AIRBNB'S NEW USERS: PREDICTION OF FIRST BOOKING COUNTRY

Pierre-Antoine Ksinant

August 3, 2019

ABSTRACT

In this report, we will study a challenge proposed through Kaggle by Airbnb in 2016, with the aim to help the company to predict where a new user will book his first travel experience. Indeed, at this time, new users had the possibility to book a place to stay in more than $34,000$ cities located across more than 190 countries. Solving this problem, thus, can help Airbnb sharing more personalized content with its community, decreasing the average time to first booking, and allowing a better forecast demand.

Keywords Kaggle ◇ Airbnb ◇ Machine Learning ◇ Random Forest Classifier ◇ Adaptive Boosting Classifier ◇ Extra-Trees Classifier ◇ Gradient Boosting Classifier ◇ Multi-Layer Perceptron Classifier ◇ Voting Classifier ◇ Scikit-Learn

1 INTRODUCTION

1.1 Project Overview

In 2016, through Kaggle[1], Airbnb[2] organized a challenge[3] with the aim to build a model able to predict the first booking destination country for Airbnb's new users.

Indeed, at the time of the challenge, new users had the possibility to book a place to stay in more than $34,000$ cities located across more than 190 countries, and, thus, solving this problem can help Airbnb sharing more personalized content with its community, decreasing the average time to first booking, and allowing a better forecast demand.

As an online company managing a lot of users and their relative data, over the years, Airbnb has been able to collect a great amount of information about their users activities on the marketplace, and, so, has developed the objective to enhance its processes thanks to this data, trying to improve its users satisfaction and to optimize its business model.

In this way, this challenge has been part of a more global approach from Airbnb, to put data (and its analysis and processing) as one of its major growth levers to conduct its business processes (e.g., an interested reader can consult [10] to observe how Airbnb tackle another problem in the scope of its core business: Its pricing suggestion algorithm for the rental properties registered in its online marketplace).

1.2 Problem Statement

To allow participants to build their solution to tackle the proposed challenge, Airbnb provided 6 tabular data files on which the prediction model will be constructed, and selected 1 quality metric to evaluate and rank the different solutions it will receive.

For the challenge, participants have been given a list of users—all the users are from the USA—along with their demographics, web session records, and some summary statistics, and they were asked to predict which country a new user's first booking destination will be.

There are 12 possible outcomes for the destination country: "US", "FR", "CA", "GB", "ES", "IT", "PT", "NL","DE", "AU", "NDF"[4], and "other"[5].

Finally, in respect to the dates, it can be noted that, depending the data files, some datasets date back to 2010 while others go until 2015.

The design of our study plan is largely inspired by [8], technically supported by Scikit-Learn [13], the free software machine learning library for the Python programming language [2], and run on a machine (see Table 1) with following hardware main characteristics:

[1] Kaggle is an online community of data scientists and machine learners, owned by Google LLC: https://www.kaggle.com.

[2] Airbnb is an online marketplace and hospitality service brokerage company based in San Francisco (CA, USA): http://www.airbnb.com.

[3] The challenge was entitled "Airbnb New User Bookings: Where will a new guest book his first travel experience?": https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings.

[4] "NDF" stands for "No Destination Found", and means there wasn't a booking.

[5] The label "other" means there was a booking, but is to a country not included in the list.

Table 1: Hardware main characteristics

Model Name	MacBook Pro
Model Identifier	MacBookPro9,2
Processor Name	Intel Core i5
Processor Speed	2.5 GHz
Number of Processors	1
Total Number of Cores	2
L2 Cache (per Core)	256 KB
L3 Cache	3 MB
Memory	4 GB
Boot ROM Version	224.0.0.0.0
SMC Version (system)	2.2f44
Serial Number (system)	C02JC2HXDTY3
Hardware UUID	F0F7E293-C121-5E28-87C0-D169FDA41B45
Sudden Motion Sensor State	Enabled

1.3 Data

The core data on which the prediction model will be constructed is reported on 6 tabular data files:

- *age_gender_bkts.csv*: Summary statistics of users' age group, gender and country of destination;
- *countries.csv*: Summary statistics of destination countries in the dataset and their locations;
- *sample_submission_NDF.csv*: Correct format for submitting predictions for the challenge;
- *sessions.csv*: Web session logs for users;
- *test_users.csv*: Testing set of users;
- *train_users_2.csv*: Training set of users.

Nonetheless, if tabular data files *sample_submission_NDF.csv* and *test_users.csv* had their interest in the context of the challenge, here, we can discard them of our analysis: They don't contain usable information, and, so, are not relevant anymore.

The same can be said of tabular data file *sessions.csv*: If the possibility to exploit its information exists, here, we are going to discard this option. Indeed, only approximately a third of the users present in *train_users_2.csv*—the main base to build our prediction models—can be found in *sessions.csv*, this threshold is too "low" to be considered, for a first look, as interesting, and thus, we will discard, too, this tabular data file for our study.

At last, tabular data files *age_gender_bkts.csv* and *countries.csv* are going to be widely exploited to enhance our comprehension of the situation, learn about Airbnb's users, and recover information in relation with the possible first booking destination countries.

As tabular data file *train_users_2.csv* will be the main base to build our prediction models, below (see Table 2), we are going to detail it more precisely:

Table 2: Detail of *train_users_2.csv*

Label	Type
id	string
date_account_created	date
timestamp_first_active	date
date_first_booking	date
gender	string
age	float
signup_method	string
signup_flow	integer
language	string
affiliate_channel	string
affiliate_provider	string
first_affiliate_tracked	string
signup_app	string
first_device_type	string
first_browser	string
country_destination	string

1.4 Quality Metric

Airbnb provided 1 quality metric to evaluate prediction models: *normalized Discounted Cumulative Gain (normalized DCG)*[6].

To calculate it, first, it is necessary to determine *Discounted Cumulative Gain (DCG)*: For that, if we consider a prediction situation in which the real first booking destination country is d, the ordered list of possibilities proposed by the considered predictor is $\{\hat{d}_i\}_{i\in[\![1,k]\!]}$, with $k \in \mathbb{N}^\star$ (for the challenge, k was fixed to 5), and if we note $\forall i \in [\![1,k]\!], rel_i = \widetilde{1}_d\left(\hat{d}_i\right)$, the relevance of the prediction result at ranking i, then, we have:

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

Then, *normalized Discounted Cumulative Gain (normalized DCG)* can be calculated like this:

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

Where $IDCG_k$ is the *Ideal Discounted Cumulative Gain (Ideal DCG)*, the maximum possible (ideal) *DCG* for a given set of queries.

Obviously, here, considering the exposed conditions, *Ideal DCG* is obtained, for example, with a prediction $\{\hat{d}_1\}$, with $\hat{d}_1 = d$, which leads to $IDCG_k = 1$.

It can be noted that all $nDCG_k$ calculations are relative values on the interval $[0, 1]$.

It can be noted, too, that if a predictor provides more than 5 predictions (k is fixed to 5), only its first 5 predictions will be considered to calculate its *normalized DCG* evaluation (in a similar way, if a predictor provides less than 5 predictions, only

[6] In another context, this metric is classically used to measure effectiveness of web search engine algorithms (e.g., see [3]).

its number of predictions will be used to calculate its *normalized DCG*).

As application example, if for a given user the first booking destination country is "FR", then:

- A {"FR"} prediction gives:

$$\begin{aligned} nDCG_5 &= \frac{DCG_5}{IDCG_5} \\ &= DCG_5 \\ &= \frac{2^{rel_1} - 1}{\log_2(1+1)} \\ &= \frac{2^1 - 1}{\log_2(2)} \\ &= 1 \end{aligned}$$

- A {"US", "FR"} prediction gives:

$$\begin{aligned} nDCG_5 &= \frac{DCG_5}{IDCG_5} \\ &= DCG_5 \\ &= \frac{2^{rel_1} - 1}{\log_2(1+1)} + \frac{2^{rel_2} - 1}{\log_2(2+1)} \\ &= \frac{2^0 - 1}{\log_2(2)} + \frac{2^1 - 1}{\log_2(3)} \\ &\simeq 0.630930 \end{aligned}$$

2 DATA COMPREHENSION

2.1 *Exploration*

2.1.1 *age_gender_bkts.csv*

This first tabular data file lists, for the year 2015, the volume of new users who have chosen between 10 possible destination countries—Australia (see Figure 1), Canada (see Figure 2), Germany (see Figure 3), Spain (see Figure 4), France (see Figure 5), Great Britain (see Figure 6), Italy (see Figure 7), Netherlands (see Figure 8), Portugal (see Figure 9) and USA (see Figure 10)—as their first Airbnb booking, segmented by gender and age repartition.

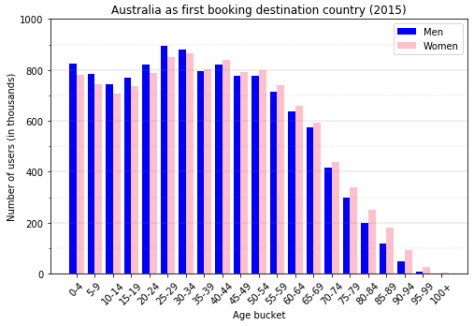

Figure 1: Australia, 1st booking destination (2015)

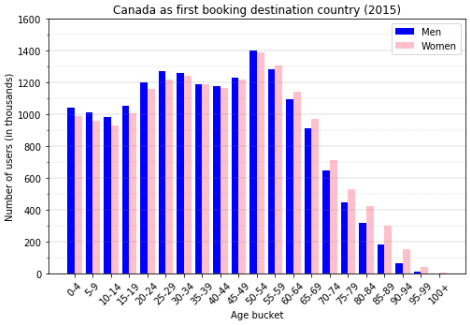

Figure 2: Canada, 1st booking destination (2015)

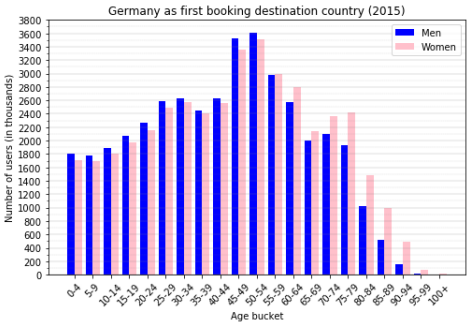

Figure 3: Germany, 1st booking destination (2015)

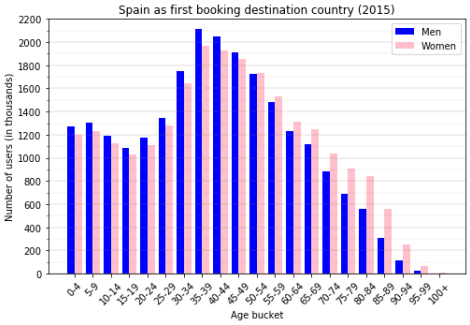

Figure 4: Spain, 1st booking destination (2015)

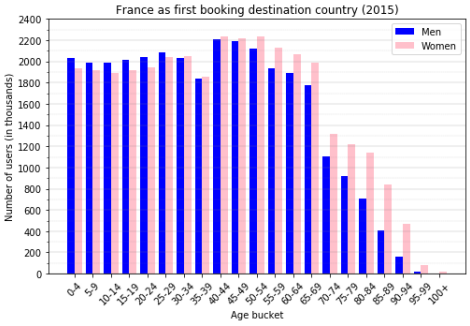

Figure 5: France, 1st booking destination (2015)

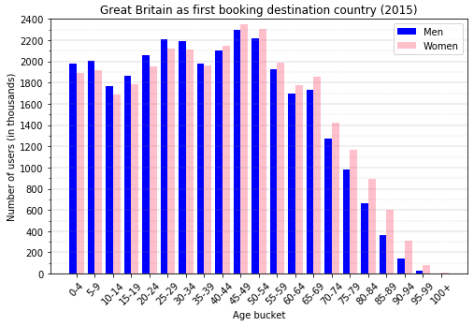

Figure 6: Great Britain, 1st booking destination (2015)

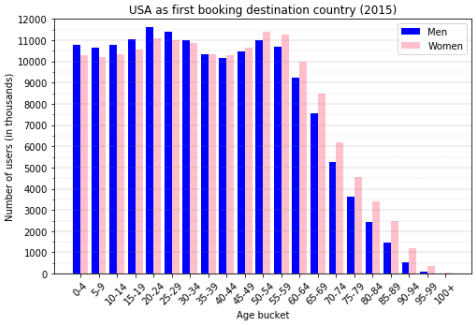

Figure 10: USA, 1st booking destination (2015)

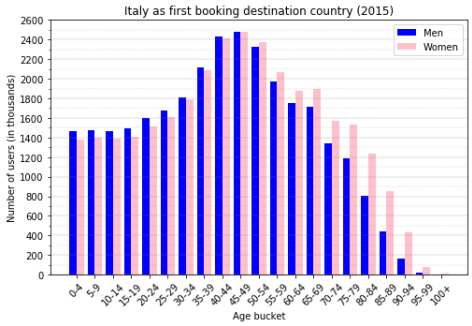

Figure 7: Italy, 1st booking destination (2015)

2.1.2 countries.csv

In this second tabular data file, 3 elements seem interessant to observe:

- The distance between each possible first booking destination country for Airbnb's panel new users and the USA (see Figure 11);
- The area of each possible first booking destination country for Airbnb's panel new users (see Figure 12);
- The Levenshtein distance between the language of each possible first booking destination country for Airbnb's panel new users and the USA language (see Figure 13).

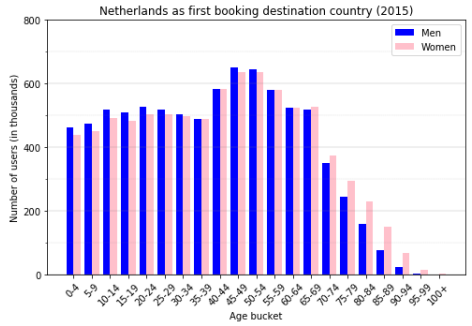

Figure 8: Netherlands, 1st booking destination (2015)

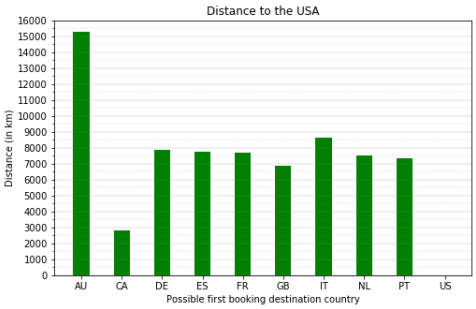

Figure 11: Distance to the USA (in km)

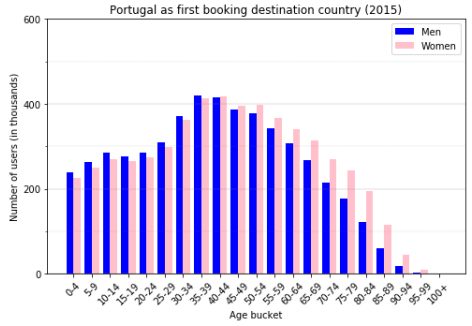

Figure 9: Portugal, 1st booking destination (2015)

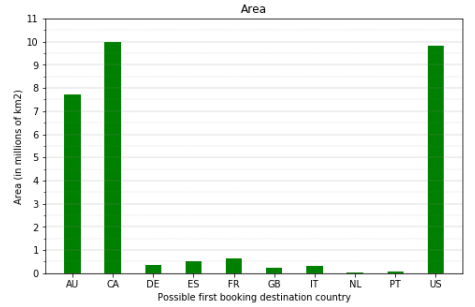

Figure 12: Area (in millions of km^2)

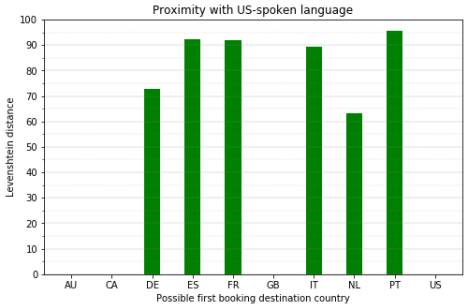

Figure 13: Proximity with US-spoken language

2.1.3 train_users_2.csv

In this third—and last—tabular data file, as major information, it can be noted that it is structured around $213,451$ Airbnb's new users, with a first booking destination country repartition (see Figure 14) as it can be observed in Table 3:

Table 3: First booking destination country repartition

Country	Number of users
No Destination Found	$124,543$
USA	$62,376$
Other	$10,094$
France	$5,023$
Italy	$2,835$
Great Britain	$2,324$
Spain	$2,249$
Canada	$1,428$
Germany	$1,061$
Netherlands	762
Australia	539
Portugal	217

Clearly, for the target variable of our problem for this study, the dataset classes are imbalanced, which is, by far, not the best situation.

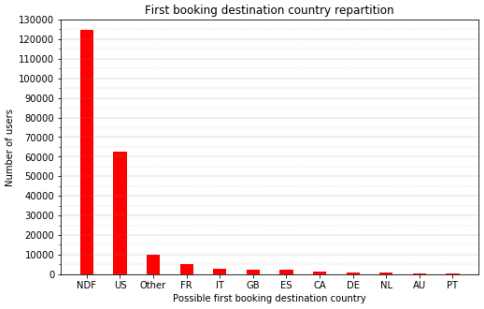

Figure 14: First booking destination country repartition

As complementary information extracted in this tabular data file, the following can be visualized:

- The account creation dates for Airbnb's panel new users (see Figure 15);
- The age repartition of Airbnb's panel new users (see Figure 16);
- The gender repartition of Airbnb's panel new users (see Figure 17).

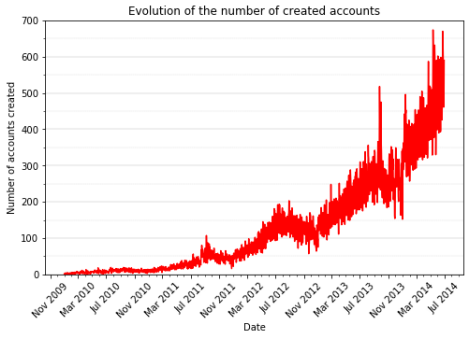

Figure 15: Evolution of the number of created accounts

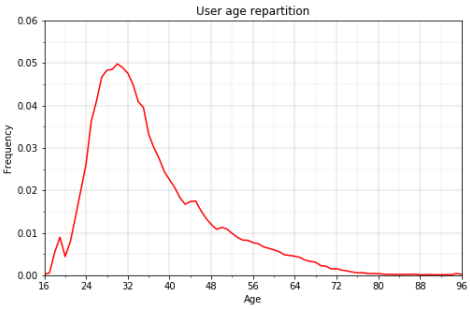

Figure 16: User age repartition

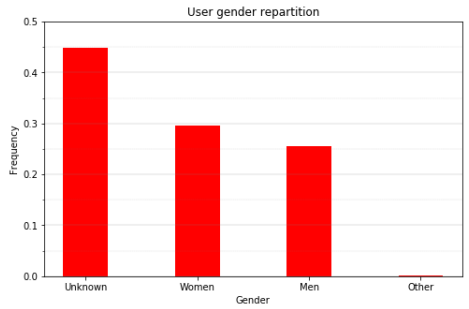

Figure 17: User gender repartition

2.2 Analysis

Respectively to *age_gender_bkts.csv*, which lists, for the year 2015, the volume of new users who have chosen between 10 possible destination countries (Australia, Canada, Germany, Spain, France, Great Britain, Italy, Netherlands, Portugal and USA) as their first Airbnb booking, segmented by gender and age repartition, 3 main conclusions can be made:

- Firstly, in the Airbnb's panel new users, there is an important variation between volumetries of travellers

corresponding to each first booking destination country, with important proportions for the USA and a marked difference between this destination and the second in the list (Germany);

- Secondly, it appears that within each possible first booking destination country, for each age bucket, there is no significative difference between volumetries of travellers corresponding each one of the two genders (nevertheless, it can be noticed that, approximately from 75 years old, the proportion of women within travellers tends to become impacting, independently from the first booking destination country, element which can probably be put in relation with a life expectancy more important for women than for men in the USA, country from where Airbnb's panel new users come from).

- Thirdly, we can report that approximately from 50 years old, the volumetries of travellers for each possible first booking destination country tend to tumble down significantly.

Respectively to *countries.csv*, which lists some characteristics of each possible first booking destination country for Airbnb's panel new users, once again, 3 main conclusions can be made:

- Firstly, within the possible first booking destination countries, we can note 4 categories corresponding to the distance to the USA: The USA itself, Canada which is a border country, the European countries (Germany, Spain, France, Great Britain, Italy, Netherlands and Portugal), and, finally, Australia, the most distant country. We can conjecture that the nearest countries to the USA (and the USA itself per se) are going to be favored by the Airbnb's panel new users, taking into account that they come from the USA.

- Secondly, in respect to the areas of each possible first booking destination country, we can consider 2 main categories: A a first category of "big" countries (USA, Canada and Australia), and a second one of "small" countries (the European ones, Germany, Spain, France, Great Britain, Italy, Netherlands and Portugal). It's difficult to make assumptions on this aspect: Indeed, a country like Australia has an important part of its area that is desert or semi-arid (the center of the country) meanwhile the European countries, despite their "small" areas, are reputed for their touristic attractivity.

- Thirdly, considering the Levenshtein distance between the language of each possible first booking destination country and the USA language, we can note 3 categories: English-speaking countries (The USA itself, Great Britain, Canada and Australia), Germanic-language-speaking countries (Germany and Netherlands) and Latin-language-speaking countries (Spain, France, Italy and Portugal). Here, we can make the assumption that English-speaking countries will be favored, then Germanic-language-speaking countries, and then Latin-language-speaking countries, by the Airbnb's panel new users, taking into account that they come from the USA.

2.3 Preprocessing

With an enhanced comprehension of the situation involved in the challenge, a better idea about Airbnb's user profiles, and now a clear vision of the positionment and ranking in users mind of each possible first booking destination country, we can now explain how we are going to preprocess the tabular data file *train_users_2.csv*, which is going to be the main base to construct the future training and testing sets we will use to build our prediction models:

1. The target variable necessary for this project is clearly identified: *country_destination*.
2. We will perform some checks on the *id* variable to make sure no error is present (we need to check that only unique ids can be counted), and then, we will drop this feature (it is not relevant for our study).
3. The *age* feature needs to be treated to eliminate outliers.
4. All missing values (*NaN* values) and imprecise values (e.g., *-unknown-*) will need to be treated carefully for each impacted variable: More precisely, we are going to drop the variable *date_first_booking* because it counts too numerous missing values, concerning the variable *age*, we will treat missing values as outliers, and replace them by -1, and, lastly, for the variable *first_affiliate_tracked*, missing values will be replaced by *untracked*, creating this way a new category inside the variable. Then, in order to "penalize" imprecise values (features *gender* and *first_browser* counts some *-unknown-* values) and now transformed missing values (i.e., features *age* and *first_affiliate_tracked*), we are going to create a new variable, *nans*, which counts the total number of these occurrences, and add it to the dataset.
5. We need to transform the date features: *date_account_created*, *timestamp_first_active* and *date_first_booking*, and, too, we will create a new variable based on dates, *time_lag*, the time lag between first activity date and account creation date for a given Airbnb new user.
6. We will need to handle the categorical features: *gender*, *signup_method*, *language*, *affiliate_channel*, *affiliate_provider*, *first_affiliate_tracked*, *signup_app*, *first_device_type* and *first_browser* (we will equally need to treat *signup_flow* as a categorical feature, though it contains numeric values, as these ones correspond to categories).
7. Lastly, we will need to scale the continuous variables we are going to introduce: *nans*, the number of missing values and unprecise values for a given Airbnb new user, and *time_lag*, the time lag, for a given Airbnb new user, between first activity date and account creation date.

Preprocessing the tabular data file this way, we get a consolidated dataset—to split between a training set (80%) and a testing set (20%) in a stratified mode to handle the problem of the imbalanced classes—composed by $213,451$ data points with 160 feature variables, 1 target variable each, and, more globally, 0 missing values.

3 PROBLEM MODELING

3.1 General Methodology

To build a prediction solution able to tackle the problem proposed in this challenge, as this is naturally implied, we are going to construct machine learning classifier models thanks to a supervised learning approach relying on the training and testing sets formed from the preprocessing of the tabular data file *train_users_2.csv*.

As it has been specified, we will evaluate these various models we are going to build thanks to the *normalized Discounted Cumulative Gain (normalized DCG)* quality metric, and we will benchmark them and consider their performance using a naive predictor as reference.

Below, we are going to follow 3 main approches to try to solve the challenge's problem:

- Firstly, we will adopt a global approach of the situation, handling all the first booking destination country classes at the same time, discarding the fact that they are widely imbalanced.
- Secondly, we will still adopt a global approach of the situation, but this time, we will add 1 step, building a new machine learning classifier model on the predictions proposed by the first machine learning classifier model, trying this way a correction step, with the aim to catch patterns on first model predictions to improve them.
- Thirdly, trying to handle better the widely imbalanced first booking destination country classes, we will propose a pipeline of 4 machine learning classifier models, and check if this approach is satisfying in comparison with a global approach of the situation.

Finally, we will analyse the results and propose a solution to the challenge's problem.

3.2 Naive Predictor

To benchmark the prediction models we will build, here, as reference, we are going to consider a naive predictor: For each prediction it has to perform, it returns the following list of classified first booking destination country possibilities:

{"NDF", "US", "other", "FR", "IT"}

As it has been observed, this list corresponds to the ordered top 5 first booking destination country possibilities in the consolidated dataset considered in this project.

Below, in Table 4, we can observe the main results obtained with this naive predictor:

Table 4: Naive predictor *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.806766
testing	global	0.806763
	Australia	0.000000
	Canada	0.000000
	Germany	0.000000
	Spain	0.000000
	France	0.430677
	Great Britain	0.000000
	Italy	0.386853
	NDF	1.000000
	Netherlands	0.000000
	Portugal	0.000000
	USA	0.630930
	other	0.500000

3.3 Global Approach of the Situation

For the global approach of the situation which has been explained above, we have tried 6 types of classifier models.

3.3.1 Random Forest Classifier

The first family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of random forest classifiers (see [12]).

As precised in Scikit-Learn's documentation, we can note that:

"A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting."

"Each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. Furthermore, when splitting each node during the construction of a tree, the best split is found either from all input features or a random subset."

"The purpose of these two sources of randomness is to decrease the variance of the forest estimator. Indeed, individual decision trees typically exhibit high variance and tend to overfit. The injected randomness in forests yield decision trees with somewhat decoupled prediction errors. By taking an average of those predictions, some errors can cancel out. Random forests achieve a reduced variance by combining diverse trees, sometimes at the cost of a slight increase in bias. In practice the variance reduction is often significant hence yielding an overall better model."

Performing a random search to tune hyperparameters, we have found that, for a random forest classifier, the best[7] tuning of key hyperparameters is as expressed in the following Table 5:

[7] This qualifier has to be considered as relative, considering the fact that the hardware at our disposal for the project had very limited computation capabilities.

Table 5: Random forest classifier best key hyperparameters

Number of trees in the forest	77
Min number of samples to split internal node	10
Min number of samples to be at a leaf node	1
Max depth of the tree	Pure leaves
Weighted classes	Yes
Whole datset is used to build each tree	Yes

Below, in Table 6, we can observe the main results obtained with such random forest classifier:

Table 6: Best random forest classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.906901
testing	global	0.816485
	Australia	0.007164
	Canada	0.035858
	Germany	0.023998
	Spain	0.106723
	France	0.375889
	Great Britain	0.111050
	Italy	0.166975
	NDF	0.965516
	Netherlands	0.013758
	Portugal	0.000000
	USA	0.742169
	other	0.474328

3.3.2 *Adaptive Boosting Classifier*

The second family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of adaptive boosting classifiers (see [6]).

As precised in Scikit-Learn's documentation, we can note that:

"An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases."

"The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction. The data modifications at each so-called boosting iteration consist of applying weights w_1, w_2, ..., w_N to each of the training samples. Initially, those weights are all set to $w_i = \frac{1}{N}$, so that the first step simply trains a weak learner on the original data. For each successive iteration, the sample weights are individually modified and the learning algorithm is reapplied to the reweighted data. At a given step, those training examples that were incorrectly predicted by the boosted model induced at the previous step have their weights increased, whereas the weights are decreased for those that were predicted correctly. As iterations proceed, examples that are difficult to predict receive ever-increasing influence. Each subsequent weak learner is thereby forced to concentrate on the examples that are missed by the previous ones in the sequence."

Performing a random search to tune hyperparameters, we have found that, for an adaptive boosting classifier, the best tuning of key hyperparameters is as expressed in the following Table 7:

Table 7: Adaptive boosting classifier best key hyperparameters

Max number of estimators (for terminated boosting)	70
Learning rate	1.0

Below, in Table 8, we can observe the main results obtained with such adaptive boosting classifier:

Table 8: Best adaptive boosting classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.823412
testing	global	0.821829
	Australia	0.000000
	Canada	0.017891
	Germany	0.012309
	Spain	0.088466
	France	0.434981
	Great Britain	0.067387
	Italy	0.233823
	NDF	0.949711
	Netherlands	0.002545
	Portugal	0.000000
	USA	0.783837
	other	0.495794

3.3.3 *Extra-Trees Classifier*

The third family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of extra-trees classifiers (see [7]).

As precised in Scikit-Learn's documentation, we can note that:

"An extra-trees classifier is a meta estimator that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting."

"In extremely randomized trees, randomness goes one step further in the way splits are computed. As in random forests, a random subset of candidate features is used, but instead of looking for the most discriminative thresholds, thresholds are drawn at random for each candidate feature and the best of these randomly-generated thresholds is picked as the splitting rule. This usually allows to reduce the variance of the model a bit more, at the expense of a slightly greater increase in bias."

Performing a random search to tune hyperparameters, we have found that, for an extra-trees classifier, the best tuning of key hyperparameters is as expressed in the following Table 9:

Table 9: Extra-trees classifier best key hyperparameters

Number of trees in the forest	55
Min number of samples to split an internal node	10
Min number of samples to be at a leaf node	1
Max depth of the tree	100
Weighted classes	Yes
Whole datset is used to build each tree	Yes

Below, in Table 10, we can observe the main results obtained with such extra-trees classifier:

Table 10: Best extra-trees classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.882097
testing	global	0.813936
	Australia	0.007570
	Canada	0.040835
	Germany	0.025496
	Spain	0.133674
	France	0.350670
	Great Britain	0.111953
	Italy	0.165946
	NDF	0.975789
	Netherlands	0.013014
	Portugal	0.000000
	USA	0.715075
	other	0.466886

3.3.4 *Gradient Boosting Classifier*

The fourth family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of gradient boosting classifiers (see [1]).

As precised in Scikit-Learn's documentation, we can note that:

"Gradient Tree Boosting or Gradient Boosted Regression Trees (GBRT) is a generalization of boosting to arbitrary differentiable loss functions. GBRT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems."

"The advantages of GBRT are natural handling of data of mixed type (heterogeneous features), predictive power and robustness to outliers in output space (via robust loss functions), while the disadvantages of GBRT are scalability, due to the sequential nature of boosting it can hardly be parallelized."

Due to the fact that the hardware at our disposal for the project had very limited computation capabilities, we have not been able to tune the keys hyperparameters that define these classifiers[8].

Below, in Table 11, we can observe the main results obtained with an *out-of-the-box* boosting classifier:

[8]The fine tuning of the numerous hyperparameters that can set gradient boosting classifiers has a very high cost in terms of computation capabilities.

Table 11: *Out-of-the-box* gradient boosting classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.830643
testing	global	0.824181
	Australia	0.003988
	Canada	0.018133
	Germany	0.014461
	Spain	0.100437
	France	0.432548
	Great Britain	0.084488
	Italy	0.225514
	NDF	0.946023
	Netherlands	0.006696
	Portugal	0.000000
	USA	0.798475
	other	0.496727

3.3.5 *Multi-Layer Perceptron Classifier*

The fifth family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of multi-layer perceptron classifiers (see [11]).

As precised in Scikit-Learn's documentation, we can note that:

"Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f : \mathbb{R}^m \longrightarrow \mathbb{R}$ by training on a dataset, where m is the number of dimensions for input and 1 is the number of dimensions for output. Given a set of features $X = x_1, x_2, \ldots x_m$ and a target y, it can learn a non-linear function approximator for either classification or regression."

"The advantages of Multi-layer Perceptron are capability to learn non-linear models and capability to learn models in real-time (on-line learning), while the disadvantages of Multi-layer Perceptron (MLP) include MLP with hidden layers have a non-convex loss function where there exists more than one local minimum (therefore different random weight initializations can lead to different validation accuracy), MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations, and MLP is sensitive to feature scaling."

Performing a random search to tune hyperparameters, we have found that, for a multi-layer perceptron classifier, the best tuning of key hyperparameters is as expressed in the following Table 12:

Table 12: Multi-layer perceptron classifier best key hyperparameters

Training data proportion for validation set	20%
Solver	Adam
Max number of epochs without improvement	10
Max number of iterations	200
Initial learning rate	0.0001
Number of neurons by hidden layers	80, 40, 20
Early stopping	Yes
Activation function	ReLU

Below, in Table 13, we can observe the main results obtained with such multi-layer perceptron classifier:

Table 13: Best multi-layer perceptron classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.822968
testing	global	0.822270
	Australia	0.000000
	Canada	0.000000
	Germany	0.000000
	Spain	0.413072
	France	0.411407
	Great Britain	0.000832
	Italy	0.010234
	NDF	0.935384
	Netherlands	0.000000
	Portugal	0.000000
	USA	0.818191
	other	0.491116

3.3.6 *Voting Classifier*

The sixth—and last—family of models we have mobilized to tackle, with a global approach of the situation, the problem proposed in the challenge, is the family of voting classifiers, which allows, generally, to reach better results than the ones obtained by the classifiers on whose they are built (e.g., in another context, see [14]).

In our study, we have built a voting classifier using the best random forest, gradient boosting and multi-layer perceptron classifiers presented above[9].

Below, in Table 14, we can observe the main results obtained with such voting classifier:

Table 14: Voting classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.869396
testing	global	0.823524
	Australia	0.000000
	Canada	0.006763
	Germany	0.007919
	Spain	0.328300
	France	0.427147
	Great Britain	0.018735
	Italy	0.054544
	NDF	0.948974
	Netherlands	0.003289
	Portugal	0.000000
	USA	0.793151
	other	0.497145

[9]Voting methods work all the better as the predictors on whose they are built are really independent from each other: In order to obtain sufficiently varied classifiers, one solution is to use very different training algorithms, this increases the chances that they make very different errors, thereby improving the accuracy of the set.

3.4 *Stratified Global Approach of the Situation*

As a second attempt to tackle the problem presented in the challenge, we have, like for the first attempt, adopted a global approach of the situation, but this time, we have added 1 step, building a new machine learning classifier model on the predictions proposed by the first machine learning classifier model, trying this way a correction step, with the aim to catch patterns on first model predictions to improve them.

As core machine learning classifier model, we have used the voting classifier presented above, and on its predictions on the training set, considered as probabilities for each first booking destination country class, we have built a gradient boosting classifier[10] to try to correct them, and to propose this way corrected predictions.

Below, in Table 15, we can observe the main results obtained with such corrected voting classifier:

Table 15: Corrected voting classifier *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.951813
testing	global	0.780407
	Australia	0.059779
	Canada	0.078116
	Germany	0.100146
	Spain	0.231286
	France	0.386274
	Great Britain	0.125668
	Italy	0.136344
	NDF	0.887426
	Netherlands	0.041941
	Portugal	0.000000
	USA	0.772363
	other	0.441702

3.5 *Decomposed Approach of the Situation*

As a third attempt to tackle the problem presented in the challenge, we have tried to handle better the widely imbalanced first booking destination country classes, and for that, we have proposed a pipeline of 4 machine learning classifier models, more precisely gradient boosting classifiers[11], structured around 4 steps:

1. A first step where a first gradient boosting classifier will have the task to determine exclusively if the first booking destination country is "NDF".
2. If the prediction provided by this first gradient boosting classifier is different from "NDF", then, a second step will be performed, where a second gradient boosting classifier will have the task to determine exclusively if the first booking destination country is "USA".

[10]The results obtained by this classifier, in comparison with the ones obtained by the other tried classifiers, led us to mobilize it, as it seems to handle better the data present in the consolidated dataset.

[11]As it has been noted previously, it seems that it is this type of classifiers which handle the best the data present in the consolidated dataset.

3. If the prediction provided by this second gradient boosting classifier is different from "USA", then, a third step will be performed, where a third gradient boosting classifier will have the task to determine exclusively if the first booking destination country is "other".

4. Lastly, if the prediction provided by this third gradient boosting classifier is different from "other", then, a fourth step will be performed, where a fourth gradient boosting classifier will have the task to determine the first booking destination country among the remaining possibilities for first booking destination country ("FR", "IT", "GB", "ES", "CA", "DE", "NL", "AU" and "PT").

Below, in Table 16, we can observe the main results obtained with such gradient boosting classifiers pipeline:

Table 16: Gradient boosting classifiers pipeline *nDCG* mean scores

Set		*nDCG* mean score
training	global	0.823940
testing	global	0.822982
	Australia	0.000000
	Canada	0.000000
	Germany	0.002976
	Spain	0.004444
	France	0.433142
	Great Britain	0.002001
	Italy	0.388995
	NDF	0.926442
	Netherlands	0.000000
	Portugal	0.000000
	USA	0.832884
	other	0.499009

3.6 Results Analysis

To analyse the various results we have obtained and to be able, thus, to determine the best machine learning classifier model to tackle the problem proposed in the challenge, we need to consider the following criteria, by order of importance:

1. The *nDCG* mean score on testing set (see Figure 18);

2. The *nDCG* mean score on testing set for each possible first booking destination country (see Figures 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30);

3. The *nDCG* mean score on training set (see Figure 31), principally to put in perspective with the *nDCG* mean score on testing set, and observe if the various compared predictors are concerned by overfitting.

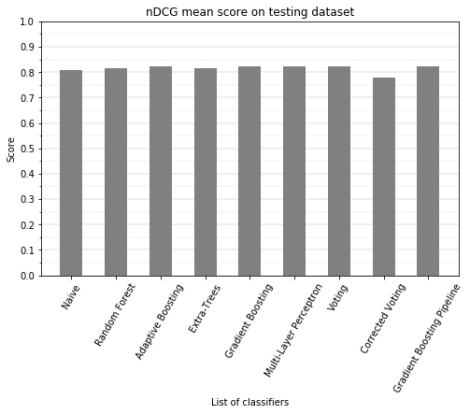

Figure 18: Global *nDCG* mean score on testing set

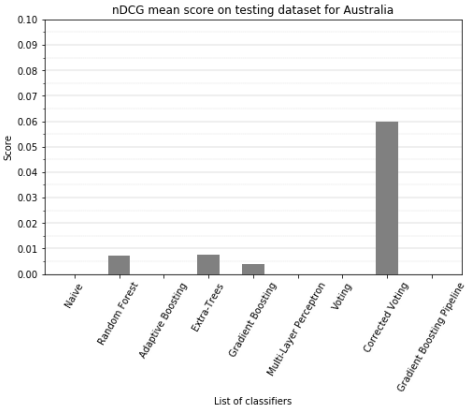

Figure 19: Australia *nDCG* mean score on testing set

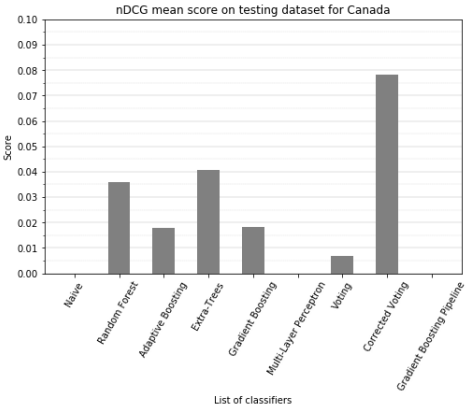

Figure 20: Canada *nDCG* mean score on testing set

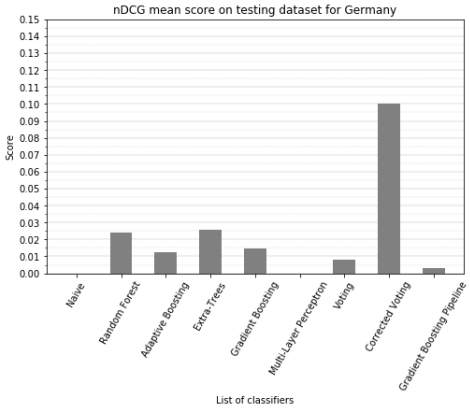

Figure 21: Germany *nDCG* mean score on testing set

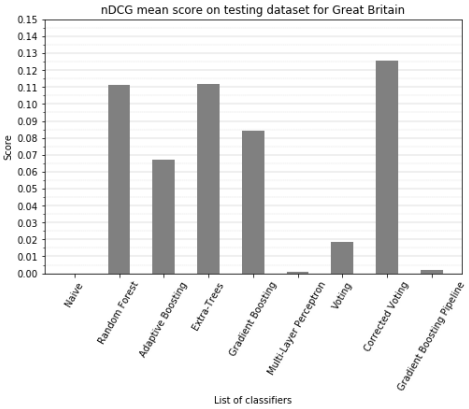

Figure 24: Great Britain *nDCG* mean score on testing set

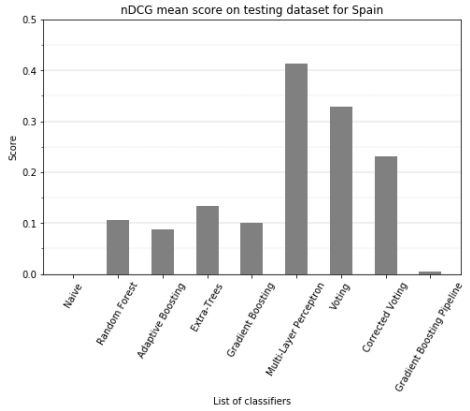

Figure 22: Spain *nDCG* mean score on testing set

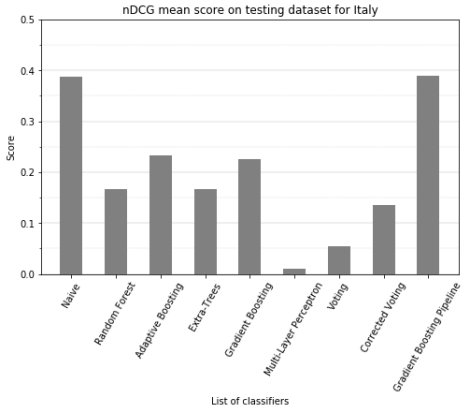

Figure 25: Italy *nDCG* mean score on testing set

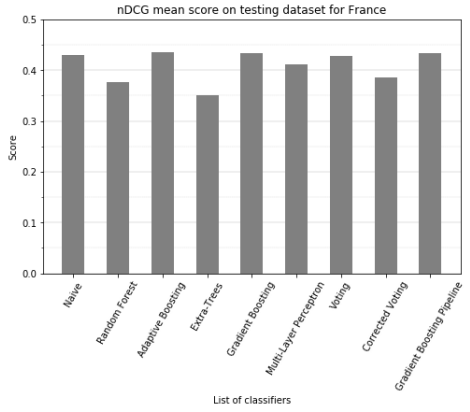

Figure 23: France *nDCG* mean score on testing set

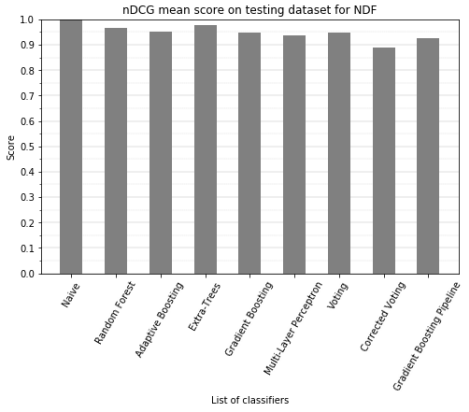

Figure 26: NDF *nDCG* mean score on testing set

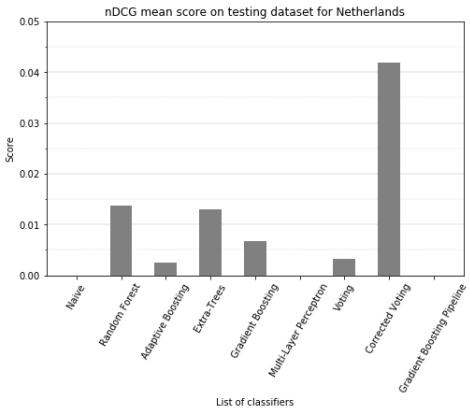

Figure 27: Netherlands *nDCG* mean score on testing set

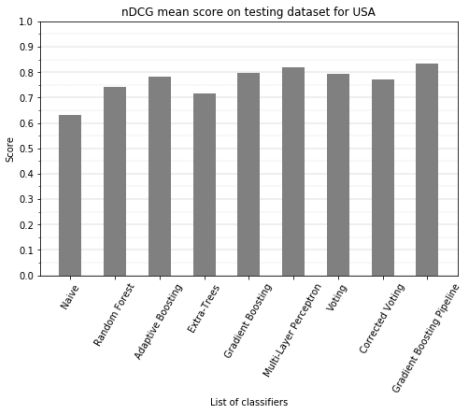

Figure 30: USA *nDCG* mean score on testing set

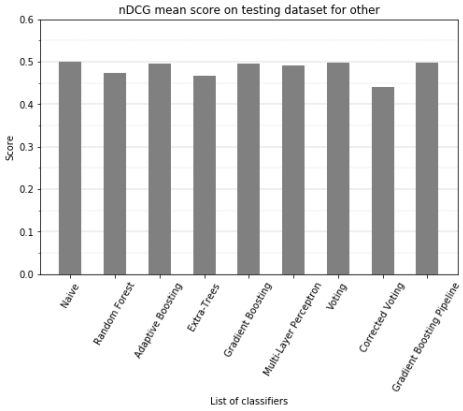

Figure 28: Other *nDCG* mean score on testing set

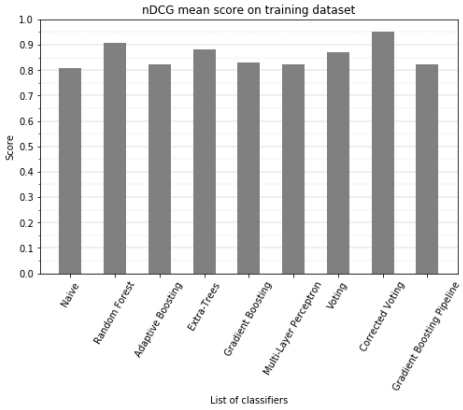

Figure 31: Global *nDCG* mean score on training set

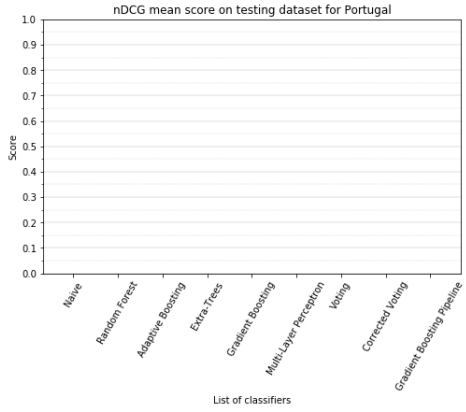

Figure 29: Portugal *nDCG* mean score on testing set

Respectively to the first criteria, we can note that the gradient boosting classifier we have built reaches the best *nDCG* mean score on testing set, and if we compare it with the one obtained on training set, there no significative difference, and, thus, this classifier doesn't appear to be concerned by overfitting on the data considered in the context of the problem proposed by the challenge, which is clearly a good point.

Nonetheless, it is important, too, to remark that the *nDCG* mean scores on testing set, for the classifier models we have built, are not significantly better than the one reached by the naive predictor, which constitutes a real concern.

In regards to the *nDCG* mean scores on testing set for each possible first booking destination country, it appears very difficult to establish a clear hierarchy among the classifiers mobilized for this study. For sure, at least, it can be said that the situation is not a success, notably for underrepresented first booking destination country classes (e.g., none of the classifiers has been able to score points for Portugal as first booking destination): this imbalanced situation is really a tricky issue to handle. It is also interesting to note that the stratified global and decomposed approaches of the situation, specially designed for this characteristic of the problem, revealed themselves to be failures, presenting no real improvement in comparison with the global approach which has been firstly developed.

Respectively to this point, nevertheless, we can however note that the gradient boosting classifier that has been built has scored points for each one of the possible first booking destination, with the exception of Portugal, which is an indication that it has catched patterns in the data for each of these destinations, which is a good point, too.

4 PROPOSED SOLUTION

With all the elements established during our study, and despite the weaknesses noticed in the previous subsection, naturally, we are led, to tackle the problem involved by the challenge, to propose a gradient boosting classifier, following a global approach of the situation, which has, furthermore, the benefit to constitute a simplified workflow in comparison with the 2 other approaches we have explored too.

That being said, we can surmise 2 main tracks to follow to try to improve the current results:

- In machine learning projects, the most import aspect, by far, is the data (e.g., an interested reader can consult [9]). Here, the most tricky element about the data at our disposal is the widely imbalanced situation of the classes corresponding to the target variable. These situations are nevertheless not uncommon, and some methods can be followed to handle this situation (e.g., see [4]). Finally, we can equally work on the dataset we have considered to build the various models we have tried, integrating the information contained in tabular data file *sessions.csv*, and performing more feature engineering on the data (e.g., in tabular data file *age_gender_bkts.csv*, it has been observed that users gender has no significative incidence on first booking destination country, regardeless of age buckets).
- Concerning the gradient boosting classifier implemetation we have mobilized for this project, it would be an interesting idea to put in action the XGBoost implementation (see [5]), and, for that, it can be hoped to benefit from hardware with better computation capabilities than the ones we have benefited for our study, notably to set precisely the tuning of its hyperparameters, which is very costy in respect to these aspects.

REFERENCES

[1] Leo Breiman. Arcing the edge. Technical report, Statistics Department, University of California at Berkeley, USA, 1997.

[2] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. Vanderplas, A. Joly, B. Holt, and G. Varoquaux. Api design for machine learning software: Experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002.

[5] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. 2016.

[6] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, August 1997.

[7] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63:3–42, April 2006.

[8] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, chapter Machine Learning Project Checklist. O'Reilly Media, 2017.

[9] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24:8–12, 2009.

[10] Dan Hill. How much is your spare room worth? *IEEE Spectrum*, 52:32–58, August 2015.

[11] Geoffrey E. Hinton. *Machine Learning: an Artificial Intelligence Approach*, volume 3, chapter Connectionist Learning Procedures, pages 555–610. Morgan Kaufmann Publishers, 1990.

[12] Tin Kam Ho. Random decision forests. In *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, 1995.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] Hans van Halteren, Jakub Zavrel, and Walter Daelemans. Improving data driven wordclass tagging by system combination. In *Proceedings of the 17th International Conference on Computational Linguistics*, 1998.