HOTEL BOOKING ANALYSIS

❖ PROBLEM STATEMENT:

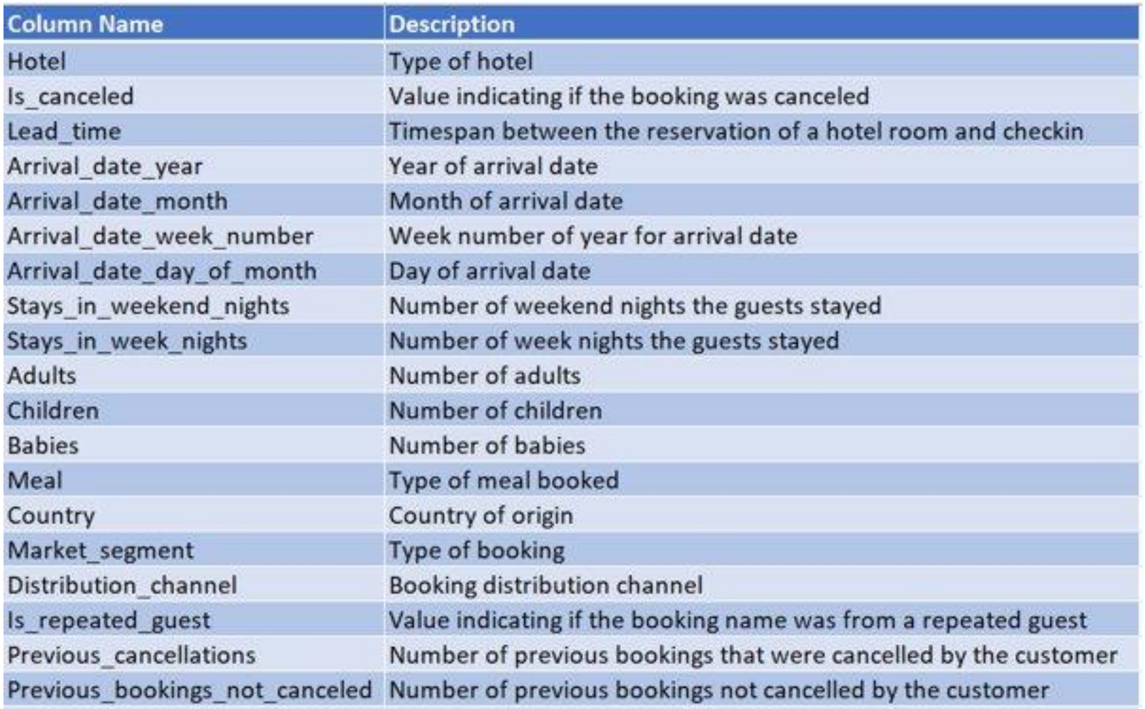
There are two aspects of this project that we are focusing to solve. First, we are forecasting the **Average Daily Rate** for a set of hotels based in Lisbon, Portugal. Secondly, we are predicting **whether a particular booking will be canceled or not**.

❖ INTRODUCTION:

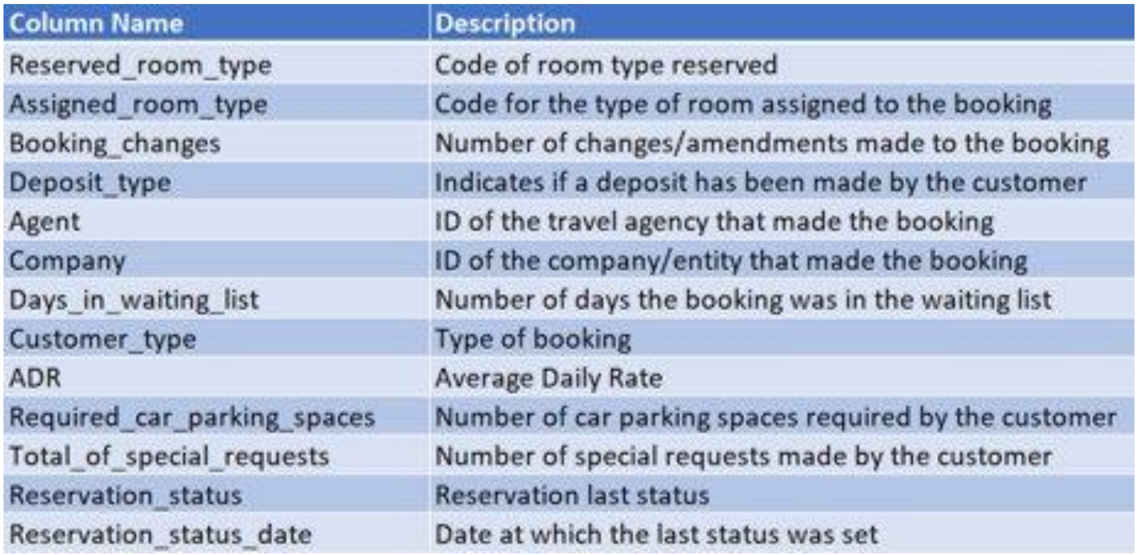
The tourism industry is very dynamic and is easily affected by many factors such as natural disasters, economic situations of varying countries, pandemics, etc. Therefore, any insight on what to expect in the coming days can be very beneficial for hotel owners and managers. Forecasting the average daily rate will give hotel owners some idea of what their profits or losses will look like. The same goes for having a prediction for how many bookings they can expect to retain and therefore they can then estimate what their profits may look like.

❖ DATA DESCRIPTION:

We are given one csv file with several columns as shown in the images below. As mentioned before, the hotels are all located in **Lisbon, Portugal**. Also, there are two main types of hotels that are included in this dataset: **city hotel** and **resort hotel**. Every hotel that has been included falls under one of these two categories.

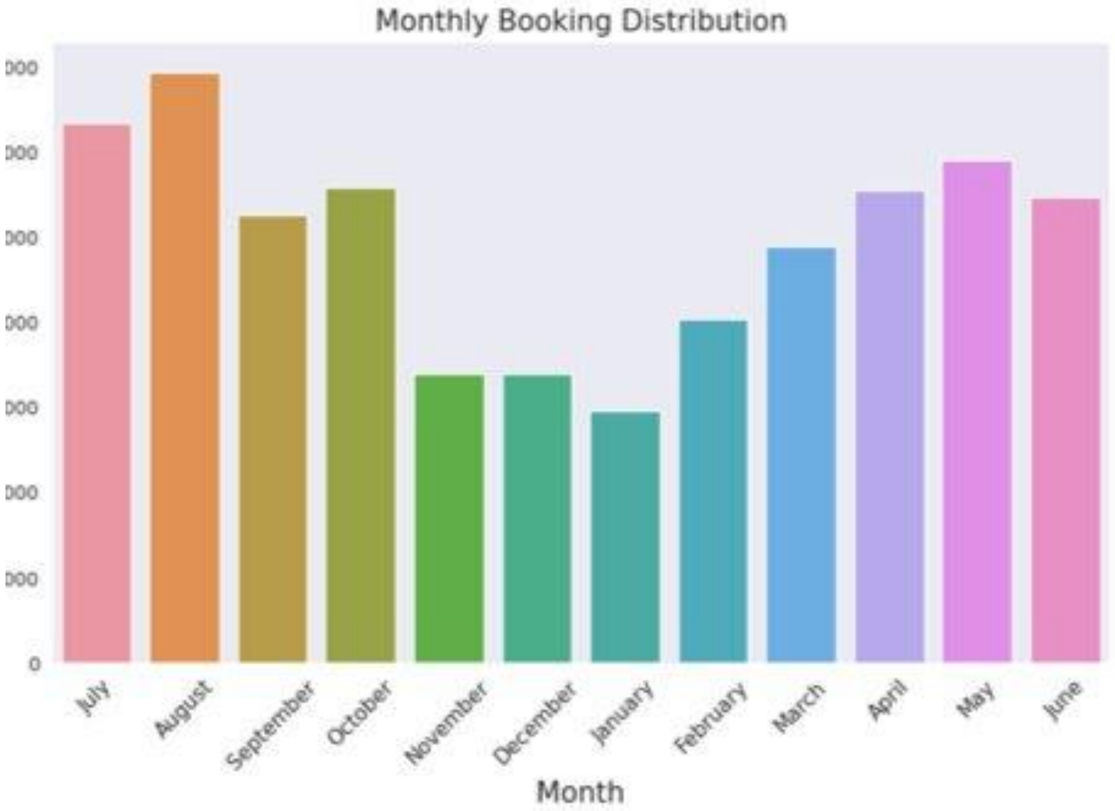


Column Name	Description
Hotel	Type of hotel
Is_canceled	Value indicating if the booking was canceled
Lead_time	Timespan between the reservation of a hotel room and checkin
Arrival_date_year	Year of arrival date
Arrival_date_month	Month of arrival date
Arrival_date_week_number	Week number of year for arrival date
Arrival_date_day_of_month	Day of arrival date
Stays_in_weekend_nights	Number of weekend nights the guests stayed
Stays_in_week_nights	Number of week nights the guests stayed
Adults	Number of adults
Children	Number of children
Babies	Number of babies
Meal	Type of meal booked
Country	Country of origin
Market_segment	Type of booking
Distribution_channel	Booking distribution channel
Is_repeated_guest	Value indicating if the booking name was from a repeated guest
Previous_cancellations	Number of previous bookings that were cancelled by the customer
Previous_bookings_not_canceled	Number of previous bookings not cancelled by the customer

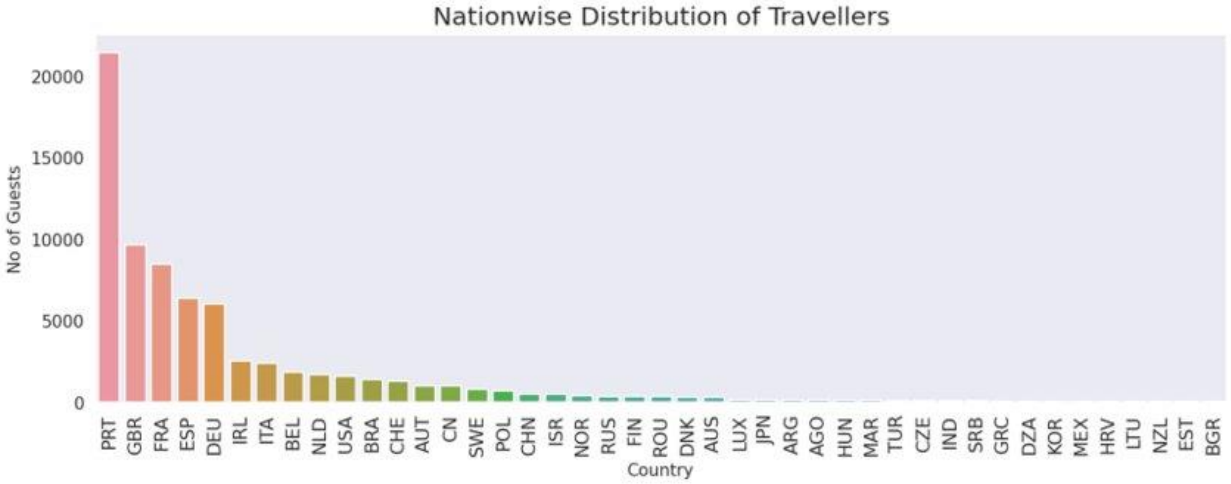


Column Name	Description
Reserved_room_type	Code of room type reserved
Assigned_room_type	Code for the type of room assigned to the booking
Booking_changes	Number of changes/amendments made to the booking
Deposit_type	Indicates if a deposit has been made by the customer
Agent	ID of the travel agency that made the booking
Company	ID of the company/entity that made the booking
Days_in_waiting_list	Number of days the booking was in the waiting list
Customer_type	Type of booking
ADR	Average Daily Rate
Required_car_parking_spaces	Number of car parking spaces required by the customer
Total_of_special_requests	Number of special requests made by the customer
Reservation_status	Reservation last status
Reservation_status_date	Date at which the last status was set

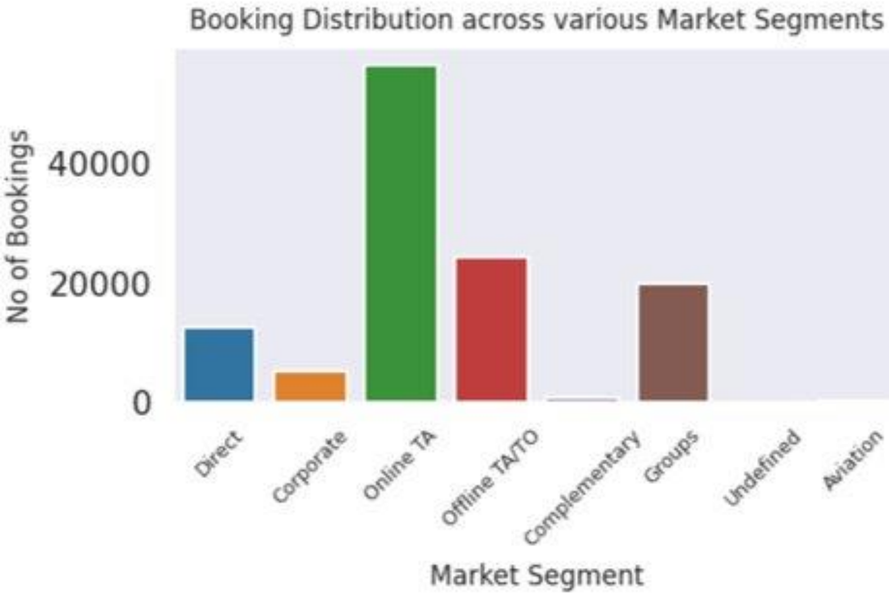
❖ **DATA VISUALIZATIONS:**



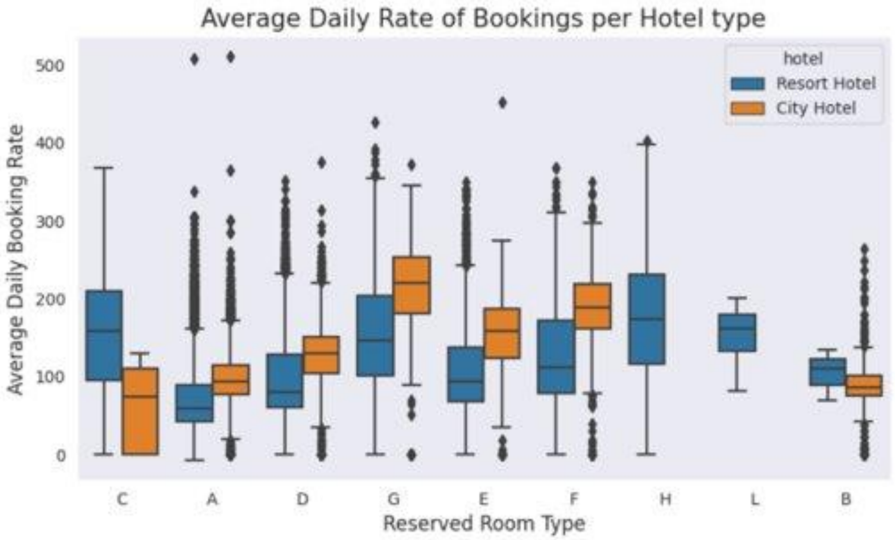
This bar graph shows how the bookings are spread out based on each month. As we can see, August has the highest number of bookings most likely due to the summer months attracting more tourists in Lisbon. Conversely, we can see that January has the lowest number of bookings.



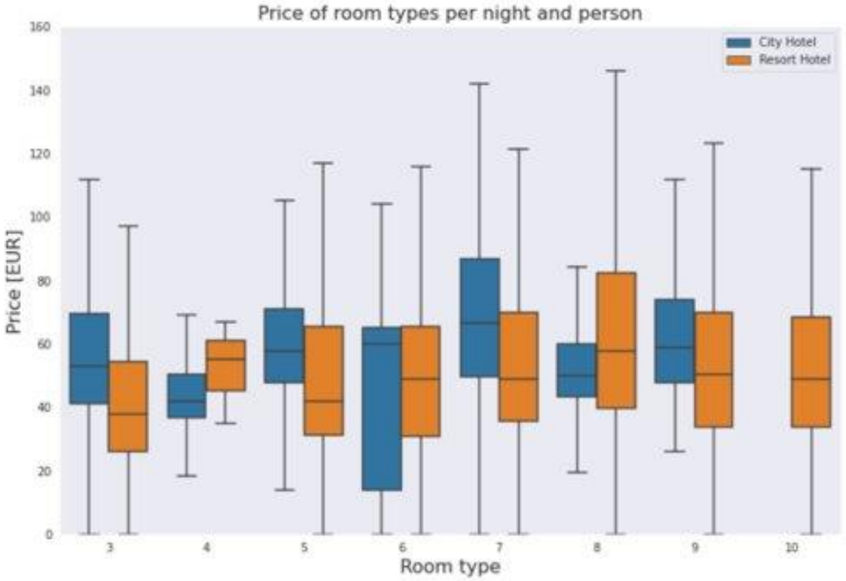
This bar plot shows us where most of the travelers to these Lisbon based hotels are coming from. It seems like most guests are from Portugal itself and then from Great Britain.



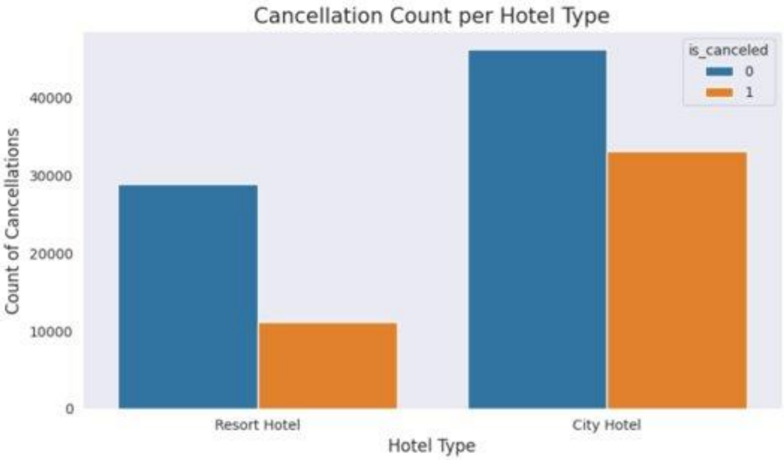
Most of the bookings are being made by travel agents, most of which are online.



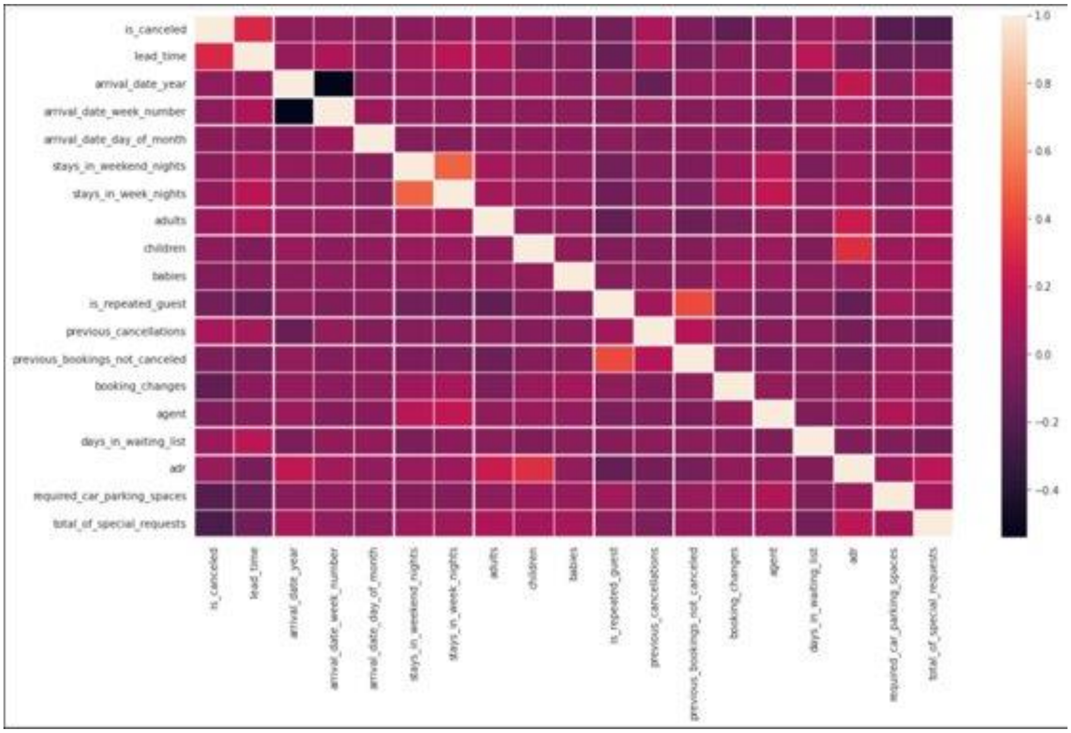
There are nine types of rooms. City hotels have less types of rooms as it is missing room types H and L. Additionally, for type C, we can see that city hotels have significantly less rooms than resort hotels which may be a selling point for resort hotels.



These box plots show us the daily rate in euros depending on the room type in city hotels vs. resort hotels.



These bar plots compare the number of cancellations for city hotels and resort hotels. Since there are more bookings for city hotels, we also see more cancellations for city hotels.



This is a heat correlation map that shows us the correlation between the features. Here we see a relationship between ADR, adults, children, and total number of special requests.

PREDICTIVE MODEL

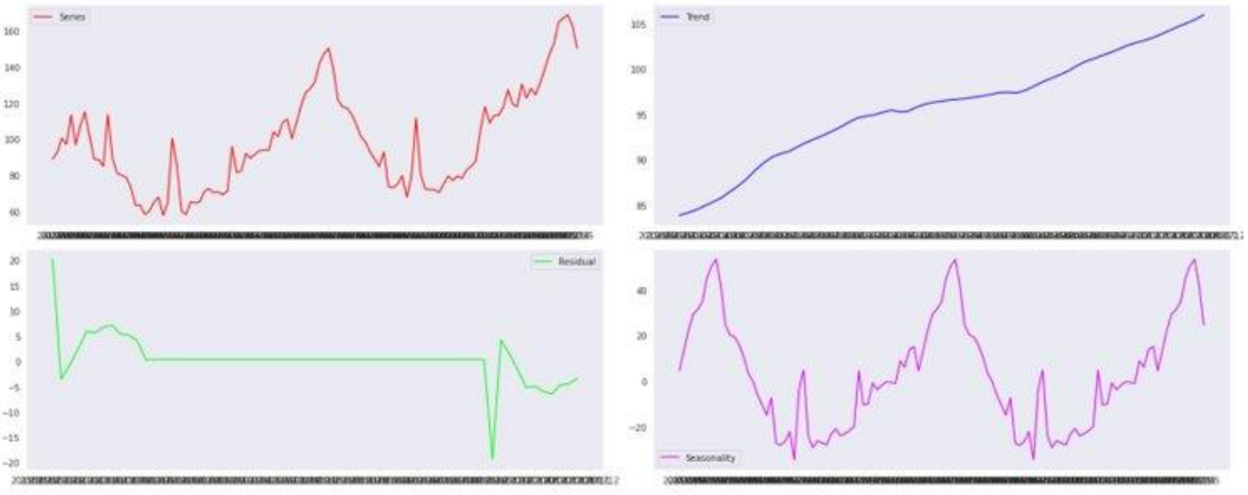
Forecasting the weekly Average Daily Rate

❖ DATA PRE-PROCESSING:

We start off by merging the arrival_date_year column with the arrival_date_week_number column into a new column that has been named **“FullDate”**. For instance, “201543” would belong to week 43 of year 2015. Next, we create a time series object that aggregates the Average Daily Rate on a weekly basis by averaging it. This is converted into a dataframe so we can split it into a train and test set. The first 100 rows are allocated to the training set while the rest of the 15 rows are given to the test set.

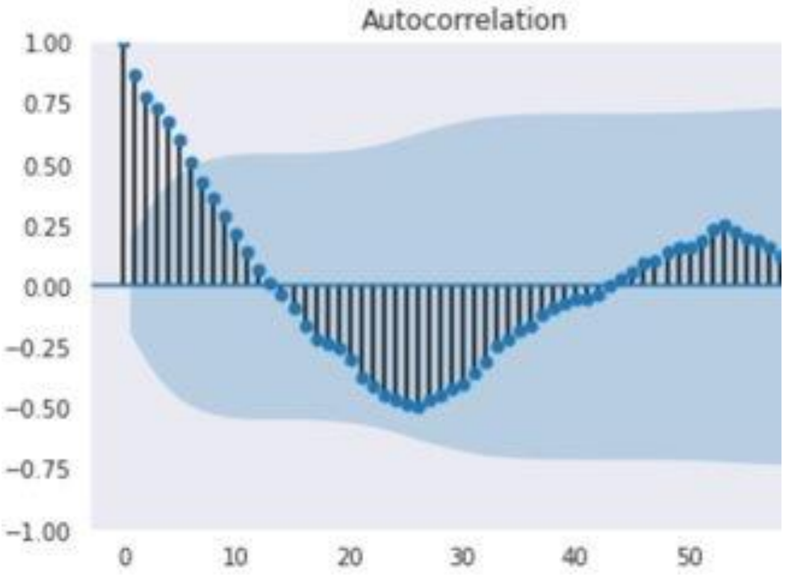
❖ DECOMPOSITION:

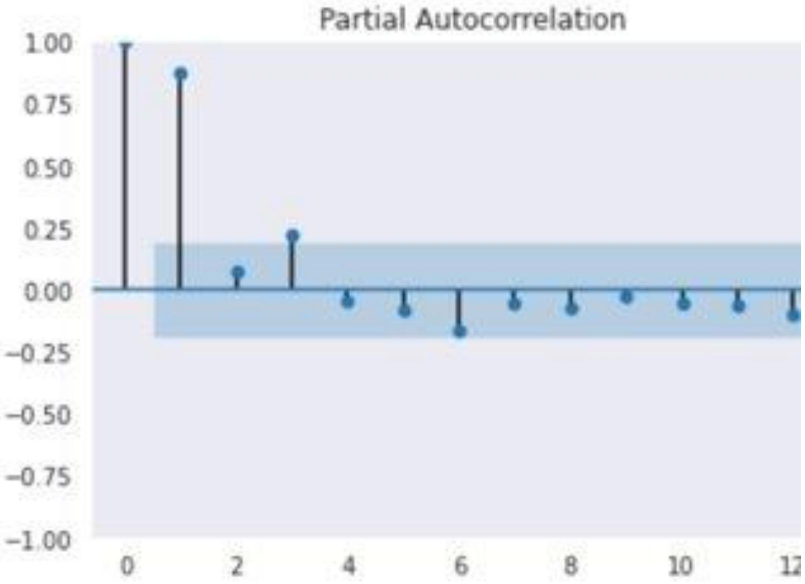
We decompose the data to find that there is both trend and seasonality present in the data. Additionally, by looking at the residual plot, we see that the decomposition perfectly covers the trend and seasonality for certain time periods.



❖ MODEL SELECTION:

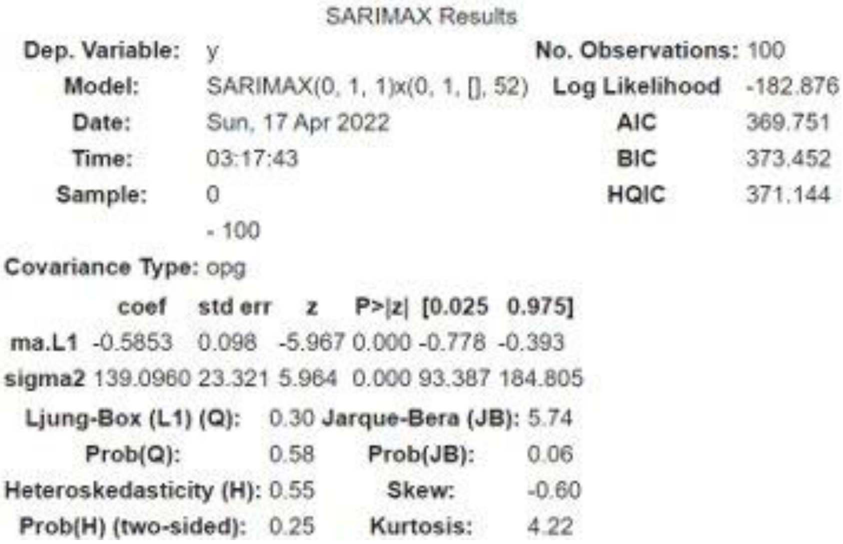
We create the ACF and the PACF plot for the train data which is shown below which tells us that they will be AR(1) models. But as we performed the ADF test on the train data, it was non-stationary. After taking the differencing with seasonality and with a period of 1, we fit a more relevant SARIMAX model after using Auto-Arima. Also, we can say from this that the seasonality is 52. We difference the data to remove any trend that is present. Then after performing the ADFuller test, we find that the data is **stationary**.



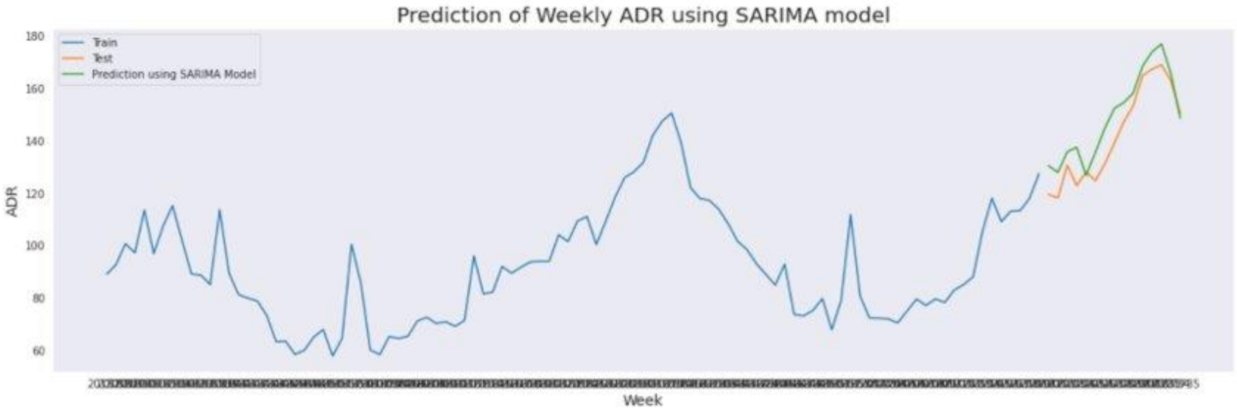


❖ DATA MODELING:

To find the best model for our data, we use the help of the Auto-Arima function. For the (p, d, q) X (P, D, Q) [S] values, it returns to us ARIMA(0, 1, 1)(0, 1, 0)[52] based on the lowest AIC scores. Next, we use the SARIMAX function on the model we received from the Auto-Arima function. The result from the SARIMAX function is given below. We then perform the Ljung-Box Test which shows us that the residuals are not correlated. Finally, we plot the actual vs. the predicted values of the **weekly average daily rate** which is shown below, and we find that the **RMSE is 8.75**.

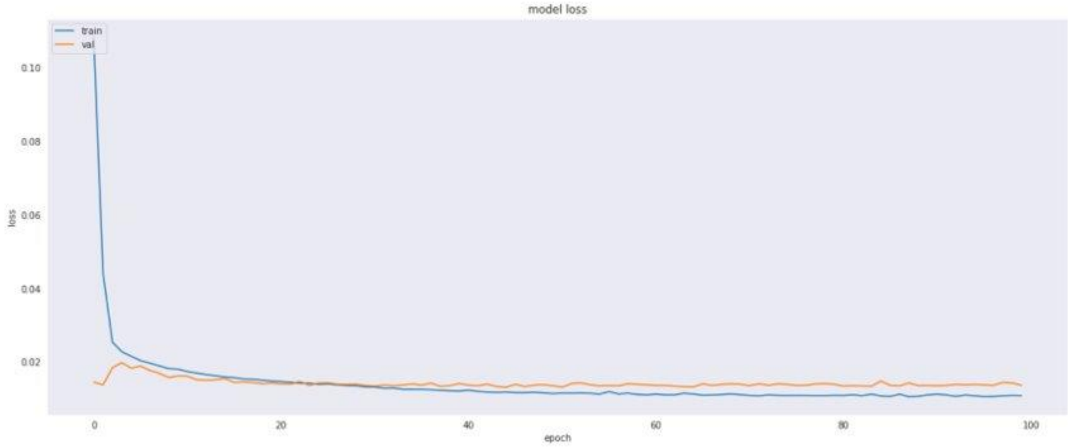


```
                              SARIMAX Results
 Dep. Variable:   y                            No. Observations: 100
     Model:       SARIMAX(0, 1, 1)x(0, 1, [], 52)  Log Likelihood  -182.876
      Date:       Sun, 17 Apr 2022                       AIC       369.751
      Time:       03:17:43                               BIC       373.452
     Sample:      0                                      HQIC      371.144
                  - 100
Covariance Type: opg
             coef    std err    z     P>|z|  [0.025   0.975]
 ma.L1   -0.5853     0.098   -5.967  0.000  -0.778   -0.393
sigma2   139.0960   23.321    5.964  0.000  93.387   184.805
  Ljung-Box (L1) (Q):        0.30  Jarque-Bera (JB):  5.74
        Prob(Q):             0.58      Prob(JB):      0.06
Heteroskedasticity (H):      0.55        Skew:        -0.60
  Prob(H) (two-sided):       0.25      Kurtosis:      4.22
```

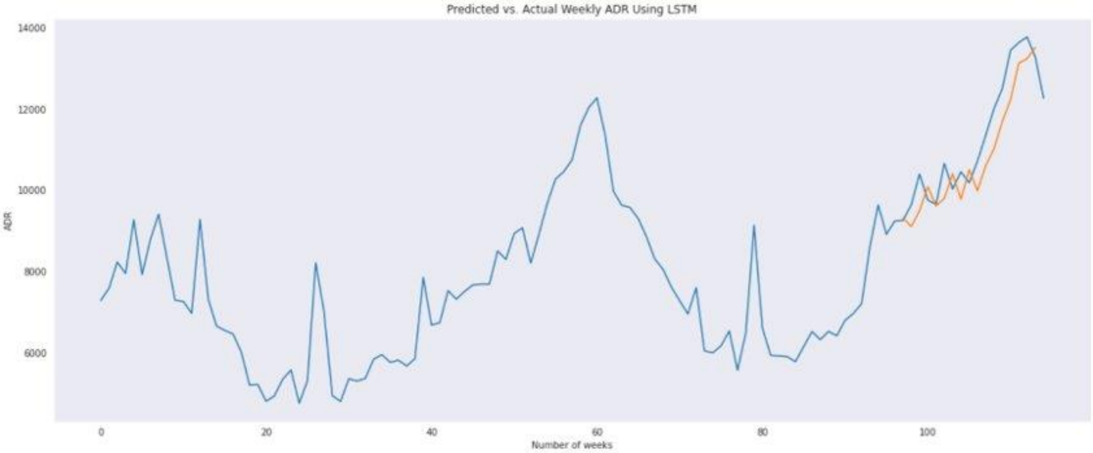


❖ PREDICTION USING LSTM:

Our next approach is to use LSTM in predicting the average daily rate. For this, we first convert our data frame into an array. We again split the data into training and test sets. 80% goes towards the train set and the remaining 20% goes towards the test set. We use the **MinMaxScaler** to scale the data and set the **lookback period** to **5**. The model is then trained for **100 epochs**. The **loss function** used is **mean square error** and the **optimizer** used is **Adam**. We also plotted the train and validation loss, the graph of which is given below. We again plot the actual vs. the predicted values of the **weekly average daily rate** which is shown below and find that the **RMSE is 8.17** which is slightly lower than what we were getting before.



This graph tells us that we could use early stopping at around 22, but as we had the computational power, we went ahead and ran the model through all 100 epochs.

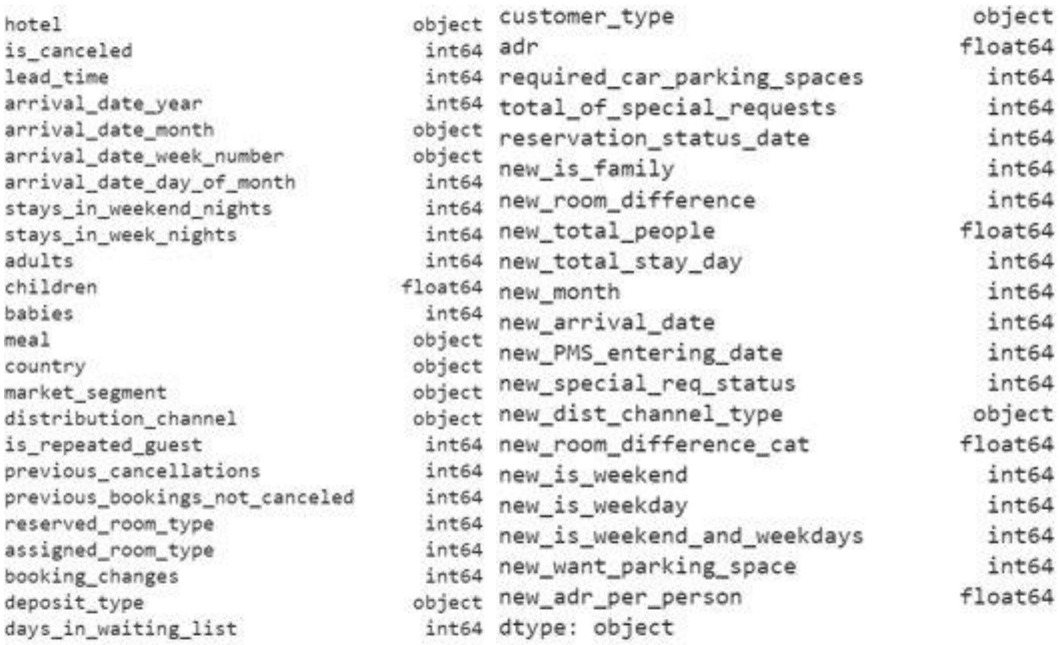


BINARY CLASSIFICATION

Predicting whether a booking will be canceled or not

❖ FEATURE ENGINEERING:

We created 15 new features to get essential information. We added “new_” to identify these new features. For example, the variable “new_is_weekend” was created by extracting the day of the week from the given data and was set to 0 if it was a weekday, otherwise, it was set to 1. Additionally, the variable “new_is_family” captures whether there are children or babies present in the booking. The total 44 features are listed below.



```
hotel                              object
is_canceled                         int64
lead_time                           int64
arrival_date_year                   int64
arrival_date_month                 object
arrival_date_week_number           object
arrival_date_day_of_month           int64
stays_in_weekend_nights             int64
stays_in_week_nights                int64
adults                              int64
children                          float64
babies                              int64
meal                               object
country                            object
market_segment                     object
distribution_channel               object
is_repeated_guest                   int64
previous_cancellations              int64
previous_bookings_not_canceled      int64
reserved_room_type                  int64
assigned_room_type                  int64
booking_changes                     int64
deposit_type                       object
days_in_waiting_list                int64
```

```
customer_type                        object
adr                                 float64
required_car_parking_spaces           int64
total_of_special_requests             int64
reservation_status_date               int64
new_is_family                         int64
new_room_difference                   int64
new_total_people                    float64
new_total_stay_day                    int64
new_month                             int64
new_arrival_date                      int64
new_PMS_entering_date                 int64
new_special_req_status                int64
new_dist_channel_type                object
new_room_difference_cat             float64
new_is_weekend                        int64
new_is_weekday                        int64
new_is_weekend_and_weekdays           int64
new_want_parking_space                int64
new_adr_per_person                  float64
dtype: object
```

The next step was to remove the unnecessary or redundant features which left us with **29 features** at the end.

❖ DATA ENCODING:

We have some columns that have too many classes, for example, the “Country” column that has more than 300 classes. So, we use **label encoding** for the columns that are of type object and can have binary values instead. We also use **one-hot encoding** for certain columns that have anywhere from **3 to 12** classes. Additionally, we used the **Standard Scaler** for numeric columns.

❖ MODEL SELECTION:

We tried six different models:

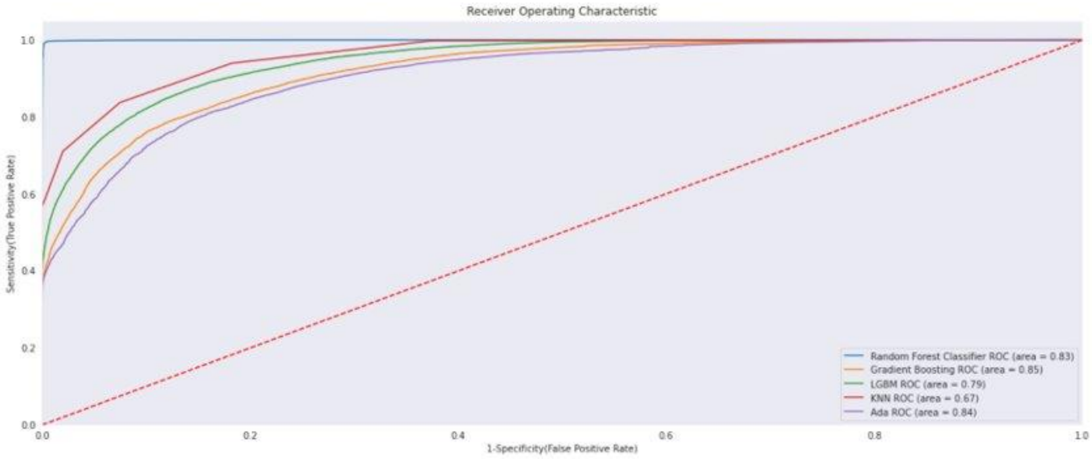
1. Classification And Regression Tree (CART)
2. K-Nearest Neighbors (KNN)
3. Random Forest (RF)
4. AdaBoost
5. Gradient Boosting Machine
6. Light Gradient Boosting Machine

We selected the best set of parameters for the above models by passing a range of parameters to a user-defined function **best_parameters()** to give out the best parameters that minimize RSS loss. The results of this function gave us the following parameters:

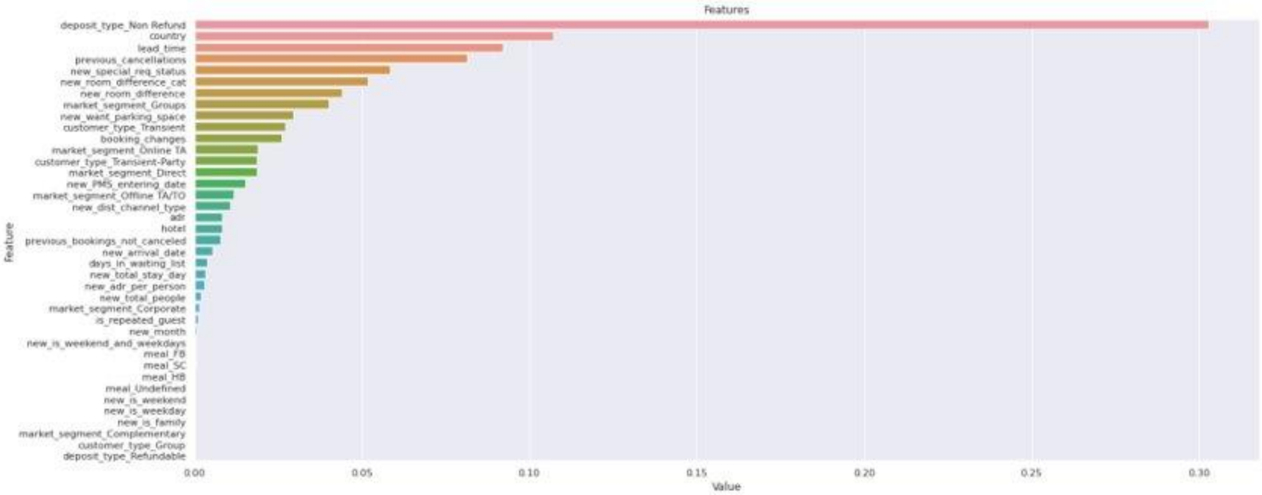
1. For the **CART** model, the parameter for the maximum depth was 1 and for the minimum samples split it was 2. The F1 score for this model came out to **0.4775**. The accuracy of the CART model is **74.98%**. The ROC AUC came out to **0.6634.** Also since the value of AUC under ROC is less even when the accuracy is high, we can say that the data is imbalanced. In this case, Specificity and Sensitivity would give us more viable results and would give a better picture of model performance on this data.
2. For the **KNN** model, the parameter for the N neighbors was set to **50**. The F1 score for the KNN model came out to **0.4783**. The accuracy of this model is **62.08%**. Meanwhile, the ROC AUC is **0.6676.**
3. For the **Random Forest** model, the parameter for the maximum depth was **5** and for the maximum features, it was **5**. The n estimators were set to **100** while the minimum sample

split was **20**. The F1 score for this model came out to **0.4945**. The accuracy for the RF model is **74.82%**. The ROC AUC came out to **0.8346**.

4. For the **AdaBoost** model, the parameter for n estimators was set to **10**. The learning rate was set to **0.98** and the algorithm used was **SAMME**. The F1 score for the AdaBoost model came out to **0.5875**. The accuracy of this model is **75.51%**. Meanwhile, the ROC AUC is **0.8450**.
5. For the **Gradient Boosting Machine** model, the parameter for n estimators was set to **100** and for the learning rate, it was set to **0.01**. The minimum sample split was **0.7**. The F1 score for this model came out to **0.4882**. The accuracy for the Gradient Boosting Machine model is **75.25%**. The ROC AUC came out to **0.8504**.
6. For the **Light Gradient Boosting Machine** model, the parameter for n estimators was set to **100,** for the learning rate it was set to **0.01,** and for the column sample by tree parameter, it was set to **0.7**. The F1 score for the Light Gradient Boosting Machine model came out to **0.5607**. The accuracy of this model is **69.86%**. Meanwhile, the ROC AUC is **0.7895**.



We would be using the AdaBoost model for our predictions as it has a relatively high ROC AUC score along with a high accuracy as well.



This feature plot tells us which features are most important when it comes to determining whether a booking will be canceled or not. As we can see, the most important feature is whether the deposit is non-refundable which fits as people would be reluctant to cancel if they could not get their money back.

❖ CONCLUSION:

For the first part of this project, we can conclude that LSTM performed better than SARIMA when it came to forecasting the average daily rate. For the latter half of the project, AdaBoost performed better than the other 5 models that we ran alongside it. Over the test dataset’s time period, our model predicted **18,000 bookings** in total out of which **4,000** would be **canceled**. We could find the average daily rate for these bookings through the LSTM model that we created in the first half of this project. For further future use, we could use this model on augmented datasets for hostels, Airbnbs, Vrbos, etc. in various locations to predict their weekly average daily rates.