Airbnb New User Booking Patterns

Kaitlin Helfter
Purdue University
West Lafayette, USA

Namrata Vivek Raghavan
Purdue University
West Lafayette, USA

Himanshu Phul
Purdue University
West Lafayette, USA

Jonatan Nyström
Purdue University
West Lafayette, USA

ABSTRACT

In this report, we will discuss our findings with the Airbnb New User Bookings dataset. First we will describe the contents of the data, discuss our preprocessing methods and creation of new features. Then we will detail our selected models and evaluate the performance of these models. Finally we will discuss any new insights found from the data.

1 INTRODUCTION

Over the past two decades, the phenomenon of tourists staying in rooms rented out by local hosts has seen a tremendous growth, especially with the advent of the internet age. But the unprecedented rise of Airbnb, an online marketplace for short-term lodging, and similar "peer-to-peer short-term rental" (PSR) services, has marked a qualitative transformation in the tourism industry.

Statistics reveal that Airbnb has had a 153% global compound growth rate since 2009 and is on track to generate $8.5 billion in revenue by 2020. The quick rise in Airbnb's popularity begs the questions as to why customers choose its services, the answer to which lies in the vast data-centric business justifications the company employs for strategic executions.

Through our models we plan to accurately predict where a new user will book their first travel experience using data mining techniques in contrast to traditional research methods to better forecast demands and positively influence the bottom-line of a $7.6 trillion sharing-economy industry.

2 PROJECT TOPIC

The primary aim of this project was to predict which country new users who have recently registered on the Airbnb website would go to travel for their first trip. The goal of our project was two fold. Firstly the team wanted to conduct an exploratory data analysis on the raw data set and process the raw data set so as to gain meaningful insights. Secondly our team wanted to use to processed data set to generate optimized models that would effectively predict the country to which a new user on Airbnb would book their first trip.

We saw this as a perfect project for a data mining challenge as it includes different data sets that correlate to one another. The various data sets consists of different dimensions as well as abnormal data, thus there is a need for large amounts of feature engineering before we can use the data set to train the data mining algorithms.

While processing the attributes, we also believed separate correlations could be further inferred. For example, how to improve the advertisement's efficiency for different groups on different , etc.

In the following sections, we will discuss the preprocessing of the data sets, performance of the algorithms that were implemented to predict the destinations of a new Airbnb users and well as insights the team gained while implementing the models.

3 DESCRIBE THE DATA SET

Our data sets have been sourced through Kaggle, an open source website for data driven competitions. The data set we used contains 2 subsets of data. The first subset contains demographic data such as gender, age and browser data such as type of browser and their method of signup for 200,000 unique user ids. It also contained the user's chosen destination, which could have been 1 from among 12 possible destinations with 'no destination found' being an option. The 12 classes in the response variable are, 'AU', 'CA', 'DE', 'ES', 'FR', 'GB', 'IT', 'NDF', 'NL', 'PT', 'US', and 'other'.

Table 1: Description of columns in Data set 1

Feature Name	Description
Id	User id
Date Account Created	The date of account creation
Timestamp First Active	Timestamp of first activity
Date First Booking	Date of first booking
Gender	Gender
Age	Age
Signup Method	Sign up location
Signup Flow	The user's origin page from
Language	International language preference
Affiliate Channel	What kind of paid marketing
Affiliate Provider	Where the marketing is

First Affiliate Tracked	First marketing interaction
Signup App	Application used for sign up
First Device Type	The device used when signing up
First Browser	The browser used at sign up
User Destination	Target prediction variable

The second data set contains over 10 million entries on web session data which each entry containing information such as the user's actions and time spent performing those actions. There about 135,000 unique user ids in this data set.

Table 2: Description of columns in Data set 2

Feature Name	Description
User Id	Value identifying a user's actions
Action	The action the user took
Action Type	The physical action the user took
Action Detail	Description of the user's action
Device Type	Device the user was using.
Secs Elapsed	The time elapsed for each action

4 PREPROCESSING

In order for the data to be used with our selective models, a significant amount of feature engineering had to be done. This feature engineering occurred in two steps: Merging the data sets and, imputing abnormal entries and creating additional features. Either one hot encoding or label encoding was then applied to the categorical columns depending upon the model implemented.

Merging the Data Sets

In order to merge the two data sets, we had to go through the web sessions data and consolidate it. For each user, we took the most common entry for the 'action', 'action type', 'action detail' and 'device type' features. For the 'secs elapsed' feature, we took the average of each user's sessions . Thus by applying window functions, the sessions data was consolidated and we performed an inner join between the two data sets on the primary key user id to merge them together.

Creation of Additional Features

In total 8 features of the combined data sets had null values. These 8 features were 'date first booking', 'age', 'first affiliate tracked' and all of the session's features.

We chose to drop the 'Date First Booking' feature in order to remain true a prediction model. This feature was added after the user booked their first trip and thus gave away too much information on the data set. For the 'age' feature, we cleaned up the data as values ranged between 0 and 2000. For values above 1000, we subtracted them from the current year, as this is the case where the user entered their birth year instead of their age. We replaced null age values and those beyond the range of 18-90 with the average age 34. For all the missing values in 'secs elapsed' field, we replaced with 0. For the remaining categorical features containing null values, we replaced those missing values with -1.

In order to keep track of each feature's missing values, we created additional features that would act as flags.

Action Fields

There are 3 fields in the session data set that have to do with the users' actions: 'action', 'action type' and 'action detail'. Since these fields are categorical, it would need to be encoded using either label encoding or one hot encoding. However, due to the large amounts of different possible actions, it would lead to too many additional columns which is not practical for spatial storage. Handling all 3 features separately would have added upwards of 450 new features after encoding them. Hence, we grouped actions with very low frequencies so as to reduce the amount of added features during one-hot encoding. By choosing to group every action used less than .01% of the time, we reduced the total number of needed features by 63%.

5 KPI METRICS USED

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where: TP = True Positive, TN = True Negative, FP = False Positive, and FN = False Negative

The first metric that our team had decided to use to evaluate the performance of the model was accuracy. The reason our team decided to calculate the accuracy of the models was because accuracy is a widely used metric that can be easily calculated for classification problems. However, the data set generated after the preprocessing steps described in section 4 consists of 12 highly imbalanced classes. Since accuracy is not a good metric for evaluating the performance of a model with imbalanced classes, our team decided to also use the nDCG score of each model to determine the best over all model.

nDCG - Normalized Discounted Cumulative Gain

$$nDCG = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(i+1)}$$

where *reli* is the relevance of the result at position *i*.

For each new user, our model outputs 5 predictions on the country of the first booking in decreasing order of probability. The ground truth country is marked with relevance = 1, while the rest have relevance = 0.

So in totality, the nDCG KPI awards bonus points to our prediction if it contains the ground truth (i.e with *reli*) and in proportion to its position in the prediction list (i.e *i*)

For each user prediction, a nDCG score between 0 and 1 is generated. An average nDCG score is then calculated across all of the users in the data set to obtain the model's final nDCG score. The higher the score, the better the model's ability to predict the users' intentions.

As a scoring function, nDCG shows its strength in evaluating a model when the input between the samples is similar but have different output labels. In this case the model will predict the input to belong to the label that occurs most frequently in the dataset. If the dataset is uneven, then one specific label will be over represented as the prediction output. An underrepresented label will then show up as second or third most likely prediction to the input. nDCG, as the scoring function, will take this into consideration and give this prediction, even if it was wrong, a value that closer represents how close the models were to predict correctly. This in return makes nDCG give us a better understanding on how well a model fits an uneven dataset. This is why Airbnb specified nDCG as the scoring function for their competition.

6 METHODS USED IN PROJECT

Random Forests

Random Forests is a supervised classification algorithm consisting of a large number of individual decision trees that are then bagged together and operate as an ensemble. The power of random forests lies in the fact that it uses a large number of relatively uncorrelated models to generate highly accurate predictions. The reason our team decided to implement the Random Forests algorithm is because of its robustness to outliers and non-linear data and its ability to handle unbalanced, high dimensional data. In addition, following literature surveys that our team had conducted, Random Forests was one of the algorithms that generated accurate predictions and had a high Normalized nDCG score.

In order to implement the Random Forests algorithm, we first conducted feature engineering following the steps outlined in section 4. Once the data was preprocessed and additional features were added, we then conducted label encoding for all the categorical columns in data set using the Scikit-Learn libraries in Python. Label encoding was chosen instead of one hot encoding as one hot encoding results in a sparse decision tree.

Once the data was processed and label encoding was conducted, we implemented the Random Forests algorithm with a 5 fold K cross validation using Scikit-Learn libraries in Python. The hyper parameter tuning of the model was conducted in order to optimize the performance of the model. We tuned the hyper parameters, number of trees and the max depth of the trees to optimize the average nCDG score of the test data set after 5 fold K cross validation. The optimal number of trees was chosen from a set of 5, 20, 40, 60, 100, 150, 200, 300 trees, while the optimal max depth of a tree was chosen from a set of 5, 10, 15, 20.

Lastly, once we narrowed down on the model with the best parameters, we compared the performance of this Random Forests algorithm to the performance of the other models implemented in this project, as well as to the algorithms discussed in the literature survey.

Neural Networks

Neural Networks is a machine learning method based on imitating the functionality of neurons in the human brain. It can both work with supervised or unsupervised learning. Since in this project our data contained labels, we decided to go with the supervised learning method.

The power of Neural Networks lies in its ability to capture complex relationships between input and output. A Neural Network would be able to fit 100% of the training data if it is created with enough nodes. A Neural Network is also very versatile, meaning there are a lot of different variables and functions that can be tweaked to improve the results.

However, all of the strengths of a Neural Network come at a cost. It is very computationally expensive. It recalculates the weights for every sample, meaning there is no good way to optimize the iterations through the whole data set at one time. A Neural Network is not only time consuming when it comes to training, but it is also time consuming when it comes to hyper-fitting it to a data set. Because there are so many variables, and retraining it is required to see the results, it is a model that requires a lot of time to implement correctly.

The reason our team decided to implement a Neural Network is because of its power of solving complex problems and our knowledge that the data set is very complex in nature due to its human elements. We discovered in our literature survey that Neural Networks were used in previous papers [1], though in that paper, the authors used a neural network only as a binary predictor to decide if the user was going to the U.S. or to another country. This differentiates a bit from how we chose to implement a Neural Network, as we wanted it to predict a destination out of 12 different labels.

The feed-forward Neural Network is constructed using 4 hidden layers with 100, 75, 75 and 50 nodes and an output layer with 12 nodes. The hidden layers used *ReLU* (rectified linear unit [2] as an activation function while the output layer used *Softmax* [3]. The optimizer used to train the network was *Adam* (Adaptive Moment Estimation), which is a version of *RMSProp*[4]. The loss method used for training was *Sparse categorical cross entropy*, which just uses one label to measure

how close it is to the correct prediction and is therefore commonly used with Softmax.

For software we used TensorFlow (1.14.0) as the back end neural network and Keras (2.3.1) as the API to program the neural network. The programming language used was Python 3.7.3.

XGBoost - Extreme Gradient Boosting

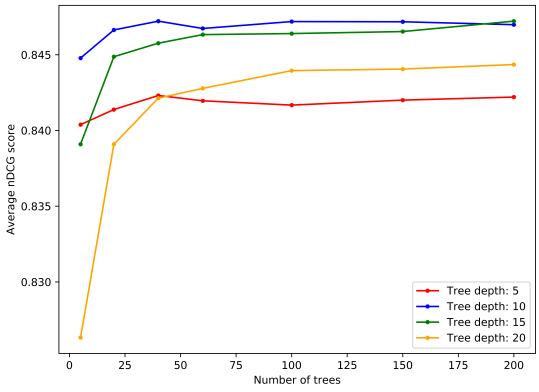
XGBoost is a scalable and accurate implementation of gradient boosting machines and it has proven to push the limits of computing power for boosted tree algorithms as it was built and developed for the sole purpose of model performance and computational speed. XGBoost and Gradient Boosting Machines (GBMs) are both ensemble tree methods that apply the principle of boosting weak learners (CARTs generally) using the gradient descent architecture. However, XGBoost improves upon the base GBM framework through systems optimization and algorithmic enhancements. The reason why the team decided to use XGBoost was to make use of its power to harness not only computational but hardware optimization abilities since when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now and also befits our chosen data.

While implementing the XGBoost model, some changes were made while preprocessing the data which is discussed later on in the report. After carrying out cross validation on our training data set, an optimum XGBoost model was chosen and compared with the previous models used during the course of the project.

7 PERFORMANCE OF METHODS USED

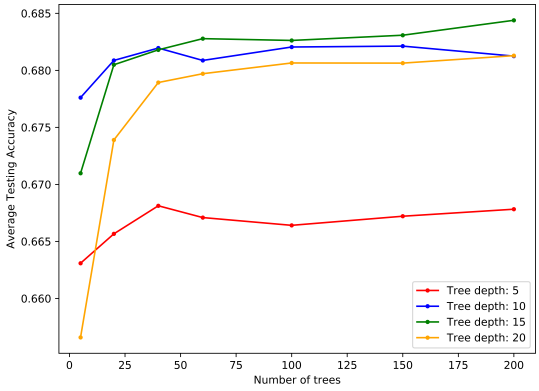
Random Forests

As mentioned in the Section 5 (Methods Used - Random Forest), hyper parameter tuning was conducted to choose the best model that optimizes the nDCG score. The graph below shows how the average nDCG score of the test data set varies with the number of trees as well as the maximum depth of the trees.



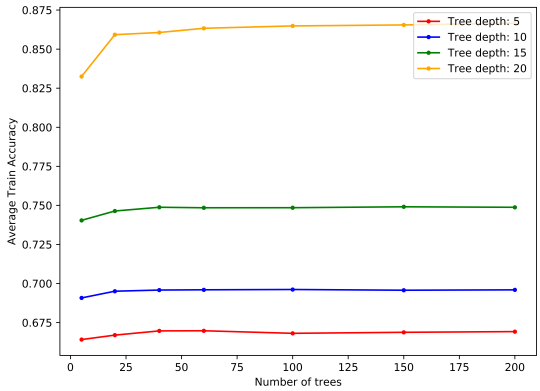
From the learning curves of the average nDCG score of the test data sets, we can see that optimal hyper parameters are setting the number of trees to 200 and the maximum depth of the tree to 15. The best model is seen to have an average nDCG score 0.847.

The following are the learning curves for the average testing accuracy versus the number of trees and the max depth of each tree.



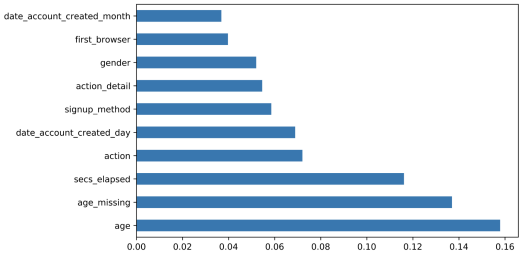
From the graph above, we can see that the best model (200 trees with maximum depth set to 15), has an approximate testing accuracy of 68.74%

The following graphs are the learning curves for the average training accuracy versus the number of trees and the max depth of each tree.



From the graph above, we can see that the best model (200 trees with maximum depth set to 15) has an approximate testing accuracy of 75%. The graph also shows that the model with maximum depth of 20, tends to over fit the data as its training accuracy is seen to be much higher than the other models. However, on the test data set, it does not perform very well.

Lastly, Python's Scikit-Learn library allows us to plot the relative importance of the features to another in the Random Forests model. The following plot shows the relative importance of the 10 most important feature in the data set.



From the feature importance plots, we can see that 'age' as well as 'secs elapsed' are the most important features in determining the country in which a new user is most likely to book their first trip via Airbnb.

Comparing the performance of the Random Forests model to Neural Networks and XGBoost, it is seen that Random Forest is seen to have an average nDCG score higher than the feed forward Neural Network, and an average nDCG score slightly lower than that of XGBoost. This is consistent with the literature survey that was done by the team.

Comparing the performance of the Random Forest model implemented above to the one implemented in [5]. we see that the current model has a test accuracy much lower than the accuracy of the model implemented in the paper (88%). A deep dive into the paper shows that the authors of the paper, included the feature, "date of first booking in their model". Our team had decided to exclude this feature in generating the model since in the spirit of creating a model that predicts the country to which a new user books their first trip via Airbnb, it would not make sense to include the feature, "date of first booking in their model", as this feature is generated after the user books their first trip via Airbnb. In addition, other implementations [6], [7] did not include this feature. Including this feature in the above generated Random Forests model pushes the model performance to 88%, similar to the performance seen in [5]. Thus, the implementation in the paper [5] does not capture the true spirit of predicting the country to which a new user books their first trip via Airbnb.

Neural Networks

The Neural Network model under performed when compared to the expectations we had on the model. With a reputation of being very versatile and able to fit most datasets, we thought it would be the same for this case as well.

The Neural Network has an accuracy of 62.9% and nDCG score of 83.4, making it the lowest score of the three models.

Even though it scored lowest in accuracy, it was only a point or two off from the other scores when it came to nDCG. This could be explained that the Neural Network is better at picking out top candidates but is very unsure about which is the best classification.

Our Neural Network scored way lower than the network in [1] did. In their paper they reported a nDCG score of 87.5%, which is significantly better than our score. They also had a different way to preprocess the data which could have contributed to the difference in score.

There are three reasons that we can come up with when we try to explain these results.

One is that the Neural Network is very bad at handling outliers. There are 12 different labels with half of them having less than 1% of the samples representing. This can make the Neural Network focus more on the larger datasets and less on the outliers.

Second is that we did not have the processing power needed for a large enough network to fit the data.

Third is that we do not have enough experience to make a Neural Network adjust to the dataset. It could be attributed to us being unable to master Keras nor TensorFlow. It could also be that we do not have enough knowledge. As mentioned before, there are few models that have as many parameters as a Neural Network that can be changed. We might have used the wrong values or the wrong methods in the network which could have prevented it from working properly for this scenario.

Different activation functions, number of nodes in each layer and optimizers were tested, though any major improvements were not seen.

XGBoost

The XGBoost model that obtained a score of 0.85 was the best nDCG score achieved amongst all the algorithms that we tested. The parameters that achieved this score had a max depth of 10, a 0.1 learning rate and 5 estimators. Training the classifier and cross-validating every combination of parameters took 35 minutes. The data set contained a total of 278 features where 85 came from the session's data set.

	train-mlogloss-mean	train-mlogloss-std	train-ndcg-mean	train-ndcg-std	test-mlogloss-mean	test-mlogloss-std	test-ndcg-mean	test-ndcg-std
0	1.179530	0.000989	0.836595	0.000184	1.247239	0.008934	0.822143	0.000690
1	1.343045	0.016489	0.766590	0.000839	1.471470	0.016626	0.739538	0.000419
2	1.030715	0.029164	0.848549	0.001430	1.201143	0.022316	0.815235	0.000558
3	2.920725	0.338127	0.816481	0.008332	3.148104	0.379141	0.776174	0.009195
4	1.296313	0.051722	0.856132	0.002401	1.513065	0.040170	0.812447	0.000704

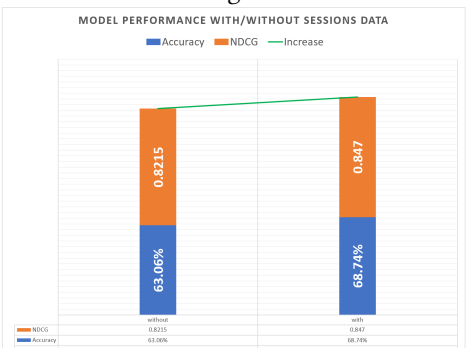
In comparison to the XGBoost model used in one of the Literature Survey done by the team in the beginning [6], our model fell short by 3 points in terms of nDCG score. It is worth mentioning here that the data used for the XGBoost model was preprocessed differently than the previous models. Taking hints from the Random Forests model, all the important features were retained as it is while the relatively low-priority features were grouped under a feature called "Others". This grouping helped reduce model dimensionality and memory complexity leading to a faster training time and may also be the reason for the increased scores over other models.

In terms parameter tuning, changes were made only to the tree depth and number of estimators used during training. However, after every cross validation, little increase was observed in our KPI's. Although the team was constrained for time, it is recommended that a grid search be carried out over a range of all the parameters to obtain the best set of parameters.

8 DISCUSSION OF INSIGHTS

One of the main insights that we gained through working on the various data mining algorithms is the importance of the web session data features in increasing the performance of the models while predicting which country a new user will make their first booking to on Airbnb. During the initial stages of model generation, our team was only using the data set 1 since data set 2 was extensive with over 10 million rows and only 5 additional features. In addition, data set 2 had information for only about half of the unique users in data set 1.

While training the models using only the information given in data set 1, we were getting relatively low average testing accuracies of 63% and average nDCG scores of 0.82. However, once we consolidated the two data sets together using the steps mentioned in section 4, and trained the models on the consolidated data set, we were able to improve the models' overall performances. The graph below depicts the overall improvement in the models' performance average before and after training it on the web session data set.



In addition, through analysis of the most important features returned by the Random Forest model in section 6, we can see that the features, 'secs elapsed' as well as 'action' are very important in predicting which country a new user is most likely to book their first trip to via Airbnb. This shows us that web session data helps to improve the over all performance of the aforementioned data mining algorithms.

9 EVALUATION OF THE OUTCOME OF YOUR PROJECT

As highlighted earlier on in our midterm report, our team was able to successfully process the data and gain valuable insights from our exploratory data analysis. Our team was also able to successfully implement a range of data mining algorithms to make the best possible predictive model and in doing so, we were able to compare and contrast three different models based on three underlying algorithms (i.e NN, RF, XGBoost) and adjudge their performance based on the key process indicator chosen (nDCG score), testing accuracy and convergence time. We, therefore, were able to meet our project goal within due time. However, we also believe that there are several different ways of possible improvement that can be used in the future to better predict user intentions but could not be implemented during the scope of this project due to time constraints. A few suggestions that the team proposes include:

Data Addition

It is possible for the team to generate additional features which might improve the overall performance of the models. For example, we can calculate the approximate distance of a user to his/her choice of destination and include it as a feature to better gain insight into booking patterns. We could have also added the time when a user creates their Airbnb account to study if the time of day affects the the destination of first booking.

Feature Analyses

As described in the Random Forest sub-section under 'Performance of Methods', we can clearly see how some of the features have more impact in predicting a user's intentions than others. These "high-priority" features can be further exploited and used to filter out redundant features and form a more condensed model by grouping related and low priority features into one group.

10 MEMBER CONTRIBUTION

Each member was expected to – and did – attend group meetings, whether physically or virtually, throughout the course of this project. In addition each member was expected to aid in writing and proofreading all project submissions. Below are the specifics each member did outside of the aforementioned expectations.

Kaitlin was solely responsible for training and testing the SVM algorithm. This algorithm, while implemented correctly, achieved a very low accuracy and a lower NDCG score in comparison to the other models, so it was not chosen as one of the selected models. She was also responsible for all project submissions.

Namrata worked on first summarizing the web session data set from over 10 million entries to about 150000 entries. She also worked on consolidating the two data sets together and preprocessing the consolidated data set. Lastly, she was solely responsible for training and testing the Random Forest algorithm.

Jonatan was solely responsible for training and testing the Neural Network algorithm. He also worked on analyzing the training dataset as well as feature extracting for preprocessing the dataset.

Himanshu performed exploratory data analysis on the training data set and was also responsible for carrying out feature engineering, pre-processing and consolidating user and web session data sets used in the XGBoost model. In addition, he was solely responsible for training and testing the XGBoost algorithm.

REFERENCES

[1] Shah, B. Travel Destination Prediction for Airbnb.

[2] Wikipedia contributors, Rectifier (neural networks) — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Rectifier_(neural_networks)&oldid=923288576, 2019; [Online; accessed 7-December-2019].

[3] Wikipedia contributors, Softmax function — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Softmax_function&oldid=928536872, 2019; [Online; accessed 7-December-2019].

[4] Wikipedia contributors, Stochastic gradient descent — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Stochastic_gradient_descent&oldid=929421698, 2019; [Online; accessed 7-December-2019].

[5] Avireddy, S.; Ramamirtham, S. N.; Subramanian, S. S. Predicting Airbnb user destination using user demographic and session information. *UC San Diego* **2015**,

[6] Ulfsson, H. Predicting Airbnb user's desired travel destinations. 2017.

[7] Zhang, K.; Pan, Z.; Shi, S. The prediction of booking destination on Airbnb dataset. *UC San Diego. http://cseweb. ucsd. edu/~ jmcauley/cse255/reports/fa15/03 8. pdf*