Airbnb New User Bookings

Team: SotA

Karthik Hegde
IMT2018509
IIITB
karthik.hegde@iiitb.org

Jishnu V K
IMT2018033
IIITB
vinodkumar.Jishnu@iiitb.org

Aman Kumar
IMT2018006
IIITB
aman.kumar@iiitb.org

I. Introduction

In this challenge, Airbnb wants us to predict in which country a new user will make his or her first booking. Predicting the region of booking will help provide better content to the users with regard to their choice of destination, decrease the average time to first booking, and better forecast demand, thus benefitting both the customers and Airbnb.The revenue for Airbnb increases and also the customers get satisfied by their service.

We are given a dataset and using classical ML models we need to predict the top 3 countries where the user will book his flight. The dataset is collected for the US citizens.

The countries which user books are as follows:-

- US
- PT(Portugal)
- NL(Netherlands)
- IT(Italy)
- GB(United Kingdom)
- FR(France)
- ES(Spain)
- DE(Germany)
- CA(Canada)
- AU(Australia)
- Other(other countries)

II. Dataset

We are given 5 .csv files as dataset to help us predict the top 3 countries where the user is most likely to book.

The 5 .csv files are as follows

- age.csv
- countries.csv
- sessions.csv
- test.csv
- train.csv

The train data frame has 170137 records and 15 features plus one target column(country destination).

The sessions dataframe has nearly 1 crore records describing the activity of users.

The ages dataframe describes about the people in that particular country destination.

The country dataframe describes about the country destination like area of the country, it is distance from US etc.

Finally we have a test dataframe where we need to predict country destination for 43314 records.

We have columns like

- user_id: The id of the user who is using the site.
- date_account_created: The date when the account was created.
- timestamp_first_active: time stamp
- date_first_booking :The date when user booked first time.(This columns was useful in seperating out the NDF countries).
- gender: The gender of the user is mentioned.
- age: The age of the user is mentioned.
- signup_method: The signup method of the user is mentioned.
- signup_flow: The signup flow is mentioned.
- and the other columns are affiliate_channel, affiliate_provider, first_affiliate_tracked, signup_app, first_device_type, first_browser.
  Using these columns we need to predict the top 3 destinations for the users in the test dataframe.

III. EDA

We have done EDA on the dataset given to us and found interesting results which helped us predict the top destinations for the users.

The results we got from EDA are as follows

- Let us see the ages of people in the given countries. We see that it follows similar to gaussian distribution with Male population more than females till age 50-54 and then women live longer.

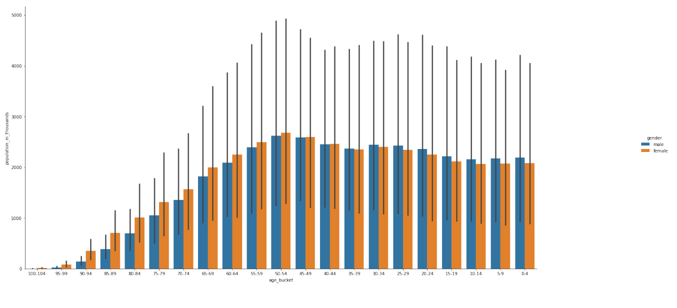

Fig. 1. Ages of people

- We see that Australia is farthest from USA and Canada is nearest to USA.
- We see that portugal is farthest in terms of language distance followed by other european countries except United Kingdom.
- We see that Canada is the biggest country in terms of area followed by US and Australia.
- We see that most users just explored the site and didn't book as NDF takes the first spot.

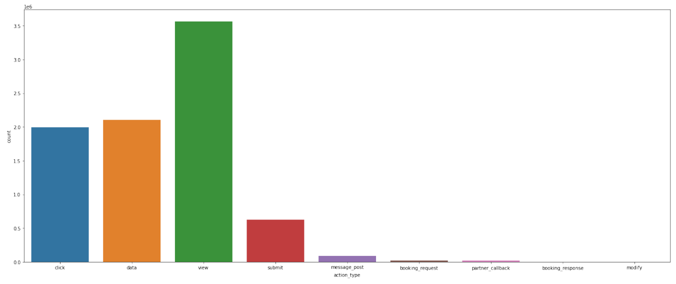

Fig. 2. Action of users

- We can see that Desktop is used more and then Phones. Also Apple users are more in case of Desktop and phones.These are the device types used by the users.
- We see that basic is the most preferred sign_up method and then FB.
- We see that Web is most preferred for signup and then comes iOS and then Android and Moweb.
- We see that direct is the most common affiliate_channel.We also see that direct is the most common provider and then comes google.
- It seems like Spain is preferred most by younger people Also United Kingdom is preferred most by older people.So age plays an important role in determining the country destination.
- We can see that only for Germany, Canada and 'Other', Males prefer them more over females.So there is gender bias among the countries.

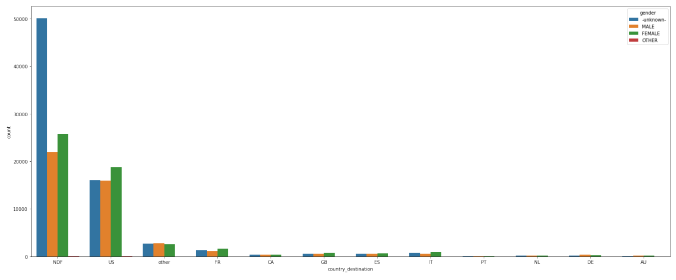

Fig. 3. Preference by males and females

- There is no bias between sign_up method, sign_up app and device type used by the user.
- We also see the popularity of countries that is the most common destinations booked by users.

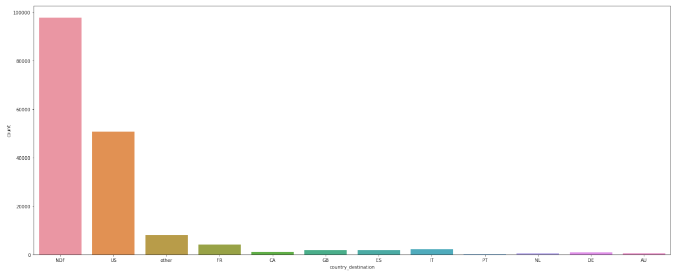

Fig. 4. Popularity of countries

The above inferences from the dataset has proven useful for us to select the features for training the model to get accurate predictions.

IV. Pre-processing & Feature Extraction

Before pre-processing, we separated the instances which belonged to the NDF class from those belonging to the non-NDF class. The pre-processing and feature extraction and training were essentially done only on the instances that belonged to the non-NDF class. The reason being that missing values of 'date_first_booking' is guaranteed discriminator between NDF and non-NDF.

The pre-processing steps include Missing value treatment, treating outliers, combining the categorical classes that have a small frequency count, and followed by Encoding of categorical features. We tried extracting feature 'month' from the feature - 'date_first_booking' - the primary reason being that some countries are more likely to be visited on some particular season. We also tried extracting features from 'sessions' data.

Following the conclusions from EDA, we decided to train the model on the features - 'gender', 'signup_method', 'signup_flow', 'language', 'affiliate_channel', 'affiliate_provider', 'first_affiliate_tracked', 'signup_app', 'first_device_type', 'first_browser', 'month', 'age'. The missing values in the 'age' were filled with value 30. The missing values in 'first_affiliate_tracked' were filled with value 'none'. The outliers in age were discarded. The age limits were above 18 and below 80. The outliers in 'age' of test set was changed to value of 30.

The categorical columns - 'gender', 'signup_method', 'signup_flow', 'language', 'affiliate_channel', 'affiliate_provider', 'first_affiliate_tracked', 'signup_app', 'first_device_type', 'first_browser' were all either label encoded or one hot encoded. For logistic regression and SGD continuous features - 'month', and 'age' were Normalized.

Before encoding the categorical features we observed that many features had some categorical classes that had very few counts. We combined such categorical classes. But the combinations of unique categorical classes, used for training, was regarded as yet another hyperparameter. Therefore we changed these combinations several times to see improvements over time. Even the features used for training were changed several times checking for improvements. We first trained our model using one-hot encoding and later we switched to label encoding because the label encoding scheme was performing better, and then again we tried switching as per the improvement in the performance.

As a feature engineering, we tried the polynomial feature transformation. we took the top 10 to 12 features that were regarded as important by our earlier trained XGBoost model and fed them to a polynomial feature transformer of degree 2. We also tried with degree 3 but this was not performing so well. The list of features, from feature importance, was changed several times according to changes made in the encoding scheme and the categorical class combination.

V. Feature extraction from sessions.csv

We first imputed the columns in 'sessions' data. The categorical columns were imputed with a new value - 'Missing' and the continuous column ('secs_elapsed') was imputed with the mean value. This data was grouped according to different 'user_id' and applying different functions on the available columns. For categorical columns - 'action', 'action_type', 'action_detail' - we took count of unique values, and total count. For continuous column - 'secs_elapsed' - we took count of unique values, total count, mean, median, mode, standard deviation, the measure of skewness, the measure of kurtosis, minimum value, maximum value, total sum of the values. The table so formed was merged with the train and test dataset. With the train set, this was merged using the inner join method and with the test set, we merged it using the outer join method. The train set so formed, after using inner join, shrunk the training data available to 11994, which is much smaller than the test set size (16578). To cover this up we also tried smote - oversampling all the classes. The train set so formed was then cleaned by treating the missing values. The features which had less than 500 unique values were classified as categorical features and the rest as continuous features. The continuous features were imputed with the corresponding mean values. The categorical features were imputed with the corresponding mode values. The cleaned data, including earlier chosen features, was then trained without smote and with smote applied. The one without smote did perform well with the public leaderboard score of 0.92183 and the private leaderboard score of 0.92267. But the one with smote applied was heavily overfitting (private leaderboard score of 0.89649).

Another strategy employed later was using the different action types to try to boost the scores of the model. Action types like "south-europe", "rest-of-world", "north-america" were present which could help us. However, since the intersection of sessions data and the train set was very small, we could not reliable train on them. Finally, we decided to use only action types related to google translate to train as these features had a better intersection with train data.

VI. Model Selection, Cross Validation and Training Process

Initially, we started of with simpler models like logistic regression. These models suffered from the imbalanced nature of the dataset preventing them from learning anything useful. We decided to switch to decision tree based models since they handle imbalanced data better. Models, we tried are LGBM, XGboost, adaboost, random forests with and without application of smote. XGboost gave us the best baseline accuracy so we decided to go ahead with hyper parameter tuning.

For cross validation, we initially started by using a hold out validation set. We soon switched to k-fold cross validation as we were able to get better results using this. With k-fold cross validation, we applied a grid-search CV to find an approximate value to the best hyperparameters and manually fine tuned them around these values(at this stage our best model was scoring 0.92185 on the public leaderboard).

Later we also experimented with bayesian optimization of hyperparameters, which provided a faster convergence to the best hyper parameters. We also switched from k-fold CV to time-series split which we felt was better due to the chronological nature of the dataset(at this stage our best model was scoring 0.92197 on the public leaderboard).

After we finished tuning XGboost, we decided to try model stacking. Models included for model stacking were adaboost, xgboost, LGBM, random forest, KNN, SGD. We tried many combinations but none of them were able to out perform the tuned XGboost model(the best stacking model score 0.92174 on public leaderboard).

The model(xgboost) trained on the sessions data performed very well in the public leaderboard but did not generalize well to the private leaderboard(public lb:0.92231 private lb:0.92251).

Out of all the models, the final model was xgboost trained on language, signup flow and affiliate data using one hot encoding. It scored 0.92197 on the public leaderboard and 0.92268 on the private leaderboard.

VII. Conclusion

By varying the features used, the feature engineering and the model families, we were able to try a wide range of models. Each of these had varying success in terms of accuracy and generalizability.

In the end, we were able to create a fairly accurate model for the task. The use of machine learning to predict the future actions of customers can prove useful, not only for companies like Airbnb, but also to the broader range of companies involved in businesses like food delivery, online fashion and groceries etc. Studies using machine learning can be used to save costs, advertise more effectively and much more to maximize profits.

Overall, by undertaking this project, we have learnt a great deal in practical machine learning. The imbalanced classes, noisy data etc posed many challenges that we feel have provided us with valuable experience. It has encouraged us in delving deeper into the subject and learn more about the specialized fields of machine learning.

VIII. Acknowledgement

We would like to thank Professor Srinivasaragavan and Professor Neelam for giving us a thorough introduction to the field of machine learning. The overview of the various sub divisions of machine learning and the detailed explanation about how things work generated lots of interest and helped us gain clarity.

We would also like to thank the teaching assistant team, for the many great sessions on practical machine learning that helped us learn how to effectively go about our own ML projects.

Special thanks to our TA, Tanmay Jain for guiding us through the project and giving us useful feedback and encouragement throughout the course of this project. It helped us a lot.

REFERENCES

[1] Documentation for XGBoost. Link: https://xgboost.readthedocs.io/en/latest/python/python_api.html
[2] Analytics vidhya article on XGBoost tuning.
[3] Documentation and user guide on scikit-learn cross validation
[4] Medium article on Bayesian optimization for efficient hyperparameter tuning.
[5] Time series split Link: https://towardsdatascience.com/time-series-nested-cross-validation-76adba623eb9
[6] LightGBM documentation. Link: https://lightgbm.readthedocs.io/en/latest/