped into clusters, it is straightforward to predict the corresponding hotel cluster for each record.

Collaborative filtering methods are based on the user level. It assumes that similar users prefer similar items. Therefore, in this case, booking history data was used in model building. In addition, K-means clustering algorithm was used to look for similar users and identify the most popular hotel cluster decision in each user cluster. The most popular hotel cluster or the booking history were integrated in the model through the feature engineering process, which proved that they were every helpful in terms of hotel cluster prediction.

Description of Data

The dataset is accessed via Kaggle (https://www.kaggle.com/c/expedia-hotel recommendations/data), which includes what customers' log activities were in 2013 and 2014. The train dataset alone has 37,670,293 records and 24 columns. Since the dataset is too large for general pandas processing, in this research project, 1% of the records are randomly sampled for analysis. In later part of this report, some techniques to handle big data are also discussed.

The sample, which is 1% of the original dataset, has 376,703 records. Most of the columns are appeared in numeric format but they actually function as a label or identification, representing different locations and users. Below is the summary of the unique value for each column.

Column	Unique Value
date_time	375,047
site_name	42
posa_continent	5
user_location_country	213
user_location_region	871
user_location_city	18,645
orig_destination_distance	223,465
user_id	262,797
is_mobile	2
is_package	2
channel	11
srch_ci	1,105
srch_co	1,106
srch_adults_cnt	10
srch_children_cnt	10
srch_rm_cnt	9
srch_destination_id	15,938
srch_destination_type_id	9
is_booking	2
cnt	37
hotel_continent	7
hotel_country	197
hotel_market	2,045
hotel_cluster	100

Table 1

Below is the summary and descriptions of the important fields in the dataset. Distribution of other fields can be found in appendix.

orig_destination_distance

This field is the physical distance between a hotel and a customer at the time of search. A null means the distance could not be calculated. This field also has the greatest number of missing values with about 35.93% of missing values.

As we can see from the plot below, the distribution is right-skewed, with majority of the distance in shorter distance.

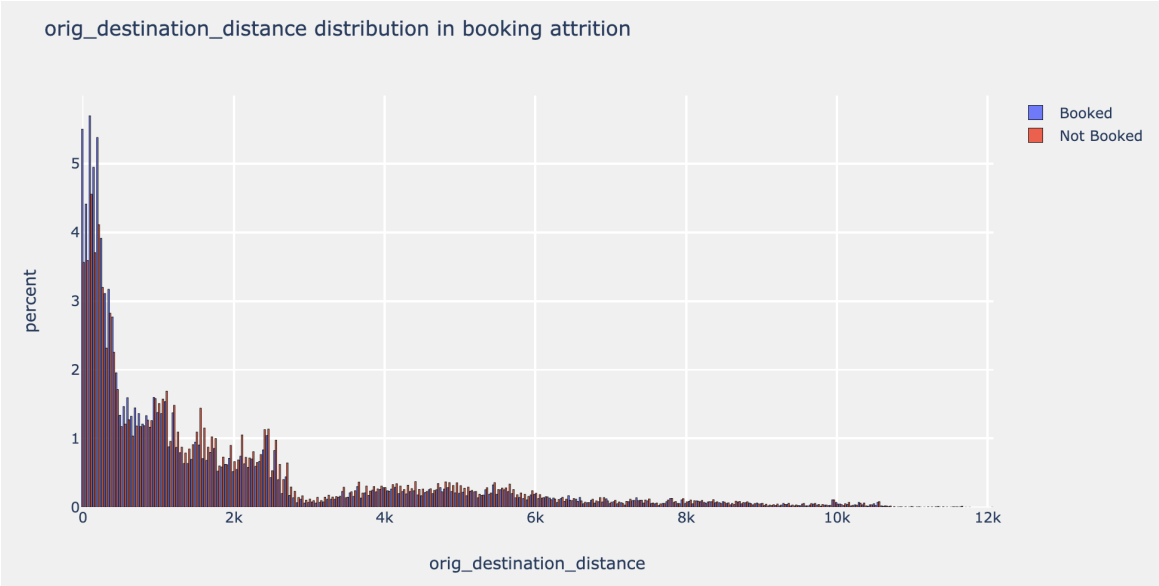

Figure 1

is_mobile

This field shows 1 when a user connected from a mobile device, 0 otherwise.

Majority of the records are not connected from a mobile device, suggesting users prefer website when access to Expedia.

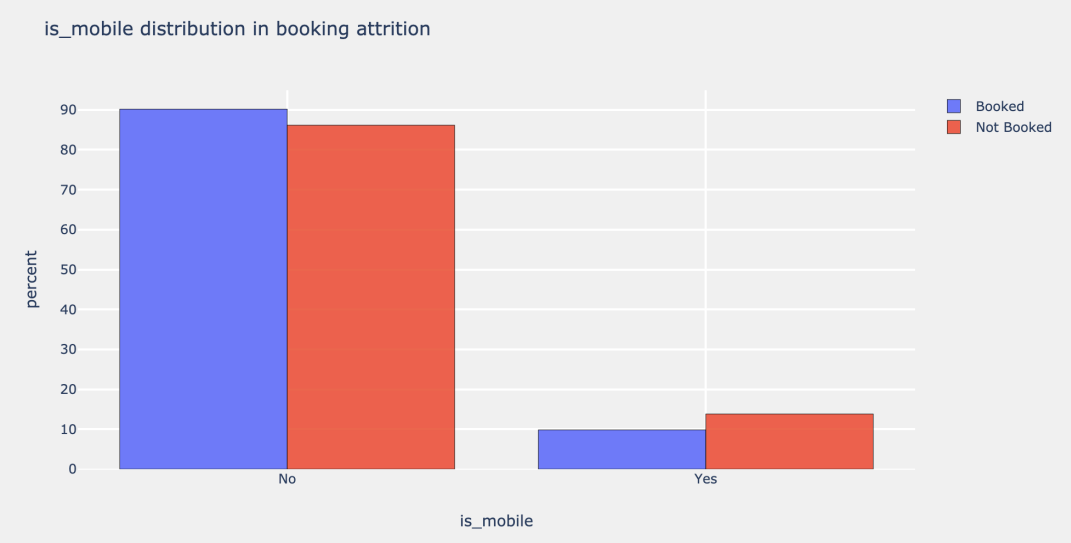

Figure 2

is_package

This field shows 1 if the click/booking was generated as a part of a package (i.e. combined with a flight), 0 otherwise.

Majority of the records are not part of a package. For those records which are in packages, most of them are not booked.

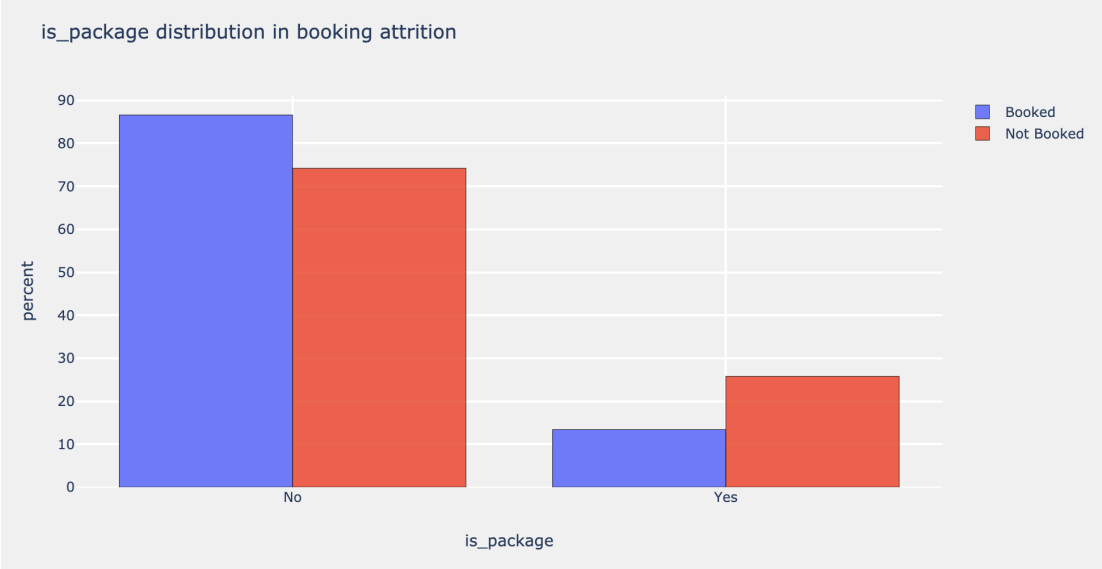

Figure 3

channel

This field shows the id of different marketing channel. Majority of the records in the dataset are from Channel 9.

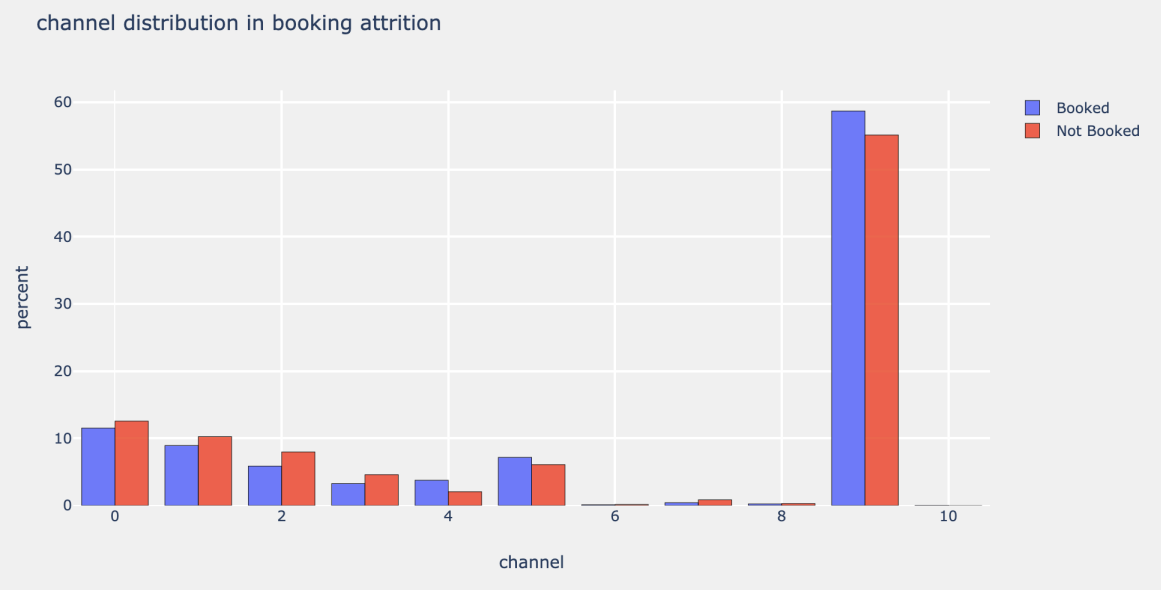

Figure 4

srch_destination_type_id

This field shows the type of destination. Most of the records are in Type 1.

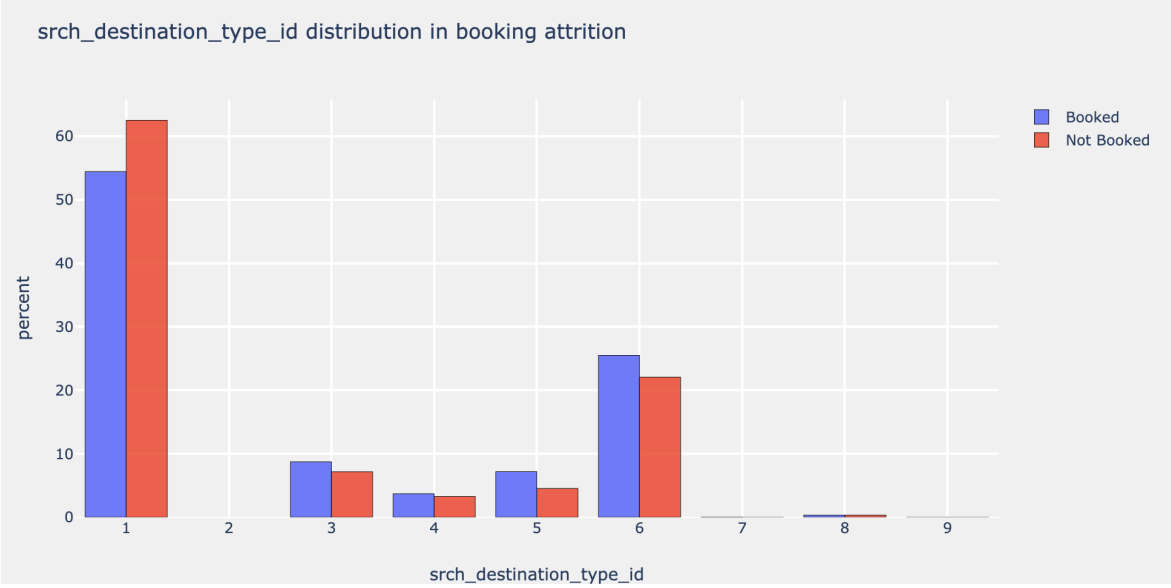

Figure 5

hotel_continent

This field shows the continent of the hotel. Most of the records are for hotels in Continent 2 and no records in Continent 1. There is no further information to label exactly which continent it is.

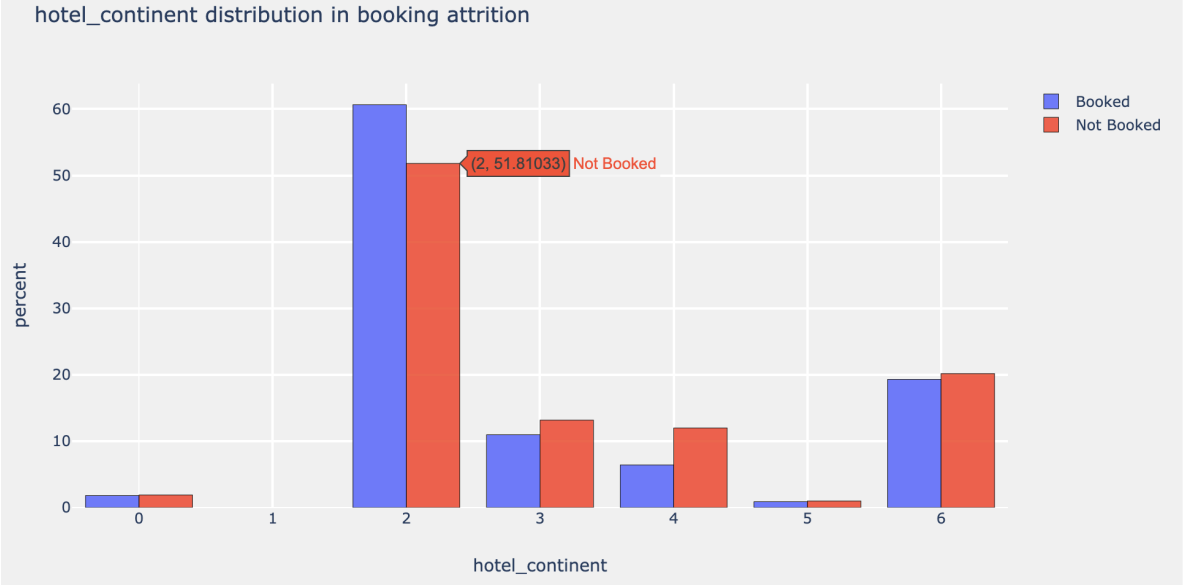

Figure 6

hotel_cluster

There are 100 unique hotel clusters in the dataset, and we can see from the distribution plot below that number of some hotel clusters are significantly larger than others. The hotel cluster that is most populated is 91, taking up about 2.75% in the whole dataset.

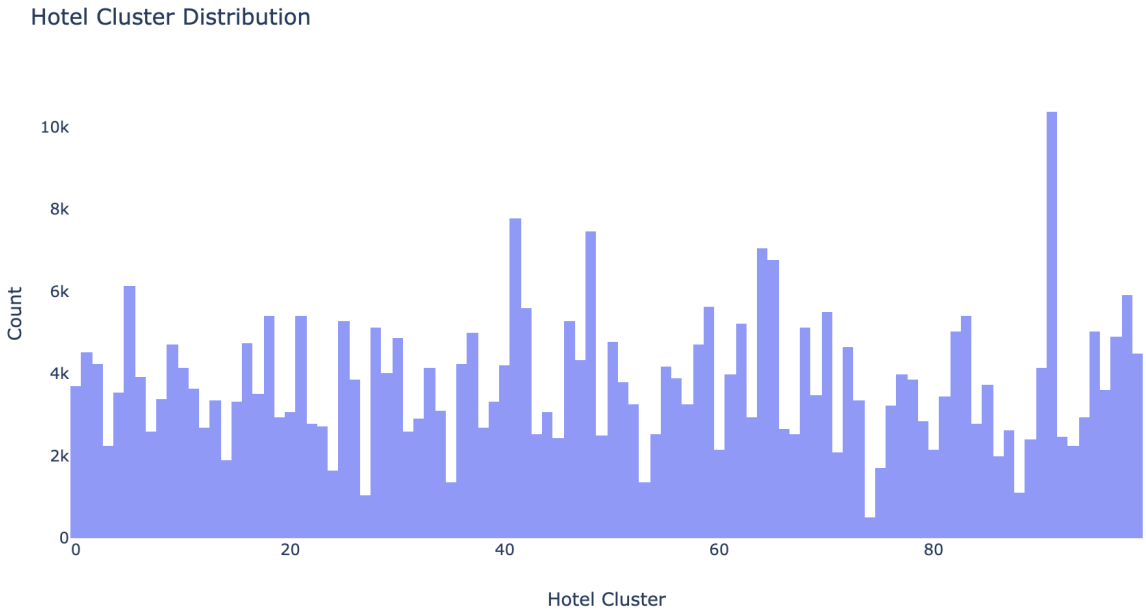

Figure 7

Data Cleaning

There are mainly 2 parts in data cleaning, correcting data types and filling missing value.

Data Type

There are 3 columns are in date format. For processing, we changed it from string to datetime format, which can be more efficient in the later feature engineering process, especially when I tried to extract weekday, month and year information.

Missing Value

This is mainly to fill the missing values in the orig_destination_distance column with 135,363 missing values. This missing value represents that the physical distance from search location to destination location are not recorded in log. Since, the missing value takes about 35.93%, I filled the column instead of dropping the NA records or dropping the whole column. There are 3 steps to fill this column.

Firstly, grouping by user location city and search destination id, I filled the missing value with the average in the group. 2.62% of the missing values were filled with first step. Secondly, grouping by user location city and hotel country, I filled the missing value with average in the group. 7% of the missing value were filled with the second step. Thirdly, grouping by user location country and search destination type id, I filled the missing value with the average in the group. 61.16% of the missing values were filled with the third step. Lastly, the remaining missing values were filled based on posa continent.

Feature Engineering

I also created 20 new candidate variables that measure user behaviors and hotel cluster history, which largely boosted the model classification performance.

Use Behavior

The candidate variables in this category were created based on the date columns.

1. Year and month were extracted from date_time, srch_ci, and srch_co columns, so we can have year and month information about the time user login, hotel check in and hotel check out.
2. Specified whether the login date falls on weekend or not, which intends to help predict type of hotels.
3. Measured how long the user were going to stay in the hotel, which is the difference between check out date and check in date in days.
4. Measured how many days the user took ahead to book or check the hotels.

Hotel Cluster

For existing users, hotel booking or browsing history can help predict which hotel cluster the users are most likely going to book. For new users, since there was no history information, most popular hotel cluster booked by similar users can be used as reference. Therefore, to create the referenced hotel cluster variable, I broke the records into 2 parts, which was also the train test split process. All the user included in training data were existing users and others were classified as new users.

I firstly found the most popular hotel cluster in training data for each user as the referenced hotel cluster. For the users in the test data, if the user id appeared in training data, the referenced hotel cluster was the same as the ones in training data. The remaining users were new users. For all the users in the dataset, I used k means clustering to group users into different cluster to figure out who were the similar users to any specific user. Attributes like site name as well as user location country were used to group similar users. Also, elbow plot method was used to decide the number of clusters desired.

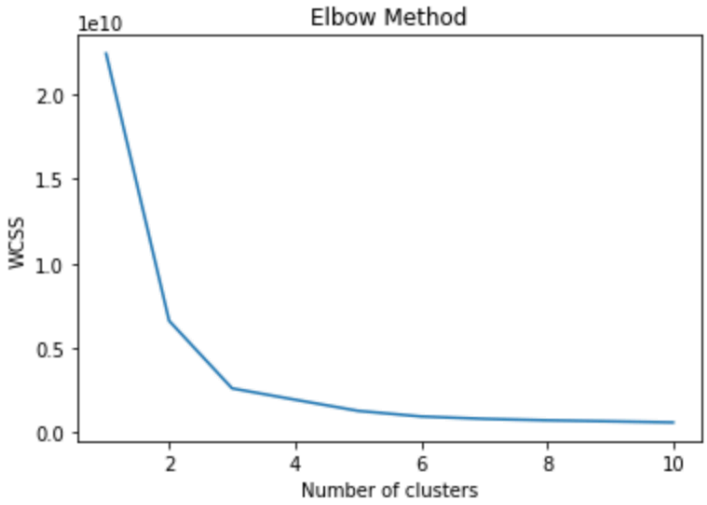

Figure 8

As shown in Figure 8, the elbow appears when there are 3 clusters. Therefore, all the users in the dataset were grouped in 3 clusters. For those new users, I checked which cluster they are in and made the most popular hotel cluster in that user cluster as the referenced hotel cluster.

Other Candidate Variables

Some other candidate variables are created by grouping the records with the same srch_destination_id, hotel_country, hotel_market, referenced_hotel_cluster, and the combination of these 4 attritions, and looking for the sum as well as the mean of the number of bookings. This is based on the intuition that the users can search for hotels for multiple times before booking the hotel they want, and the desired hotel cluster should be corresponding to a successful booking (is_booking = 1).

Other Data Processing

There are other data processing conducted, including data scaling and PCA.

Data Scaling

Data Scaling helps make all the column value in the same scale, which is helpful for the classification for some of the models. In this project, all the column values except for target variable (hotel_cluster) were z-scaled to have mean value as 0 and standard deviation as 1.

The formula we used for Z scale is : $x' = (x-\mu)/\sigma$

x': z score for each variable
x: value of each record on each variable μ: mean for each variable
σ: standard deviation for each variable

PCA

PCA can help find the dominated direction in the data and reduce dimensions. It is an unsupervised method which only looks at the independent variables. It rotates the coordinate system along the dominant directions. All the PC then are orthogonal and ordered by the variance or spread in their directions. As each PC is rotated axis and is thus a linear combination of the original variables which helps solve collinearity issue.

In this project, 25 PCs were taken, which explains 90.02% of the variance.

Machine Learning Algorithm and Result

7 types of machine learning models were used in this project, including Naïve Bayes, K-Nearest Neighbors, Random Forest, Boosted Tree, Logistic Regression, Support Vector Machine, and Neural Network, and each of the model were cross-validated 10 times to reduce variance.

The best performing model was Random Forest without PCA. The accuracy was about 50.56%, which outperformed many of the results shared on Medium and Kaggle. The Random Forest Classifier with PCA had lower accuracy and this may due to the loss of information happened in PCA.

Summary of the Model Performance(accuracy) is shown below.

	Naïve Bayes	**K-Nearest Neighbors**	**Random Forest**	**Boosted Tree**	**Logistic**	**SVM**	**Neural Network**
Without PCA	4.77%	9.33%	50.56%	17.18%	22.89%	26.80%	29.90%
PCA	13.09%	8.14%	20.34%	5.10%	16.73%	21.63%	29.50%

Table 2

Summary of the top 5 feature importance from the best model is shown below. The Top 5 features are all created from the feature engineering process. The Top 3 features are related to the referenced hotel cluster feature which captures the hotel cluster booking history.

Feature	Importance
rec_hotel_cluster_booking_mean	0.211335
rec_hotel_cluster	0.185523
rec_hotel_cluster_booking_sum	0.177068
group_booking_sum	0.054899
hotel_market_booking_sum	0.043229

Table 3

Conclusion

In this project, we tried to recommend the correct hotel cluster for each search record by building both supervised models and unsupervised model. The supervised model was to predict the corresponding hotel cluster by learning the attributes provided during each search such as time, destination, and market. The unsupervised model was used during the feature engineering process to look for similar users and the most popular hotel cluster decisions made in each user group.

As shown in the random forest feature importance table, user booking history and booking preference by similar users significantly help with the hotel cluster prediction. The accuracy of the best random forest model is 50.56% and if we excluded those booking preference variables, the accuracy is only about 10%. This indicates that users are generally consistent with their preference and similar users tend to make similar decisions.

For feature improvement, more candidate variables can be created to better measure booking preference, such as the most popular hotel cluster in recent one month, half year or one year. In addition, big data tool can be leveraged to study the whole dataset to learn more about the information and user pattern, which is also discussed in the Appendix 2.

Appendix 1 – Distribution of Other Columns

srch_adults_cnt

This field represents the number of adults specified in the hotel room. The number of adults are ranging from 0 to 9, and most of the rooms specify 2 adults.

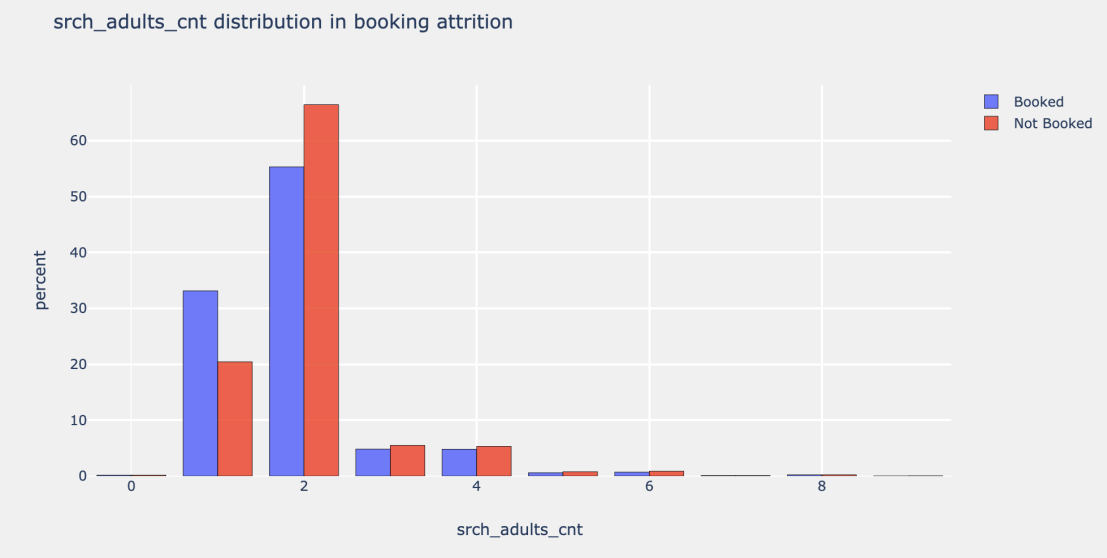

Figure 9

srch_children_cnt

Similar to the previous field, this field represents the number of children specified in the hotel room. The number is ranging from 0 to 9 and majority of rooms specified 0 children, which indicated that most of the rooms do not have the number of children specified.

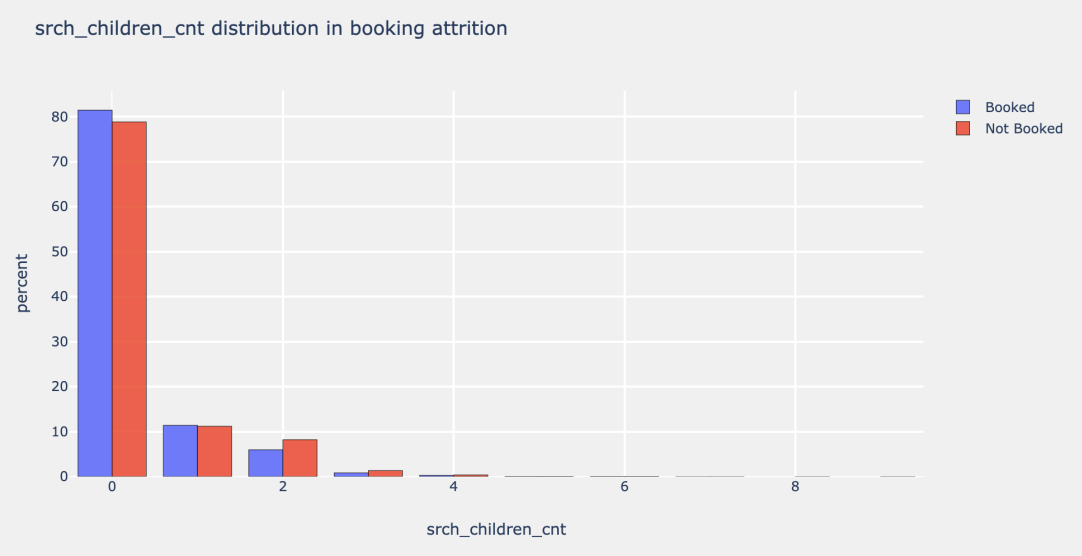

Figure 10

srch_rm_cnt

This field represents the number of hotel rooms specified in the search. The number of rooms ranging from 0 to 8. Most of records shows only 1 room in each search.

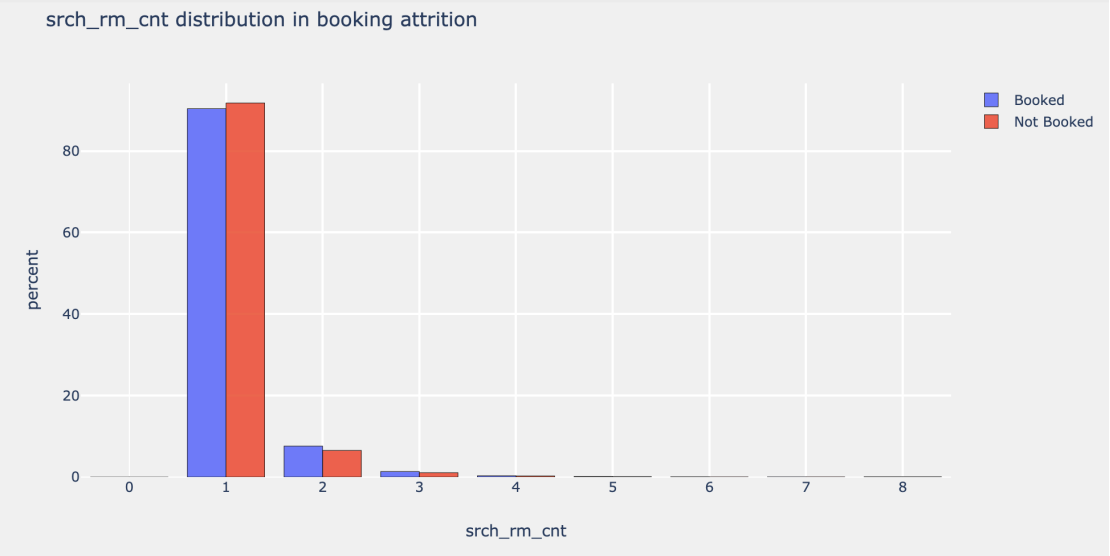

Figure 11

Appendix 2 – Big Data Technique

The original train data alone as downloaded from Kaggle is as large as 5 GB, with over 37 million records. To handle the big data, PySpark was tried for data loading and processing. PySpark is an open sourced apache spark package that integrated with Python. In addition to PySpark, cloud storage was used to better store data, which can help reduce loss in data and improve data loading speed.

Spark's core data structure is the Resilient Distributed Dataset (RDD), which is a low-level object that enables Spark work its magic by splitting data across multiple nodes in the cluster. PySpark is a great language to leverage Spark by using Python language.

Specifically, in this project, data were loaded to AWS S3. After connecting AWS S3 with local Jupyter Notebook, data then can be read and processed with PySpark on Jupyter Notebook. By Comparison, loading the data locally took about 3 minutes and loading the data from AWS S3 took only 567 ms. Further data processing like data cleaning and feature engineering as mentioned earlier is complicated but applicable using PySpark. Therefore, further research on PySpark will be conducted. To have better improvement, instead of creating clusters locally, clusters can be set up using virtual machine, cloud providers (AWS EMR or GCP) or vendor solutions (Databricks or Cloudera).

The key data type used in PySpark is the Spark dataframe, which can be thought of as a table distributed across a cluster and it functions similar to Pandas.

In short, PySpark is a powerful tool to handle big data in terms of exploratory data analysis, building machine learning model pipelines and handling ETL.

References

Reference	Links
Data	https://www.kaggle.com/c/expedia-hotel-recommendations/data
Model Performance	https://towardsdatascience.com/a-machine-learning-approach-building-a-hotel-recommendation-engine-6812bfd53f50
Recommender System	https://en.wikipedia.org/wiki/Recommender_system#Collaborative_filtering
PySpark	https://towardsdatascience.com/a-brief-introduction-to-pyspark-ff4284701873

Table 4