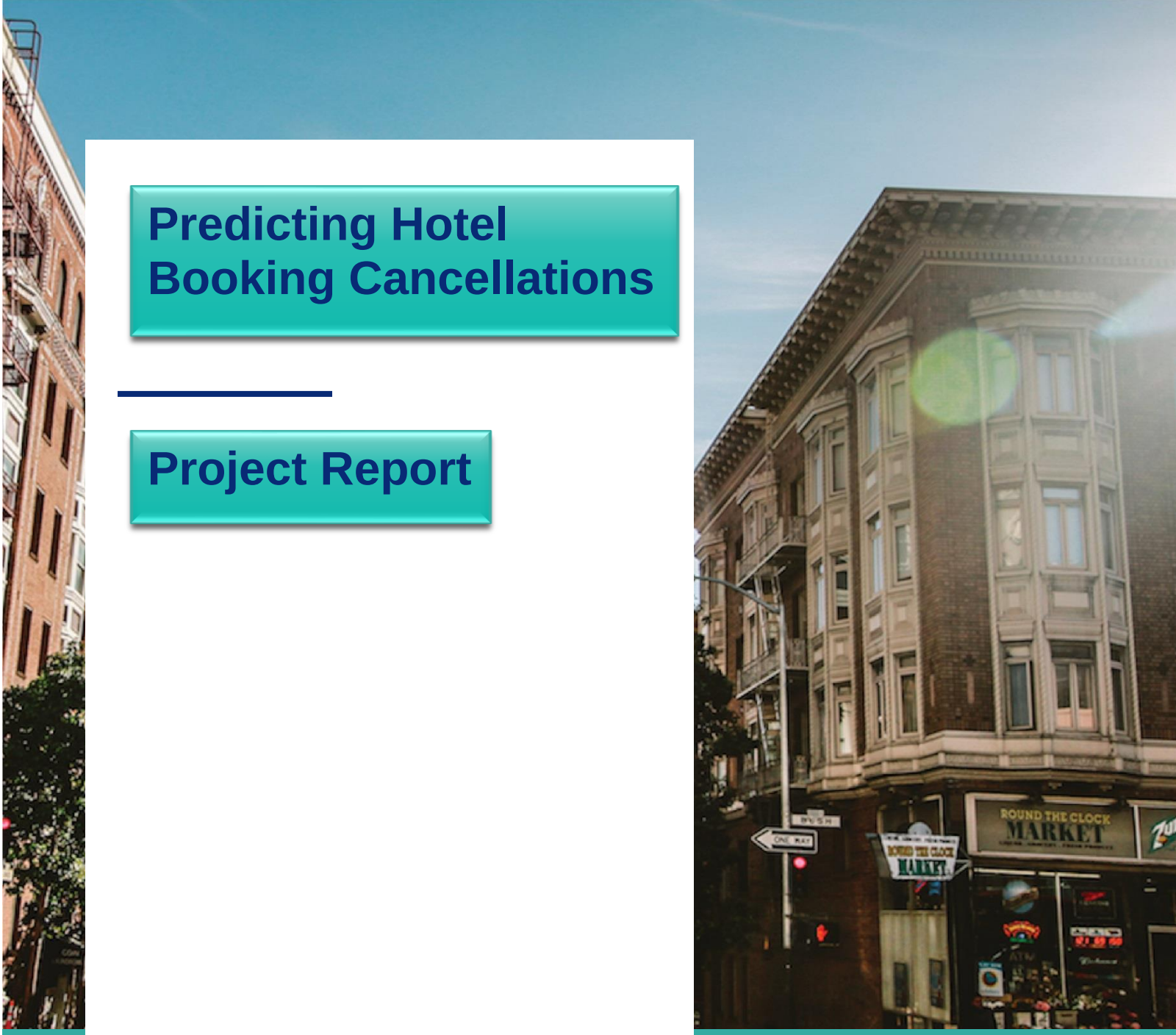
Predicting Hotel Booking Cancellations

Project Report



JUNE 2021

Springboard Capstone
Author: Razvan Nelepcu
Mentor: Dhiraj Khanna

Table of Contents

Introduction

Context

I believe we are at a point where nobody can say that their lives were not affected by the Coronavirus pandemic. Even if it was having someone close getting sick or even die, losing the job or having to change the working style completely to working from home, or just the fact that you had to see the friends and relatives over video calls and not in person and if you did see them, everyone was wearing masks.

The businesses were also heavily impacted by the pandemic. And one of the immediate economic impacts that happened after COVID-19 was declared a world-wide pandemic was the full stop in everything that was related to tourism. Bars and restaurants, airlines and hotels were the businesses that took the hardest hit.

Therefore, with limited customers willing to travel and to book a hotel room and plenty availability, it became crucial for hotels to attract new customers, and even more important to identify those who might cancel their reservation and establish a business strategy to have them keep their reservation before they cancel. For their survival in a pandemic and post-pandemic world, hotels must adapt and use all tools at hand to maximize profits.

Problem Statement

Booking cancellations are without a doubt the biggest headache of any revenue manager or hotel manager nowadays. The simplicity of canceling the reservation and the fact that most of the time there is no fees for making the cancelation makes it extremely easy for customers to cancel their reservations. And when this happens in high percentages it creates a big lost opportunity for the hotels.

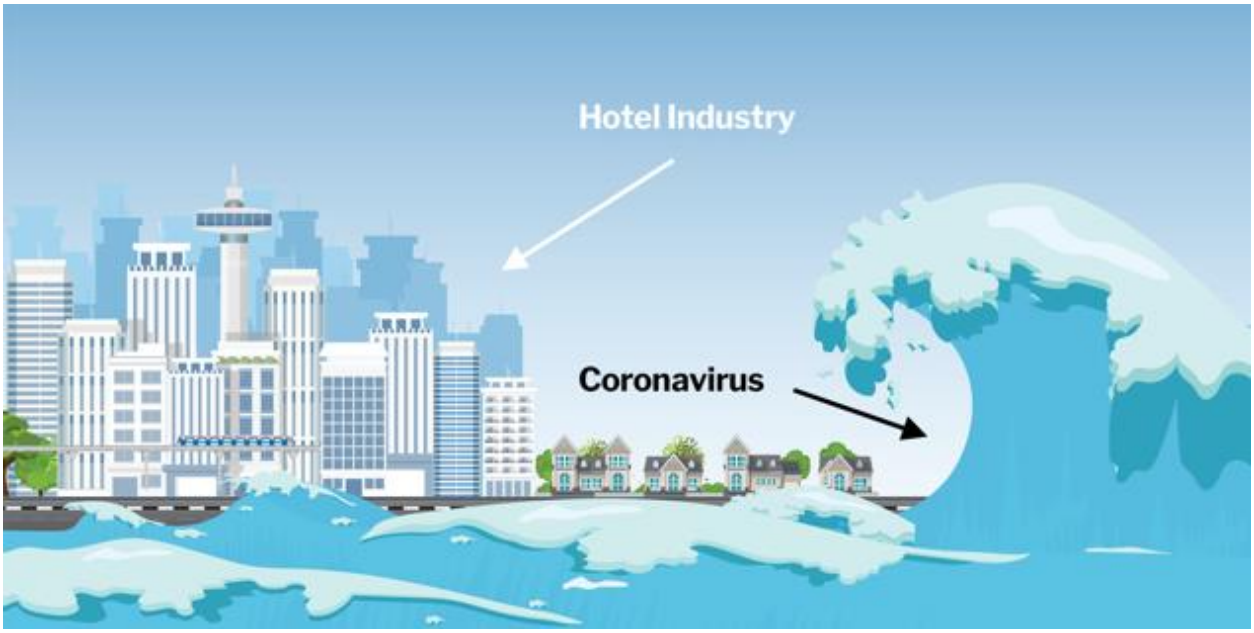


Figure 1 – COVID-19's impact on hospitality

If before 2020 booking cancellations was a huge issue for hotels with percentage of cancelations at 40%, we can only imagine the struggle hotels had during the pandemic. But as of June 2021, the moment of this report, the lockdown situation is relaxing, and customers are starting to travel again and to use hotels again. It will take long before all things go back to normal, but it is a sign of recovery.

And while Marketing Campaigns are starting to bring as many customers as possible to hotels and are offering various discounts and amenities, having a tool to predict customers booking cancelations is becoming vital for the success or even survival of these businesses.

These reasons have made me consider working on a Project that uses hotels booking data and the end goal was to apply the Data Science Methods in order to predict future bookings cancelations, respectively the cancelations percentages for each booking created. For this Classification problem we will be using a series of Machine Learning Classifiers.

The project follows all steps of Data Science workflow divided in the following major steps:

- Data Wrangling
- Exploratory Data Analysis
- Pre-processing and training data development
- Modeling
- Documentation

Data Wrangling

Data collection and organization

The data was downloaded from ScienceDirect. It contains two datasets with hotel bookings registered at a City Hotel in Lisbon and at a Resort Hotel in Algarve between 1st of July of 2015 and the 31st of August 2017. Both datasets share the same structure, with 31 variables describing the 40,060 observations of the Algarve Resort Hotel and 79,330 observations of the Lisbon City Hotel.

All steps described here can be found in this Jupiter Notebook. We started by joining the two datasets then checking the feature types to make sure the data type matches the intended feature type. We completed this step having 32 features out of which 1 datetime, 14 categorical and 17 numeric.

And before moving to data cleaning where we dealt with missing values and outliers, we considered necessary to have a deeper look into our data. And this was completed separately for numerical and for categorical features.

Exploring the numerical features distributions and outliers

The purpose of the univariate analysis was to take a deeper look into the features from our data. We were looking at the shape of our distributions, ranges of values and outliers.

Based on the summary (***Figure 2***) of our numerical features we could identify some interesting facts:

- We have an average **37.04% cancelation rate** for the 2 hotels.
- The average lead-time is 104 days, so on average it takes more than 3 months from the booking date until the arrival date.
- We have some surprisingly long stays(19 weekend nights, respectively 50 weeknights)

- We also have some surprisingly large reservations groups(55 adults, or 10 children and 10 babies)
- We also have an unexpected high number of previous cancelations for one record – 26.

	count	mean	min	25%	50%	75%	max	std
is_canceled	119390	0.370416	0	0	0	1	1	0.482918
lead_time	119390	104.011	0	18	69	160	737	106.863
arrival_date_year	119390	2016.16	2015	2016	2016	2017	2017	0.707476
arrival_date_week_number	119390	27.1652	1	16	28	38	53	13.6051
arrival_date_day_of_month	119390	15.7982	1	8	16	23	31	8.78083
stays_in_weekend_nights	119390	0.927599	0	0	1	2	19	0.998613
stays_in_week_nights	119390	2.5003	0	1	2	3	50	1.90829
adults	119390	1.8564	0	2	2	2	55	0.579261
children	119386	0.10389	0	0	0	0	10	0.398561
babies	119390	0.00794874	0	0	0	0	10	0.0974362
previous_cancellations	119390	0.0871178	0	0	0	0	26	0.844336
previous_bookings_not_canceled	119390	0.137097	0	0	0	0	72	1.49744
booking_changes	119390	0.221124	0	0	0	0	21	0.652306
days_in_waiting_list	119390	2.32115	0	0	0	0	391	17.5947
adr	119390	101.831	-6.38	69.29	94.575	126	5400	50.5358
required_car_parking_spaces	119390	0.0625178	0	0	0	0	8	0.245291
total_of_special_requests	119390	0.571363	0	0	0	1	5	0.792798
reservation_status_date	119390	2016-07-30 00:24:47.883354368	2014-10-17 00:00:00	2016-02-01 00:00:00	2016-08-07 00:00:00	2017-02-08 00:00:00	2017-09-14 00:00:00	NaN

Figure 2 Numerical features description

By plotting the numerical features distribution, we observed immediately that while there are a few features that have values distributed somewhat uniformly, like lead time, arrival date year, arrival date week number or arrival date day of month, the other features are having most of their values towards 0 and some outliers. So next we had a look at some of these outliers to see if any of them are an effect of wrong records entries or just natural outliers of the distributions.

To explore the outliers of our numerical data we formulated a series of questions pertaining to these features: high values on length of stay, high number of customers per booking(adults, children, or babies), high numbers of reserved parking spaces, and high ADR(average daily rate). These were some conclusions:

- The highest ADR value was 5400, and the seconds and third highest were 510 and 508. We identified that here the high value was due to an data entry error, so we removed the booking.
- We can see that all the top 10 bookings with the highest number of adults were canceled. The number of adults was between 55 and 26, they were all Groups, no babies or children, and the lead_time was always over 300 days. I think these were not errors in entries and these were all real bookings that got canceled. You can find these bookings in ***Figure 3***.

None of the other outliers or distributions seemed different from what was expected from the features' descriptions, so we concluded our numerical features initial exploration.

	2173	1643	1539	1917	1962	2003	1752	2164	1587	1884
hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel	Algarve Resort Hotel
is_canceled	1	1	1	1	1	1	1	1	1	1
lead_time	338	336	304	349	352	354	340	361	333	347
arrival_date_year	2015	2015	2015	2015	2015	2015	2015	2015	2015	2015
arrival_date_month	October	September	September	September	September	September	September	October	September	September
arrival_date_week_number	41	37	36	39	39	39	37	40	36	38
arrival_date_day_of_month	4	7	3	21	24	26	12	3	5	19
stays_in_weekend_nights	2	1	0	1	1	2	2	2	2	2
stays_in_week_nights	0	2	3	3	3	5	5	5	5	5
adults	55	50	40	27	27	26	26	26	26	26
children	0	0	0	0	0	0	0	0	0	0
babies	0	0	0	0	0	0	0	0	0	0
meal	HB	BB	BB	HB	HB	BB	BB	BB	BB	BB
country	PRT	PRT	PRT	PRT	PRT	PRT	PRT	PRT	PRT	PRT
market_segment	Direct	Direct	Direct	Direct	Direct	Offline TA/TO	Offline TA/TO	Offline TA/TO	Offline TA/TO	Offline TA/TO
distribution_channel	Direct	Direct	Direct	Direct	Direct	TA/TO	TA/TO	TA/TO	TA/TO	TA/TO
is_repeated_guest	0	0	0	0	0	0	0	0	0	0
previous_cancellations	0	0	0	0	0	0	0	0	0	0
previous_bookings_not_canceled	0	0	0	0	0	0	0	0	0	0
reserved_room_type	A	A	A	A	A	A	A	A	A	A
assigned_room_type	A	A	A	A	A	A	A	A	A	A
booking_changes	0	0	0	0	0	0	0	0	0	0
deposit_type	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit	No Deposit
agent	NULL	NULL	NULL	NULL	NULL	96	96	96	96	96
company	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
days_in_waiting_list	0	0	0	0	0	0	0	0	0	0
customer_type	Group	Group	Group	Group	Group	Group	Group	Group	Group	Group
adr	0	0	0	0	0	0	0	0	0	0
required_car_parking_spaces	0	0	0	0	0	0	0	0	0	0
total_of_special_requests	0	0	0	0	0	0	0	0	0	0
reservation_status	Canceled	Canceled	Canceled	Canceled	Canceled	Canceled	Canceled	Canceled	Canceled	Canceled

Figure 3 – The top 10 bookings with the highest number of adults were all Canceled

Target feature and categorical feature exploration

Our target feature is the binomial feature ***is_canceled***. As we presented above, the overall cancelation rate at the 2 hotels is at **37.04%.**

Next, we continued exploring the target feature and how did it influence our categorical data. We will present just some exploration, for more features please consult the Jupiter Notebook.

One of this exploration showed us that the most bookings are during the summer months but also that July and January are having the highest cancelations percentages with over 40%(***Figure 4 and 5***).

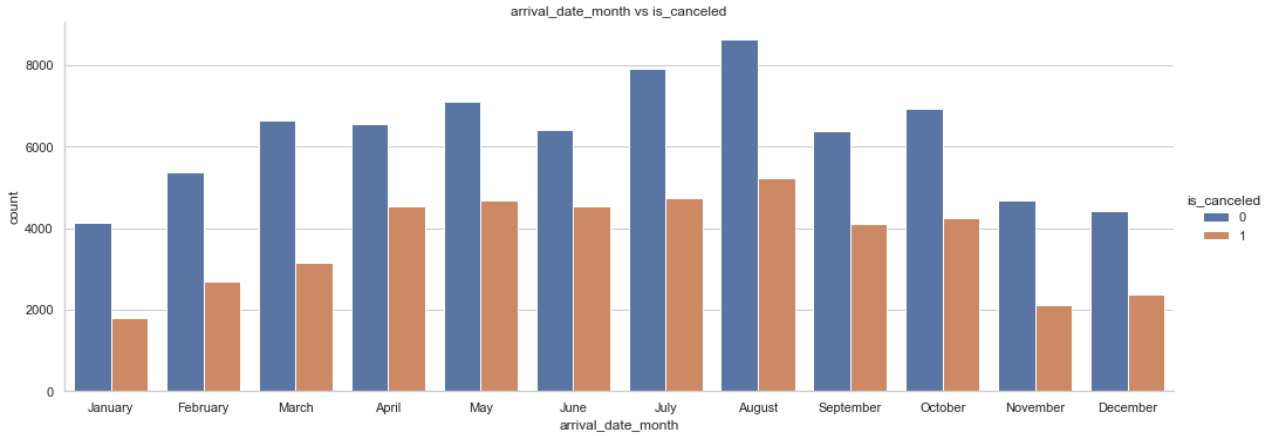


Figure 4 – Number of Bookings canceled by month

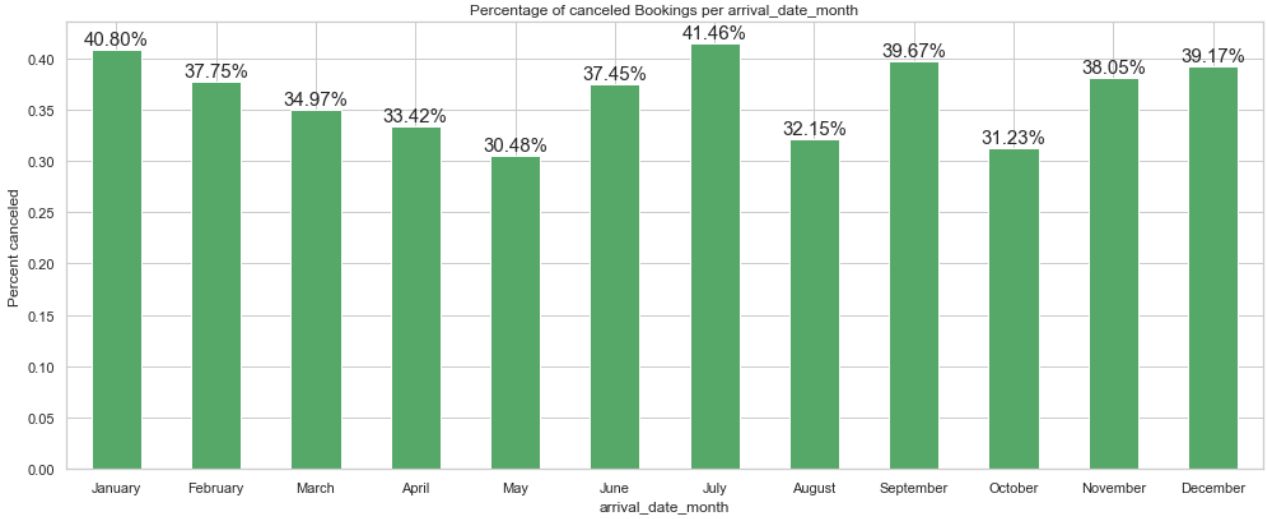


Figure 5 – Percentage by month

Another interesting aspect was that when analyzing the Market_segment we can see that the majority of the bookings were attributed to Online TA segment(***Figure6***) and that the Groups segment accumulated a 61.06% cancelation rate(***Figure7***).

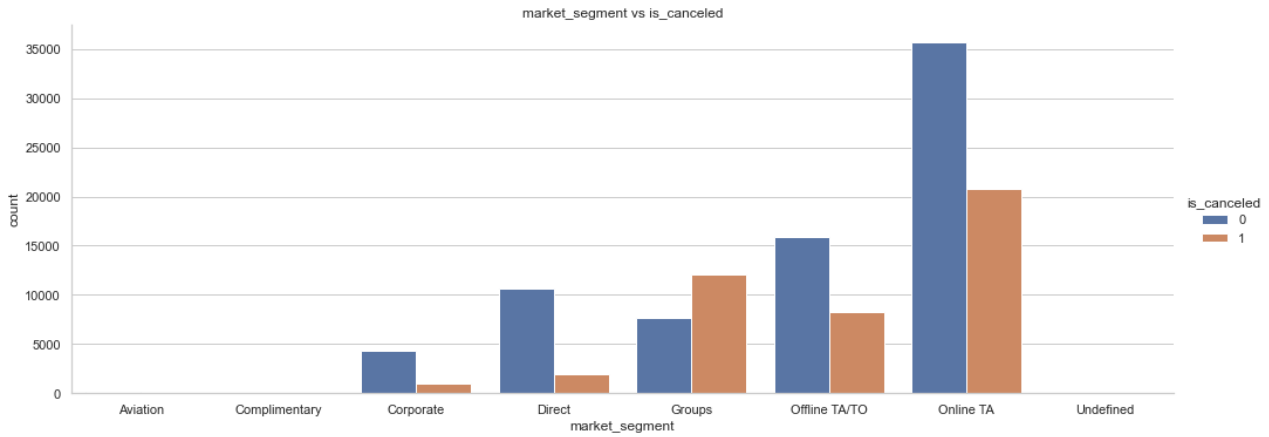


Figure 6 – Bookings canceled by Market Segment

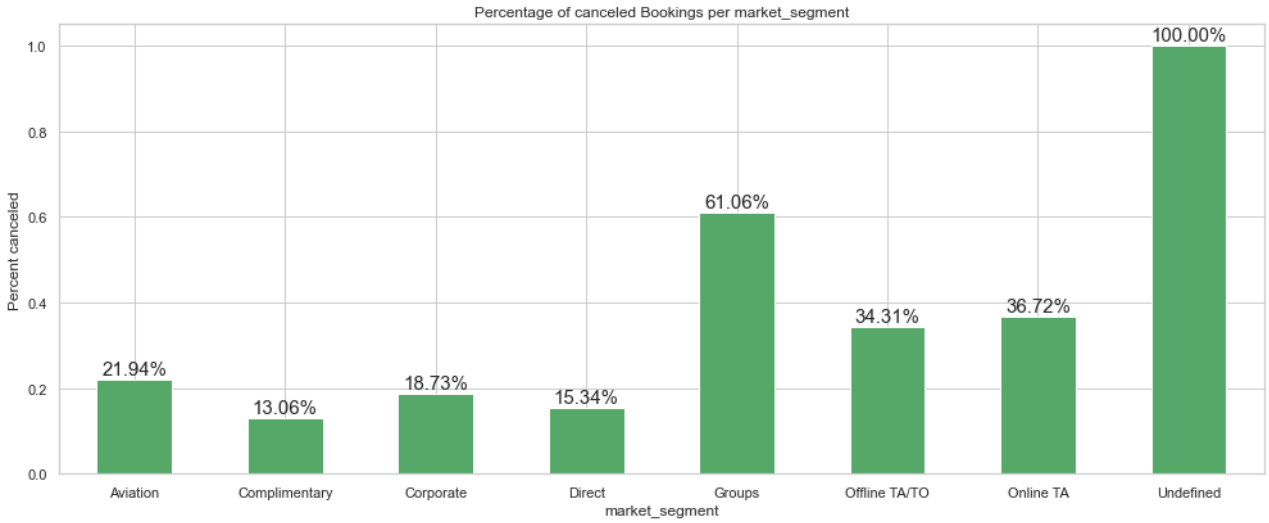


Figure 7 – Percentage of cancelations by Market Segment

Finally, another interesting finding was that while not accounting for a huge amount of the bookings, a very high percentage(**99.36%)** of customers that made a non-refundable deposit eventually canceled their booking(***Figure 8 &9***).

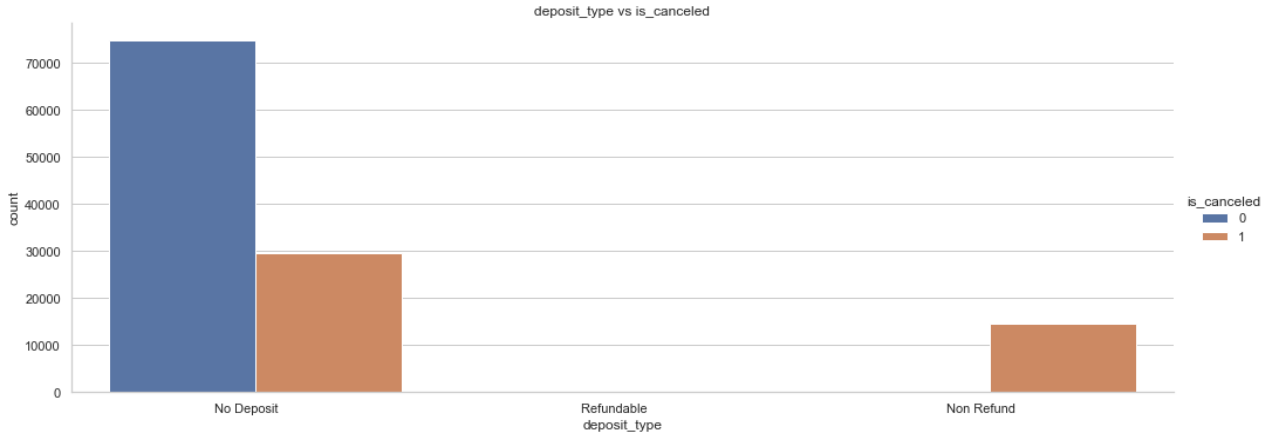


Figure 8 - Bookings canceled by Deposit Type

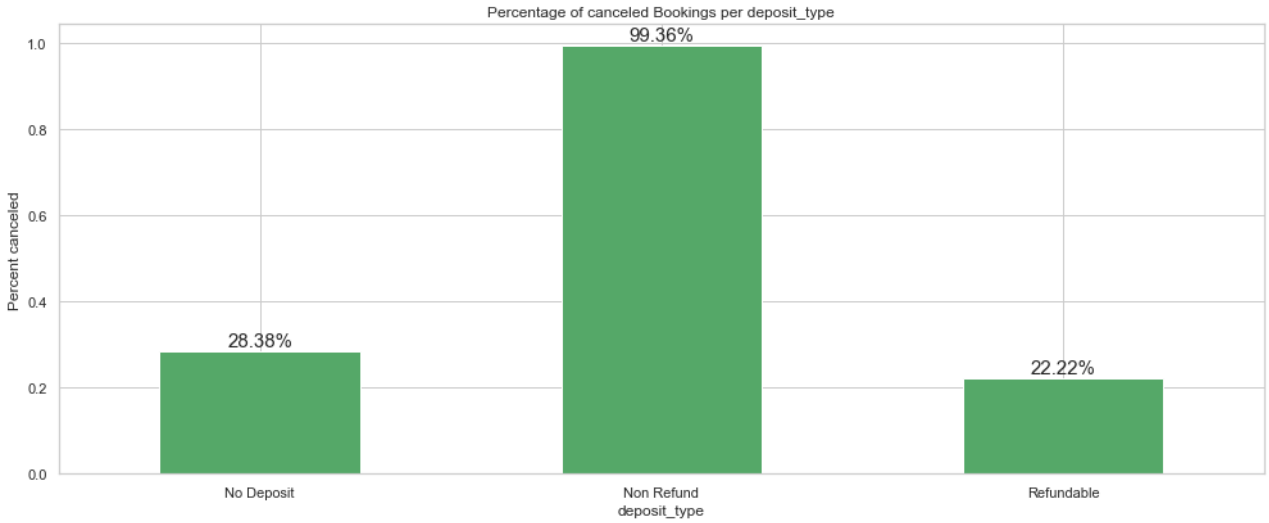


Figure 9 – Percentage cancelations by Deposit Type

Cleaning and saving data

When running an **isnull** count we saw that we have only 2 features with missing values - 488 for countries and 4 for children.

	count	%
country	488	0.408748
children	4	0.003350
hotel	0	0.000000
agent	0	0.000000
reserved_room_type	0	0.000000

We replaced the missing country values with 'UNK'. Another option would have been to replace those with the most common value, in this case PRT(Portugal) but maybe those missing values are especially from tourists that came from outside Portugal, so we do not want to lose any specific info. For the missing values in our children feature, we replaced the missing data with the median value.

We were also missing 16340 agent values and 112593 company values. Which made sense, since there might be some visitors that made the booking without an agent, and most of the people staying in the hotel are paying for the stay themselves and not from their employer's money.
We replaced those NULL values with no_agent or no_company.
The cleaned resulting data was saved for future exploration and processing.

Exploratory Data Analysis

Exploratory Data Analysis(EDA) is one of the most important steps of the Data Science Method because of two main reasons. Firstly, an EDA done properly will tell the story of the data to everyone - from a scientist to a stakeholder or even a regular person without technical or domain knowledge. Secondly, EDA allows us to see patterns, create assumptions and test them as well.

We believe that one of the most complete definitions of EDA is in NIST/SEMATECH e-Handbook of Statistical Methods.

> "Exploratory Data Analysis (EDA) is an approach/philosophy for data analysis that employs a variety of techniques (mostly graphical) to:
> 1. maximize insight into a data set;
> 2. uncover underlying structure;
> 3. extract important variables;
> 4. detect outliers and anomalies;
> 5. test underlying assumptions;
> 6. develop parsimonious models; and
> 7. determine optimal factor settings.
>
> The EDA approach is precisely that - an approach - not a set of techniques, but an attitude/philosophy about how a data analysis should be carried out." - ***e-Handbook of Statistical Methods***

And while we performed some exploration in the first Notebook of our project, some questions remain that we explored in this notebook:

- Does the origin of the booking matter when it comes to Booking Cancelations? In other words, are there significant differences in bookings, and more specific in booking cancelations, coming from the Algarve Resort Hotel or from Lisbon City Hotel?
- With data being spread over 3 years, are there any trends that can be identified correlated to the year or the arrival?
- Is a heatmap of our features going to reveal any unexpected correlations?
- In the previous notebook we identified that almost all customers(99.36%) who had a "No Refund"(full pay of the stay) decided to cancel their booking which seems very surprising. Are there other features that could explain this very high percentage?

Hypothesis testing - Are the two hotels records similar?

A first question that we considered is how different are the 2 hotels? Was there a significant difference between the two so that we must consider which records came from one or another, or can we ignore this fact and simply disregard the origin of these records and perhaps remove having the hotel feature entirely?

With this purpose in mind, to see if records from the 2 hotels are similar or not, we conducted a Hypothesis Testing regarding the cancelations' distribution of the two hotels.

Our **Null hypothesis**:

- **H_{null}**: the observed difference in the mean cancelations per day of Algarve Resort Hotel and Lisbon City Hotel is due to chance (and thus not due to the hotel).

The opposite hypothesis is the **Alternate hypothesis**:

- **$H_{alternative}$**: the observed difference in the mean cancelations per day of Algarve Resort Hotel and Lisbon City Hotel is due to chance (is due to the hotel).

We are also going to pick a **significance level** of 0.01 - A confidence level of **99%**.
To check this hypothesis, we had to create a distribution for each hotel containing relevant indicators regarding booking cancelations.

So as a first step of the Hypothesis Testing, from our data we created a new feature containing the percentage of bookings canceled per day for each of the two hotels.

We then had to check if any of the 2 distributions were normal distributions using stats.normaltest(), and the p-value for the 2 distributions were almost 0, resulting non-normal distributions.

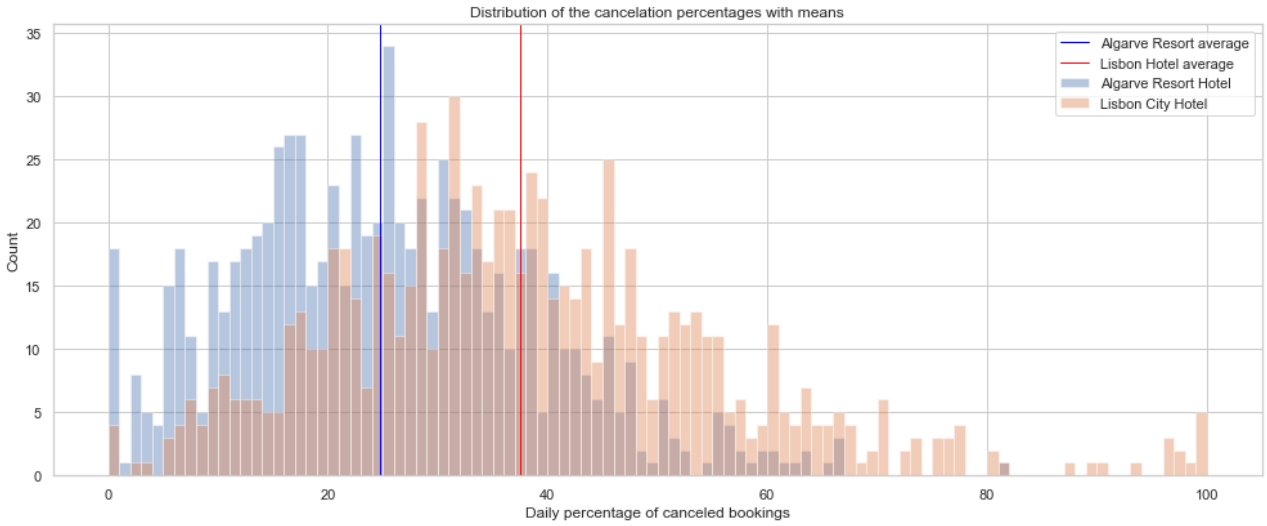


Figure 10 – Distribution of the percentages of canceled bookings per day for each of the 2 hotels

Since the data was not normally distributed, we used a non-parametric test. This is simply a label for statistical tests used when the data is not normally distributed.
We created 1000 permutations on our data(the daily percentages) and we calculated the distributions of mean differences between the 2 hotels' values. The observed difference between the 2 initial percentages was 12.66, plotted with the red line on the mean difference distribution after 1000 permutations. The p-value for our observed difference was 0, with zero differences from our permutations being at least as extreme as our observed difference!

It did not matter which significance level we picked; the observed mean difference was statistically significant, and we rejected the Null Hypothesis. As somewhat expected, we concluded that hotels have an influence on the is_canceled target feature.

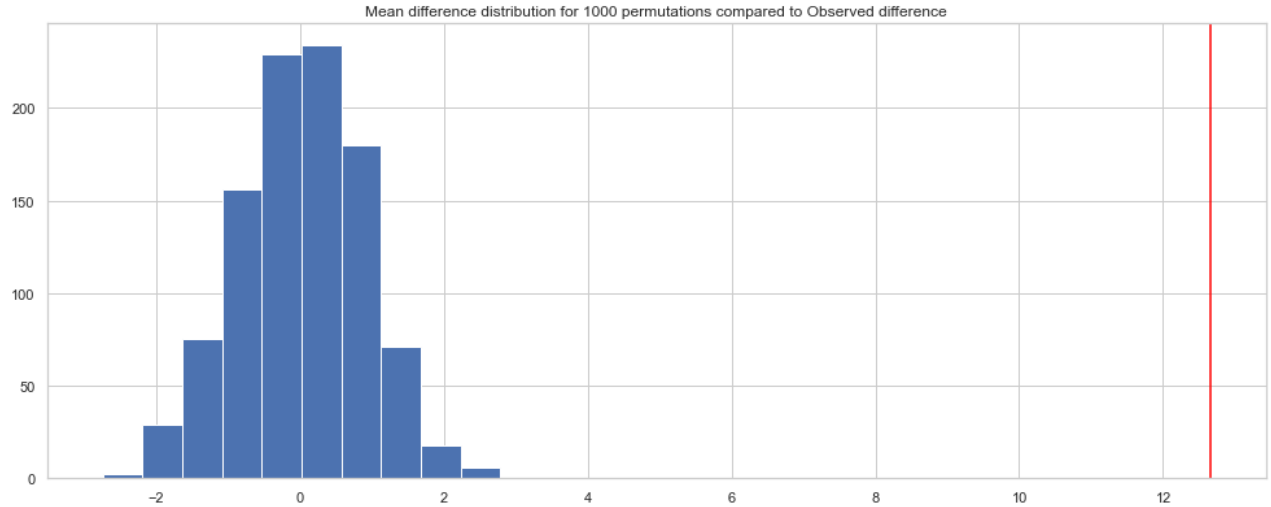


Figure 11 – Observed Value compared to 1000 Permutation distribution

Feature analysis

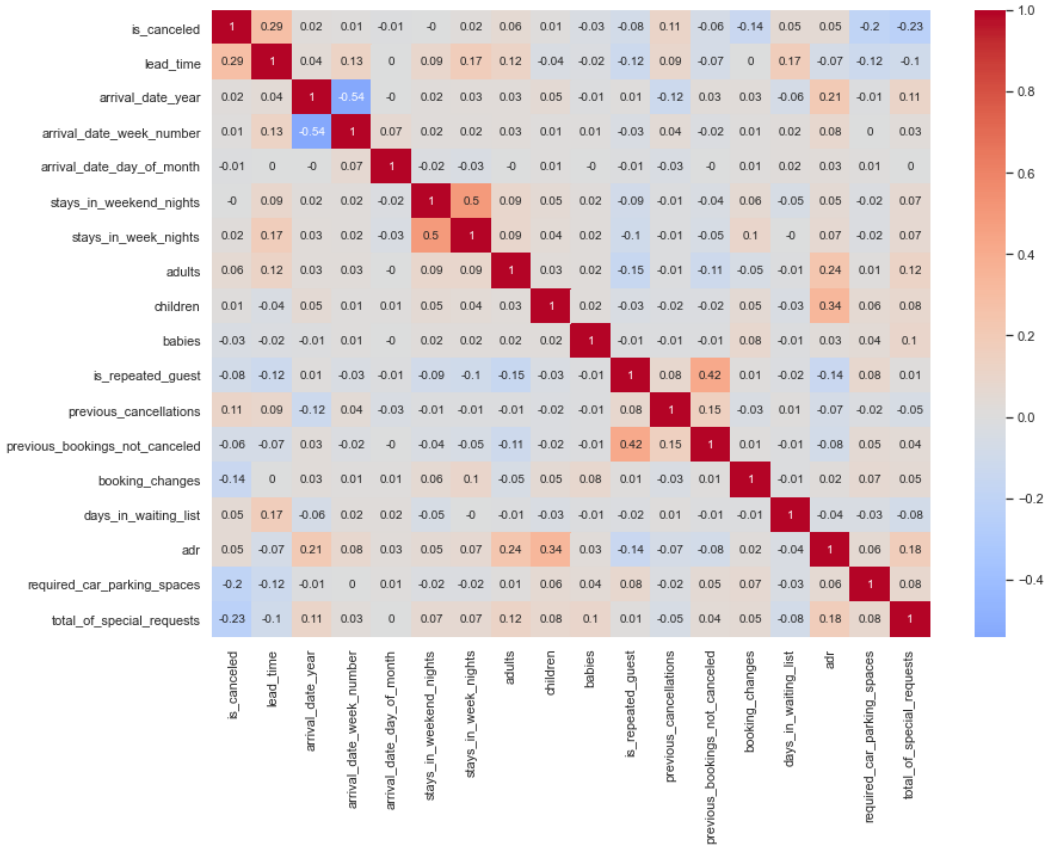
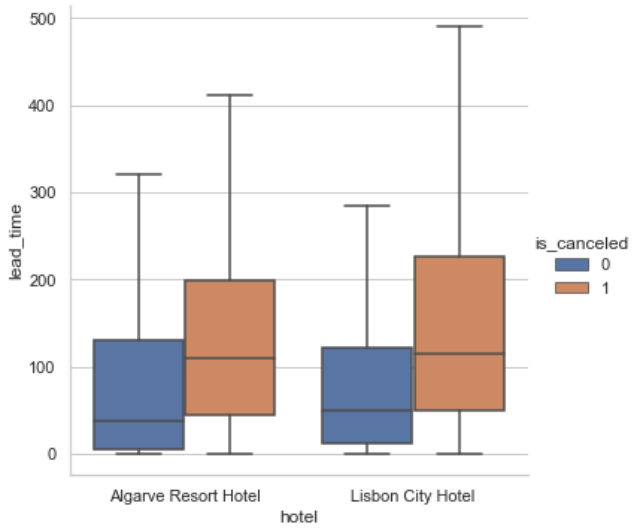


Figure 12 - Heat Map

We continued our EDA with a heatmap analysis, and we came to the following findings:

- is_canceled positively correlated with lead_time and negatively correlated with required_car_parking_spaces and total_of_special_requests
- ADR positively correlated with arrival_date_year, adults, and children

So naturally, we wanted to check the positive correlation between is_canceled and lead_time and ADR and lead_time.



As we can see in the boxplots, the higher the lead_time the higher chances to have a canceled booking.

Another interesting finding is that while for confirmed bookings adr and lead_time are not correlated, for those canceled the 2 features are negatively correlated.

Figure 13 - Lead time for confirmed and canceled bookings per hotel

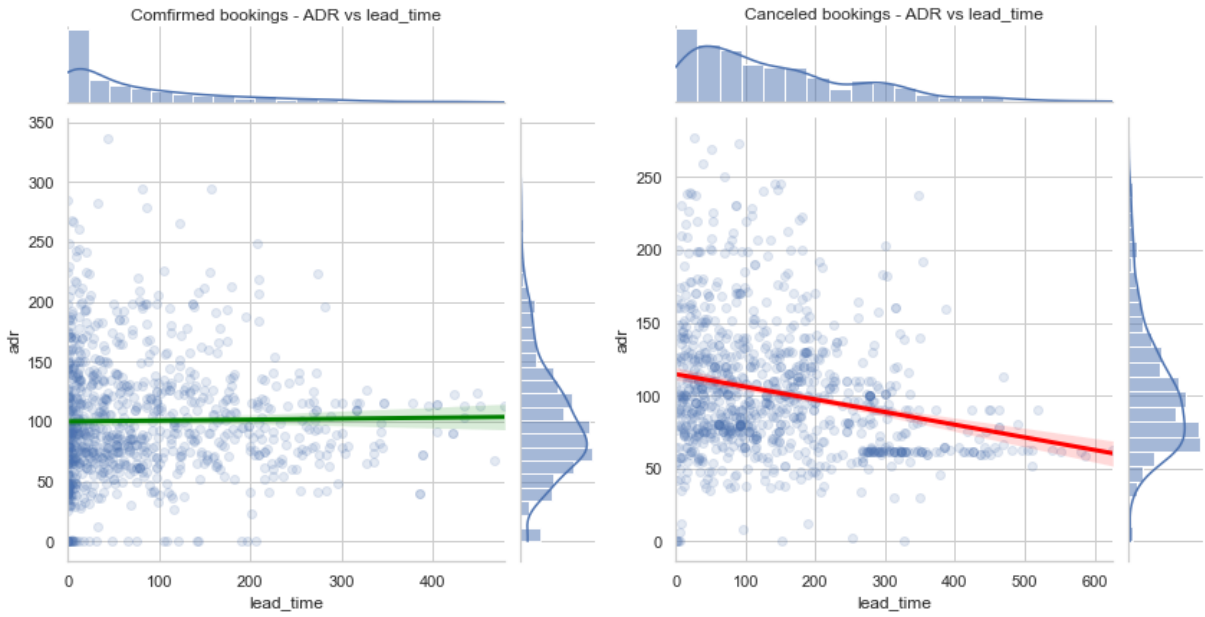
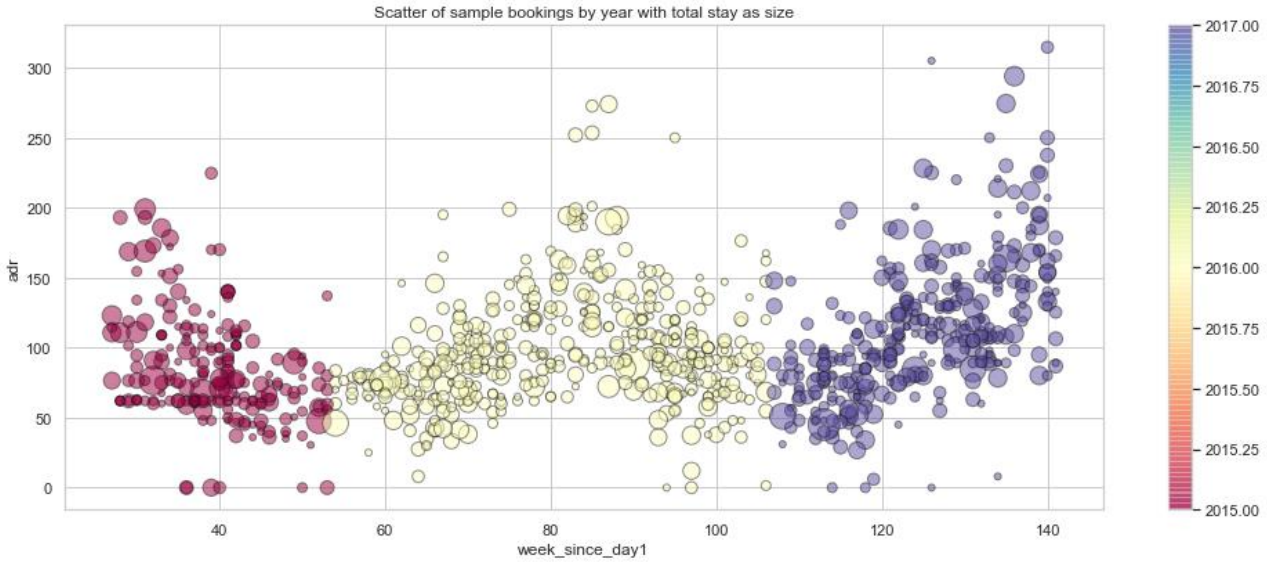


Figure 14 – Confirmed bookings vs Canceled bookings

We then analyzed a sample of 1000 bookings to see if we find any interesting findings. We had a look at:

- week number on **x** axis and **year of arrival** as color - for the temporal aspect
- **total stay** - we will add the weekend nights with weeknights - as points size
- **adr** on **y** scale.



Deposit question

In the last part of our EDA we wanted to explore and see if we can identify a reasoning behind one strange finding in our initial exploration in the Data Wrangling:

- *Why most of the customers with non-refundable deposits are canceling their bookings?*

We found out that there are 14,586 customers who did a non-refundable deposit, and according to the description it was a full pay for their stay. The big surprise was that 99.36% of these customers ended up canceling their booking.

It goes against the normal expectations the fact that 99% of the customers that already paid in full for their stay, knowing that they will lose all that money if they will cancel, will still eventually cancel the booking. Therefore, we continued exploring this situation, to see if we could identify what might have caused this.

Apart from the fact that none of the booking had any babies in it, nothing else seems a clear reason for this high cancelation rate. We checked the correlation heatmap for the customers who had non-refundable Deposits compared against the whole dataset heatmap, to see if we had any high values.

Compared to the main dataset , there were a few higher correlations, mostly negative. We were mainly looking after high positive or negative correlations differences between our target feature, is_canceled and any other feature.

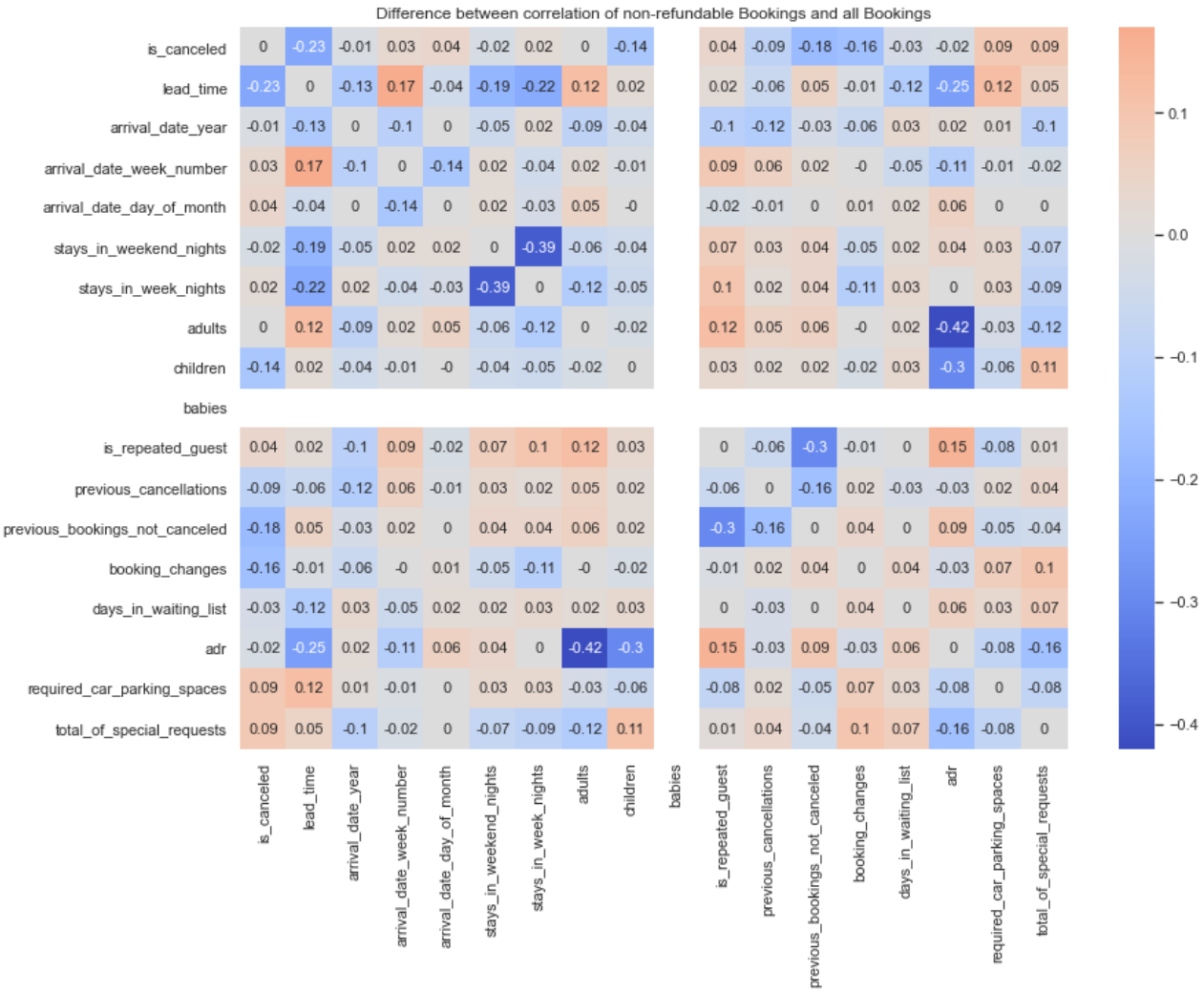


Figure 15 - Difference between Heatmaps of subset and that of all data

None of these values and correlations were not reasons enough to explain this extremely high cancelations rate in the subset.

The data we have does not contain are reasonable explanation for the fact that 99.36% of the customers who paid in full a non-refundable deposit ended up canceling their booking. We concluded that is either a company policy that caused this extremely high percentage or an error in feature description.

Pre-processing

In this part we manipulated our data to prepare it for our ML modeling from our final part.

Data Leakage Analysis

One of the main worries when using data trying to make predictions is not to have Data Leakage. To analyze that we had to go back on our data's source and analyze what the authors had to say about this.

> "One of the most important properties in data for prediction models is not to promote leakage of future information.
>
> In order to prevent this from happening, the timestamp of the target variable must occur after the input variables' timestamp. Therefore, instead of directly extracting variables from the bookings database table, when available, the variables' values were extracted from the bookings change log, with a timestamp relative to the day prior to arrival date (for all the bookings created before their arrival date)."

So, when data was collected, they tried to prevent data leakage as much as possible, but some features were affected by it because of the industry particularities.

> "In hotel industry it is quite common for customers to change their bookings' attributes, like the number of persons, staying duration, or room type preferences, either at the time of their check-in or during their stay. It is also common for hotels not to know the correct nationality of the customer until the moment of check-in. Therefore, even though the capture of data took considered a timespan prior to arrival date, it is understandable that the distribution of some variables differs between non canceled and canceled bookings."

Our initial suspicion was that some features, like total_of_special_requests, required_car_parking_spaces or booking_changes might have been affected by the changes a customer does at check-in or during their stay. To check that we had a look at the correlations between our target feature and all other numerical features, and nothing abnormally high came up.

And taking into consideration what the authors said in the comments, that the data was taken from their system with one day prior to the arrival, we believe that the correlation we see with our target feature are solely determined by the fact that customers that will keep their booking

have a tendency to make more special requests or require more car parking spaces before they check-in compared to the one who will cancel their booking.

We also checked the nationality aspects of our bookings because the author's comments were specific about a possibility of data leakage on this feature.

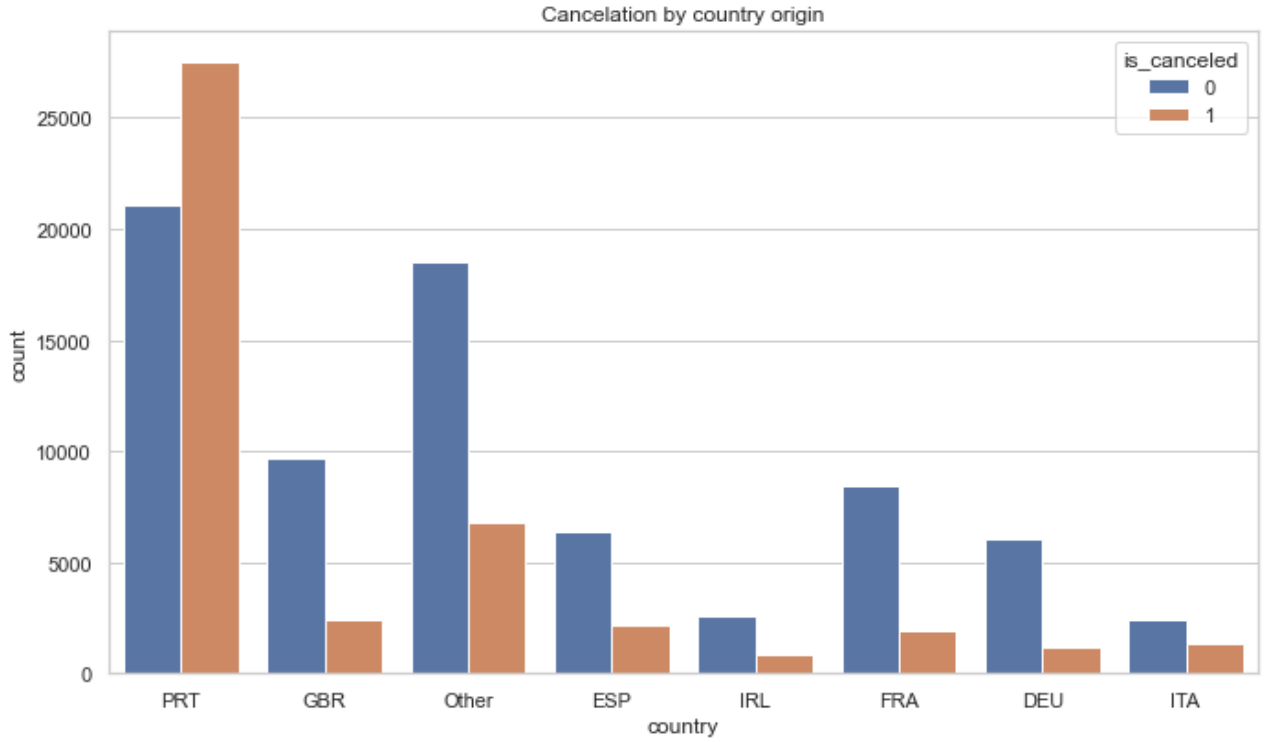


Figure 16 – Cancelation by country origin

As we can observe, a very high number of canceled bookings were being attributed to PRT. It seems that bookings that fail to show up are often marked as being from the home country. It is not until the arrival that the nationality becomes more accurate. This results in an uneven distribution of canceled bookings from the home country which is inaccurate and misleading for the model.
So we consider country to be a source of data leakage and we removed that feature.

Feature engineering

We began our Feature Engineering by creating a few other features that we considered might add value to the dataset:

1. arrival_day_of_week: using the arrival date year, month, and day we got the Day of Week.

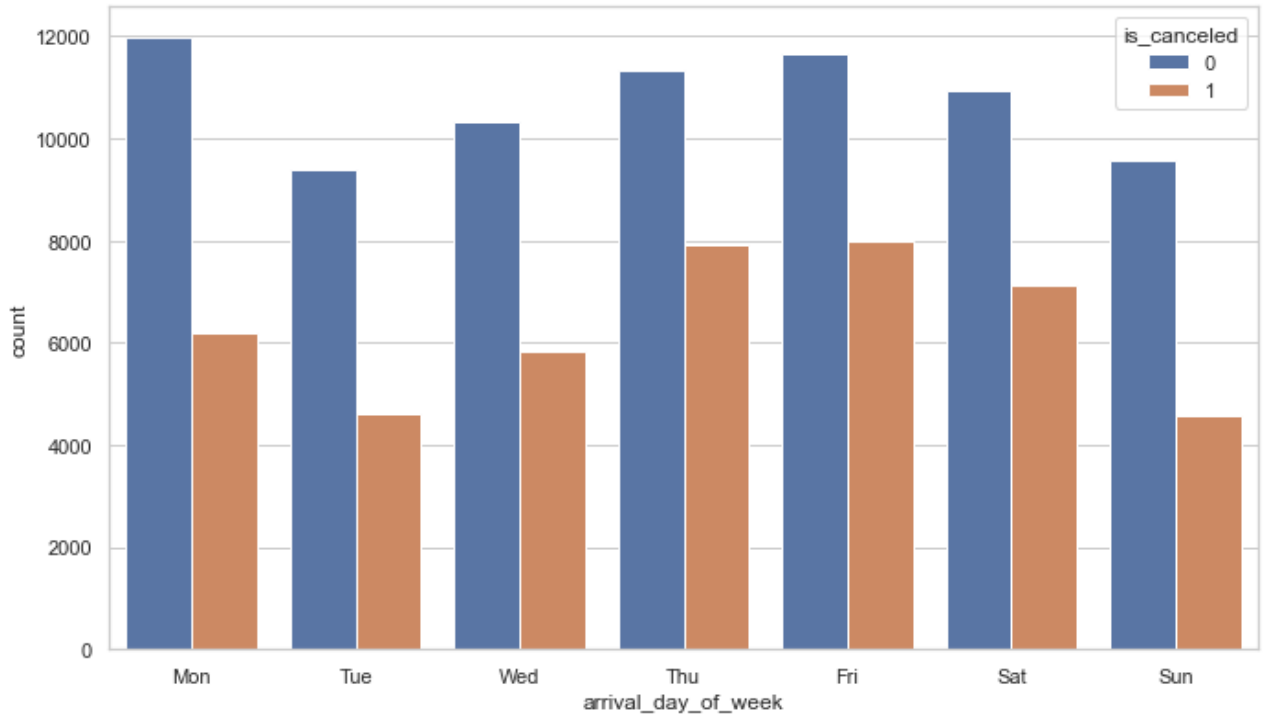


Figure 17 - Arrival Day of week

2. booking_purpose.

Using some domain knowledge and using the available data we will create a booking_purpose feature for all our bookings

While there might be a lot of different classifications depending on articles or point of views, we will just limit of classification to 5 major classes:

- **short_stay** - stays under a day - stays with 0 weeknights and 0 weekend night
- **business** - stays for a few weekdays - stays containing 1-5 weeknights and 0 weekend night
- **leisure** - stays for weekend and maybe 1 extra day - stays containing 0-1 weeknights and 1-2 weekend night
- **vacation** - stay up to 3 whole weeks - stays containing 2-15 weeknights and 1-7 weekend night
- **long_stay** - stays more than 3 weeks - stays containing 16+ weeknights or 8+ weekend night

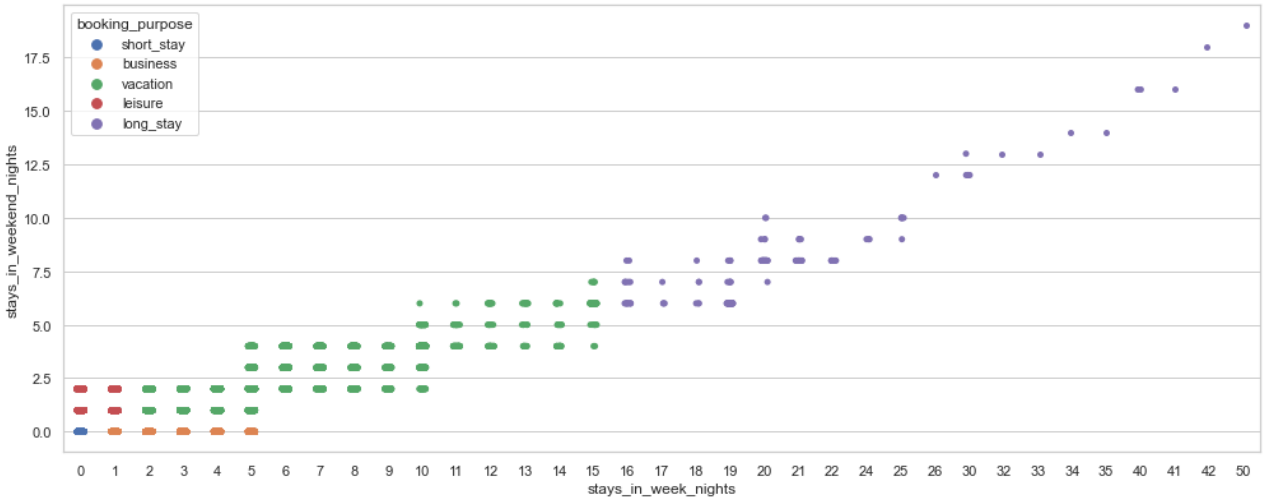


Figure 18 - Booking purpose feature

3. received_different_room

Because our dataset included both reserved and assigned room, we decided to keep just received room and create a binary feature named received_different_room.

Customer Segmentation – Unsupervised Learning

The last step in our Feature Engineering was to create a customer segmentation using an Unsupervised Learning model, and to use the created labels as input for our Supervised Learning Classification.

We used Panda's get_dummy encoder and from our initial 21 features we got a total of 50 features.

Then we split our data and trained the training set with a KMeans model and used the Elbow method and the Silhouette plotting and scores to decide on best model and parameters.

After modeling on raw data we got the following scores:

For n_clusters = 2 The average silhouette_score is : 0.5353149352905746
For n_clusters = 3 The average silhouette_score is : 0.4290141111617016
For n_clusters = 4 The average silhouette_score is : 0.39308834600714887
For n_clusters = 5 The average silhouette_score is : 0.3810518356349841
For n_clusters = 6 The average silhouette_score is : 0.3571291192764286

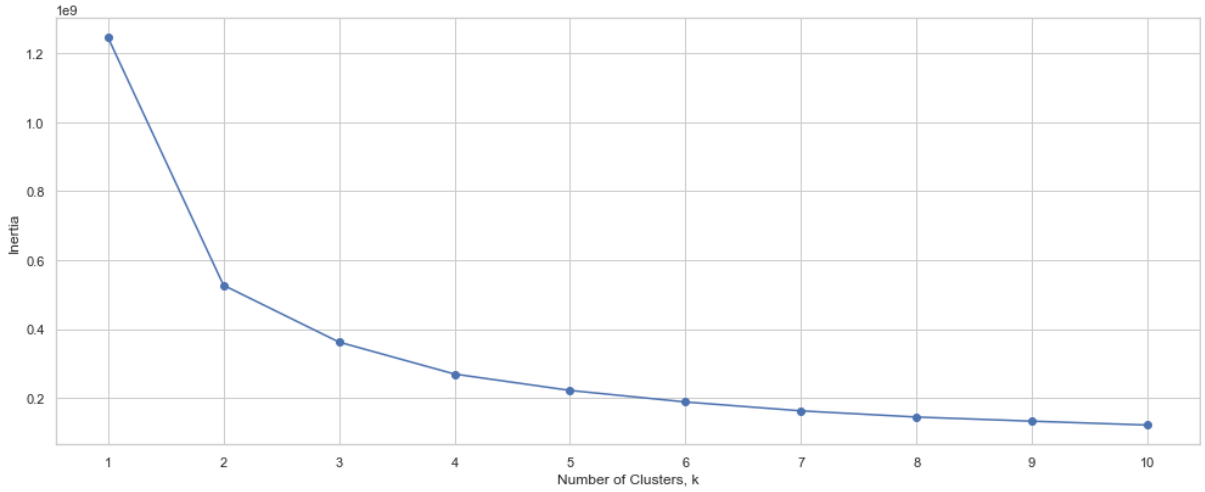


Figure 19 - KMeans on raw data

Then we used a StandardScaler and got the following scores and Elbow.

For n_clusters = 2 The average silhouette_score is : 0.2569508435615243
For n_clusters = 3 The average silhouette_score is : 0.11253907707395579
For n_clusters = 4 The average silhouette_score is : 0.13264970075366783
For n_clusters = 5 The average silhouette_score is : 0.13999638038942847
For n_clusters = 6 The average silhouette_score is : 0.1321686531093396

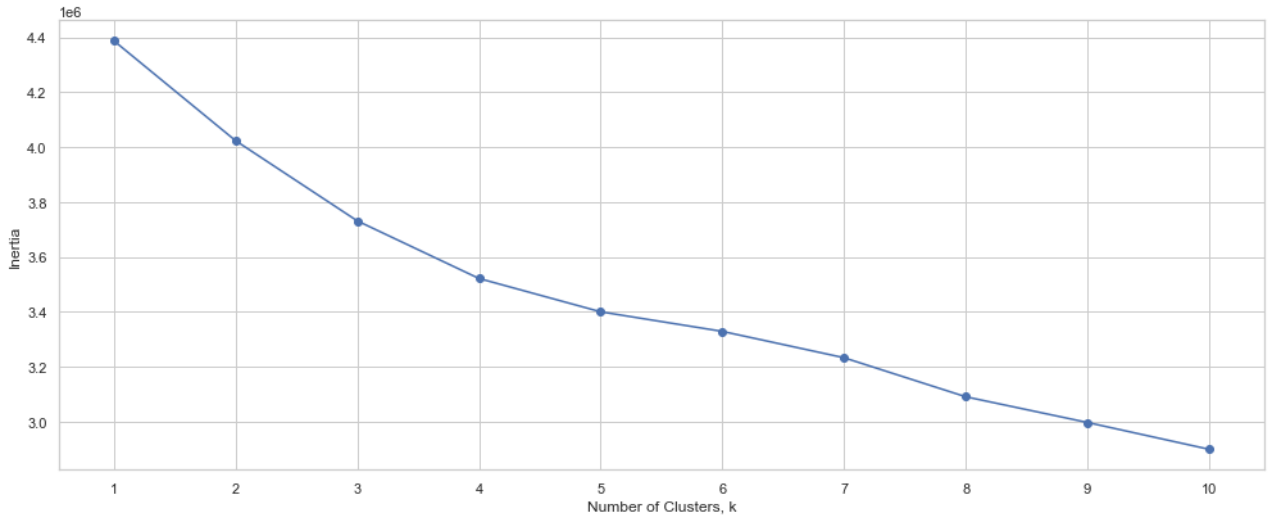


Figure 20- KMeans on Scaled data

Finally, we used PCA decomposition to reduce of dimensionality.

For our Unsupervised Clustering algorithm, we used just the first **4 Principal Components.**

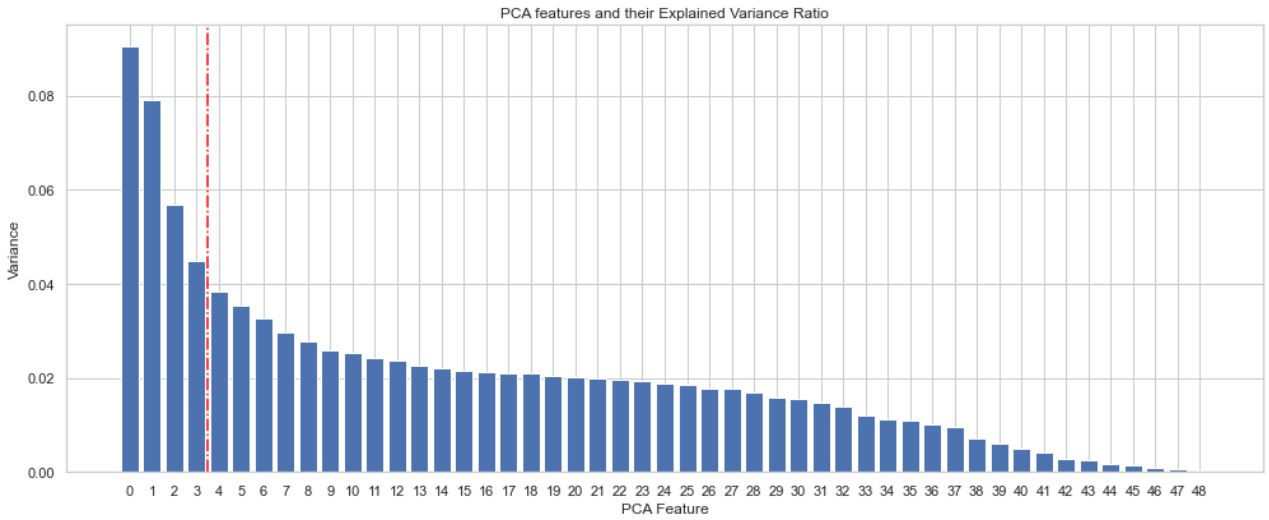


Figure 21 - Using 4 PCA features

And we got the following Elbow plot and Silhouette scores.

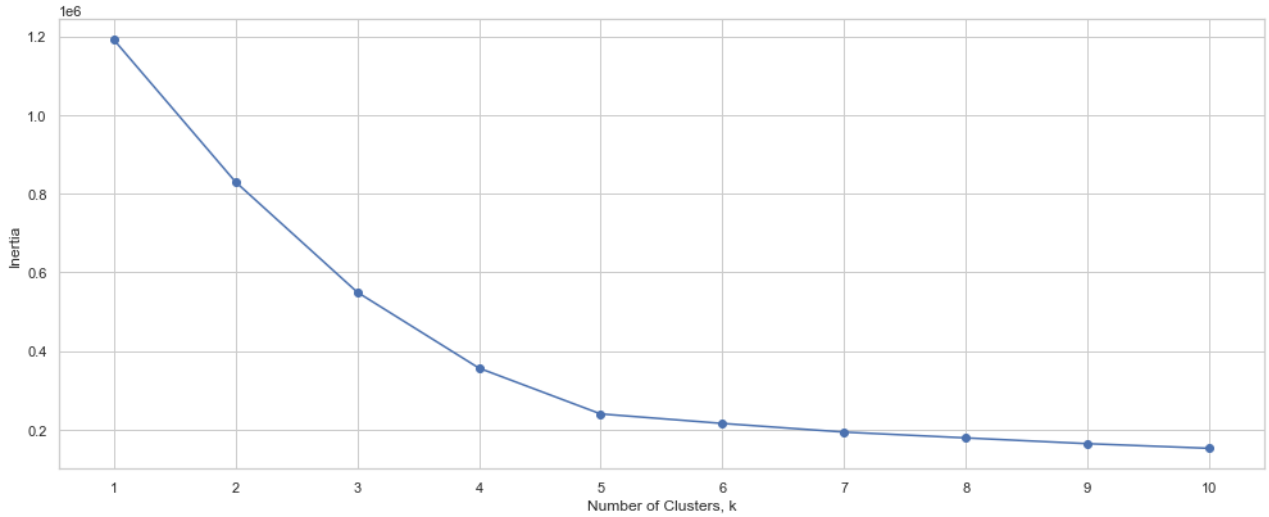


Figure 22 - Elbow after Scaler and PCA

For n_clusters = 2 The average silhouette_score is : 0.4758166007653855
For n_clusters = 3 The average silhouette_score is : 0.44452511390222194
For n_clusters = 4 The average silhouette_score is : 0.49483987143320063
For n_clusters = 5 The average silhouette_score is : 0.5427177449861779
For n_clusters = 6 The average silhouette_score is : 0.4677257498548569

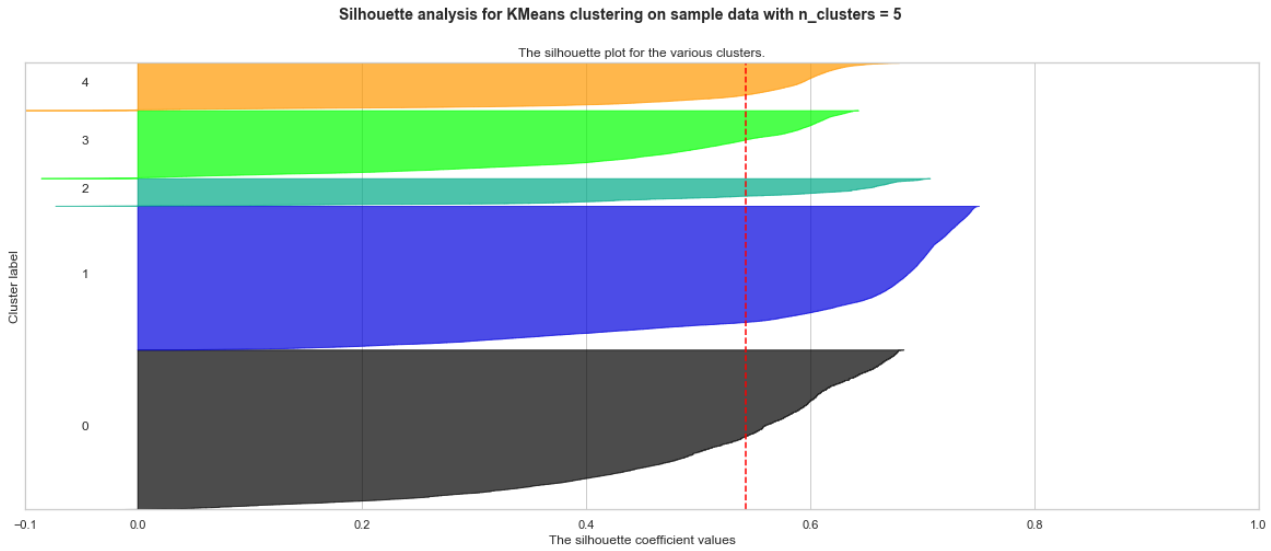


Figure 23 - Best Silhouette Score

The Silhouette scores after PCA were overall better that the scores from our raw data. The optimal clustering considering plot thickness and score is the clustering with 5 clusters, with a **0.54 Silhouette score**.

For a better representation of the clustering(segmentation) we plotted the cluster labels over the first 2 PCA features.

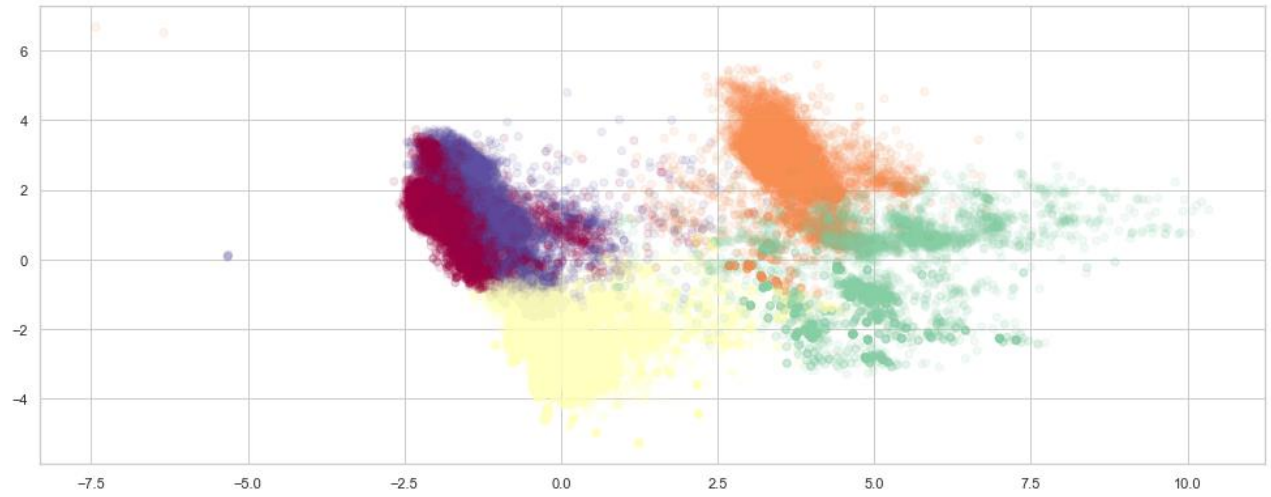


Figure 24 - Customer Segmentation

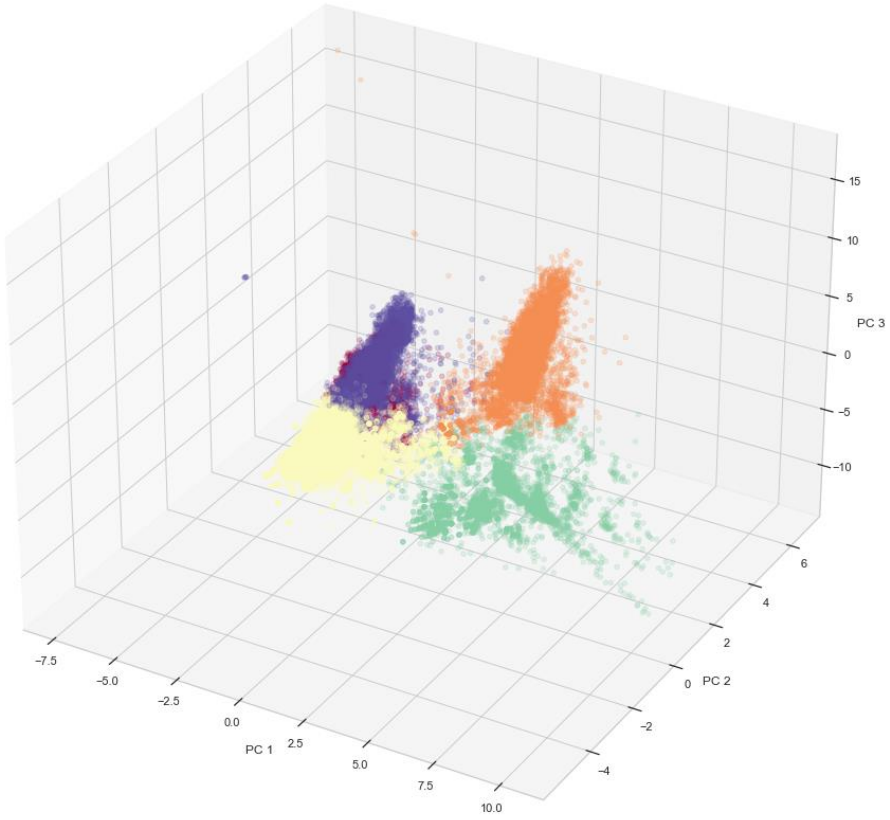


Figure 25 - Customer Segmentation plotted over first 3 PCA features

Concluding our pre-processing we saved our customer segmentation labels and saved our model and cleaned data.

Modeling

In this part we created Classification models using **SKLearn** and **Catboost** to model the data that was cleaned in the pre-processing steps, with the final goal to find the best model to predict future Hotel Bookings Cancelations.
With a wide diversity of Machine Learning models, we will start with some simple ones and then move to the Ensemble models.
We will adjust the hyper-parameters of these models to optimize them while taking into consideration overfitting, underfitting, and the bias-variance trade-off.
Finally, we will see how the model fitted on available data can predict if a new Booking will be canceled or not and with what percentage, right as the booking enters the management system.

Metrics selection

The main metric used was **Accuracy**, which gives the percentage of correct predictions, and the **area under the ROC curve** which measures how well the model can differentiate between classes.
We created a DataFrame to save the scores for both training and test sets so when we decided on choosing the right model, we were able to develop further analysis to identify models that were overfitting or underfitting on training data.
Beside the 2 mentioned metrics we also printed the classification report containing information about Recall and Precision and we plotted the normalized confusion matrix and the ROC curve.

Final pre-processing:

Before exploring the models, we can use for our classification problem we had to make sure our data was ready for this step. For that we had to consider the types of classifiers we were about to use:

- Algorithms that exploit distances or similarities (e.g., in the form of scalar product) between data samples, such as LinearRegression, k-NN and SVM and that are sensitive to feature transformations and cannot work with categorical data.
- Graphical-model based classifiers, such as Fisher LDA or Naive Bayes, as well as Decision trees and Tree-based ensemble methods (RF, XGB) are invariant to feature scaling, and can work with categorical data.

For ordinal categorical data we could have used **LabelEncoder**, but since our categorical data was nominal, we used solely **Panda's get_dummies**.
Then we split the data into **Train and Test** and Used a **StandardScaler**.

Logistic Regression

Out of the Box

We used at first an Out of the Box model.

```
Train - Accuracy: 0.8211
Test - Accuracy: 0.8224
Validation - Classification report
              precision    recall  f1-score   support

           0       0.82      0.92      0.87     22691
           1       0.82      0.66      0.73     13126

    accuracy                           0.82     35817
   macro avg       0.82      0.79      0.80     35817
weighted avg       0.82      0.82      0.82     35817
```

```
Train - Area under ROC score: 0.8761
Test - Area under ROC score: 0.8765
```

```
<Figure size 648x648 with 0 Axes>
```

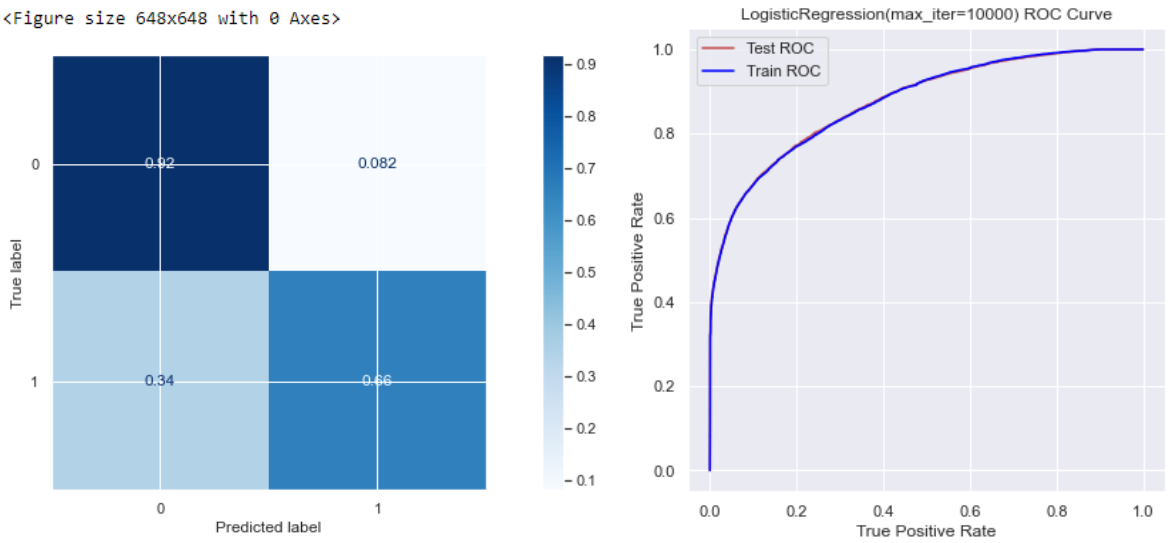


Figure 26 - Out of the box LogReg

Logistic regression with PCA

Secondly, we used PCA decomposition to see how many PCA Features we can use, and we picked 50%, 70% 90% and 99% explained variance ratio thresholds.

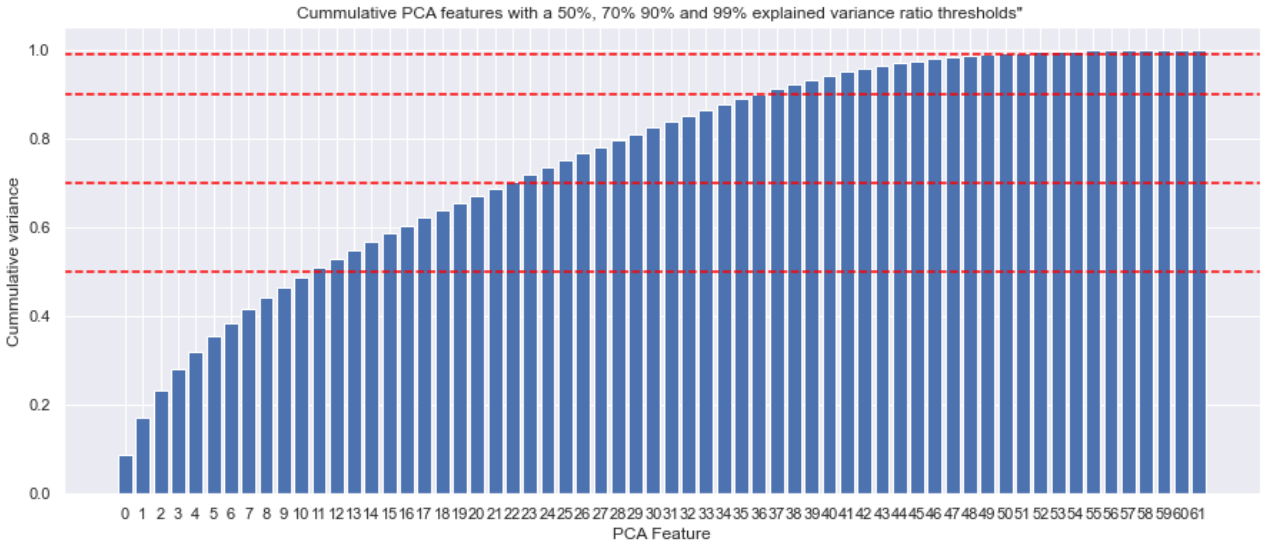


Figure 27 - PCA for LogisticRegression

Based on our PCA decomposition we choose the following number of principal components:

- 5 - for about 30% explained variance
- 11 - for >50% explained variance
- 22 - for >70% explained variance
- 37 - for >90% explained variance
- 50 - for >99% explained variance

We had to consider the fact that a model based on fewer features will be underfitting while one based on a lot of them will usually overfit.
Another aspect when considering underfitting vs overfitting is the dataset size, and in this case the training data has 80k+ records, so considering we are dealing with a rather big dataset we were able to choose more features without a high risk of overfitting. And we could check that by comparing the metrics from our training set versus the scores our models produced on the test set.

```
----------------------Train & Test scores : Log reg model with 50 PCA features-------------------
Train - Accuracy:  0.8209
Test - Accuracy:  0.8224
Train - Area under ROC score: 0.8742
Test - Area under ROC score: 0.8747

              precision    recall  f1-score   support

           0       0.82      0.92      0.87     22691
           1       0.83      0.65      0.73     13126

    accuracy                           0.82     35817
   macro avg       0.82      0.79      0.80     35817
weighted avg       0.82      0.82      0.82     35817

<Figure size 648x648 with 0 Axes>
```

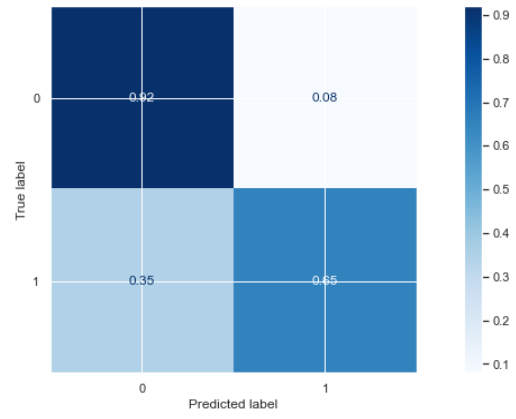


Figure 28- The Scores and Confusion Matrix for Logistic Regression with 50PCA features

We observed that the higher the number of PCA features the higher our scores are.
But there was no improvement in the accuracy of our model so for the further modeling we kept the data without the PCA transformation.

Hyperparameter Tunning with GridSearchCV and RandomizedSearchCV

For our next step we used the same model, but we did **Hyperparameter Tunning** on alpha, using both **GridSearchCV** and **RandomizedSearchCV**.
It is known that the randomized method can perform almost as well as the grid search and considering that we were working with a rather large dataset that can cause high processing times, it is likely that the RandomizedSearchCV will be preferred, but we wanted to compare how well the 2 methods perform.
But here we could see as well that the observed ROC_AUC and Accuracy scores were not significantly different from the ones observed with the out of the box model.

Logistic regression with Ridge and Lasso regularization

Furthermore, we used the 2 types of regularizations: Ridge and Lasso on or Logistic Regression classifier, and for each of these we will use different alpha values.
Even after using both types of regularizations on our LogisticRegression model, with cross validations on multiple alphas, the model's scores remain relatively constant, without improving the model performance significantly.

KNN Classifier

We used a KNeighborsClassifier with a variation of n_neighbors parameters.

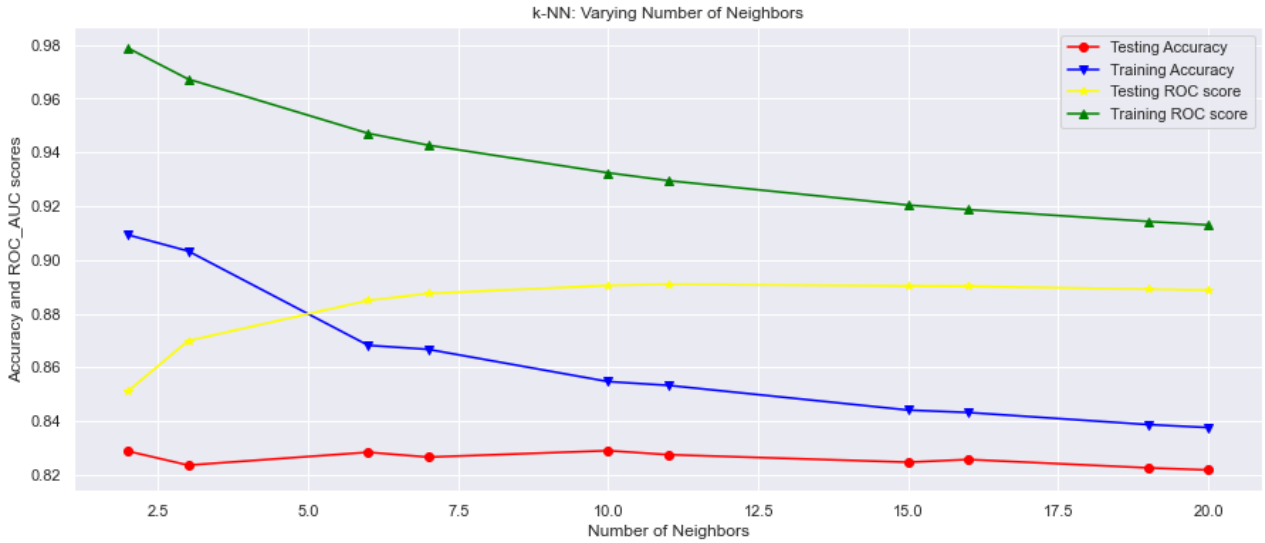


Figure 29 - KNN Scores

We picked the optimal KNeighborsClassifier model to be the model with n_neighbors = 10 because:

- it has the highest test score.
- is the number of n_neighbors where the roc_auc score begins to become constant.

The main issue with this model is that it can take around 30 minutes for it to fit to data and predict for just one value of n, considerably more than any other models used so far.
But all our KNN models performed better than the LogisticRegression.

Random Forest Classifier

Using Random Forest Classifier improves the usability of the model because of its ability to handle non-linear features and outliers. The data we can use will not have to be binned or scaled, making it useful for raw data to be used for direct model predictions.

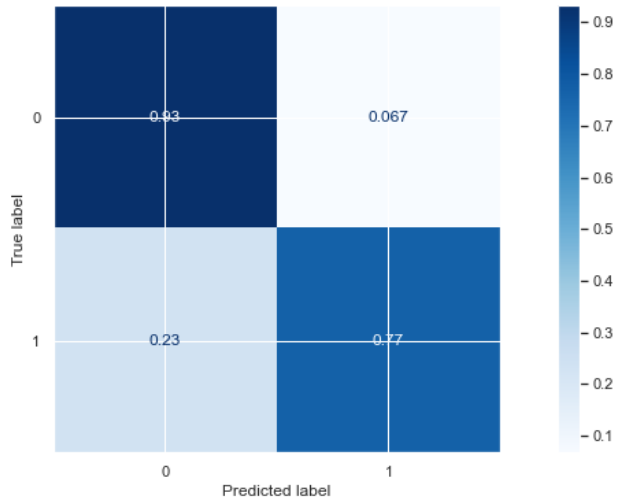
Out of the box Random Forest Classifier

```
Train - Accuracy: 0.9922
Test - Accuracy: 0.8724
Validation - Classification report
              precision    recall  f1-score   support

           0       0.87      0.93      0.90     22691
           1       0.87      0.77      0.82     13126

    accuracy                           0.87     35817
   macro avg       0.87      0.85      0.86     35817
weighted avg       0.87      0.87      0.87     35817
```

<Figure size 648x648 with 0 Axes>




```
Train - Area under ROC score: 0.9991
Test - Area under ROC score: 0.9371
```

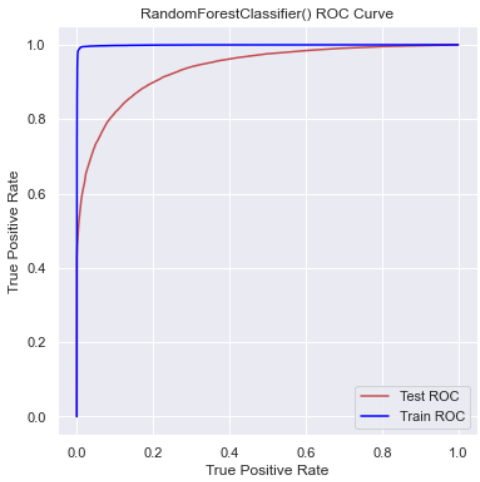


Figure 30 - Base RFC

Using an Out of the box Random Forest model we observed a big increase in scoring metrics from the KNN and LogisticRegression classifiers used previously.
We can notice as well that the out of the box model is overfitted on our training data, having extremely high accuracy and roc_auc scores.

Hyperparameter tunning with RFC

We did a grid search hyperparameter tunning for RFC on max_depth and number of estimators.

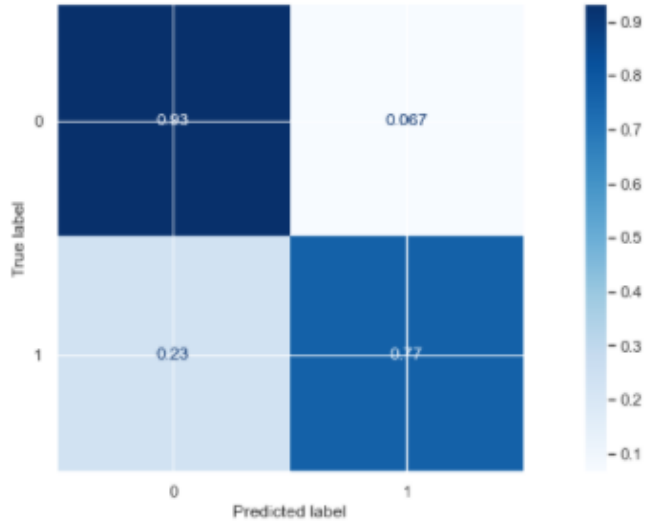
The base model slightly improved its Accuracy and roc_auc score with increased estimators.

```
---------Train & Test scores : RFC model with n_estimators = 200 and max_depth = None ---------
Train - Accuracy: 0.9922
Test - Accuracy: 0.8726
Validation - Classification report
              precision    recall  f1-score   support

           0       0.87      0.93      0.90     22691
           1       0.87      0.77      0.82     13126

    accuracy                           0.87     35817
   macro avg       0.87      0.85      0.86     35817
weighted avg       0.87      0.87      0.87     35817

<Figure size 648x648 with 0 Axes>
```




```
Train - Area under ROC score: 0.9991
Test - Area under ROC score: 0.9375
```

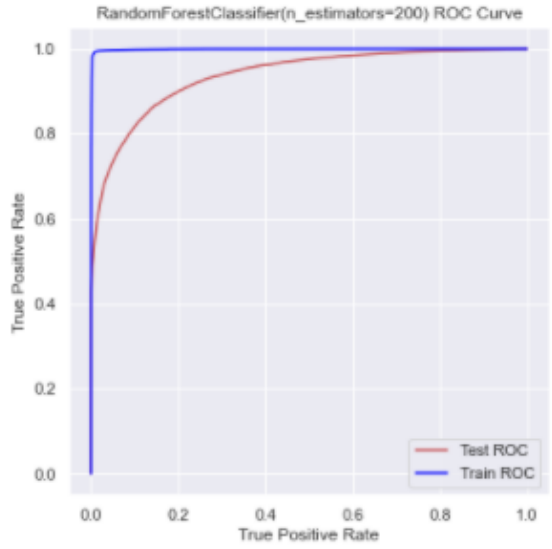


Figure 31 - Best RFC

Feature importance

To analyze the feature importance in our Random Forest Classifier we used 2 methods:

- permutation importance on our Test data.

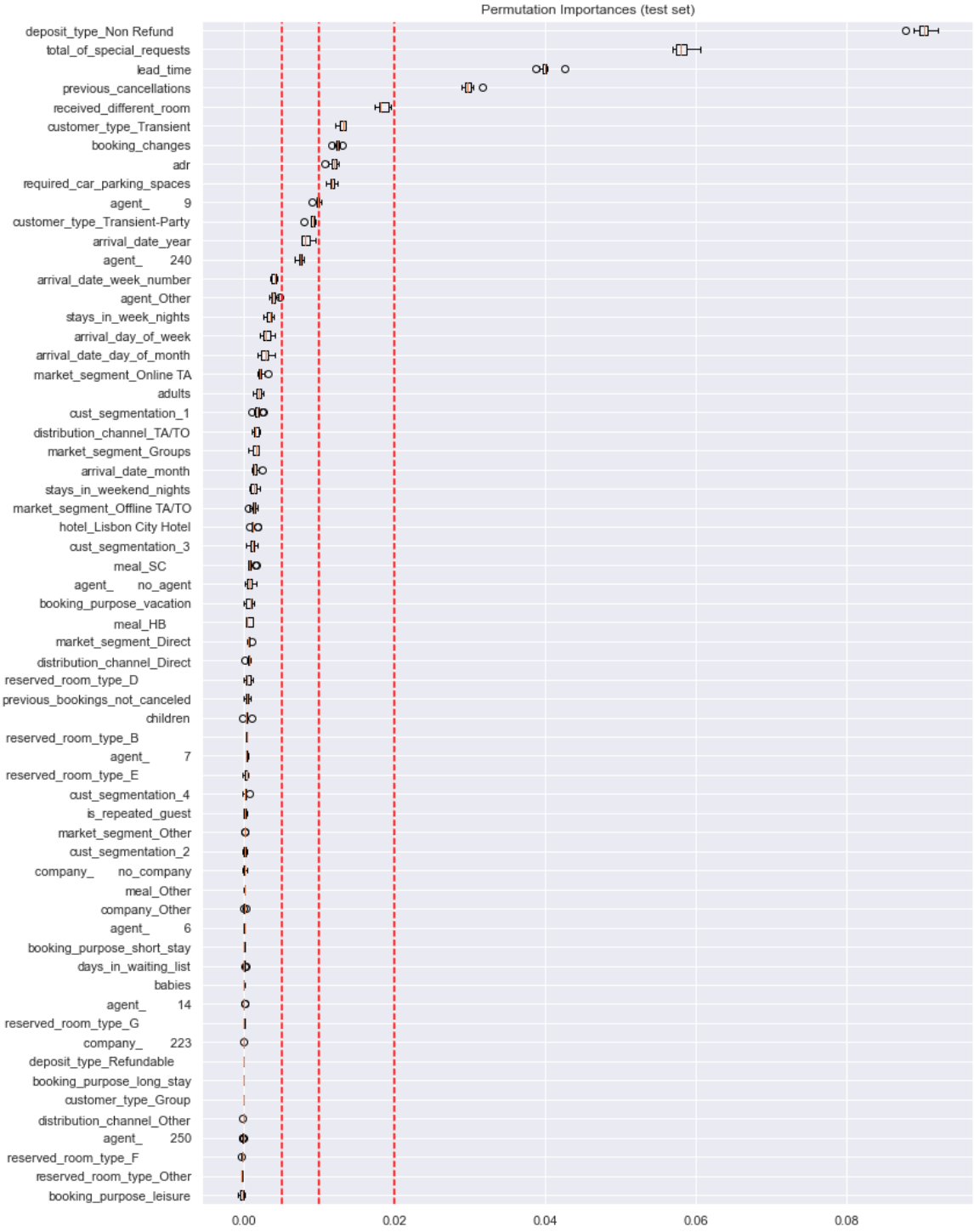


Figure 32 - Feature importance using permutation_importance

We identified 3 different thresholds on which we could select our top features based on the features' permutation test:

- 0.02 - top 4 features
- 0.01 - top 9 features
- 0.005 - top 13 features

❖ Feature_importances_

A secondary method to see how the features impacted the model is using feature_importances_. Plotting the feature importance is a way that you can gain a perspective on which features are driving the model predictions.

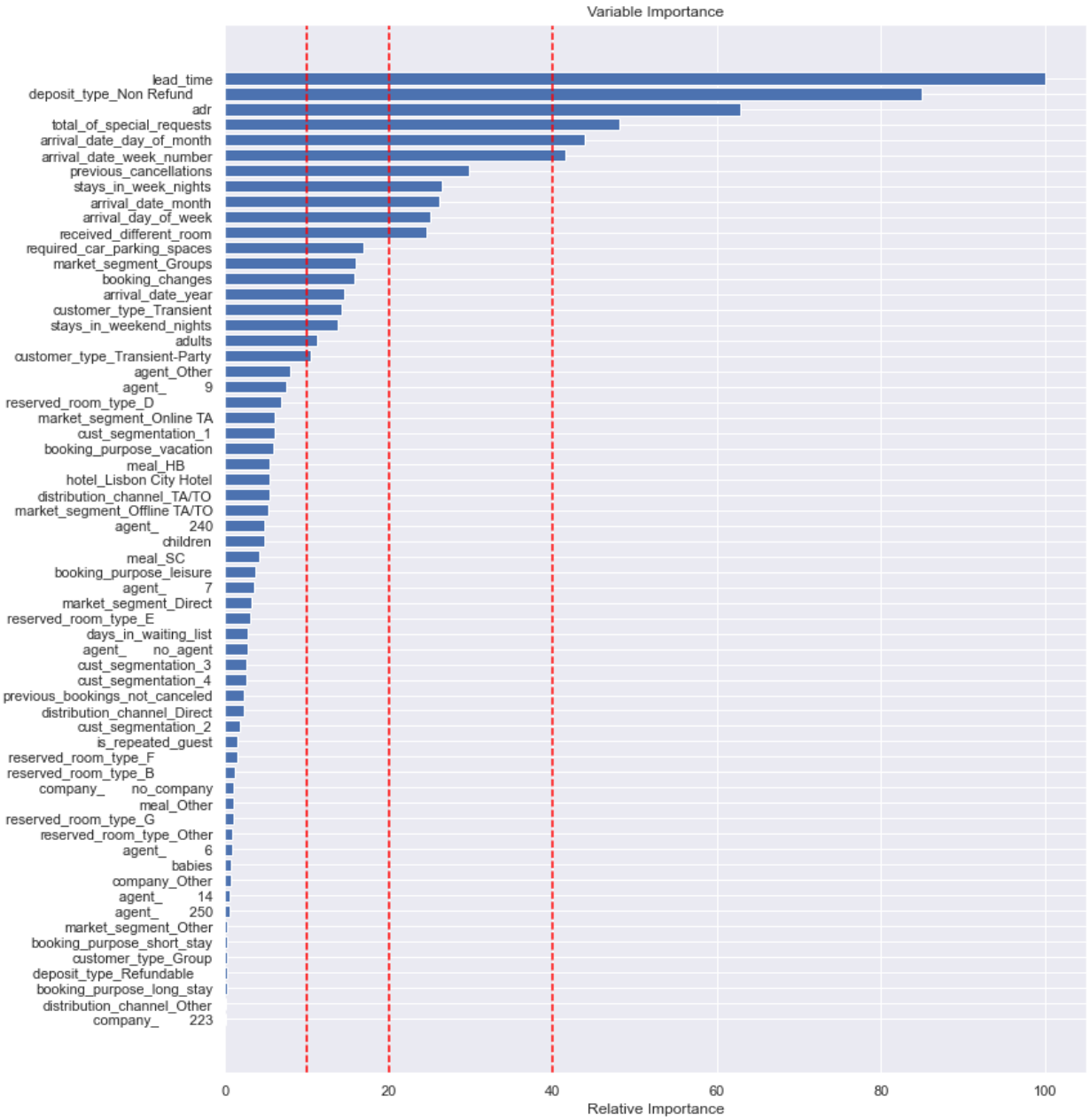


Figure 33 - Feature importances

We have some different thresholds on which we could select our top features:

- 10% - top 19 features
- 20% - top 11 features
- 40% - top 6 features

Catboost Classifier

The big advantage of this model over any model from SKlearn is the fact that it is able to work on categorical data. Other advantages are that it is faster than most other models, and generally it produces very high scores.

Figure 34 - Catboost model

Scoring Ranking and Model Selection

	model	train_accuracy	test_accuracy	train_ROC	test_ROC
28	(DecisionTreeClassifier(max_features='auto', random_state=1233902736), DecisionTreeClassifier(ma...	0.9922	0.8726	0.9991	0.9375
19	(DecisionTreeClassifier(max_features='auto', random_state=259966552), DecisionTreeClassifier(max...	0.9922	0.8724	0.9991	0.9371
27	(DecisionTreeClassifier(max_features='auto', random_state=64402675), DecisionTreeClassifier(max_...	0.9922	0.8718	0.9991	0.9369
26	(DecisionTreeClassifier(max_features='auto', random_state=1646429223), DecisionTreeClassifier(ma...	0.9920	0.8713	0.9991	0.9352
33	(DecisionTreeClassifier(max_features='auto', random_state=446869832), DecisionTreeClassifier(max...	0.9919	0.8694	0.9991	0.9332
34	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9256	0.8612	0.9853	0.9300
25	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9038	0.8604	0.9771	0.9323
24	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9034	0.8598	0.9764	0.9313
23	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9045	0.8592	0.9767	0.9300
31	(DecisionTreeClassifier(max_features='auto', random_state=446869832), DecisionTreeClassifier(max...	0.9909	0.8505	0.9988	0.9221
35	Catboost model	NaN	0.8485	NaN	NaN
32	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9199	0.8436	0.9840	0.9171
13	KNeighborsClassifier(n_neighbors=10)	0.8547	0.8290	0.9324	0.8905
9	KNeighborsClassifier(n_neighbors=2)	0.9093	0.8288	0.9788	0.8512
11	KNeighborsClassifier(n_neighbors=6)	0.8682	0.8284	0.9471	0.8849
29	(DecisionTreeClassifier(max_features='auto', random_state=446869832), DecisionTreeClassifier(max...	0.9867	0.8281	0.9981	0.8926
14	KNeighborsClassifier(n_neighbors=11)	0.8533	0.8275	0.9295	0.8909
12	KNeighborsClassifier(n_neighbors=7)	0.8667	0.8266	0.9427	0.8875
16	KNeighborsClassifier(n_neighbors=16)	0.8432	0.8257	0.9187	0.8902
30	(DecisionTreeClassifier(max_depth=20, max_features='auto',\n random_state=...	0.9327	0.8249	0.9884	0.8893
15	KNeighborsClassifier(n_neighbors=15)	0.8441	0.8247	0.9204	0.8903
22	(DecisionTreeClassifier(max_depth=10, max_features='auto', random_state=65203512), DecisionTreeC...	0.8257	0.8246	0.9092	0.9016
21	(DecisionTreeClassifier(max_depth=10, max_features='auto',\n random_state=...	0.8264	0.8245	0.9093	0.9015
20	(DecisionTreeClassifier(max_depth=10, max_features='auto',\n random_state=...	0.8249	0.8237	0.9084	0.8990
10	KNeighborsClassifier(n_neighbors=3)	0.9033	0.8236	0.9673	0.8699
17	KNeighborsClassifier(n_neighbors=19)	0.8387	0.8226	0.9143	0.8891
4	LogisticRegression(C=0.31622776601683794, max_iter=10000, solver='saga')	0.8208	0.8225	0.8760	0.8763
0	LogisticRegression(max_iter=10000)	0.8211	0.8224	0.8761	0.8765
6	LogisticRegression(C=0.01, max_iter=10000, penalty='l1', solver='saga')	0.8206	0.8223	0.8741	0.8745
1	GridSearchCV(cv=5, estimator=LogisticRegression(max_iter=10000),\n param_grid={'C': ...	0.8212	0.8222	0.8762	0.8766
7	LogisticRegression(C=0.31622776601683794, max_iter=10000, penalty='l1',\n solv...	0.8206	0.8222	0.8760	0.8764
8	LogisticRegression(C=10.0, max_iter=10000, penalty='l1', solver='saga')	0.8210	0.8221	0.8761	0.8765
5	LogisticRegression(C=10.0, max_iter=10000, solver='saga')	0.8210	0.8221	0.8761	0.8765
2	RandomizedSearchCV(estimator=LogisticRegression(max_iter=10000),\n param_distr...	0.8211	0.8221	0.8761	0.8765
18	KNeighborsClassifier(n_neighbors=20)	0.8376	0.8218	0.9130	0.8887
3	LogisticRegression(C=0.01, max_iter=10000, solver='saga')	0.8198	0.8216	0.8748	0.8752

Figure 35 Models Ranking on Test Accuracy

Concluding, we will select for our booking's cancelations prediction a DecisionTreeClassifier with the following hyperparameters: **max_depth= None** and **n_estimators=200**. The model has the highest score with **0.8726** Accuracy score and **0.9375** roc_auc scores on validation set.

Future model implementation and usability

For model implementation we must take into consideration the fact that the sooner you have the information regarding a high cancelation possibility the better. Therefore, the soonest one can have this information is at the time of booking creation. Obviously, between the time of creating the booking and the time of checking-in a lot of features will be modified and other will be created(such as lead_time, received_different_room).
But there is the crucial need for an initial assessment of the booking because it allows an early interaction with the customer.

Here are 2 possible models that can be deployed.

1. **Front Desk Cancelation Percentage model**.
The first one will be a model that takes as input just data from the moment the booking is created, and immediately returns percentages of this booking being canceled. If the percentage is within certain threshold, the concierge(or an automated bot in case of an online booking) will offer a particular discount or bonus.

2. **Monthly Cancelations Assessment.**
The second model is to be run every month and contains all other features available. The model runs all bookings with arrival data within the next month and outputs the bookings that are likely to be canceled so they can be targeted with a special marketing campaign. This second model runs on all features we have trained our model on so far, so there is no need to further explore this avenue.

```
new_booking
```

```
{'hotel': 'Algarve Resort Hotel',
 'arrival_date_year': 2019,
 'arrival_date_month': 7,
 'arrival_date_day_of_month': 12,
 'stays_in_weekend_nights': 4,
 'stays_in_week_nights': 5,
 'adults': 2,
 'children': 1,
 'babies': 0,
 'meal': 'BB',
 'market_segment': 'Direct',
 'distribution_channel': 'TA/TO',
 'is_repeated_guest': 1,
 'previous_cancellations': 1,
 'previous_bookings_not_canceled': 4,
 'reserved_room_type': 'A',
 'deposit_type': 'No Deposit',
 'agent': '250',
 'company': 'no_company',
 'arrival_day_of_week': 4,
 'arrival_date_week_number': 28,
 'booking_purpose': 'vacation'}
```

Front Desk Cancelation Percentage model

In our initial booking we will only have access to a limited number of features, so the first step was to select just those available in the moment of booking creation. Secondly, we used features engineering techniques to transform the booking information and create the extra features we will be needing in the modeling part.

Next, we manually entered a random booking, and we use our best model to make prediction on the likelihood of the cancelation.

```
booking_prediction = best_model.predict(booking_w_dummy)
print('Customer is likely to cancel the booking' if booking_prediction[0]==1 else 'Customer is likely to keep the booking')
```

```
Customer is likely to cancel the booking
```

```
y_pred_prob = best_model.predict_proba(booking_w_dummy)[:,1]
print('Customer is {} % likely to cancel the booking'.format(round(y_pred_prob[0]*100,4)))
```

```
Customer is 52.0 % likely to cancel the booking
```

Figure 36 - New booking Cancellation prediction

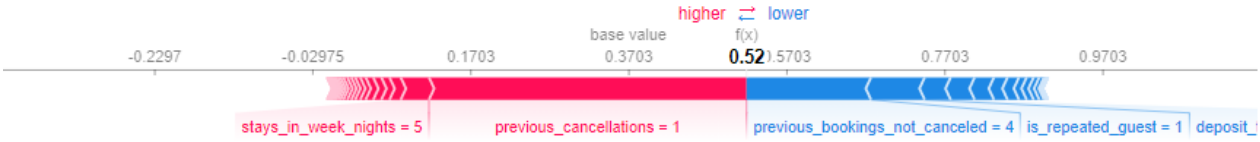
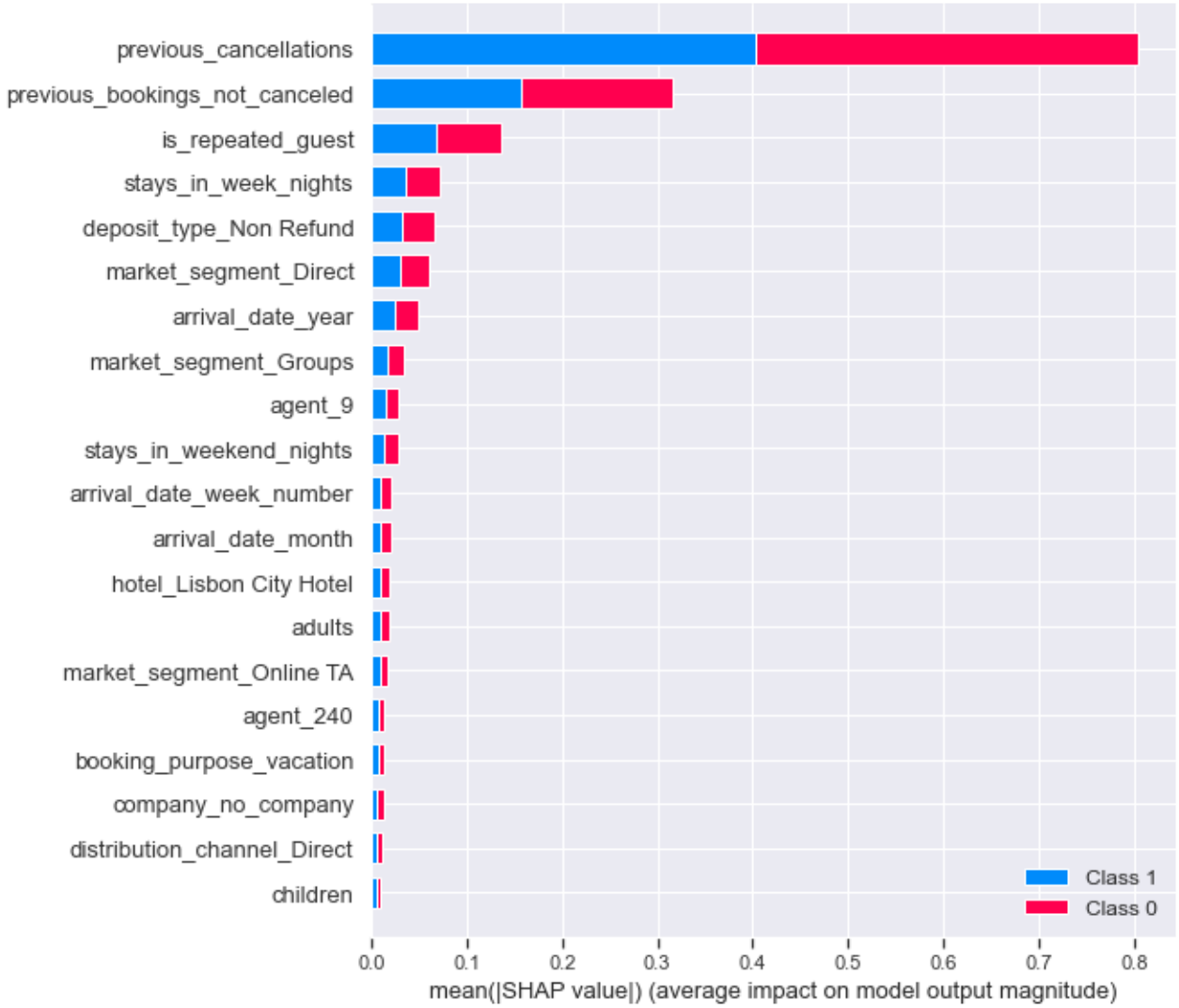


Figure 37 - Features importances on the obtained Cancelation likelihood

We can observe that for this booking, the customer is 52% likely to cancel.
Using Shap library we also identified the average impact on model output, and how the features values influenced the obtained prediction.



Recommendations

As we described earlier, we recommend implementing 2 variants of this model: the Front Desk Cancelation Likelihood model(percentage based) and the Monthly Cancelations Assessment.

For the Front Desk model, we went ahead and created a template of Marketing offers and strategies based on 2 criteria:

- Cancelations Likelihood
- Existing customer

We based our Course of Action Decision Tree on the idea that everyone likes to qualify for a discount or promotion(and therefore their attachment to the booking increases and are more likely to not cancel), so we come up with the following types of Marketing Campaigns:

- Referral Campaign
- Customer Loyalty – 5 levels.
- New Customer Discount – 4 levels.

The Course of Action after the Monthly Cancelations Assessment can be something simple like just sending an e-mail to those who are likely to cancel, something like *"Hey, here are some places to visit while staying in our hotel. […] Congratulations for qualifying for our marketing campaign. Looking forward to seeing you soon!".* Or depending on hotel availability, additional offers may be included(price reductions, amenities, room upgrades).

Future work

One big question remained unanswered in our project(see EDA section or notebook): Why are 99% of customers who are paying in full for their booking end up cancelling their reservation? Is that connected to the hotel policy? It is one agency or company that forces customers to make non-refundable deposits and then somehow almost everybody cancels their reservation? Or is the feature description from the DirectScience.com inaccurate?

Furthermore, we can continue our modelling work by including various other models to see how these models perform compared to the best model we found so far, the Random Forest Classifier:

- Light GBM
- XGBoost
- SVC
- AdaBoost

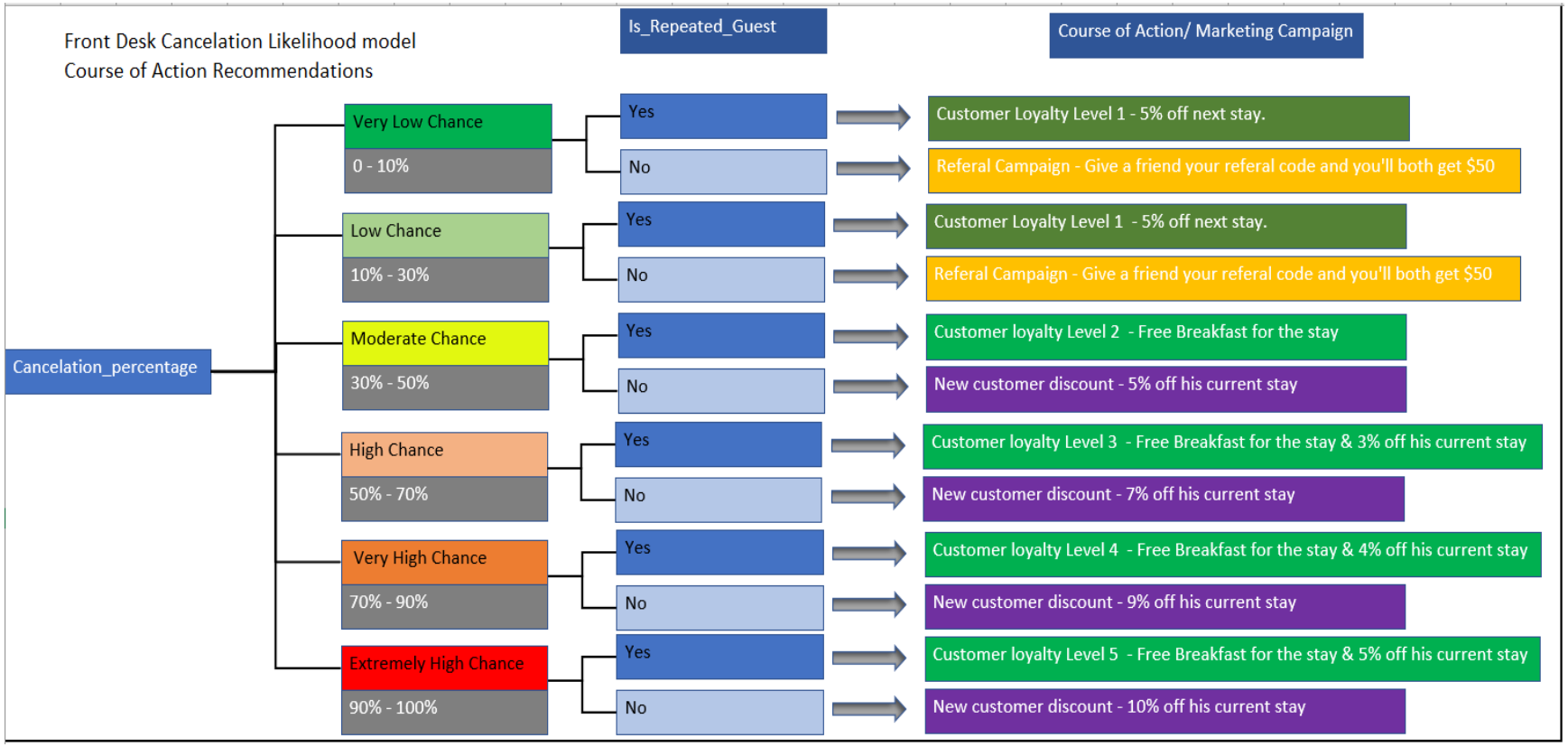


Figure 38 - Recommended Marketing Campaigns Targeting

Another way to develop our future work would be to create a better interactivity when entering a new booking or showing recommendations. Now this was done through a series of input commands in Jupiter Notebook. We can use python's library **streamlit** for further model deployment. Unfortunately, right now streamlit is incompatible with Jupyter Notebooks so further exploration must be done outside these notebooks.

Lastly, after implementing the Marketing Campaigns recommendations, how customers respond to these Campaigns and end up canceling or not their booking is data that can be used to analyze how impactful are the Campaigns and how to do better Customer Targeting using Data Science.

We are sure that implementing Booking Cancelation Prediction models will be useful used along with a wide range of Marketing and Business policies and that will certainly decrease the overall Cancelation rate leading to higher Retention rate and higher revenue while operating under the same costs.