Expedia Hotel Recommendation System

JINGRONG TIAN, XUNGE JIANG

[1]Institution (Georgetown University)

<jt1204@georgetown.edu>, <xj64@georgetown.edu>

Abstract. All the travel agencies nowadays manage to satisfy the artificial intelligence-driven characterization in order to hold its position in the rapidly consolidating market. Thus a precise and scientific algorithm of recommendation, comparison, and matching is needed for every agency. This paper intends to provide a hotel recommendation system by using the dataset from Expedia, a world's leading travel platform. The aim of this hotel recommendation system is to predict and recommend five hotel clusters to a user out of one hundred given clusters.

Keywords: Recommendation. XGBoost. Naive Bayes. Random Forest.

1 Introduction

Booking a hotel is always an essential and personalized step of planning a trip. It involves many factors to make a decision, such as country, distance and weather. Every traveler researches and customizes his/her's itinerary to find the perfect balance among price, adventure and luxury. We wish to study the behavior and common characteristics of previous customers who used Expedia online booking to implement a hotel recommendation system. The system may not only save customers time on researching millions of options, but also give personalized and satisfying hotels.

The dataset was released by Expedia on Kaggle challenge. The goal of the challenge is to offer the top five hotels that are mostly likely to be booked for each user's specific search. In this study, we include many feature engineering techniques and three machine learning models to construct our system. Three models are Random Forest, Naive Bayes, and XGBoost.

2 Dataset

The dataset is available on Kaggle under competition named "Which hotel type will an Expedia customer book?" The training set consists of 37,670,293 entries and the test set contains 2,528,243 entries. The train data is a random selection from Expedia database in 2013-2014 time frame and the test set is from data in 2015.

The other dataset 'destinations.csv' contains all id that corresponds to srch_destination_id, along with 149 columns of latent information about that destination. Those latent features are assumed to be some combination of destination characteristics. These latent features were converted to numbers, so they could be anatomized.

As noted in the data description, some entries in srch_destination_id are missing since some hotels are too new to have enough features. Also, both datasets consist of 24 features. Thus, the challenging part of constructing a good system is to deal with missing value and curse of dimensionality.

Feature Name	Feature Description	Feature Data Type
date_time	Timestamp	string
site_name	ID of Expedia point of sale	int
posa_continent	ID of continent associate with site name	int
user_location_country	the ID of the country the user is located	int
user_location_region	the ID of the region the user is located	int
user_location_city	the ID of the city the user is located	int
orig_destination_distance	physical distance between a hotel and a customer at the time of search	double
user_id	ID of user	int
is_mobile	1 when a user connected from a mobile device, 0 otherwise	tinyint
is_package	1 if the click/booking was generated as part of a package, 0 otherwise	int
channel	ID of a marketing channel	int
srch_ci	Checkin date	string
srch_co	Checkout date	string
srch_adults_cnt	The number of adults specified in the hotel room	int
srch_children_cnt	The number of children specified in the hotel room	int
srch_rm_cnt	The number of hotel rooms specified in the search	int
srch_destination_id	ID of the destination where the hotel search was performed	int
srch_destination_type_id	Type of destination	int
hotel_continent	Hotel continent	int
hotel_country	Hotel country	int
hotel_market	Hotel market	int
cnt	Number of similar events in the context of the same user session	int
is_booking	1 if a booking, 0 if a click	int
hotel_cluster	ID of a hotel cluster - ***This is what we are going to predict***	bigint

Figure 1: DataSet Overview

Feature Name	Feature Description	Feature Data Type
srch_destination_id	ID of the destination where the hotel search was performed	int
d1-d149	latent description of search regions	double

Figure 2: Frequency Plot of hotel_cluster

2.1 Data Transformation and Feature Engineering

1. Change Data Type for Future Machine Learning. Decompose "datatime"columns to extract search month, day of week, and time of day.

2. Principle Component Analysis on destination data. Since there are over 20 latent features in the destinations.csv file, Principle Component Analysis is implemented to reduce the dimension. For each unique srch_destination_id, three corresponding latent features are selected from destinations.csv file, and are later joined to the original data-frame as 'd0','d1','d2'.

3. New Feature Generation. Transform two variables "check-in"and check-out time to one variable 'length_of_stay'.

4. Feature Selection. Based on the results of Tree Based Classifiers(Figure 3) and Expedia's search input format from its website. The following features are selected: 'stay_span', 'srch_destination_id', 'hotel_country', 'srch_rm_cnt', 'hotel_market', 'd0', 'd1', 'd2', 'popularity', 'is_booking', 'hotel_cluster'

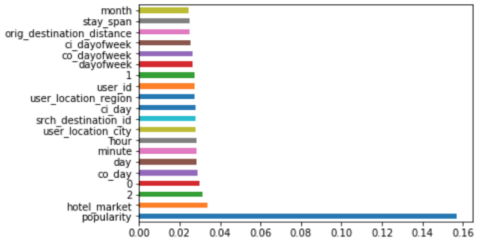

Figure 3: Tree Based Classifiers Importance

2.2 Data Exploration Analysis

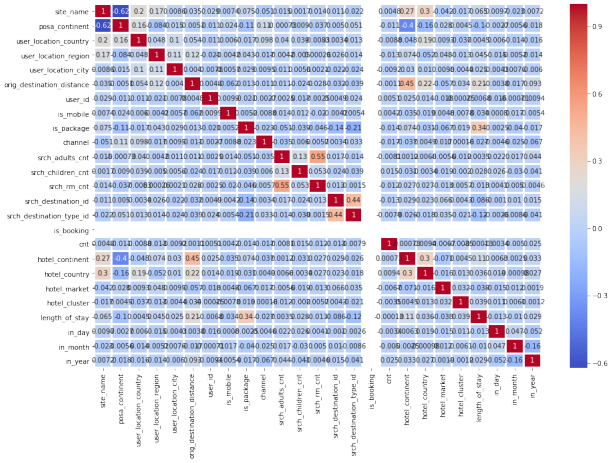

Figure 4: Frequency Plot of hotel_cluster

As the correlation plot shown in Figure 4, high correlation attributes are

- hotel_continent and orig_destination_distance
  Since orig_destination_distance contains missing value, choose hotel_continent.

- hotel_continent and site_name
  Choose hotel_continent as it has been included by previous discussion.

- hotel_country and site_name
  Choose hotel_country since site_name has high correlation with two variables.

- length_of_stay and is_package
  Choose length_of_stay since it is more representative.

- srch_destination_id and srch_destination_type_id
  Choose srch_destination_id as it is more unique and distinguishable;

- srch_rm_cnt and srch_adults_cnt
  Choose srch_rm_cnt since it summarizes both srch_adults_cnt and srch_children_cnt.

2

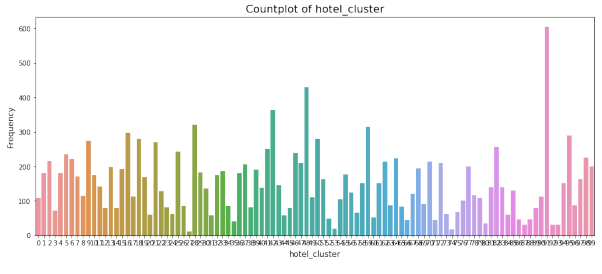

Figure 5: Frequency Plot of hotel_cluster

The plot shown in Figure 5 demonstrates the appearances count of each hotel cluster. The distribution is fairly even with only several high-frequency clusters.

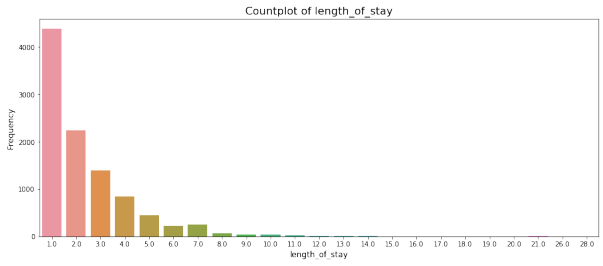

Figure 6: Frequency Plot of length_of_stay

The plot shown in Figure 6 conveys the travelers' tendency on duration of stay. People tend to stay less than a week, and mostly, they love to book for just a day.

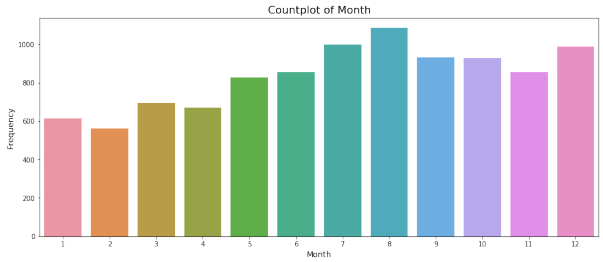

Figure 7: Frequency Plot of Check in Month

The plot shown in Figure 8 counts the frequency of check in month. As indicated on the plot, July, August and December are the peak months for traveling.

3 Methodology

3.1 Random Forest

Random Forest is a supervised machine learning method that can be applied on both regression and classification problems. To generate predictions for classification problems, the algorithm builds multiple decision trees and takes the majority votes from decision trees. A random forest classifier is a meta-estimator that fits a number of decision tree classifiers on various subsamples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.

The basic algorithm is as below:

Given a training set with n rows of data, in this project $n = 37,670,293$, which is then split-ted in X_train Y_train. For $b = 1, 2, ...B$ where $B =$ Number of sub-trees.

1. Assemble sets X_b, Y_b where $X_b \subseteq X_{train}, Y_b \subseteq Y_{train}$

2. Training regression trees on X_b, Y_b and collect the outputs

3. Take the average of all outputs.

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B}\hat{f}_b(x')$$

The value of B is adjustable. Normally for a large dataset, B is set to be around 1000 or higher.

3.2 XGBoost

XGBoost is short for Extreme Gradient Boosting, which refers to using ensembles of weak learners to make a prediction. XGBoost is based on this model. It is used for supervised learning problems, where we use the training data (with multiple features) x_i to predict a target variable y_i.

1. Creating regression trees and assemble the regression trees with their score

2. Create Objective for Tree Assemble. Assume there are k trees.

$$\hat{y_i} = \sum_{k=1}^{K} f_k(x_i)$$

Objective function is

$$L = \sum_{i=1}^{n} \text{training loss}(y_i, \hat{y_i}) + \sum_{k=1}^{K} \text{Complexity of the Trees}$$

3

for *k=1 to K* **do**
  Propose $S_k = s_{k1}, ...s_{kl}$by percentiles on feature k;
  Propose can be done per tree, or per split;
end
for *k=1 to K* **do**

$$G_{kv} = \sum_j g_j$$

  training loss ;

$$H_{kv} = \sum_j h_j$$

  Complexity loss;
end

Algorithm 1: XGBoost

3.3 Naive Bayes

Naive Bayes is a simple supervised learning probabilistic classifier that uses the Maximum A Posteriori decision rule. Simply speaking, it calculates all the probabilities $P(Class|x_1, ..., x_n)$ based on conditions and find the class label associated with the maximum probability.

It is constructed based on the assumption that the value of one attribute is independent of any other attributes, given the class/label.

Naive Bayes Classifier is based on the Bayes Rule.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

In this particular dataset, the probability of whole dataset is the same across models which the calculations are skipped for faster algorithm. Two unknowns are likelihood $P(x|c)$ and prior $P(c)$. Below briefly explains how algorithm is constructed, where x stands for attributes and c is class.

$$P(x|c) = \frac{P(x|c)P(c)}{P(x)} \tag{2}$$

$$\propto P(x_1|c) \times P(x_2|c).. \times P(x_n|c) \times P(c)$$

Predict Class = train class where $max(p)$ occurs

4 Results and Discussion

Three algorithms are implemented to build the recommendation system, including XGBoost, Naive Bayes, and Random Forest. Among the three algorithms, XGBoost gives the best result following by Random Forest and Naive Bayes.

Below is a piece of the visualized random forest.

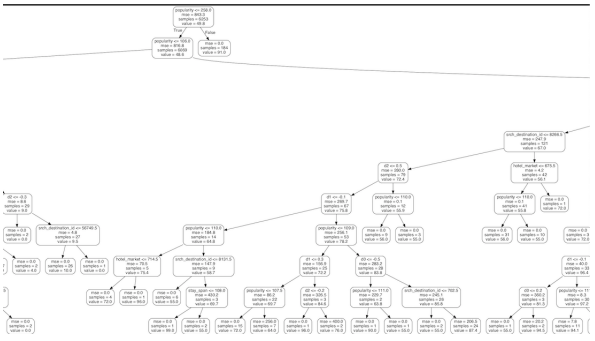

Figure 8: Zoom Part of Visualized Random Forest

Accuracy Table

Model	Accuracy
Random Forest	0.3738
XGBoost	0.4146
Naive Bayes	0.376

5 Conclusion and Limitations

We have successfully implemented a hotel recommendation system using Expedia's dataset even though most of the data was anonymized which restricted the amount of feature engineering we could do. We ranked the problem at hand as a multi-class classification problem and maximized the probabilities of each cluster, picking the top five clusters at the end.

The most important and challenging part of implementing the solutions was to create and extract meaningful features out of the 38 million data points provided to us. The exploration of data took a long time given the size of data and it helped us extract features that seemed to have high impact on predicting the hotel clusters.

Future works could focus on improving feature engineering by associating features together and creating pivot tables and pipelines for the later machine learning algorithms.

5.1 Model Limitations

Naive Bayes tends to compute the probability using method of frequency counts while the dataset contains

4

continuous variables.

The weakest aspect of Naive Bayes is the implicit assumption of independence between attributes. As the correlation heatmap suggests, many of the attributes a correlated, either low or high. It may have some impacts on classifying.

Random Forest is a strong classification method while it can be biased towards attributes with many levels. Since the data is fairly large and has some categorical variables, bias may exist in the model selection.

XGBoost is the strongest method among these three but the weakness is revealed with dealing with dummy variables. The dataset contains several dummy and binary features, XGBoost may not perform well.