HOTEL BOOKING DEMAND

A report submitted in partial fulfilment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

SUNIL KUMAR PARSIKA

Regd. No: 20B91A05M3

Under Supervision of Mr. Gundala Nagaraju Henotic Technology Pvt Ltd Hyderabad
(Duration: 7th July, 2022 to 6th September, 2022)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SAGI RAMA KRISHNAM RAJUENGINEERING COLLEGE

(An Autonomous Institution)

Approved by AICTE, NEW DELHI and Affiliated to JNTUK, Kakinada

CHINNA AMIRAM, BHIMAVARAM, ANDHRA PRADESH

SAGI RAMA KRISHNAM RAJU ENGINEERING COLLEGE

(Autonomous)

Chinna Amiram, Bhimavaram

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the “**Summer Internship Report”** submitted by **SUNIL KUMAR PARSIKA, 20B91A05M3** is work done by him/her and submitted during 2021 - 2022 academic year, in partial fulfilment of the requirements for the award of the Summer Internship Program for **Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING,** at **Henotic Technology Pvt Ltd** from *07.07.2022 to 06.09.2022*

Department Internship Coordinator **Dean -T & P Cell** **Head of the Department**

Table of Contents

Abstract

Hotel reservation System is a computerized system used to store and retrieve information and conduct transactions related to hotels and their services. The project is aimed at exposing the relevance and importance of machine learning in hotel management. It is projected towards enhancing the relationship between guests and hotel management, and thereby making it convenient for the guests to book the hotel and their specific services as when they require such that they can utilize this software to make reservations or even to cancel them.

The project is aimed at the prediction of a booking's cancellation status based on the data collected from various hotels. This enhances the management of hotel services. This data includes guests' choice of services along with extensive information on each guest including trip related information and demographic information. It includes the information like strength of family, their requested services, their previous connections with the hotel, and more.

By using this data from various hotels across, we try to train a machine learning model that predicts whether a guest cancels his hotel reservation. This helps the hotels to improve their flaws and also adjust other guests' services.

1.0 INTRODUCTION

With the increasing power of computer technology, companies and institutions can now a days store large amounts of data at reduced cost. The amount of available data is increasing exponentially, and cheap disk storage makes it easy to store data that previously was thrown away. There is a huge amount of information locked up in databases that is potentially important but has not yet been explored. The growing size and complexity of the databases makes it hard to analyse the data manually, so it is important to have automated systems to support the process. Hence there is the need of computational tools able to treat these large amounts of data and extract valuable information.

In this context, Data Mining provides automated systems capable of processing large amounts of data that are already present in databases. Data Mining is used to automatically extract important patterns and trends from databases seeking regularities or patterns that can reveal the structure of the data and answer business problems. Data Mining includes learning techniques that fall into the field of Machine learning. The growth of databases in recent years brings data mining at the forefront of new business technologies.

The goal of this program is to see how well various statistical methods perform in predicting whether or not a reservation of suite at a hotel will be cancelled based on some factors like arrival period, family strength of the booking party, changes and services, and more of the given historical data.

1.1 What are the different types of Machine Learning?

There are classified mainly into three types. They are

Supervised Learning:

Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labelled data. Even though the data needs to be labelled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.

In supervised learning, the ML algorithm is given a small training dataset to work with. This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with. The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labelled parameters required for the problem.

This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset. This means that supervised machine learning algorithms will continue to improve even after being deployed, discovering new patterns and relationships as it trains itself on new data.

Categories of Supervised Machine Learning

Supervised machine learning can be classified into two types of problems, which are given below:

- Classification
- Regression

Classification

Classification algorithms are used to solve the classification problems in which the output variable is categorical, such as Yes or No, Male or Female, Red or Blue, etc. The classification algorithms predict the categories present in the dataset. Some real-world examples of classification algorithms are Spam Detection, Email filtering, etc.

Some popular classification algorithms are given below:

- Random Forest Algorithm
- Decision Tree Algorithm
- Logistic Regression Algorithm
- Support Vector Machine Algorithm

Regression

Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables. These are used to predict continuous output variables, such as market trends, weather prediction, etc.

Some popular Regression algorithms are given below:

- Simple Linear Regression Algorithm
- Multivariate Regression Algorithm
- Decision Tree Algorithm
- Lasso Regression

Applications of Supervised Learning:

- Image Segmentation
- Medical Diagnosis
- Fraud Detection
- Spam detection
- Speech Recognition

Unsupervised Learning:

Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.

The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post-deployment development than supervised learning algorithms.

Categories of Unsupervised Machine Learning:

Unsupervised Learning can be further classified into two types, which are given below:

- Clustering
- Association

Clustering

The clustering technique is used when we want to find the inherent groups from the data. It is a way to group the objects into a cluster such that the objects with the most similarities remain in one group and have fewer or no similarities with the objects of other groups. An example of the clustering algorithm is grouping the customers by their purchasing behavior.

Some of the popular clustering algorithms are given below:

- K-Means Clustering algorithm
- Mean-shift algorithm
- DBSCAN Algorithm
- Principal Component Analysis
- Independent Component Analysis

Association

Association rule learning is an unsupervised learning technique, which finds interesting relations among variables within a large dataset. The main aim of this learning algorithm is to find the dependency of one data item on another data item and map those variables accordingly so that it can generate maximum profit. This algorithm is mainly applied in Market Basket analysis, Web usage mining, continuous production, etc.

Some popular algorithms of Association rule learning are:

- Apriori Algorithm
- Eclat
- FP-growth algorithm.

Applications of Unsupervised Learning:

- Network Analysis
- Recommendation Systems.
- Anomaly Detection
- Singular Value Decomposition

1.2 Benefits of Using Machine Learning in Hotel Bookings

When it comes to learning technology, we should be aware of the pros and cons of that technology. The reason is so that we can understand the capabilities of that subject.

That is exactly what we are doing here. Understanding the advantages and disadvantages of Machine Learning will help us to unlock many doors.

The advantages of Machine Learning are vast. It helps us to create ways of modernizing technology. The disadvantages of Machine Learning tell us its limits and side effects. This helps us to find different innovative ways to reduce these problems.

Advantages of Machine Learning:

1. Automation of Everything

Machine Learning is responsible for cutting the workload and time. By automating things, we let the algorithm do the hard work for us. Automation is now being done almost everywhere. The reason is that it is very reliable. Also, it helps us to think more creatively.

Due to ML, we are now designing more advanced computers. These computers can handle various Machine Learning models and algorithms efficiently. Even though automation is spreading fast, we still don’t completely rely on it. ML is slowly transforming the industry with its automation.

2. Wide Range of Applications

ML has a wide variety of applications. This means that we can apply ML on any of the major fields. ML has its role everywhere from medical, business, banking to science and tech. This helps to create more opportunities. It plays a major role in customer interactions. Machine Learning can help in the detection of diseases more quickly. It is helping to lift businesses. That is why investing in ML technology is worth it.

3. Scope of Improvement

Machine Learning is the type of technology that keeps on evolving. There is a lot of scope in ML to become the top technology in the future. The reason is it has a lot of research areas in it. This helps us to improve both hardware and software.

In hardware, we have various laptops and GPUs. These have various ML and Deep Learning networks in them. These help in the faster processing power of the system. When it comes to software, we have various UIs and libraries in use. These help in designing more efficient algorithms.

4. Efficient Handling of Data

Machine Learning has many factors that make it reliable. One of them is data handling. ML plays the biggest role when it comes to data currently. It can handle any type of data.

Machine Learning can be multidimensional or different types of data. It can process and analyze these data those normal systems can't. Data is the most important part of any Machine Learning model. Also, studying and handling of data is a field.

5. Best for Education and Online Shopping

ML would be the best tool for education in the future. It provides very creative techniques to help students study. Recently in China, a school has started to use ML to improve student focus. In online shopping, the ML model studies your searches. Based on your search history, it would provide advertisements. These will be about your search preferences in previous searches. In this, the search history is the data for the model. This is a great way to improve e-commerce with ML.

1.3 About Industry (Hotels sector):

A hotel is an establishment that provides paid lodging on a short-term basis. Mostly hotels are used by personals for holiday stays, for work and official stays, and more. There exist no-star hotels to five-star luxury hotels which provides different types of services to its customers.

In the recent times hotels have become more popular and affordable to public. Some of the hotels even offer stays for pet animals. People choose hotels based on many factors like their services, the area, the price, their work, their luxury.

1.3.1 AI / ML Role in Hotels Sector:

Machine Learning is a sub-set of artificial intelligence where computer algorithms are used to autonomously learn from data. Machine learning (ML) is getting more and more attention and is becoming increasingly popular in many other industries. Within the Hotels sector, there is more application of ML regarding the stays.

Hotels Dataset content:

- Hotel: The type of hotel
- Lead time: Number of days between the arrival date and registered date
- Arrival date year: Year of arrival date
- Arrival date month: Month of arrival date
- Arrival date week: Week number of arrival date
- Arrival date day: Day of arrival date

- Stays in weekend nights: Number of weekend nights
- Stays in week nights: Number of week nights
- Adults: Number of adults in the booking journal
- Children: Number of children in the booking journal
- Babies: Number of babies in the booking journal
- Meal: Type of meal booked
- Country: Country of origin of entry in the booking journal
- Market Segment: The market segment designation
- Distribution channel: The booking distribution channel
- Is repeating guest: Indicates whether that guest is a repeating one
- Previous cancellation: Indicates if the repeating guest cancelled their previous booking
- Previous booking not cancelled: Indicates if the repeating guest has not cancelled their previous booking
- Reserved room type: Code of type of room reserved
- Assigned room type: Code of type of room assigned
- Booking changes: Number of changes made to the booked service
- Deposit type: Indicates if the guest has deposited money
- Agent: Id of the travel agency that made the booking
- Company: Id of the company that made the payment of the booking
- Days in waiting list: Indicates the number of days the booking has not been allotted
- Customer type: Type of booking
- Adr: Average daily rate of transactions
- Required car parking: Indicates whether the guest required a parking lot
- Total of special requests: Indicates the number of special requests made by the guest
- Reservation status: Indicates the status of the guest's reservation
- Reservation status date: Date at which last reservation status was updated
- Is cancelled: Indicates whether or not the guest has cancelled their reservation

2.0 HOTEL RESERVATION CANCELLATION:

The project is aimed at the prediction of a hotel reservation's cancellation using the data collected from across various hotels. This enhances the relationship between guests and the hotel management. This data includes guests' choice of services along with extensive information on each guest including booking related information and demographic information. It includes the information like number of family members, their booking reason, their previous bookings, their requirements, and more.

By using this data and the cancellation status taken from the hotels' database we train the model and try to predict the cancellation of a booking. This helps the management to analyze the circumstances that promotes to not cancelling their booking.

2.1. Main Drivers for AI in Booking's Cancellation Analysis:

Predictive modelling allows for simultaneous consideration of many variables and quantification of their overall effect. When many bookings are analyzed, patterns regarding the characteristics of the cancellation that drive loss development begin to emerge.

- Type of Hotel
- Arrival period
- Adults and Children
- Meal
- Country
- Reservation status
- Customer Type
- Special Requests Made
- Room Type
- Booking Changes
- Deposit Type
- Previous Cancellation

2.2. Internship Project – Data Link

The internship project data has been taken from the website kaggle.com, and the link is https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand

This dataset contains 1,19,390 instances(rows) of 32 features(columns)

The description of the dataset is shown below.

```
In [4]: print(train0.shape)
        print(train0.describe())

(119390, 32)
         is_canceled      lead_time  arrival_date_year  \
count  119390.000000  119390.000000      119390.000000
mean        0.370416     104.011416        2016.156554
std         0.482918     106.863097           0.707476
min         0.000000       0.000000        2015.000000
25%         0.000000      18.000000        2016.000000
50%         0.000000      69.000000        2016.000000
75%         1.000000     160.000000        2017.000000
max         1.000000     737.000000        2017.000000

       arrival_date_week_number  arrival_date_day_of_month  \
count             119390.000000              119390.000000
mean                  27.165173                  15.798241
std                   13.605138                   8.780829
min                    1.000000                   1.000000
25%                   16.000000                   8.000000
50%                   28.000000                  16.000000
75%                   38.000000                  23.000000
max                   53.000000                  31.000000

       stays_in_weekend_nights  stays_in_week_nights         adults  \
count            119390.000000         119390.000000  119390.000000
mean                  0.927599              2.500302       1.856403
std                   0.998613              1.908286       0.579261
min                   0.000000              0.000000       0.000000
25%                   0.000000              1.000000       2.000000
50%                   1.000000              2.000000       2.000000
75%                   2.000000              3.000000       2.000000
max                  19.000000             50.000000      55.000000

            children         babies  is_repeated_guest  \
count  119386.000000  119390.000000      119390.000000
mean        0.103890       0.007949           0.031912
std         0.398561       0.097436           0.175767
min         0.000000       0.000000           0.000000
25%         0.000000       0.000000           0.000000
50%         0.000000       0.000000           0.000000
75%         0.000000       0.000000           0.000000
max        10.000000      10.000000           1.000000

       previous_cancellations  previous_bookings_not_canceled  \
count           119390.000000                   119390.000000
mean                 0.087118                        0.137097
std                  0.844336                        1.497437
min                  0.000000                        0.000000
25%                  0.000000                        0.000000
50%                  0.000000                        0.000000
75%                  0.000000                        0.000000
max                 26.000000                       72.000000

       booking_changes          agent      company  days_in_waiting_list  \
count    119390.000000  103050.000000  6797.000000         119390.000000
mean          0.221124      86.693382   189.266735              2.321149
std           0.652306     110.774548   131.655015             17.594721
min           0.000000       1.000000     6.000000              0.000000
25%           0.000000       9.000000    62.000000              0.000000
50%           0.000000      14.000000   179.000000              0.000000
75%           0.000000     229.000000   270.000000              0.000000
max          21.000000     535.000000   543.000000            391.000000

                 adr  required_car_parking_spaces  total_of_special_requests
count  119390.000000                119390.000000              119390.000000
mean      101.831122                     0.062518                   0.571363
std        50.535790                     0.245291                   0.792798
min        -6.380000                     0.000000                   0.000000
25%        69.290000                     0.000000                   0.000000
50%        94.575000                     0.000000                   0.000000
75%       126.000000                     0.000000                   1.000000
max      5400.000000                     8.000000                   5.000000
```

3.0 AI / ML Modelling and Results:

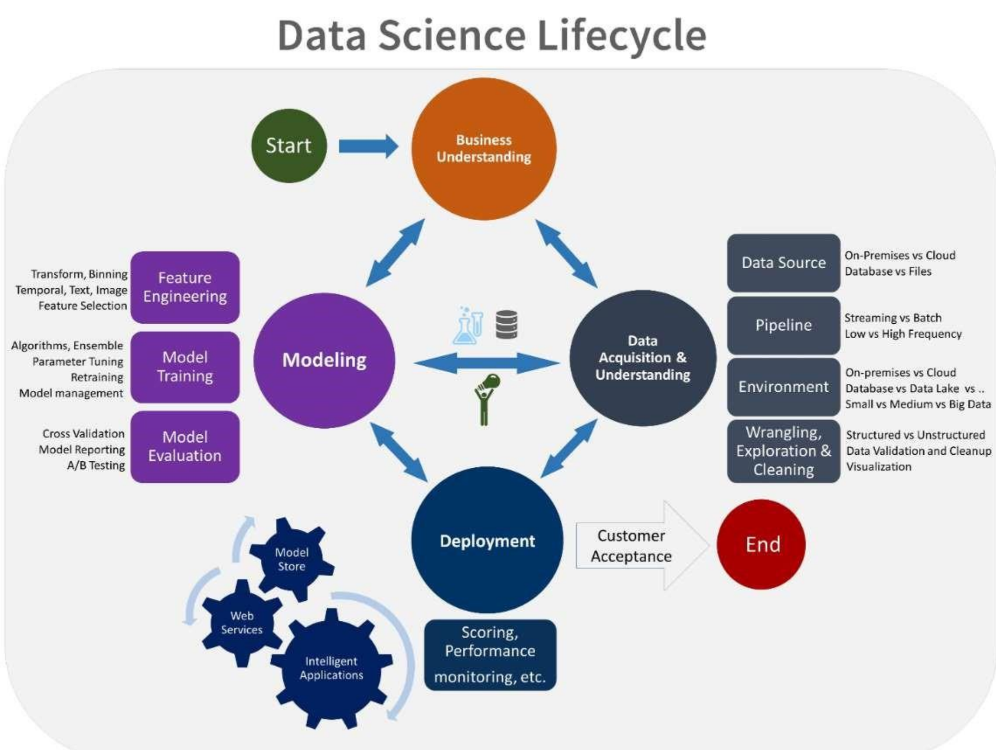
3.1. Problem Statement:

- Predictive models are most effective when they are constructed using a firm's own satisfactory data since this allows the model to recognize the specific nature of a company's exposure. The construction of the model also involves input from the company throughout the process, as well as consideration of industry leading claims practices and benchmarks.
- Predictive modelling can be used to quantify the impact to the hotels' services resulting from the failure to meet or exceed service leading practices so that in future a few lesser guests cancels their booking.
- So, our final moto is to predict whether a guest at a hotel cancels his or hers' reservation based on features provided

3.2. Data Science Project Life Cycle:

 What is a Data Science Project Lifecycle?

Data Science is a multidisciplinary field of study that combines programming skills, domain expertise and knowledge of statistics and mathematics to extract useful insights and knowledge from data.



3.2.1 Data Exploratory Analysis:

Exploratory Data Analysis refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

3.2.2. Data Pre-processing:

We removed variables which does not affect our target variable (satisfaction)as they may add noise and increase our computation time, we checked the data for anomalous data points and outliers. We did principal component analysis on the data set to filter out unnecessary variables and to select only the important variables which have greater correlation with our target variable.

3.2.2.1. Check the Duplicate and low variation data.

You have a dataset and must check there is duplicates or not. The Python pandas library has a method for it, that is duplicated (). It checks for the duplicates rows and returns True and False. For the data frame object. If you use the method sum () along with it, then it will return the total number of the duplicates in the dataset.

Now you have known that there are duplicates in the dataset and want to remove the duplicates from the dataset. There are two ways you can remove duplicates. One is deleting the entire rows and other is removing the column with the most duplicates.

3.2.2.2. Identify and address the missing variables:

Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset. In Pandas, usually, missing values are represented by NaN.

Checking the missing values:

The first step in handling missing values is to look at the data carefully and find out all the missing values.dataframe.isnull().sum() will tell about missing values in the entire column.

Figure Out How to Handle the Missing Data:

Analyze each column with missing values carefully to understand the reasons behind the missing values as it is crucial to find out the strategy for handling the missing values.

There are 2 primary ways of handling missing values:

1. Deleting the Missing values
2. Imputing the Missing Values

3.2.2.3. Handling of Outliers:

As outliers are very different values—abnormally low or abnormally high—their presence can often skew the results of statistical analyses on the dataset. This could lead to less effective and less useful models.

But dealing with outliers often requires domain expertise, and none of the outlier detection techniques should be applied without understanding the data distribution and the use case.

3.2.2.4. Categorical data and Encoding Techniques:

What is Categorical Data?

Since we are going to be working on categorical variables in this article, here is a quick refresher on the same with a couple of examples. Categorical variables are usually represented as 'strings' or 'categories' and are finite in number. Here are a few examples:

1. The city where a person lives: Delhi, Mumbai, Ahmedabad, Bangalore, etc.
2. The department a person works in: Finance, Human resources, IT, Production.
3. The highest degree a person has: High school, Diploma, Bachelors, Masters, PhD.
4. The grades of a student: A+, A, B+, B, B- etc.

In the above examples, the variables only have definite possible values. Further, we can see there are two kinds of categorical data-

- Ordinal Data: The categories have an inherent order
- Nominal Data: The categories do not have an inherent order

Label Encoding:

- We use this categorical data encoding technique when the categorical feature is ordinal. In this case, retaining the order is important. Hence encoding should reflect the sequence.
- In Label encoding, each label is converted into an integer value. We will create a variable that contains the categories representing the education qualification of a person.

Binary Encoding:

- Binary encoding is a combination of Hash encoding and one-hot encoding. In this encoding scheme, the categorical feature is first converted into numerical using an ordinal encoder. Then the numbers are transformed in the binary number. After that binary value is split into different columns.
- Binary encoding works well when there are a high number of categories. For example, the cities in a country where a company supplies its products

3.2.2.5. Feature Scaling:

Why Feature Scaling?

Real Life Datasets have many features with a wide range of values like for example let's consider the house price prediction dataset. It will have many features like no. of. bedrooms, square feet area of the house, etc.

As you can guess, the no. of bedrooms will vary between 1 and 5, but the square feet area will range from 500-2000. This is a huge difference in the range of both features.

Many machine learning algorithms that are using Euclidean distance as a metric to calculate the similarities will fail to give a reasonable recognition to the smaller feature, in this case, the number of bedrooms, which in the real case can turn out to be an important metric.

E.g.: Linear Regression, Logistic Regression, KNN

There are several ways to do feature scaling. I will be discussing the top 5 of the most used feature scaling techniques.

3.2.3. Selection of Dependent and Independent variables:

The dependent or target variable here is satisfaction Target which tells us a which tells us that the Customer
is satisfied, neutral or dissatisfied.

The independent variables are selected after doing exploratory data analysis.

3.2.4 Data Sampling Methods:

The data we have is highly unbalanced data so we used some sampling methods which are used to balance the target variable so we our model will be developed with good accuracy and precision. We used three Sampling methods

3.2.4.1. Stratified sampling

Stratified sampling randomly selects data points from majority class so they will be equal to the data points in the minority class. So, after the sampling both the class will have same no of observations.

It can be performed using strata function from the library sampling.

3.2.4.2. Simple random sampling

Simple random sampling is a sampling technique where a set percentage of the data is selected randomly. It is generally done to reduce bias in the dataset which can occur if data is selected manually without randomizing the dataset.

We used this method to split the dataset into train dataset which contains 70% of the total data and test
dataset with the remaining 30% of the data.

3.2.5 Models Used for Development:

We built our predictive models by using the following ten algorithms:

3.2.5.1. Model 01 (Logistic Regression)

Logistic uses logit link function to convert the likelihood values to probabilities so we can get a good estimate on the probability of a particular observation to be positive class or negative class. The also gives us p-value of the variables which tells us about significance of each independent variable.

3.2.5.2. Model 02 (Decision Tree Classifier)

Decision Tree Learning is supervised learning approach used in statistics, data mining and machine learning. In this formalism, a classification or regression decision tree is used as a predictive model to draw conclusions about a set of observations.

Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees.

3.2.5.3. Model 03 (Random Forest Classifier)

Random forest is an algorithm that consists of many decision trees. It was first developed by Leo Bierman and Adele Cutler. The idea behind it is to build several trees, to have the instance classified by each tree, and to give a "vote" at each class. The model uses a "bagging" approach and the random selection of features to build a collection of decision trees with controlled variance. The instance's class is to the class with the highest number of votes, the class that occurs the most within the leaf in which the instance is placed.

The error of the forest depends on:

- Trees correlation: the higher the correlation, the higher the forest error rate.
- The strength of each tree in the forest. A strong tree is a tree with low error. By using trees that classify the instances with low error the error rate of the forest decreases.

3.2.5.4. Model 04 (Extra Tree Classifier)

Specifically, it is an ensemble of decision trees and is related to other ensembles of decision trees algorithms such as bootstrap aggregation (bagging) and random forest. Specifically, it is an ensemble of decision trees and is related to other ensembles of decision trees algorithms such as bootstrap aggregation (bagging) and random forest. The Extra Trees algorithm works by creating many unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees in the case of regression or using majority voting in the case of classification.

3.2.5.5. Model 05 (KNN Classifier)

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most like the available categories’-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

3.2.5.6. Model 06(Gaussian Naive Bayes)

The name “Naïve” is used because the algorithm incorporates features in its model that are independent of each other. Any modifications in the value of one feature do not directly impact the value of any other feature of the algorithm. The main advantage of the Naïve Bayes algorithm is that it is a simple yet powerful algorithm.

It is based on the probabilistic model where the algorithm can be coded easily, and predictions did quickly in real-time. Hence this algorithm is the typical choice to solve real-world problems as it can be tuned to respond to user requests instantly. But before we dive deep into Naïve Bayes and Gaussian Naïve Bayes, we must know what is meant by conditional probability.

3.2.5.7. Model 07 (XGB classifier)

XG Boost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XG Boost models majorly dominate in many Kaggle Competitions. In this algorithm, decision trees are created in sequential form. Weights play an important role in XG Boost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

3.2.5.8. Model 08 (Light GBM)

Light GBM is a gradient boosting framework based on decision trees to increases the efficiency of the model and reduces memory usage.

It uses two novel techniques: Gradient-based One Side Sampling and Exclusive Feature Bundling (EFB) which fulfills the limitations of histogram-based algorithm that is primarily used in all GBDT (Gradient Boosting Decision Tree) frameworks. The two techniques of GOSS and EFB described below form the characteristics of Light GBM Algorithm. They comprise together to make the model work efficiently and provide it a cutting edge over other GBDT frameworks.

3.2.5.9. Model 09 (SVC)

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate ndimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

3.2.5.10 Model 10 (Gradient Boosting Classifier)

This algorithm builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n_classes_ regression trees are fit on the negative gradient of the loss function, e.g. binary or multiclass log loss. Binary classification is a special case where only a single regression tree is induced.

3.3.AI/ ML Models Analysis and Final Results:

We used our train dataset to build the above models and used our test data to check the accuracy and performance of our models. We used confusion matrix to check accuracy, Precision, Recall and F1 score of our models and compare and select the best model.

Code for Data Preprocessing

```
#Importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot
as plt import seaborn as
sns
%matplotlib inline
#Ignore harmless
warnings import
warnings
warnings.filterwarnings("ignore") #Set to
display all the columns in dataset
pd.set_option("display.max_columns",
None)
#Import psql to run queries
import pandasql as psql
#Importing the data
train0 = pd.read_csv("hotel_bookings.csv", header=0)
train0
#Copy to back_up files
train1 = train0.copy()
#Display first 5 records
train0.head()
# Display the dataset information
```

```
train0.info()
print(train0.shape)
print(train0.describe())
#heat map to look for the correlated
attributes plt.figure(figsize = (24, 12)) corr
= train0.corr() sns.heatmap(corr, annot =
True, linewidths = 1) plt.show()
train0.hist(bins=50, figsize=(20,15))
plt.tight_layout() plt.show()
#plotting each feature
wrt y for i in
train0.columns: try:
    print(i, " : ", train0[i].corr(train0['is_canceled'], 'pearson'))    print("--------
---------")
    #df0.plot.scatter(x = df0[i], y = df0['is_canceled'])
    #plt.scatter(np.array(df0[i]), np.array(df0['is_canceled']))
 #plt.show()
except:
pass
#Removing unrequired columns using correlation
Corr_Matrix = round(train0.corr(),5)
Corr_Matrix['is_canceled']
print('-----------------------------')
c=abs(Corr_Matrix['is_cancele
d']) c
train1 = train0.sample(frac=1) d = ['arrival_date_year', 'stays_in_weekend_nights', 'reservation_status_date',
'reservation_status','agent', 'adr',
'company','children','babies',
'days_in_waiting_list'] train2 = train1.drop(d,
axis='columns') train2 count =
train2['is_canceled'].value_counts()
print('class 0:',count[0])
print('class 1:',count[1])
print("pro
:",count[0]/count[1],":1")
#checking for null/ na for i in
train2.columns: print(i, " : ",
train2[i].isnull().sum())
train3 = train2.dropna()
train3.shape
#finding string
columns def
find(train, x): try:

float(train[x][1])
return False
except:
```

```
        return True
#li = [find(train2, x) for x in train2.columns]
for i in train2.columns:
    if find(train2, i):
        print("-----------------")
print(train2[i].value_counts())
encode_cols = [
'hotel',
'arrival_date_month',
'meal',
'reserved_room_type',
'distribution_channel',
'market_segment',
'deposit_type',
'customer_type',
'assigned_room_type',
'country']
train3          =          train2.copy()          from
sklearn.preprocessing import LabelEncoder
LE = LabelEncoder()
for i in encode_cols:
    train3[i]=LE.fit_transform(np.ravel(train3[[i]]))
for i in train3.columns:
    print(train3[i].value_counts())
    print("---------")
y  =  "is_canceled"
train3                    =
train3.dropna()
cols  =  []  for  i  in
train3.columns:
if i != y:

cols.append(i) X =
train3[cols]   y   =
train3[y]  for  i  in
train3.columns:
    print(train3[i].isnull().sum())
train3.shape
#splitting data intotrain and test
from sklearn.model_selection import train_test_split x_train, x_test, y_train, y_test
= train_test_split(X, y, test_size=0.2, random_state=10)
#selecting linear regresion model
from             sklearn.linear_model             import
LogisticRegression   lr   =   LogisticRegression()
lr.fit(x_train, y_train) lr.score(x_test, y_test)
#confusion  matrix  for  logistic  regression
y_pred = lr.predict(x_test)
```

y_pred_prob = lr.predict_proba(x_test) # Confusion matrix in sklearn from sklearn.metrics import confusion_matrix from sklearn.metrics import classification_report
# actual values
actual = y_test
# predicted values
predicted = y_pred # confusion matrix
matrix = confusion_matrix(actual,predicted, labels=[1,0],sample_weight=None, normalize=None) print('Confusion matrix : \n', matrix) # outcome values order in sklearn
tp, fn, fp, tn = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1) print('Outcome values : \n', tp, fn, fp, tn)
# classification report for precision, recall f1-score and accuracy C_Report = classification_report(actual,predicted,labels=[1,0]) print('Classification report : \n', C_Report)
# calculating the metrics sensitivity = round(tp/(tp+fn), 3); specificity = round(tn/(tn+fp), 3); accuracy = round((tp+tn)/(tp+fp+tn+fn), 3); balanced_accuracy = round((sensitivity+specificity)/2, 3); precision = round(tp/(tp+fp), 3); f1Score = round((2*tp/(2*tp + fp + fn)), 3);
# Matthews Correlation Coefficient (MCC). Range of values of MCC lie between -1 to +1.
# A model with a score of +1 is a perfect model and -1 is a poor model from math import sqrt mx = (tp+fp) * (tp+fn) * (tn+fp) * (tn+fn)MCC = round(((tp * tn) - (fp * fn)) / sqrt(mx), 3) print('Accuracy :', round(accuracy*100, 2),'%') print('Precision :', round(precision*100, 2),'%') print('Recall :', round(sensitivity*100,2), '%') print('F1 Score :', f1Score) print('Specificity or True Negative Rate :', round(specificity*100,2), '%') print('Balanced Accuracy :', round(balanced_accuracy*100, 2),'%') print('MCC :', MCC) # Area under ROC curve
from sklearn.metrics import roc_curve, roc_auc_score
print('roc_auc_score:', round(roc_auc_score(y_test, y_pred), 3)) HTResults = [] def plott(models): models.fit(x_train, y_train)
# Prediction
y_pred = models.predict(x_test)
y_pred_prob = models.predict_proba(x_test)
# Print the model name print('Model Name: ', models) # confusion matrix in sklearn from sklearn.metrics import confusion_matrix from sklearn.metrics import classification_report

actual
values actual =
y_test #
predicted values
predicted =
y_pred #
confusion matrix
matrix = confusion_matrix(actual,predicted, labels=[1,0],sample_weight=None,
normalize=None) print('Confusion matrix : \n', matrix) # outcome values order in sklearn
tp, fn, fp, tn = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1)
print('Outcome values : \n', tp, fn, fp, tn)
# classification report for precision, recall f1-score and accuracy
C_Report = classification_report(actual,predicted,labels=[1,0])
print('Classification report : \n', C_Report)
# calculating the metrics sensitivity =
round(tp/(tp+fn), 3); specificity = round(tn/(tn+fp), 3);
accuracy = round((tp+tn)/(tp+fp+tn+fn), 3);
balanced_accuracy = round((sensitivity+specificity)/2,
3);

precision = round(tp/(tp+fp), 3);
f1Score = round((2*tp/(2*tp + fp + fn)), 3);

Matthews Correlation Coefficient (MCC). Range of values of MCC lie between -1 to +1.
A model with a score of +1 is a perfect model and -1 is a poor model

from math import sqrt mx = (tp+fp) * (tp+fn) * (tn+fp) * (tn+fn)
MCC = round(((tp * tn) - (fp * fn)) / sqrt(mx), 3) print('Accuracy :',
round(accuracy*100, 2),'%') print('Precision :', round(precision*100,
2),'%') print('Recall :', round(sensitivity*100,2), '%') print('F1 Score
:', f1Score) print('Specificity or True Negative Rate :',
round(specificity*100,2), '%') print('Balanced Accuracy :',
round(balanced_accuracy*100, 2),'%') print('MCC :', MCC)

Area under ROC curve

from sklearn.metrics import roc_curve, roc_auc_score

print('roc_auc_score:', round(roc_auc_score(actual, predicted), 3))
# ROC Curve from sklearn.metrics import roc_auc_score from
sklearn.metrics import roc_curve logit_roc_auc =
roc_auc_score(actual, predicted) fpr, tpr, thresholds =
roc_curve(actual, models.predict_proba(x_test)[:,1]) plt.figure()
# plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' %
logit_roc_auc) plt.plot(fpr, tpr, label= 'Classification Model' %
logit_roc_auc) plt.plot([0, 1], [0, 1],'r--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True

```
Positive Rate') plt.title('Receiver operating characteristic')
plt.legend(loc="lower right") plt.savefig('Log_ROC')
    plt.show()
    new_row = {'Model Name' : models,
               'True_Positive' : tp,
               'False_Negative' : fn,
               'False_Positive' : fp,
               'True_Negative' : tn,
               'Accuracy' : accuracy,
'Precision' : precision,
               'Recall' :
sensitivity, 'F1 Score'
: f1Score,
               'Specificity' : specificity,
               'MCC':MCC,
               'ROC_Score':roc_auc_score(actual, predicted),
               'Balanced Accuracy':balanced_accuracy}
    HTResults.append(new_row)
# Build the Calssification models and compare the
results from sklearn.linear_model import
LogisticRegression from sklearn.tree import
DecisionTreeClassifier
from sklearn.ensemble import
RandomForestClassifier from sklearn.ensemble
import ExtraTreesClassifier from
sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC from
sklearn.ensemble import BaggingClassifier from
xgboost import XGBClassifier


from sklearn.ensemble import GradientBoostingClassifier
import lightgbm as lgb
# Create objects of classification algorithm with default hyper-parameters
ModelLR = LogisticRegression()
ModelDC = DecisionTreeClassifier()
ModelRF = RandomForestClassifier()
ModelET = ExtraTreesClassifier()
ModelKNN = KNeighborsClassifier(n_neighbors=5)
#modelBAG = BaggingClassifier()
ModelGB = GradientBoostingClassifier()
ModelLGB = lgb.LGBMClassifier()ans = pd.DataFrame(HTResults)
ans
ModelGNB = GaussianNB()
ModelXGB = XGBClassifier(n_estimators=100, max_depth=3, eval_metric='mlogloss')
ModelSVM = SVC(probability = True)
```

# Evalution matrix for all the algorithms
MM = [ModelLR, ModelDC, ModelRF, ModelET,ModelKNN, ModelGB, ModelLGB, ModelGNB, ModelXGB, ModelSVM] for models in MM: plott(models)

3.3.1 Logistic Regression Python Code

```
from sklearn.linear_model import LogisticRegression
ModelLR = LogisticRegression()
Train(ModelLR)
```

3.3.2 Decision Tree Classifier Python Code

```
from sklearn.tree import DecisionTreeClassifier
ModelDC = DecisionTreeClassifier()
Train(ModelDC)
```

3.3.3 Random Forest Classifier Python Code

```
from sklearn.ensemble import RandomForestClassifier
ModelRF = RandomForestClassifier()
Train(ModelRF)
```

3.3.4 Extra Trees Classifier Python Code

```
from sklearn.ensemble import ExtraTreesClassifier
ModelET = ExtraTreesClassifier()
Train(ModelET)
```

3.3.5 KNeighbors Classifier Python Code

```
from sklearn.neighbors import KNeighborsClassifier
ModelKNN = KNeighborsClassifier(n_neighbors=5)
Train(ModelKNN)
```

3.3.6 Gaussian Naïve Bayes Classifier Python Code

```
from sklearn.naive_bayes import GaussianNB
ModelGNB = GaussianNB()
Train(ModelGNB)
```

3.3.7 XGB Classifier Python Code

```
from xgboost import XGBClassifier
ModelXGB = XGBClassifier(n_estimators=100, max_depth=3, eval_metric='mlogloss')
Train(ModelXGB)
```

3.3.8 LGBM Classifier Python Code

```
import lightgbm as lgb
ModelLGB = lgb.LGBMClassifier()
Train(ModelLGB)
```

3.3.9 SVC Python Code

```
from sklearn.svm import SVC
ModelSVM = SVC(probability = True)
Train(ModelSVM)
```

3.3.10 Gradient Boosting Classifier Python Code

```
from sklearn.ensemble import GradientBoostingClassifier
ModelGB = GradientBoostingClassifier()
Train(ModelGB)
```

4.0 Conclusion and Future work:

The model results are in the following order by considering the model accuracy, F1 score and RoC AUC score.

1. Random Forest Classifier 2. Extra Tree Classifier 3. LGBM Classifier

We recommend model – Random Forest Classifier as a best fit for the given Hotel Bookings Cancellation data set. It predicts cancellation of the booking with high accuracy and F1 score based on the given parameters.

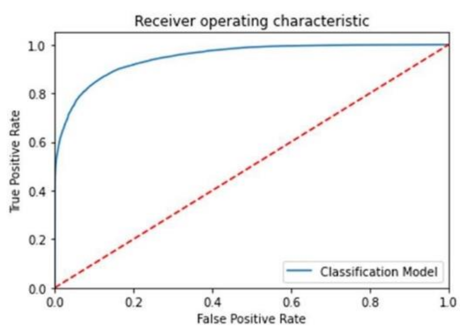
- Random Forest Classifier Result

```
Model Name:  RandomForestClassifier()
Confusion matrix :
 [[ 6909  1789]
 [ 1019 14161]]
Outcome values :
 6909 1789 1019 14161
Classification report :
               precision    recall  f1-score   support

           1       0.87      0.79      0.83      8698
           0       0.89      0.93      0.91     15180

    accuracy                           0.88     23878
   macro avg       0.88      0.86      0.87     23878
weighted avg       0.88      0.88      0.88     23878

Accuracy : 88.2 %
Precision : 87.1 %
Recall : 79.4 %
F1 Score : 0.831
Specificity or True Negative Rate : 93.3 %
Balanced Accuracy : 86.4 %
MCC : 0.743
roc_auc_score: 0.864
```




```
-------------------------------------------------------
```

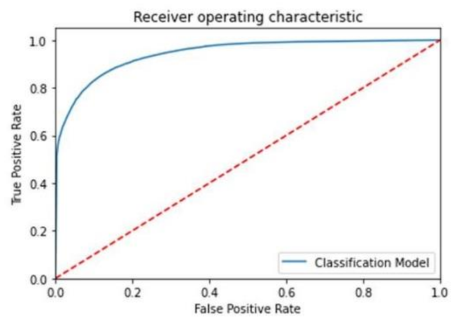
□ Extra Tree Classifier Result

```
Model Name:  ExtraTreesClassifier()
Confusion matrix :
 [[ 6805  1893]
 [ 1041 14139]]
Outcome values :
 6805 1893 1041 14139
Classification report :
              precision    recall  f1-score   support

           1       0.87      0.78      0.82      8698
           0       0.88      0.93      0.91     15180

    accuracy                           0.88     23878
   macro avg       0.87      0.86      0.86     23878
weighted avg       0.88      0.88      0.88     23878

Accuracy : 87.7 %
Precision : 86.7 %
Recall : 78.2 %
F1 Score : 0.823
Specificity or True Negative Rate : 93.1 %
Balanced Accuracy : 85.6 %
MCC : 0.731
roc_auc_score: 0.857
```



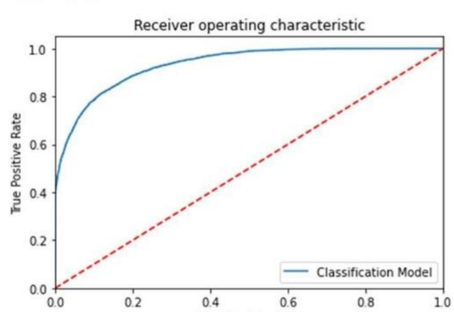
□ LGBM Classifier Result

```
Model Name:  LGBMClassifier()
Confusion matrix :
 [[ 6680  2018]
 [ 1322 13858]]
Outcome values :
 6680 2018 1322 13858
Classification report :
              precision    recall  f1-score   support

           1       0.83      0.77      0.80      8698
           0       0.87      0.91      0.89     15180

    accuracy                           0.86     23878
   macro avg       0.85      0.84      0.85     23878
weighted avg       0.86      0.86      0.86     23878

Accuracy : 86.0 %
Precision : 83.5 %
Recall : 76.8 %
F1 Score : 0.8
Specificity or True Negative Rate : 91.3 %
Balanced Accuracy : 84.0 %
MCC : 0.694
roc_auc_score: 0.84
```



5.0 References

The data is originally from the article Hotel Booking Demand Datasets, written by Nuno Antonio, Ana Almeida, and Luis Nunes for Data in Brief, Volume 22, February 2019.

- https://www.quora.com/What-are-the-valid-reasons-for-a-guest-to-cancel-a-hotel-booking-reservationwithout-paying
- https://www.reviewpro.com/blog/everything-about-guest-satisfaction-surveys/
- https://www.cdaresort.com/blog/resort-vs-hotel-whats-thedifference/#:~:text=Hotels'%20primary%20purpose%20is%20to,found%20within%20the%20resort's%2 0establishment.
- https://en.wikipedia.org/wiki/Hotel
- https://www.sciencedirect.com/science/article/pii/S2352340918315191

6.0 Appendices

6.1 Python code Results

The results of all the above ten algorithms on the Hotel Booking Cancellation Prediction dataset are documented and tabulated as below.

```
ans = pd.DataFrame(HTResults)
ans
```

	Model Name	True_Positive	False_Negative	False_Positive	True_Negative	Accuracy	Precision	Recall	F1 Score	Specificity	MCC	ROC_Score	Balanced Accuracy
0	LogisticRegression()	5058	3640	1883	13297	0.769	0.729	0.582	0.647	0.876	0.485	0.728734	0.729
1	DecisionTreeClassifier()	6849	1849	2001	13179	0.839	0.774	0.787	0.781	0.868	0.653	0.827802	0.828
2	(DecisionTreeClassifier(max_features='auto', r...	6903	1795	1038	14142	0.881	0.869	0.794	0.830	0.932	0.741	0.862626	0.863
3	(ExtraTreeClassifier(random_state=23990978), E...	6821	1877	1036	14144	0.878	0.868	0.784	0.824	0.932	0.733	0.857978	0.858
4	KNeighborsClassifier()	5866	2832	2020	13160	0.797	0.744	0.674	0.707	0.867	0.554	0.770669	0.770
5	([DecisionTreeRegressor(criterion='friedman_ms...	6401	2297	1499	13681	0.841	0.810	0.736	0.771	0.901	0.652	0.818584	0.818
6	LGBMClassifier()	6680	2018	1322	13858	0.860	0.835	0.768	0.800	0.913	0.694	0.840452	0.840
7	GaussianNB()	7773	925	9422	5758	0.567	0.452	0.894	0.600	0.379	0.293	0.636484	0.637
8	XGBClassifier(base_score=0.5, booster='gbtree'...	6625	2073	1575	13605	0.847	0.808	0.762	0.784	0.896	0.667	0.828957	0.829
9	SVC(probability=True)	4686	4012	1888	13292	0.753	0.713	0.539	0.614	0.876	0.446	0.707185	0.708

6.2 List of Charts

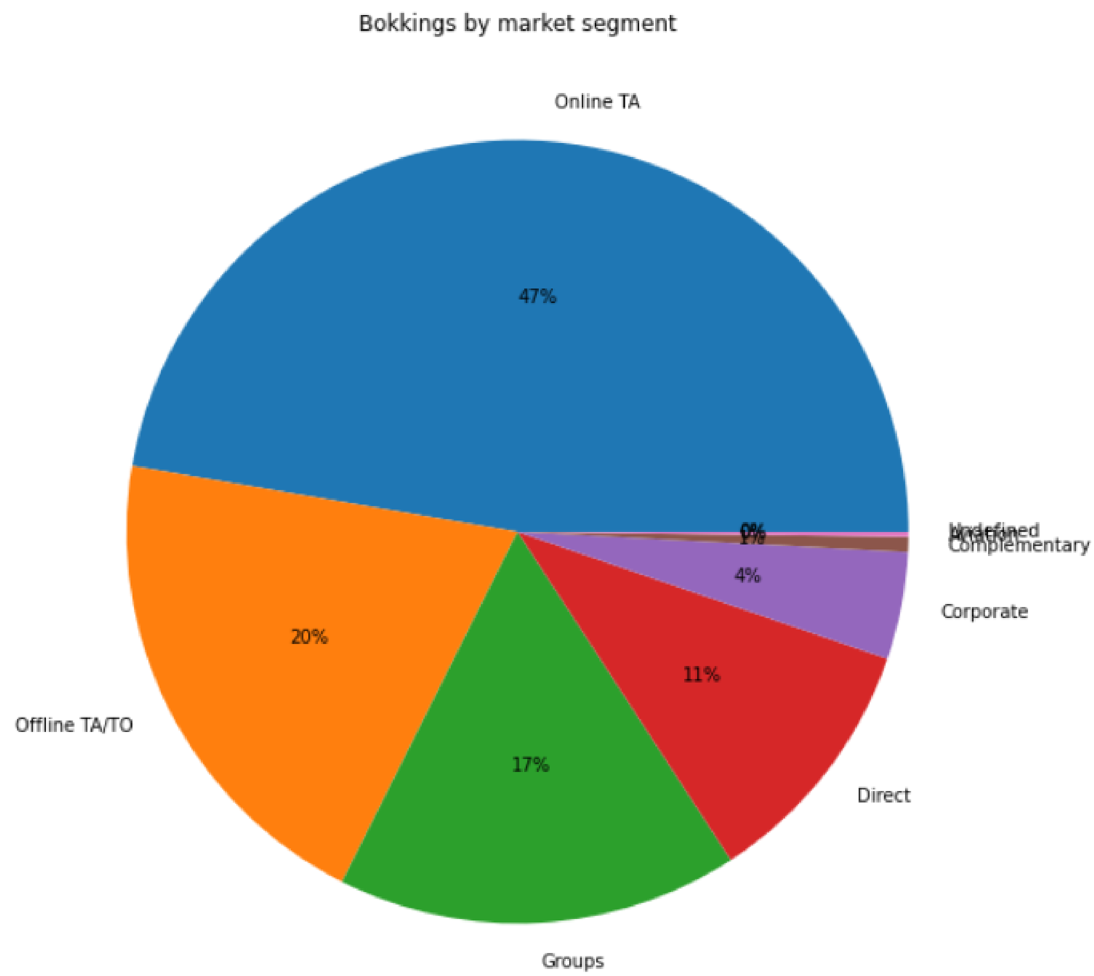
6.2.1 Chart 1: Density spread of values of each feature

```
train0.hist(bins=50, figsize=(20,15))
plt.tight_layout()
plt.show()
```

6.2.2 Booking by market segment

```
d = train0['market_segment'].value_counts()
plt.figure(figsize=(10, 10))
p = plt.pie(d, labels=d.index, autopct="%.0f%%")
plt.title("Bokkings by market segment")
```

```
Text(0.5, 1.0, 'Bokkings by market segment')
```



6.2.3 Counts of cancelled vs not-cancelled booking in different types of hotels

```
sns.countplot(x='hotel', hue='is_canceled', data=train0)
plt.legend(['Not Cancelled', 'Cancelled'])
```

```
<matplotlib.legend.Legend at 0x18febc4fee0>
```

6.2.4 Chart 2: heatmap for Correlation between variables

```
#heat map to look for the correlated attributes
plt.figure(figsize = (24, 12))

corr = train0.corr()
sns.heatmap(corr, annot = True, linewidths = 1)
plt.show()
```