|  |  |
| --- | --- |
| **HOTEL BOOKINGS ANALYSIS**  **PROJECT REPORT** | Abstract  [Draw your reader in with an engaging abstract. It is typically a short summary of the document. When you’re ready to add your content, just click here and start typing.]  Hassanat, Tamsir and Tabatha |

**INTRODUCTION**

We analysed the hotel booking data for a given company. Our aim was to enable the company to make informed decisions about marketing, leading to increased revenue. The company has two kinds of hotel: city and resort hotels. We were provided with bookings data collected from 2015 to 2017.

To make suggestions about targeted marketing, we came up with 6 questions that were answered with the data. The questions are:

1. Where do most of the guests comes from?
2. Are the guests usually families or individuals?
3. What is the distribution of the average daily rate for each hotel type?
4. Which hotel type is mostly booked and then occupied?
5. How loyal are the guests? Should marketing aim for past or new guests?
6. Which months are the hotels mostly occupied?

The first two and fifth questions helped with determining the company’s target audience. The second two questions helped determine the hotel type to be emphasised in the marketing campaign while the sixth one determined the best time to carry out the campaign.

**IMPLEMENTATION AND EXECUTION**

**DATASET:** The dataset contained 100,000+ rows and 32 columns. Each row provided details of an individual booking. As part of the preprocessing, we removed redundant rows and columns. We also had to engineer new featues such as is\_family. The following important columns were used in the analysis:

* **hotel**, categorical variable, described whether the row entries were for a city or resort hotel
* **arrival\_date** indicated the date, month, and year of the guest’s arrival at the hotel
* **adults**, **children** and **babies** columns, quantitative variables, showed their respective numbers for a given booking
* **country,** categorical variable, showed the origin of the booking
* **is\_repeated\_guest,** binary numerical variable**,** showed if the booking was from a past guest (1) or a new guest (0)
* **adr** (Average Daily Rate), quantitative variables, showed the sum of all lodging transactions divided by the total number of staying nights for a given day
* **reservation\_status**, categorical variable, indicated the last reservation status of the guest. There were three possibilities:
* *Canceled* – booking was canceled by the customer;
* *Check-Out* – customer has checked in but already departed;
* *No-Show* – customer did not check-in and did not inform the hotel of their reason
* **is\_family**, binary variable, indicated the booking that were made by families i.e adults, children and/or babies
* Other columns from Tabatha and Tamsir

Figure 1: dependencies of the python scripts

**PREPROCESSING**: We created a script which contained functions to address each cleaning process for the dataset. The script was then imported into the workflow.py for the changes to be applied to the data. The functions contained in the preprocessing.py are:

1. fillnull: this function was used to fill null values with relevant values. Columns such as ‘children’, ‘agent’ and ‘company’ contained null entries. Some columns with null entries were ignored as the nulls were useful for the analysis such as the ‘country’ column. For the children column, we used the fillna function to fill the null values with 0.
2. change\_date: this function was defined to unify the arrival details for the booking. The arrival date for each booking was split into day, month, week number and year. Firstly, the month was converted from the month name (e.g. July) to the month number (e.g. 7) using the calendar module. Then the day, month, and year were concatenated to replace the year column. Finally, the day, month and week number columns were dropped from the data.
3. change\_datatype: this function was defined to change the data type for columns with the wrong data type. Arrival\_date and reservation\_status\_date columns were changed to datetime using the pd.to\_datetime() while children column was changed to integer using the astype()
4. Other cleaning functions from Tabatha and Tamsir
5. clean\_data: This function called on all the functions defined above and produced the final cleaned dataset.

**ANALYSIS:** Unlike the preprocessing, two scripts were created for the analysis. An utils.py which contained methods defined under a given class, each class was specific to an analysis question listed above. Then an analysis.py in which the methods from utils were imported and applied. We also had a workflow.py which combined the processes from preprocessing.py and analysis.py.

1. **utils.py:** a module which contained two classes: A typical class answered an analysis question by containing methods that i) compute statistics, ii) plot visualisation and save to PDF iii) save summary statistics to CSV

class AdrStats:

def \_\_init\_\_(self, hotel\_data):

def compute\_stats(self):

return adr\_stats

def plot\_fig(self, filename):

def summary\_csv(self, filename):

…

1. **analysis.py:** here, separate functions were defined to invoke all the methods from each class in the utils.py. A typical function takes in the dataset as a parameter, the first method which computes statistics is applied to the parameter and the result is saved as a variable. Then the other methods for visualising and transforming to a csv are applied to the variable. The file path to which the visuals and CSV files are saved is specified as the parameter for the methods. A last function called ‘all\_analysis’ combined all the functions defined earlier

from utils import AdrStats, …

def analyse\_adr(bookings):

adr\_stats = AdrStats(bookings)

adr\_stats.plot\_fig(‘filepath')

adr\_stats.summary\_csv(filepath’)

…

def all\_analysis(bookings):

analyse\_adr(bookings)

analyse\_reservation\_status(bookings)

1. **workflow.py:** this is the final script which combines the data cleaning and analysis processes. It imports the clean\_data function from preprocessing.py and the all\_analysis function from analysis.py. It reads the data using pandas read method and assigns it to a variable. Then the clean\_data and all\_analysis functions are applied to the variable to produce all the analysis results (statistics and graphs) at once. In this case, the cleaned data is not produced independently. The cleaned data is only produced when the preprocessing.py is directly executed.

from preprocessing import clean\_data

from analysis import all\_analysis

if \_\_name\_\_ == "\_\_main\_\_":

hotel\_data = pd.read\_csv(r'hotel\_bookings.csv', encoding='utf-8')

clean\_data(hotel\_data)

all\_analysis(hotel\_data)

**RESULTS**

● ii) the achieved results, *i.e.,* statistics and graphs.

**CONCLUSION**