CS/DS 866: Machine Learning Project Report
Expedia Hotel Recommendations

Amit Gupta IMT2012003
Madhumathi K IMT2012020
Tanmayee Narendra IMT2012046

June 20, 2016

Abstract

The main objective of the Expedia Hotel Recommendations competetion on Kaggle was to provide personalized hotel recommendations to users of Expedia, an online travel company. In this paper, we describe our approach and results.

1 Problem Description

According to Expedia, there is not enough customer specific data in order to personalise hotel recommendations for every user. However, they have logs of customer behavior. These include what customers searched for, how they interacted with search results (click/book), whether or not the search result was a travel package.

Expedia was interested in predicting which hotel group a user is going to book. Expedia already has in-house algorithms to form hotel clusters, where similar hotels for a search (based on historical price, customer star ratings, geographical locations relative to city center, etc) are grouped together. These hotel clusters serve as good identifiers to which types of hotels people are going to book, while avoiding outliers such as new hotels that don't have historical data.

The goal of this competition was to predict the booking outcome (hotel cluster) for a user event, based on their search and other attributes associated with that user event. The train and test datasets were split based on time: training data from 2013 and 2014, while test data are from 2015. Training data includes all the users in the logs, including both click events and booking events. Test data only includes booking events. destinations.csv data consists of features extracted from hotel reviews text.[1]

2 Data Description

The following table gives the names and descriptions of the attributes provided.

Column name	Description
date_time	Timestamp
site_name	ID of the Expedia point of sale
posa_continent	ID of continent associated with site_name
user_location_country	ID of the country the customer is located
user_location_region	ID of the region the customer is located
user_location_city	ID of the city the customer is located
orig_destination_distance	Physical distance between a hotel and a customer at the time of search
user_id	ID of user
is_mobile	1 when a user connected from a mobile device, 0 otherwise
is_package	1 if the click/booking was generated as a part of a package
channel	ID of a marketing channel
srch_ci	Checkin date
srch_co	Checkout date
srch_adults_cnt	number of adults specified in the hotel room
srch_children_cnt	number of children specified in the hotel room
srch_rm_cnt	number of hotel rooms specified in the search
srch_destination_id	ID of the destination where the hotel search was performed
srch_destination_type_id	Type of destination
hotel_continent	Hotel continent
hotel_country	Hotel country
hotel_market	Hotel market
is_booking	1 if a booking, 0 if a click
cnt	Numer of similar events in the context of the same user session
hotel_cluster	ID of a hotel cluster

3 Approach

This section describes various approaches that we used.

3.1 Aggregation Method

In the first approach, we tried to find the most common hotel clusters across the data, then used them as the predictions. We found out top 5 popular clusters in the data. For creating a better algorithm, we ran correlation method to check if anything correlates well with hotel_cluster. We found out that no column is linearly correlated with hotel_cluser. This is because there is no linear ordering to hotel_cluster. This suggests that techniques that relies on linear correlations between predictors and target, such as logistic regression, linear regression, will not work for this problem.

The next method was to find the most popular hotel clusters for each destination (i.e aggregation based on srch_destination_id). We used that to predict

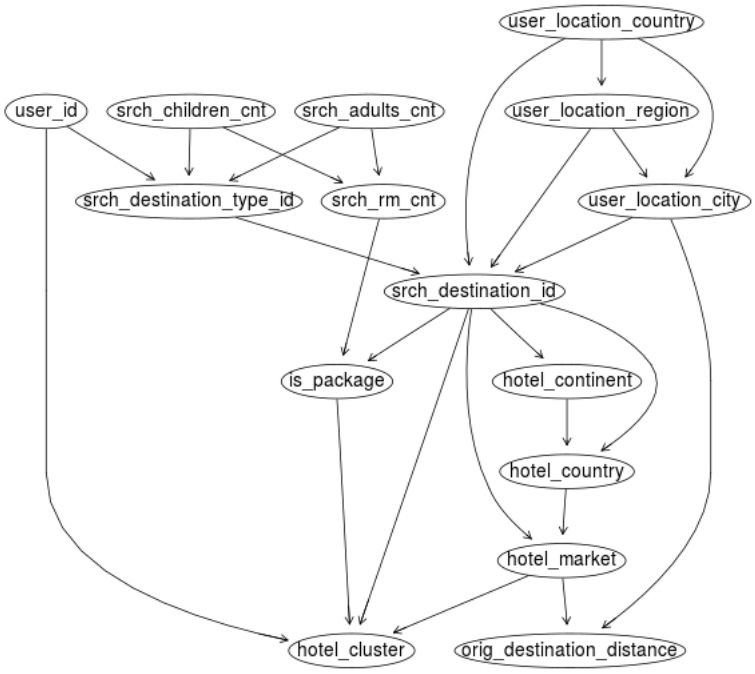

Figure 1: Initial Bayesian Network

that a user who searches for a destination is going to one of the most popular hotel clusters for that destination.

It was clear that aggregation methods were performing better than any ML method for this problem.

3.1.1 Data Leak

In this competition there was a data leak. The 'Data Leak' consists on the fact that the distances to the hotel from a given location are incredibly precise and unique. This is the same as stating directly which hotel was booked, and, ultimately, the hotel cluster. Close to 33% of the test data has a valid orig_destination_distance.

3.2 Bayesian Network Approach

Next, we tried to model this as a Bayesian network. Using our common sense and intuition about hotel bookings, we made the following Bayesian network. (See Figure 1) Recall that an arrow in a Bayesian Network represents some sort of dependency or influence. The basic principle behind the structure is that, each node is conditionally independent of its non descendants given its parents.

Note that for a particular record in the test file, we are given all values of the nodes, except hotel_cluster. Quantile binning was used to convert the attribute orig_destination_distance to discrete values.

The maximum likelihood estimate of the hotel_cluster value is computed as below. Basically, we are required to count the number of rows with a particular configuration of values, for all possible hotel_cluster values. The first five, in decreasing order of counts are given as the top 5 hotel_clusters.

From the local independence assumption, we know that the hotel_cluster depends only on its parents. Hence, the rest of the network (all other variables) becomes irrelevant.

3.2.1 Refining the Solution

Previously, we built the network based on our intuition. In order to improve the model, we next decided to exhaustively check for all possible parent configurations for hotel_cluster. However, this turned out to be computationally expensive. Fields such as srch_destination_id and user_location_city have unique values which are of the order 10^5. Storing these in main memory was a problem.

A work-around for this problem was found. Initially, the training data was sorted on the required attribute (say a) that has a large cardinality, and the initial and final indices of contiguous rows with the same attribute was stored separately. Later, only rows of a particular value of the attribute a were read and the count for the required parent configuration was computed.

We also tried different weights for counting. Previously, we were giving equal weights for all rows with a particular parent configuration. Now, we gave a higher weight for rows that had is_booking value of 1 and a lower weight for those with a 0 value.

For the final solution we used a parent configuration comprising of srch destination id, hotel_market, orig_destination_distance, user_location_country and user_location_city. Orig_destination_distance was not binned (so that we make full use of the data leak), and continuous values were used. These attributes were decided upon after training on 50% of the data, and checking on which provided a better results.

4 Implementation

The solution was implemented in Python using GraphLab library.

5 Results

Submissions were evaluated with Mean Average Precision (MAP) @ 5.

1. Destination_id : 0.24142
2. Destination_id and hotel market : 0.30350

3. Destination_id, orig_dest_dist and hotel market : 0.47831

4. Destination_id, User location_city, user_location_region, orig_dest_distance and hotel_market : 0.49654

5. Destination_id, User location_city, user_location_region, orig_dest_distance and hotel_market with different weightage for is_booking: 0.50177

6 Conclusion

The winning solution, with an accuracy of 0.61 was computationally extremely intensive; It used 700 GB of RAM and took more than 36 hours to run. A combination of field aware factorisation machines and gradient boosting was used, in addition to a gradient descent routine that used 10^{11} iterations. A majority of the top solutions used distance matrix completion strategies to exploit the data leak.

References

[1] Expedia Hotel Recommendations. *Kaggle.* https://www.kaggle.com/c/expedia-hotel-recommendations/

[2] Barber, David. *Bayesian reasoning and machine learning.* Cambridge University Press, 2012.