Predicting first user booking destination, an Airbnb case study

Tiago Craveiro. Jorge Ferreira

1 University of Porto. FEP. 4200-465 Porto. PT

up2009008673@fep.up.pt Up201204029@fep.up.pt

Abstract. On this work it's requested that a prediction model is created to predict which will be the first destination that a given traveler is going to select after its sign up. The focus of this report will strongly rely on feature engineering techniques and its powerfulness on what concerns prediction accuracy and efficiency. The analysis will firstly be focused on data exploration where the datasets will be described, data cleaned and some context will be retrieved from the main features. Then the core study of this report will be on the feature engineering techniques, where categorical data will be encoded and some feature manipulation will be done either by creating new features from the original dataset or the elimination of non-relevant data. This is a dataset with an extended number of features and rows and therefore the manipulation of this data demonstrated key to achieve better prediction accuracy. Lastly some learning models were applied namely the AutoML pipeline, name H2O, and specifically XGboost tree with some hyper parametrization optimization using grid search. In the end it was possible to achieve 87,4% of accuracy on predicting the destination selected by first time user of Airbnb.

Keywords: *Airbnb, feature engineering, hyper-parameters optimization.*

1. Problem context

The use case addressed on this report is based on a Kaggle competition that presents the problem of predicting the destination of first-time travelers of a short-time accommodation startup called Airbnb.

Airbnb is an online marketplace that connects people who want to rent out their homes with people who are looking for accommodations in that locale. It currently covers more than 100,000 cities and 220 countries worldwide. New users on Airbnb can book a place to stay in 34,000+ cities across 190+ countries. By accurately predicting where a new user will book their first travel experience, Airbnb can share more personalized content with their community, decrease the average time to first booking, and better forecast demand.

The objective of this work is to achieve the highest possible accuracy on predicting the destination country of the first-time travelers by focusing mostly on feature engineering techniques and the impact that those have on getting better predictions. The prediction variable is the destination country and the main metric of evaluation to be used, since is the one considered on the competition, is the accuracy of the model.

The problem is composed by 4 train distinct dataset: the train dataset with 15 distinct features (10 categorical and 5 numerical) and 213.451 observations about users and its characteristics such as gender, language, device user for booking a travel, among others. A second dataset about user sessions with 5 features (4 categorical and 1 numerical) that records all interactions made by users on Airbnb's digital applications such as navigation, search that was linked with the first dataset by the user id. The sessions dataset had more than 10 million sessions stored. Lastly the remaining two datasets were about countries and age gender distribution figures. These details are shown aggregated on the figure bellow.

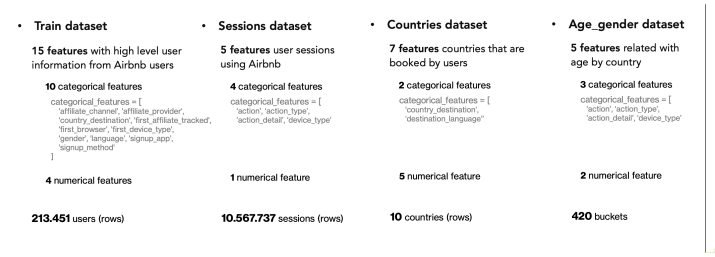

Figure 1 - Problem dataset categorization

The test dataset that should be used for submission on the Kaggle competition had 62.096 entries with users that never booked a travel with Airbnb and, therefore, the destination was unknown.

On the next chapter we will present the data exploration, then on the third chapter feature engineering techniques will be studied in depth, to then on chapter 4 focus on the machine learning modelling and results evaluation. Chapter 5 contains the conclusions of the work and the next steps.

Commented [GU1]: A imagem está com o sublinhado a vermelho dos erros

Commented [TC2R1]: done

2. Exploratory analysis

In this section it will be described the main features of the dataset starting with the characterization of the prediction variable: country destination. This variable is a categorical one that represent a list of possible destination countries that a given booking can have. There are ten main countries as displayed on the figure bellow, then all destination with less frequency are considered "other" and finally there is a label used for travel users that never booked a travel with Airbnb "NDF – not booked". As it can be observed on the figure bellow there is a quite significant unbalance between classes of this variable with a considerable higher frequency of for the label "NDF" followed by United states.

Country destination
AU
CA
DE
ES
FR
GB
IT
NL
PT
US
NDF – not booked
Other

Figure 3 - List of possible country destination classes

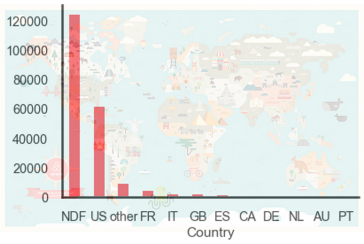

Figure 2 - Distribution of classes in the country destination variable

In the train dataset the numerical variable age has 42% of the observed instances with empty values. This variable also contains some outliers as shown on the density graph below, with a strange occurrence of ages around 2000 years.

Statistics	
count	125461
mean	49.668
std	155.66
Min	1
25%	28
50%	34
75%	43
max	2014

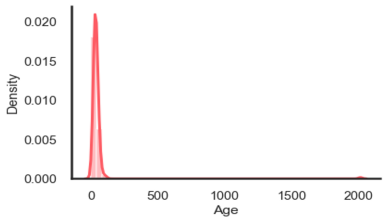

Figure 4 - Statistics table and Density distribution graph

In order to do some data cleaning, outliers where removed and it was transformed ages higher than 100-year-old and bellow 18 years old into NA values, the first since the expected life longevity is lower than that threshold and the second since travelers below 18 years old are not allowed according with Airbnb policies on age limiting. The output can be seen on the graph bellow.

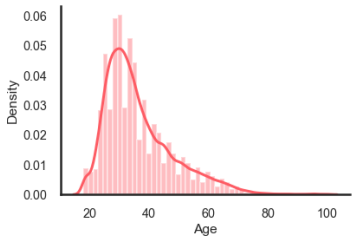

Figure 5 - Age Density distribution plot after data cleaning

Looking at other feature from the train dataset, the gender, there are four distinct values: “Male”, “Female”, “Other” and “nan”. The majority of the observations, around 44% have nan value. Despite that, there seems to be a balance between the Male and Female class and displayed on the figure below, with a residual amount of people with “other”.

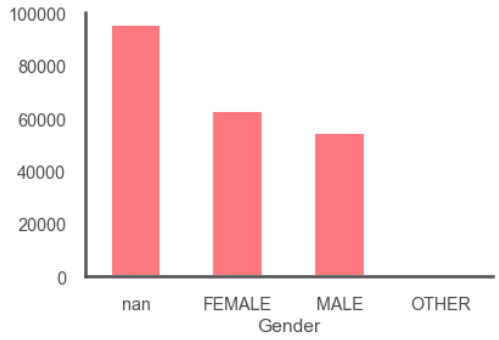

Figure 6 - Class distribution of the Gender feature

The dataset also contains 3 distinct variables with dates: the date of account creation, the date of first activity and the date of first booking. One of those dates is in Unix timestamp (the first activity), while the others are in date format. To the variable in UNIX timestamp, it was applied conversion methods to date-time in a new variable, as well it was made sure that the original variable was numeric.

Looking at the date account creation variable on the chart below, it is displayed its distribution of new accounts created per day. By analyzing it, one can conclude that the number of new user accounts have constantly and significantly increased from 2009 until 2014.

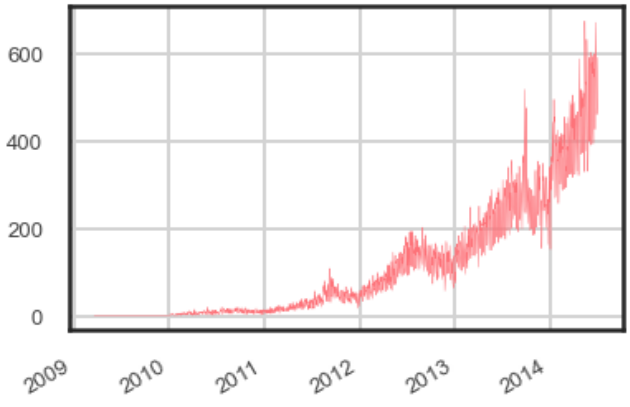

Figure 7 - Distribution of user account creation by year

3. Feature Engineering

Less frequent classes aggregation

There were several categorical variables with a long tail of observations in some classes. This might be quite impactful when modelling and predicting the country of destination since categorical variables need to be converted into numerical ones by using encoding techniques, since most of the models only deal well with numerical data type. If this long tail classes were converted into dummy variables using One-Hot encoding, for example, the number of variables would increase a considerably and the dataset would be really scattered. Therefore, classes with a frequency lower than a threshold of 200 were grouped into "other" class, as it can be seen on the language variable in the figure bellow that before had 25 distinct classes and after this data treatment just 11.

Figure 8 - Group less frequent classes into "other" in Language feature

It is possible to see that the most frequent language used on the app is the English accounting for more than 96% of the users on the training set.

Categorical variable	Number of classes before	Number of classes after
language	25	11
affiliate_channel	7	7
affiliate_provider	18	12
signup_app	4	4
first_device_type	9	9
first_browser	52	10
signup_method	3	3
first_affiliate_tracked	7	6
gender	3	3
Total	**128**	**65**

Figure 9 - Table with feature reduction after applying grouping to less frequent classes

Other variable with a considerable dispersion in terms of values is the first browser. The most common browser is chrome with 63845, followed by Safari with 45169, Firefox and IE are also browsers with a big concentration. There are a considerable number of users with unknown

browser: 27266. Some examples of browsers aggregated on the class “other” are Pale Moon, and SeaMonkey with only 12 and 11, users respectively.

Missing values

By using One-Hot encoding to transform the categorical variables into numerical dummy variables, the missing data on those columns is ignored. Since that will be the strategy of encoding the only concern is regarding the numerical variables like age. Here, there are several options like replacing for a constant, using a central metrics to replace, or creating intermediate regressors (like k-NN) to predict the value of the missing data and replace it, among others.

On this job, both the replacing with a constant, the use of central metrics and the use of a k-NN was tested. The constant value used was -1, the central metric used was the mean, and the k-NN used was configured with k=3.

Date variables creation

After removing outliers, missing values and the transforming the less frequent classes of categorical variables in an aggregated class named “other”, the understanding of the data lead to the creation of new variables regarding dates. The dates’ variables were decomposed into day, month and year making 9 different variables out of 3 original ones. On the test data, the date_first_booking does not exist, i.e., the value is always null, which makes sense since we are predicting the destination country for users that never booked a stay. Hence, all the data regarding this variable was removed from the train dataset, leading to 6 different variables in terms of dates, day_account_created, month_account_created, year_account_created, day_first_active, month_first_active, year_fisrt_active. The authors suspect that the only date varaibles that contain important information for the learners to abstract are the ones related with the month, since it may inform about some yearly seasonality of the bookings connected with seasons. The year should not contain relevant information since the test dataset contains only users registered in 2014, while the train data contains older users. The day’s variables will be relevant if there is some monthly seasonality.

Creation of aggregated variables with session dataset

The session dataset is composed by 6 columns: the user id, the action made on the session, the classification of the action type, some detail about the action, the device used, and the seconds passed on that action.

	user_id	action	action_type	action_detail	device_type	secs_elapsed
0	d1mm9tcy42	lookup	NaN	NaN	Windows Desktop	319.0
1	d1mm9tcy42	search_results	click	view_search_results	Windows Desktop	67753.0
2	d1mm9tcy42	lookup	NaN	NaN	Windows Desktop	301.0
3	d1mm9tcy42	search_results	click	view_search_results	Windows Desktop	22141.0
4	d1mm9tcy42	lookup	NaN	NaN	Windows Desktop	435.0

Figure 10 - Train Session original dataset

The same user shows several times in the session’s dataset, since a user can perform several actions, as can be seen on the table above. However, the prediction should be done at the user level, so the data was converted at user level using aggregated sessions data transformation. This was done by aggregating the number of actions done of each action_type and summing the seconds that took to perform each action_type. Lastly, dummy variables were created for each action_time.

```
session_group=sessions.groupby(['user_id','action_type']).agg({'action':'count','secs_elapsed':'sum'}).reset_index()
session_df=pd.get_dummies(session_group,columns=['action_type']).groupby(['user_id']).sum().reset_index()
session_df.head()
```

Figure 11 - Python code for session features group by and dummy variables creation

The sessions dataset at the user level can be seen on the table below. It is composed by: action – total number of actions performed; secs_elapsed - total seconds spent by that user performing session actions; and a dummy variable for each one of the action types the user can perform. This information can be really useful to understand the user navigation on the Airbnb website and to abstract information of how it influences the first destination country booked.

	user_id	action	secs_elapsed	action_type_-unknown-	action_type_Other	action_type_booking_request	action_type_booking_response	action_type_click
0	00023iyk9l	39	738079.0	0	1	1	0	1
1	0010k6l0om	63	586543.0	1	1	0	0	1
2	001wyh0pz8	90	282965.0	1	1	0	0	1
3	0028jgx1x1	31	297010.0	1	0	0	0	1
4	002qnbzfs5	782	6463327.0	1	1	1	0	1

action_type_data	action_type_message_post	action_type_modify	action_type_partner_callback	action_type_submit	action_type_view
1	0	0	1	0	1
1	0	0	1	0	1
1	0	0	0	1	1
1	0	0	0	1	1
1	1	0	0	1	1

Figure 12 - Output train data set after groupby

Feature selection

The feature selection was done through experimentation of removal of variables while training the models. The technique applied was a backward feature selection, meaning that we started with all variables and at each loop performed one variable was removed in such aa way that the performance increased. As it will be explained on the next chapters, the timestamps, the years and the days variables when removed contributed for a higher performance. All the remaining wither ensured the same or worst performance when removed.

4. Problem modelling, training, and evaluation

As explained before the core focus of this report is to establish a relation between the feature engineering component and the accuracy of the predicting model. Furthermore, to understand the process in place the train dataset was adjusted and fine tunned in the feature engineering stage, then it was applied distinct training models and after that the output with the predicted destination countries would be uploaded to Kaggle platform to access the accuracy of the prediction.

On the first iteration it was just used the train dataset with outliers' removal, Nan values converted to -1 and categorical variables converted into numerical as explained in section 3. Then it was used the *H2O AutoML* model training pipeline, where 10 models were tested with 70% of the train data. The output of the AutoML shown that the model with higher performance was *XGBoost* and the result of the first submission into *Kaggle* was 81% accuracy, that although good, didn't outperform most of the other applicants.

To further explore the *XGBoost* it was applied a Grid Search where the hyper-parameters such as the number of estimators, the max tree depth and the learning rate parameter were iterated in order to optimize the model performance. Along with this optimization it was also included the most complete version of the training dataset with the inclusion of the session dataset features and the removal of the non-relevant variables. The result was a significant increase in the prediction accuracy to 87,4% in the public competition, achieving that way the 350th place out of 1458 applicants – top 24%. The results are displayed in the figure bellow.

To further explore the *XGBoost* it was applied a Grid Search where the hyper-parameters such as the number of estimators, the max tree depth and the learning rate parameter were iterated in order to optimize the model performance. Along with this optimization it was also included the most complete version of the training dataset with the inclusion of the session dataset features and the removal of the non-relevant variables. The description of the best result is summarized on the table below.

Feature Engineering Techniques	**Best results**	**Alternatives tested**
Features selection and engineering	Sessions Features created by transforming sessions dataset Backward feature selection. Features dropped: - Day_account_creation, - Day_first_active, - Year_account_cretion, - Year_first_active, - timestamp_acccount_activation, - timestamp_first_active Creation of days, month and year for date's variables Consolidation of less frequent classes in 'other' class for categorical features	Using all original features on main datset Do not consolidate less frequent classes in 'other' class for categorical features
Missing values replacement strategy	One-Hot enconding ingnoring missing values for categorical Constant value for numerical	One-Hot enconding ingnoring missing values for categorical k-NN or mean for numerical
Models	XGBoost	h2o tested: - Gradiant-Boost Machine (GBM) - Distributed Random Forest (DRF) - Extremely Randomized Trees (EXT)
Hyper-parameter tuning on best model	n_estimaters_param = 48 max_depth_param=6 learning_rate_param=0.25	n_estimaters_param=[28, 38, 48, 58] max_depth_param=[3,4,5,6] learning_rate_param= [0.1,0.25, 0.3]

The result was a significant increase in the prediction accuracy to 87,4% in the public competition, what would represent achieving the 350th place out of 1458 applicants – top 24%. The results are displayed in the figure below. Keep in mind that the rank is virtual since the competition already finished.

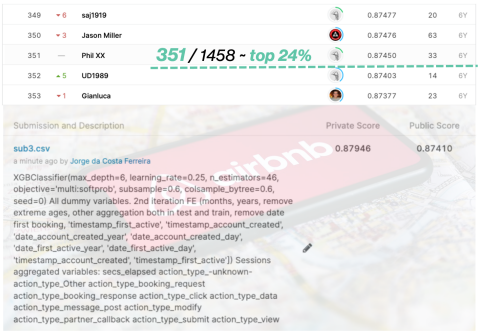

Figure 13 - Kaggle results using XGboost and hyper parameters optimization

5. Conclusions and next steps

The results achieved through the modelling of this problem were quite satisfactory and proved the importance of focusing on the exploration of the data and on the feature engineering for the models to create better results. The first submissions of the results were based on simply taking the original data and creating learners with auto-ML. As the project evolved, new features were created, the data was cleaned with outliers' removal and missing values replacement. This made the accuracy metric to reach 87.4% while it started at approximately 81% on the first submission.

There is some further work that can be done in order to achieve even higher accuracy results on prediction and most of them are related with feature engineering: PCA and creation of new features. Apply principal component analysis (PCA) to the training dataset already encoded in order to find the Principal Components (linear combination of features) that mostly explain the predictive variable. This not only might have impact over accuracy but also to the training and prediction performance. On the feature creation standpoint there are several new features that might bring better results for example transforming the months variable to a season one, meaning grouping "June", "July", "August" and "September" into "Summer" class. This is a variable quite related with the travelling problem, since there are destinations that are highly related with the year season. Other interesting variable transformation would be the grouping of certain devices into two classes, meaning "iPhone", "Samsung" etc.. are "mobile" devices where "Macbook pro" and "Lenovo" are "desktop devices".

Lastly, there was an additional feature that might bring significant value called "date to first booking" that is not defined as well as the "destination country" the predicting variable. Being this variable unknown, it can be applied a predicting model for this feature that have values on the train dataset and predict what could be the date first booking of the test dataset, creating this way a clear link between the date a given user books the first travel and the associated destination because a given date has a month and therefore an associated season. This date can be also transformed into another numerical variable "time to first booking" meaning with the "data account created" and the estimated "date to first booking" one can extrapolate how many days does a user takes since its sign up until making the first booking.