CENTRE FOR
CONTINUING EDUCATION

CEBD 1260

Hotel Booking

25.03.2020

Cristian Nitu

Shiva Elhamian

Guillermo Campos

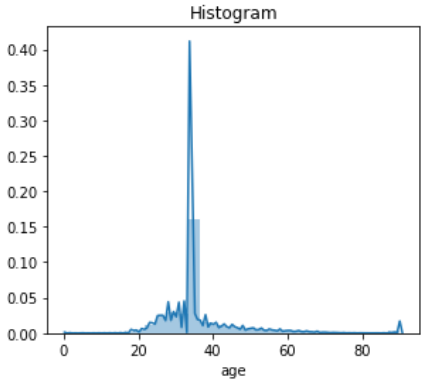
Overview

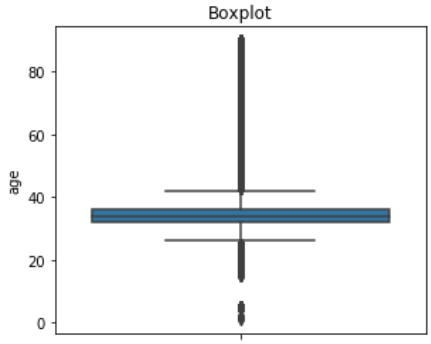
The ability to collect exponentially growing information from customers and companies, combined with the ability to perform automatically complex mathematical calculations to this data, give us the opportunity to extract unique insights and be able to predict information with upcoming data. The purpose of this project is to predict where a new guest will be booking their first travel destination based on the user booking behaviors and demographic information. We performed exploratory data analysis, data cleaning, feature engineering and machine learning on the project.

Our final validation score is multi log loss of **1.139**. We also found out that the three most determinant features are *age, week_first_active, week_first_booking*, two of which are new features added in the feature engineering step.

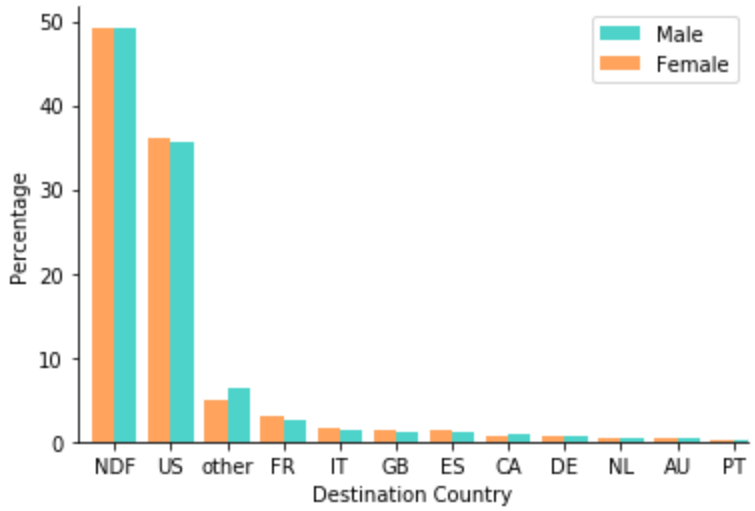
Exploratory Data Analysis

After checking all the users in the training and test data sets we decided to fill all the missing values with the median value. We found that people in their thirties make their first reservation in AirBnB

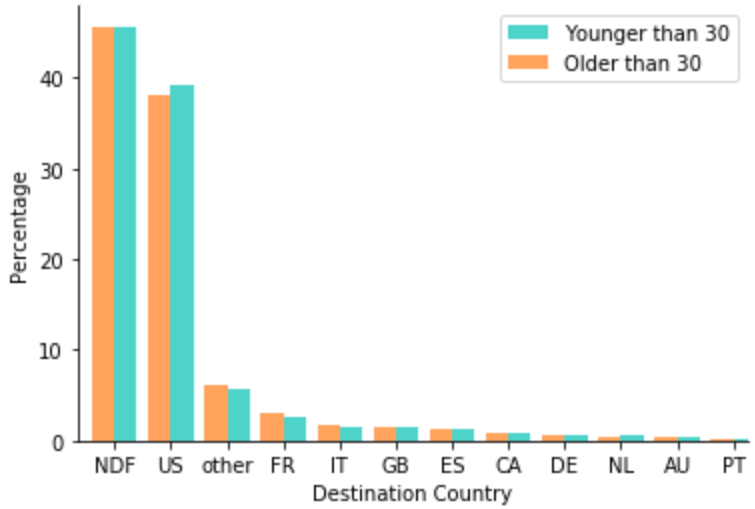




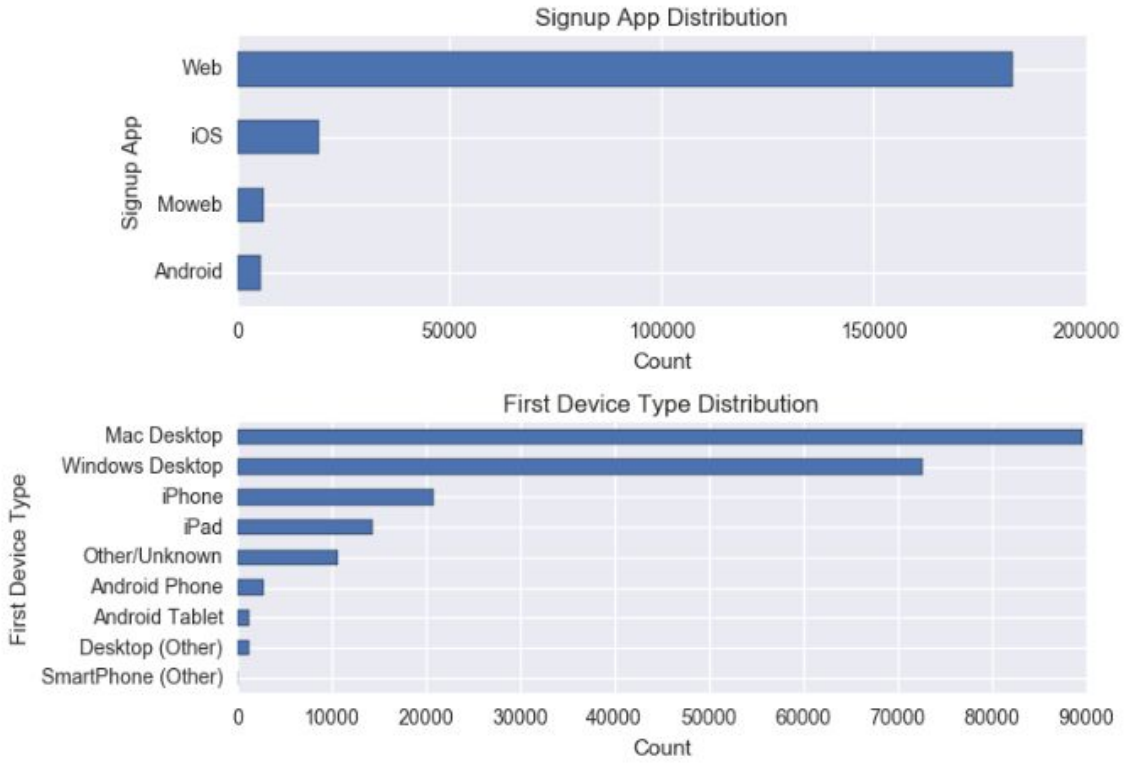
Then we start looking for some distinctive features that allow us to establish unique relationships between the data. We started by looking at the relationship between gender and the country of destination.



Because we didn't find a real impact on gender we decided to keep exploring the data and see if there was some incidendence between the age and the country of destination.



After that we decided to identify the preferences at the moment of the initial sign up to the platform and we found that the people eager to sign up and eventually book keep using the most traditional tools like desktops.



System Pipeline

Data preprocessing

Handling Outliers

- **Age**: replaced values outside the range 18 to 99 with 18 and 99. Note that the minimum age for booking on Airbnb is 18.
- **Secs_elapsed**: caped valued less than 0 And more than 20766. To calculate 20766, we used Interquartile Range Rule:
  IQR = 75th quartile - 25th quartile
  Upper boundary = 75th quartile + (IQR * 1.5)

Dealing with missing values

```
find_missing(df_users)
```

	column	NAN_percentage	dtype
0	age	0.229181	float64
1	first_affiliate_tracked	0.019548	object

- **Age**: Using Mean Imputation technique, we filled in the missing age values with median age, 34.
- **First_affiliate_tracked**: Since the number of records with missing values was relatively low, we dropped the records with missing value.

```
find_missing(df_sessions)
```

	column	NAN_percentage	dtype
2	action_type	0.106570	object
3	action_detail	0.106570	object
4	secs_elapsed	0.012872	float64
1	action	0.007535	object
0	user_id	0.003264	object

- **Action_type, action_detail**: Filled in the missing values with *-unknown-*. It is one of the values in the dataset.
- **Secs_Elapes, action, user_id**: Since the total number of records with missing values was relatively low (less than 5 percent of all the data), we dropped instances with missing value.

Feature Engineering

Datetime fields

We extracted year, quarter, week, and day of week from time_first_active, date_account_created and date_first_booking.

Handling categorical variables

We performed one-hot encoding on all categorical variables in the df_users data set.

dataset	Column Name	Categorical/ Numerica	Number of Categorie
df_users	signup_method	Categorical	3
df_users	gender	Categorical	4
df_users	language	Categorical	25
df_users	affiliate_channel	Categorical	8
df_users	affiliate_provider	Categorical	8
df_users	first_affiliate_tracked	Categorical	8
df_users	signup_app	Categorical	4
df_users	first_device_type	Categorical	8
df_users	first_browser	Categorical	8

The total number of features added is 76.

Aggregating features

We aggregated the features in sessions data set grouped by *user_id* and calculated new features:

```
# feature aggregation
agg_dict = {
    'secs_elapsed': ['sum', 'mean','min','max','median'],
    'action': ['nunique','count'],
    'device_type': ['nunique','count'],
    'action_type': ['nunique','count'],
    'action_detail': ['nunique','count']
}
```

Models

From all the models we tested, tree based models gave us the best accuracy. In this case the LightGBM gave us the best performance.

We performed hyperparameter optimization on our model, using the following param grid.

```
lgb_params_grid = {
    "objective": ["multiclass"],
    "metric": ['multi_logloss'],
    "num_class": [12],
    "max_depth": [32, 64, 128, 256],
    "num_leaves": [255, 400, 1000],
    "bagging_fraction": [0.5, 0.9],
    "feature_fraction": [0.5, 0.9],
    "bagging_freq": [5, 10],
    "max_bin": [250, 400],
    "learning_rate": [0.05, 0.1],
    "bagging_seed": [seed],
    "verbosity": [-1]
}
```

Methodologies

We splitted our data into train and test sets by 80 and 20 percent.

The train set is then splitted into train and validation sets again by 80 and 20 percent. We also evaluated our model using multi log loss function.

Results:

Model: LightGBM

Loss function: Multi log loss

Validation Score: **1.139**

Feature importance: