The Prediction of Booking Destination of New User on Airbnb

Chien-Wei Lin, Po-Kang Huang

April 25, 2017

Abstract

New users on Airbnb can book a 34,000+ cities across 190+ countries. By accurately prediction where a new user will book their first travel destination, Airbnb can share more information with their community, decrease the average time to first booking. This project aims to analyze Airbnbs dataset and design a machine learning flow to predict booking destination for new users on Airbnb. The task was assigned by Kaggle competition, which also provided training and test dataset for this project. It is an important real-life practical problem and can be implemented by various ways. Participants in this project will learn how to turn real-life information into instances suitable for machine learning, and learn to design models able to handle imbalanced data.

Introduction

The goal of this project is to predict the most possible booking destination country given the information of a particular user on Airbnb. In this project, we start from observing and analyzing the given dataset, trying to find out useful features for learning and prediction. Then we design the models appropriate for this problem and select those with the best performance. Since it is a project competition on Kaggle, we also learn the models and learning processes used by several winners in the competition. In the data pre-processing part, some previous materials show that how to transfer real-life data into training data. For the model selecting part, multi-layer models are also recommended to handle imbalanced data in this project.

Methodology

1. Dataset
   (a) Description of dataset
   The dataset provided by Airbnb is a list of users information, including their demographics, web session records, and some statistics data. The whole dataset contains 213451 samples. The properties contain: *id*, *date-account-created*, *date-first-booking*, *gender*, *age*, *signup-method*, *signup-flow*, *language*, *affiliate-channel*, *affiliate-provider*, *first-affiliate-tracked*, *signup-app*, *first-device-type*, *first-browser*, *country-destination*, *timestamp-first-active*.
   (b) Observation and analysis of Data
      i. Country Destination
      Based on Figure 1, we found that 58% of the users ended up booking nothing, which indicated as NDF (no destination found) in *country-destination*. Furthermore, there are 97.97% of those users who did not booked in the Airbnb did not have the record of *date-first-booking*. Therefore, we turn to the users who had booked in Airbnb to ensure the effectiveness. The result is that all the users who did not book in Airbnb (missing *date-first-booking* record) have NDF on the *country-destination*.

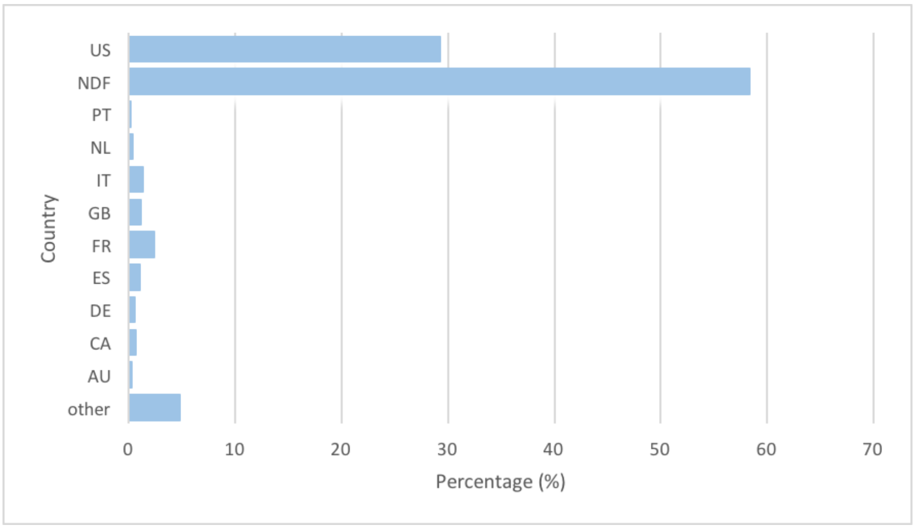

Figure 1: Distribution of the destination countries

ii. Range of Age
From Fig. 2, we can observe that most of the users who booked on Airbnb are in the range between 20 to 45 years old.

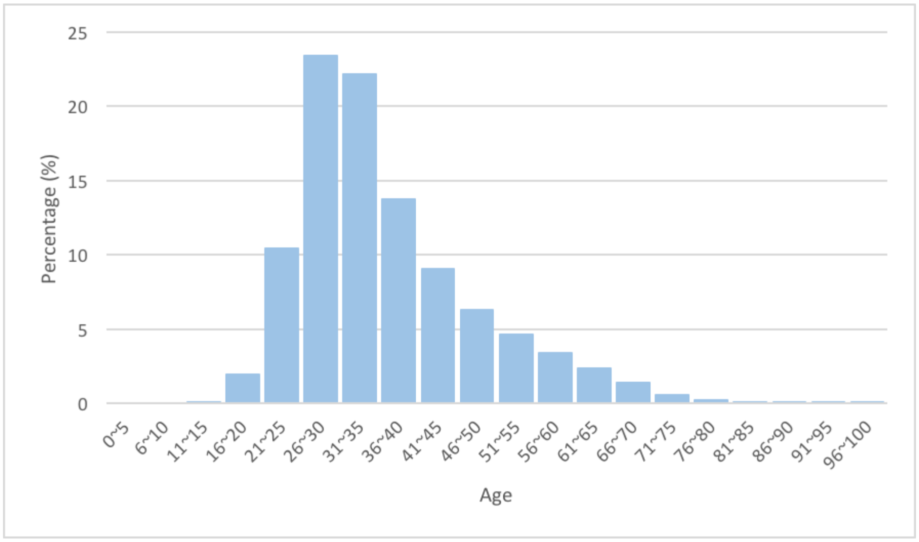

Figure 2: Age of users without missing data and outliers

(c) Data pre-processing

i. One-Hot Encoding
One hot encoding transforms categorical features to a format that works better with classification and regression algorithms. For all features in our data, they are belonging to some categories. We can encode those to normal values as Table 1. However, this would not make sense from the machine learning perspective. We cannot say category of Female is greater or smaller than Male.
It will not be proper to assign 0, 1, 2, 3 to represent male, female, other, and unknown, respectively. Instead of using numerical value, we translate this feature from 1-dimension vector into 4-dimension vector. Each dimension will be 1 if it matches the description of the feature, and will be 0 otherwise (show as Table 2). For most of our features, we translate each 1-dimension features into multiple

Table 1: Normal values.

Sample	Gender	Value
0	Male	1
1	Female	2
2	Other	3
3	Male	1

dimension binary features. This seems to work well with most machine learning algorithms.

Table 2: One-Hot Encoding values.

Sample	Male	Female	Other	Unknown
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	1	0	0	0

ii. Age
We first represented the age with its number directly, which is a trivial result. Then, we found out that it was the age distribution which matters. From Fig. 2, we assume that children and seniors have less probability to use the Airbnb to plan a trip. After discarding the data with age < 5 and age ≥ 100, we create 19-dimension vector, which counted the index as $\frac{age}{5}$.

iii. Royalty of Using Apple Product
We first treated these three features, *first-device-type*, *first-browser*, and *signup-app*, separately, which did not improve our performance. After analysis, we noticed that they have some relations when it comes to Apple product, such as iPhone, iPad, Macbook, or Safari. As a result, we shrunk the three features into one feature. If one of the features contains the iOS, Mac, or other Apple related product, then we set this feature to 1, or 0 otherwise.

iv. Group Features
Instead of using the original language, we grouped the language according to the area of origin and linguistic, such as Asia language, North Germanic, and Latin. We also compared the date difference between the *account-created*, *date-first-booking*, and *date-first-active.* The total features in our data set is 91.

2. Learning Model

(a) Model selection
One can notice that the distribution of destination country in the whole dataset is extremely unbalanced. There are two destinations countries account for a large proportion of the dataset, NDF (no destination found) and US. NDF account for about 58%, and US account for about 29% of the whole data. We would get a poor performance if we simply use a single-layer multi-class classifier to predict the destination country.
Moreover, 97% users with NDF have no records of *date-first-book*, so we have a strong confidence to predict NDF for all users whose date-first-book is missing. By this reason, we directly predict users without *date-first-book* to NDF, and then focus on the remaining data with date-first-book.
The remaining data has a large proportion whose destination countries are US (70%).

We develop a two-layer classifier to handle this. The first layer is a binary classifier, whose goal is to distinguish the destination country between US and non-US. The second layer is a multiclass classifier, whose goal is to predict the most 5 possible destination countries for each user whose destination countries are non-US.

i. Binary Classifier
The binary classifier consists of SVM, Naive Bayes and logistic regression models. We trained the three models by cross-validation separately, choosing the one with the best performance, and the prediction is made by majority voting of the three models. We first tried using linear regression model, but the linear regression is not appropriate for binary classification, because the linear regression model assumes that the outcome is continuous, binary outcomes obviously violate the assumption. The reason why not using polynomial regression is it is time-expensive.

ii. Multiclass Classifier
For the remaining non-US users, the multiclass classifier will predict the top 5 possible booking destination countries for each user. We use SVM and logistic regression model together as our multiclass classifier. For the SVM model, we directly calculate the probability of each destination countries for a user. For the logistic regression model, the one-vs-all classification is used to calculate the probability of each classes. Then we could get 10 probabilities of each countries by normalizing the probabilities of all the classes for a user. Combining two lists of probabilities from two models for a user, we sorted the probabilities and pick 5 countries with highest probabilities as its prediction outcome.

(b) Evaluation model

i. Binary Evaluation
For the binary classification, where we try to separate US-users and non-US-users, we simply give the score of 1 to the user whose prediction country matching the ground truth country, otherwise, we give the score of 0 to the incorrect predictions.

ii. Multiclass Evaluation
For the multiclass evaluation, we follow the evaluation metric for the Kaggle competition, NDCG (Normalized Discounted Cumulative Gain) @k where k=5. The NDCG evaluation is shown as:

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

$$nDCG_k = \frac{DCG_k}{IDCG_k} \tag{2}$$

where rel_i is the relevance of the result at position i, in our case $rel_i \in \{1, 0\}$. The term $IDCG_k$ is the maximum possible ideal DCG_k for a given set of queries. All $nDCG_k$ calculations are relative values on the interval $\{1, 0\}$. Using this evaluation metric, we predict a list of 5 destination countries with highest probabilities, and compare the countries, in order, on the list with the ground truth country to calculate its evaluation score. The ground truth country is marked with relevance = 1, while the rest have relevance = 0. For example, if for a user with the prediction list $\{US, FR, GB, CA, AU\}$ and the ground truth country is FR, the evaluation score becomes:

$$DCG_5 = \frac{2^0 - 1}{log_2(1+1)} + \frac{2^1 - 1}{log_2(2+1)} + 0 + 0 + 0 = 0.6309 \tag{3}$$

The maximum score for each user is 1 and minimum 0. The score is determined by the order of the correct prediction country in the prediction list. If the correct prediction is at first position, the score will be 1, and the score decreases if the position of the correct country goes larger. This method is often used to measure effectiveness of web search engine algorithms or related applications, which avoids the diaadvantages of arbitrarily giving the score 1 or 0.

(c) Cross validation and testing data set
After removing the instances without *data-first-booking*, we segmented 15% of the total data as the testing set and the remaining data as training set shown in Fig 3.

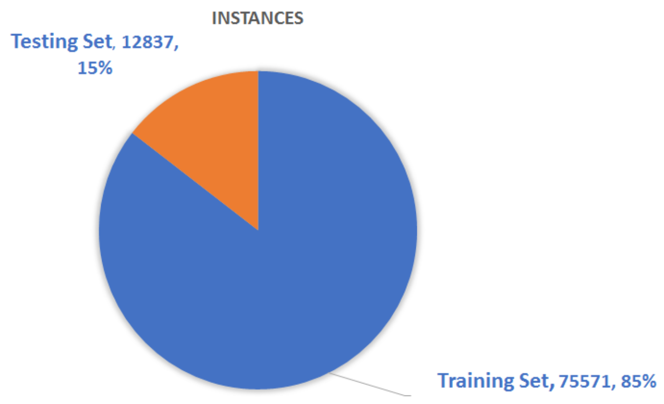

Figure 3: Ratio of testing instances and training instances.

To evaluate and validate our model, we use the 10-fold cross-validation. In this way, the training data set can both be used to train and test our model.

(d) Bootstrap aggregating
For both the binary classifier and multiclass classifier, we use bagging to increase the accuracy with 9 bags. The number of instances in each bag is followed by Gaussian distribution.

(e) Description of entire model
From above, our system can be shown as Fig. 4, which consists of three main parts: pre-processing, binary classification and multiclass classification.

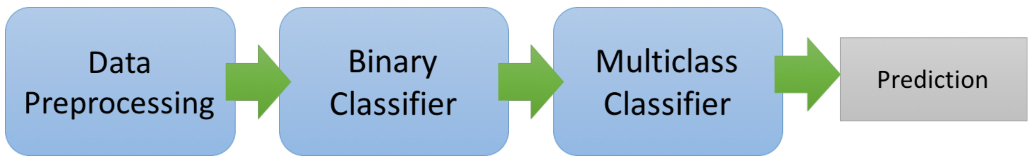

Figure 4: The flowchart of our training (and predicting) model.

For the data pre-processing, we analyze the data distribution and search for confident relationships between features and destination countries.
The goal of binary classifier is to separate US-users and non-US-users. To improve the performance, we also used bagging algorithm for our binary classifier. The bagging (shown as Fig. 5) consists of several sub trainers, which have randomly different numbers of training data followed by the Gaussian distribution. Each small binary trainer is composed of three models: SVM, logistic regression, and Naive Bayes

(shown as Fig. 5). The majority voting is used to the predict outcome of each binary trainer.

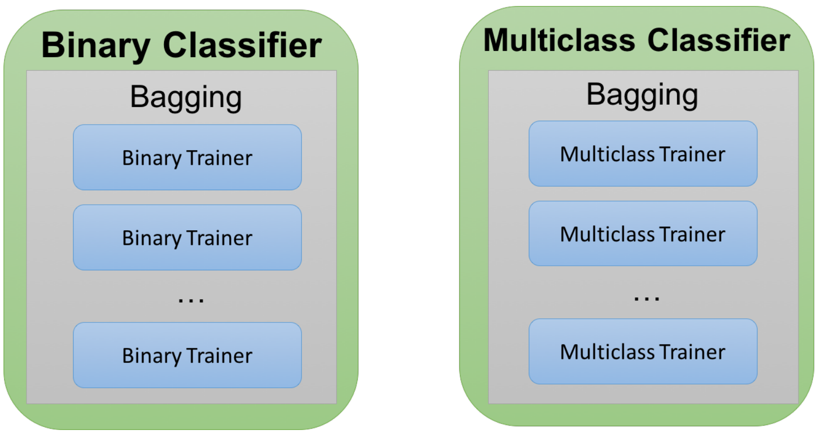

Figure 5: Binary classifier and Multiclass classifier.

The multiclass classifier is used to predict the destination countries of the non-US users. The bagging algorithm is also used for multiclass classifier. Each small multiclass trainer in the bagging consists of two training models: SVM and logistic. Each model will predict a probability list of all the destination countries, and top 5 countries with highest probability will be chosen as result (show as Fig. 6).

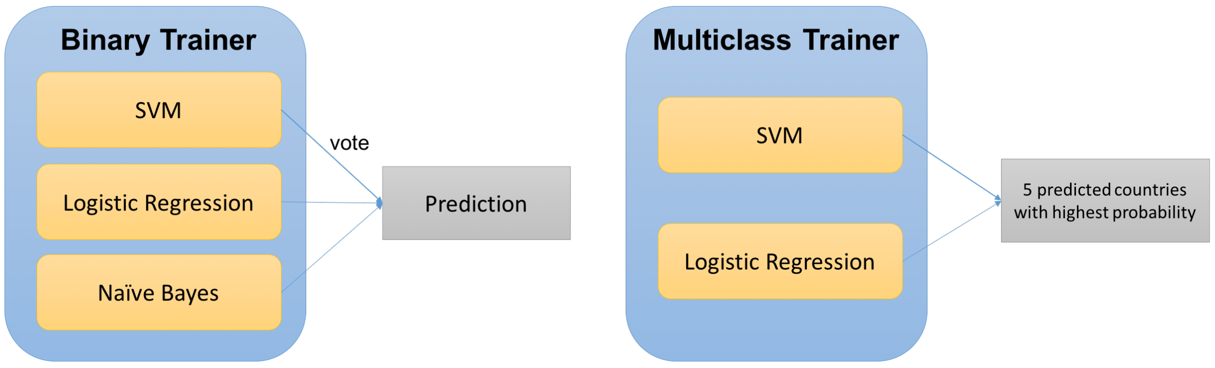

Figure 6: Weak Binary trainer and Multiclass trainer.

In addition, we also used 11-fold cross-validation to average prediction of a single model in binary Trainer and Multiclass Trainer. For each SVM, logistic, and NB, we have 11 different learners. The prediction will be based on the voting results from 11 different learners.
We used Scikit-learn, which is an open source Python library for machine learning, to implement SVM model, logistic regression model, and Naive Bayes model in this project.

Results

1. Binary Classification
   The binary classifier consists of several weak binary trainers, and each trainer consists of three learning models. The accuracies of the weak binary trainers are from 48% to 62%,

which have a large variance. The overall accuracy of the big binary classification layer is 60.04%, and it can reach 64.28% after we apply up-sampling for all training procedures. The validation data used here is 50% US-users and 50% non-US-users, so the accuracy is higher than random guess, and the evaluation is simply calculating the rate of correct predictions.

2. Multiclass Classification
   The multiclass classifier is composed of several weak multiclass trainers as well. Each multiclass trainer consists of two learning models. The accuracies of the weak multiclass trainers are from 42% to 58%. The overall accuracy of the big multiclass classification layer is 56.31%. There are total 10 classes in the validation data, and the class with the largest rate in the validation data is less than 32%. The evaluation model used here is NDCG.

3. Overall
   The overall accuracy is 91.71% for the entire learning model. The prediction of NDF is simply determined by whether a user has *date-first-active* or not, and the amount of users missing *date-first-active* accounts for 58.34% of all the dataset. For the users with *date-first-active*, where US-users accounts for 70.14%, the accuracy is 80.09%. NDCG is used for evaluation here.

Discussion

1. Class Imbalance (70% US-users v.s. 30% non-US-users)
   (a) Disparity in Frequency of the Observed Classes
   Since the performance of binary classifier will influence the outcome of multiclass classifier in our model, first step was to increase the accuracy of the binary classifier. At first, we had 70% validation accuracy on US/non-US binary classifier. To have better result, we used cross-validation and bagging. Other than increased the accuracy, the performance became worse. Based on this observation, we noticed that our destination prediction on the validation set were all US. The 70% accuracy was just the percentage of US in validation set.
   (b) Up-Sampling and Down-Sampling
   In our classification problem, the class imbalance had a significant negative impact on model fitting. Our solution for this problem was to use up-sampling, which randomly sampled the instance from minor class with replacement to have the same size of the majority class, and down-sampling, which randomly select the subset of the majority class to have the same size of minority class.

2. Overfitting
   (a) Ensemble Learning
   Due to our training accuracy had out performed the validation accuracy, we tried to replace our original model with ensemble methods, such as Random Forest, Ada Boost, Gradient Boost, and XGBoost. Unfortunately, the training accuracy improved, but not the validation accuracy.
   (b) Insufficient Feature Information
   It was reasonable to suspect our data is not separable. To confirm this suspicion, we tried to find the Support Vectors in the SVM. Not surprised, 99% of our instances were support vectors. Consequently, SVM somehow found a hyperplane that could separate the data, but it did not make any sense to other validation data. This

made our task to classify become difficult. To fix this problem, we revised our feature and add more information with group features which we mentioned in data-preprocessing. Dimension of our data increased from 79 to 91. Surprisingly, the percentage of support vectors dropped to 80% after adding those features, and the accuracy improved as well.

Conclusion

1. Performance
   In binary classification, the accuracy of each individual model is about 50% to 60%, which is just slightly better than random guess (50%). The accuracy could be even worse before we apply up-sampling into training, the accuracy could be just equal to the random guess. Even we apply ensemble algorithms into learning procedure, the improvement of the performance is still limited. The possible reasons for this problem could be in the feature selection. The feature vector we used might lack the critical information to determine which destination countries a user would choose. We observed the parameters of trained models, and found that parameters with higher weights correspond to the age features. It could be elaborated that younger people tend to go to US and middle-aged people tend to travel to non-US regions. But other features with high training weight are hard to find consistence and have large variance among different sections of training data.
   The accuracy of multiclass classification is 56%, which is much higher than the random guess. According to the NDCG evaluation, we can say that we are able to predict the correct country among the first three predictions.
   Considering both binary and multiclass classifiers, the accuracy would be higher than 80% for the users with *date-first-booking*, which means it is highly probably at least one prediction would be correct among the first two prediction countries. This performance could be useful enough for the affiliate provider to share useful information to users.

2. Future Directions
   The features have significant effects on our model performance. It might be helpful if we find some other connection between the raw data. For example, the age might have a strong relationship with the type of device used by users, because teenagers have a longer average time using smart phones or tablets than elder people. Further, combing few features together could bring some useful information. For another example, the difference between the date of users account is created and the time when the account session is active could represent some other hidden information for the users decision. If we could find out these hidden information, the accuracy could be better. We can also try some other feature selection technique to reduce our feature dimensions.

Individual effort

1. Chien-Wei Lin
   Chien-Wei is responsible for features extracting, data pre-procession, cross validation, and Bootstrap aggregating algorithm.

2. Po-Kang Huang
   Po-Kang is responsible for implementing binary classifier and multiclass classifier, evaluation models, and Bootstrap aggregating algorithm.

References

[1] Nabil Abdellaoui. *Airbnb Recruiting: New User Bookings.*

[2] Jason Brownlee. *Bagging and Random Forest Ensemble Algorithms for Machine Learning.*

[3] Tom Fawcett. *Learning from Imbalanced Classes.*

[4] Kaggle.com. *Airbnb New User Bookings (https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings).*

[5] Sichao Shi Ke Zhang, Zhengren Pan. *The Prediction of Booking Destination On Airbnb Dataset.*

[6] Yannick Kimmel Michael Winfield Rob Castellano, Zi Jin. *Airbnb New User Bookings.*

[7] Mehreen Saced Saba Arslan Shah. *Predicting Purchased Policy for Chustomers in Allstate Purchase Prediction Challenge on Kaggle.*

[8] Karlijn Willems. *Python Machine Learning: Scikit-Learn Tutorial.*