ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

Business Analytics and Data Science
Πρόγραμμα Μεταπτυχιακών Σπουδών στην
ΑΝΑΛΥΤΙΚΗ ΤΩΝ ΕΠΙΧΕΙΡΗΣΕΩΝ ΚΑΙ ΕΠΙΣΤΗΜΗ ΤΩΝ ΔΕΔΟΜΕΝΩΝ
Τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων

Πρόγραμμα Μεταπτυχιακών Σπουδών
στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Τμήμα Οργάνωσης και Διοίκησης Επιχειρήσεων

<u>Διπλωματική εργασία</u>

Predicting hotel booking cancellations using predictive analytics methods

του
Τζελέπη Δημήτρη

Υποβλήθηκε ως απαιτούμενο για την απόκτηση του μεταπτυχιακού διπλώματος στην Αναλυτική των Επιχειρήσεων και Επιστήμη των Δεδομένων

Επιβλέπων Καθηγητής: Άγγελος Μάρκος

Δεκέμβριος 2023

Abstract

The aim of this dissertation is to provide a comprehensive examination of the impact of hotel booking cancelations and to evaluate the suitability of various machine learning algorithms for predicting and mitigating their effects. It begins by delving into the significance of cancellations for hotels, examining their influence on aspects like revenue loss, diminished room occupancy, and the costs of handling last-minute alterations. The study identifies the various ways in which cancelations can impact hotel revenue and operations, including loss of income, reduced room occupancy, and the costs associated with accommodating last-minute changes. Following this, the study conducts a comparative analysis of various machine learning algorithms for their effectiveness in predicting booking cancelations. The comparison is based on factors such as accuracy, interpretability, and ability to identify the factors that contribute to cancelations. The results of the comparison provide valuable insights into the relative performance of different machine learning algorithms in the context of hotel booking cancelations. The findings suggest that certain algorithms perform better than others in terms of accuracy and interpretability and highlight the importance of considering these factors when selecting an appropriate algorithm for hotel revenue management. Finally, the conclusion of the study highlights the practical applications of the research findings for hotels. By utilizing the insights gained from this study, hotels can improve their revenue management and customer satisfaction by better predicting and managing booking cancelations.

Acknowledgements,

I extend my heartfelt thanks to Professor Angelos Markos for his exceptional guidance and steadfast encouragement throughout my thesis project.

Contents

1 Introduction

The hospitality industry is constantly facing challenges such as fluctuations in demand, changes in consumer preferences and economic factors. One of the major challenges for hotels is managing cancelations and trying to predict future bookings. To effectively navigate these challenges, hotels are turning to forecasting models and revenue management strategies.

Cancelations have a significant impact on demand-management decisions in the context of revenue management. When a hotel receives a cancellation, it loses the revenue from that booking and must also deal with the added challenge of trying to fill that room again. This can lead to a decrease in overall revenue, as well as a decrease in occupancy rate. One way to mitigate the impact of cancelations is by using forecasting models that can predict future bookings. Machine learning models can be used to analyze historical data and identify patterns that can help hotels anticipate changes in demand for rooms. By using these models, hotels can better prepare for the ups and downs of the market and make more informed decisions about their pricing and inventory.

For example, according to the findings of Morales and Wang's study in 2010, the results undeniably indicate that utilizing features obtained and derived from hotel bookings in their property management system (PMS) databases is a highly reliable method for predicting outcomes with great precision. In simpler terms, the research conclusively proves that information extracted from hotel bookings can be effectively used to make highly accurate predictions (Morales & Wang, 2010).
In addition to forecasting models, hotels can also use machine learning algorithms to identify patterns in booking cancelations. For example, a hotel may use machine learning to identify patterns in the timing of cancelations, such as a tendency for guests to cancel closer to their arrival date. By identifying these patterns, hotels can take proactive steps to mitigate the impact of cancelations, such as by offering incentives for guests who reschedule their reservations or by implementing more flexible booking policies.

In conclusion, the hospitality industry is constantly facing challenges and hotels are always looking for ways to adapt and improve their operations. By using machine learning models for forecasting and identifying patterns in booking cancelations, hotels can better predict and adapt to changes in demand. Additionally, by offering flexible booking policies and incentives, hotels can mitigate the impact of cancelations and encourage guests to reschedule their reservations. With the right approach, hotels can improve their revenue management and occupancy rate.

The primary purpose of this study is to conduct a detailed comparative analysis of various machine learning algorithms to determine their effectiveness in predicting hotel booking cancelations and offer valuable insights into their relative performance. This analysis will include, but not be limited to, algorithms such as decision trees and random forests. These insights will not only contribute to academic knowledge in the field of hospitality management and predictive analytics but also offer practical applications for hoteliers.

By identifying the most effective algorithms, hotels can improve their revenue management strategies, adopt more flexible booking policies, and enhance overall customer satisfaction. Ultimately, this study seeks to bridge the gap between advanced data analytics and practical revenue management solutions in the hospitality industry.

The thesis is structured as follows. Chapter 2 provides a comprehensive overview of hotel revenue management. It delves into the historical development of revenue management and outlines its basic concepts. The role of revenue management in the hospitality industry is discussed, along with real-world examples of its implementation. A significant focus is placed on the concept of overbooking, exploring both its advantages and disadvantages. Chapter 3 is dedicated to exploring various machine learning approaches for predicting hotel booking cancellations. It begins with a general overview and then delves into different types of learning: supervised, unsupervised, and reinforcement learning. Detailed sections are devoted to various classifiers, including Decision Tree, Random Forest, Extra Tree, Gradient Boosting, XgBoost, LightGBM, Logistic Regression, KNeighborsClassifier, and Support Vector Machine. Each classifier is examined in terms of its mathematical foundation, advantages, limitations, and practical implications. The methods and tools used in the study are described in Chapter 4. It includes a detailed description of the dataset, the programming language and environment used, with a specific focus on PyCaret, and the application of predictive analytics to hotel booking cancellations. Chapter 5 presents the data analysis carried out in the study. It includes the handling of missing values, the detection and treatment of outliers, exploratory data analysis, and predictive modeling. The chapter further breaks down the predictive modeling section into machine learning model comparison, analysis of model performance, and examination of feature importance.

The final chapter summarizes the findings of the thesis, drawing conclusions from the research and suggesting implications for the field of hotel revenue management. This chapter also offers recommendations for future research based on the study's outcomes.

2 Introduction to Hotel Revenue Management

2.1 History of Revenue Management

To delve deeper into the concept of Revenue Management, it is necessary to understand its core principle. Kimes encapsulated this principle succinctly, stating, "Revenue management or yield management is a process by which the appropriate room is sold to the appropriate customer at the right price at the right time" (Protopapadakis, 2013). Revenue Management revolves around analyzing consumer behavior, understanding their perception of the product or service's value, and shaping the pricing strategy to maximize revenue.

Originating in the early 1980s in America, Revenue Management emerged from the aviation industry's need to maintain competitiveness. The underlying concept was to sell enough seats at discounted prices to cover operating and fixed costs. Once these discounted seats were taken, the remaining seats would be sold at higher prices, thereby maximizing revenue. This principle proved to be highly effective and soon found applications across different industries, including hotels, car rentals, and more recently, the wider service sector (Kimes et al., 1989a).

The primary objective of Revenue Management is to optimize revenue and profit by meticulously analyzing customer behavior and market conditions, and then adjusting prices and inventory in response to these insights. However, to fully appreciate its impact, one must understand that Revenue Management goes beyond just pricing. It is a comprehensive approach that encompasses demand forecasting, inventory control, and overall business strategy.

For instance, in the context of hotels, Revenue Management is not just about setting room rates. It involves analyzing booking patterns, seasonality, market segments, and competitive set, amongst others. It is about understanding when to sell a room at a premium price and when to offer a discount to ensure maximum occupancy. It also includes optimizing non-room revenue from ancillary services like food and beverage, spa services, etc.

Furthermore, technological advancements have greatly enhanced the capabilities of Revenue Management. Modern Revenue Management Systems (RMS) use sophisticated algorithms and artificial intelligence to analyze past data, monitor real-time market conditions, and predict future trends. These tools provide actionable insights that empower decision-makers to make data-driven strategic decisions. For instance, RMS can help identify when demand is likely to peak and suggest optimal pricing to maximize revenue during such periods (Bounatirou & Lim, 2020).

The evolution of Revenue Management has also seen it move beyond a purely tactical function to a strategic role. It is now a critical part of the decision-making process at the highest levels of management. It informs strategic decisions like property development, renovations, and branding, amongst others.

2.2 Basic ideas of revenue management

Revenue Management hinges on the basic tenets of economics – supply and demand – to shape an effective pricing strategy. This concept, though seemingly simple, influenced by a complex array of factors. Creating a strategy that maximizes revenue and optimizes resource utilization requires a nuanced understanding of the market and competition It also demands a keen awareness of industry trends, prevailing socio-political-economic conditions on both local and global scales, and potential external influences.

To implement Revenue Management effectively, certain prerequisites must be met.

• The product or service must be perishable or exhaustible.

• The amount of product or service available for sale must be fixed.

• The product or service should have a degree of price flexibility.

Given these stipulations, it's evident that the hospitality sector – and hotel businesses, in particular – are ideally suited to apply Revenue Management strategies. In the hotel industry, a room unsold for a specific day represents lost revenue that cannot be recouped. Thus, the room is considered a perishable product. Moreover, the number of rooms available for sale is constant, given the fixed infrastructural capacity of a hotel. Finally, hotels have significant flexibility in terms of pricing.

Further extending this, Revenue Management's power doesn't end with mere room pricing. It also applies to other aspects of hotel operations such as function rooms, catering services, spas, and other amenities. Each of these services follows the same fundamental principles and can be optimized using revenue management techniques. This approach ensures that every revenue-generating aspect of a hotel's operation is optimally priced and utilized, thereby maximizing the overall revenue.

In the context of dynamic pricing, advanced Revenue Management systems employ machine learning and artificial intelligence to analyze past data, track real-time demand, and predict future trends. This technology-driven approach provides more precise and actionable insights, enabling hoteliers to make informed pricing decisions.

Moreover, integrating Revenue Management with Customer Relationship Management (CRM) can yield even more profound results. By understanding customer preferences, buying behaviors, and sensitivity to price changes, hotels can personalize their offers and maximize the value provided to each guest. This not only increases revenue but also enhances customer satisfaction and loyalty.

2.3 The role of Revenue Management

In today's complex digital age, where variables shift within seconds and consumer demand fluctuates unpredictably, managing and pricing accommodations has become a critical aspect of hospitality. Amid the recent pandemic, the demand for accommodation changed radically, causing a significant hit on the tourism industry worldwide. Consequently, the market has been flooded with a myriad of opportunities and challenges, escalating the urgency for efficient, real-time strategic pricing.

With the market in constant flux, it's clear that static pricing models are obsolete. The need for a dynamic pricing strategy – one that takes into account various external and internal factors such as demand, competition, and booking patterns – has become paramount. Such a strategy allows for maximum yield, ensuring profitability and sustainability even in the face of adversity.

Revenue management technology has evolved from a 'nice-to-have' tool to an absolute necessity for survival and growth in the tourism industry. These technologies help in analyzing large amounts of data, predicting customer behavior, and accordingly adjusting prices. The powerful analytics not only yield valuable insights but also empower decision-makers to make informed choices. Hence, these technologies play a pivotal role in filling any potential gap in the revenue management planning of a hotel, particularly one that aims for long-term growth and sustainability.

Investments in advanced revenue management technologies have skyrocketed over the years. Hotels are now leveraging these technologies, along with expert human resources and strategic partnerships, to predict and respond to market changes more effectively. The combination of skilled revenue management professionals and cutting-edge technology helps extract valuable insights from the vast amount of data available today. The results are accurate forecasts that inform strategic decisions and create competitive advantage.

2.4 Cases of Revenue Management implementation

In the hospitality industry, the application of Revenue Management (RM) is often observed in situations where there is uncertainty on the part of management regarding the optimal timing for selling a product. Specifically, hotel accommodation is sold before they are consumed by the customer, leading to a lack of guarantee that the product will indeed be consumed, especially in cases where the customer does not show up. This results in a loss of revenue, as the unconsumed product is considered a lost opportunity. Effective use of Revenue Management allows for accurate determination of customer numbers and minimizes revenue and profit loss (Marcus & Anderson, 2008).

Furthermore, forecasting is one of the most potent tools employed by Revenue Management. In instances where hotels are unaware of customer needs, it is advisable to gather customer history using appropriate technological means. This enables the prediction of demand, which in turn informs pricing strategies and room availability (Badinelli, 2000).

Additionally, most businesses in the hospitality sector frequently experience fluctuations in demand, necessitating the application of Revenue Management. RM techniques contribute to balancing these demand fluctuations by specifying the time periods during which price increases and decreases, promotions, and sales of the appropriate product to the right customer at the right price should occur. This is also useful in addressing emergency situations.

2.5 Overbooking

In hospitality enterprises, reservation control activities are generally divided into two categories: room allocation and overbooking. Room allocation refers to 'booking limits' as per demand, which typically has a dynamic character (Baker & Collier, 1999), while overbooking is a process that determines the total number of rooms (Toh & Dekay, 2002).

Overbooking is a practice implemented by hotel businesses with the aim of maximizing revenues. This practice, known as a way, to recover or minimize losses resulting from cancellations or customers not showing up, has become widespread (Kimes et al., 1998). The reservation department of hotel businesses must address key issues, including early departures, cancellations of rooms or services, and customer no-shows.

To prevent financial losses due to the above circumstances, the overbooking tactic is commonly used. In instances where the implementation of overbooking is not feasible, a 'first-come-first-served' approach is employed. The major disadvantage of this method is customer dissatisfaction, as it can foster a sense of discrimination among guests (Choi & Mattila, 2004).

Regarding the decision taken by hotel businesses on which customers have priority in cases of overbooking and room shortages, a method is predominantly used that is based on hierarchy. More specifically, priority is given to frequent travelers and regular customers, followed by guests of professional events and their organizers, and groups traveling via airlines with the maximum number of overnight stays (DeKay et al., 2004).

However, there are drawbacks that, with the right management of the board and human resources of hotel businesses, can be converted into advantages, making the use of the overbooking method essential and beneficial.

2.5.1 Overbooking Advantages

Overbooking as a management strategy is crucial to the operations of hotel accommodations. This strategy is usually crafted and executed by a team of revenue and sales managers. Their goal is to ensure the hotel maximizes its profit while maintaining guest satisfaction.

One of the chief benefits of the overbooking technique lies in its ability to generate revenue by allowing a greater number of rooms to be booked than what the hotel physically has. This method takes into account the potential for last-minute cancellations and no-shows. This is especially critical in the hotel industry, where booking cancellations or no-shows could lead to significant revenue loss. By overbooking, hotels can effectively cushion themselves against such unforeseen circumstances, turning potential losses into opportunities for increased revenue (Wangenheim & Bayón, 2007).

However, the overbooking strategy must be implemented with careful forecasting. Poor execution can lead to customer dissatisfaction and potential reputational damage. Thus, hotel managers aim to make their reservations equivalent to the number of cancellations or no-shows they expect (Phumchusri & Maneesophon, 2014). They utilize historical data, algorithms, and even machine learning tools to improve the accuracy of their forecasts (Weatherford & Kimes, 2003).

The role of the management and human resources is pivotal in implementing overbooking effectively. They need to maintain a keen focus on demand forecasting. This means understanding and predicting customer behavior to ensure supply meets demand (Pullman & Rodgers, 2010). Successful forecasting not only helps in implementing overbooking but also in achieving operational efficiency and superior customer service. In other words, by accurately predicting the demand, they can guarantee that all customers are serviced adequately.

2.5.2 Overbooking Disadvantages

While overbooking may initially seem like a strategy to maximize revenue, it can actually have significant negative implications on both customer satisfaction and the hotel's financial situation.

The sense of injustice felt by customers due to the lack of service that results from overbooking is a substantial issue. It leads to customer dissatisfaction, and in turn, loss of the establishment's credibility (Wirtz, 2003).

Another noteworthy concern arises when a hotel, due to overbooking, is forced to find alternative accommodation for customers. This situation not only incurs the cost of arranging accommodation in a nearby hotel but also includes transportation expenses for the guest to the alternative location (Hwang & Wen, 2009).

The most significant costs generated by overbooking are the loss of dissatisfied customers and the potential loss associated with prospective customers. Clients negatively affected by overbooking are less likely to select the same hotel in the future and are inclined to post negative reviews that can influence potential customers' decisions (Ivanov, 2007).

3 Machine learning approaches for predicting hotel Booking Cancelations

3.1 Overview

Machine Learning is a subset of Artificial Intelligence that allows applications to yield precise prediction outcomes without the need for task-specific programming. It is also a branch of Computer Science with the main objective to construct models predicated on algorithms, apply these models to data, and subsequently make predictions or decisions. Machine Learning has a wide spectrum of applications, including but not limited to, text processing, natural language processing, and speech recognition.

Mathematically, machine learning models are often represented by complex functions or data structures, with several adjustable parameters that are optimized during the training process. The principal objective of this learning process is to create a model that not only fits the given data well but also can generalize effectively to unseen data, making accurate and reliable predictions.

To measure the performance of these models, and ensure they are meeting their intended objectives, various evaluation metrics are used. These can range from accuracy, precision, recall, and F1 score for classification tasks, to mean squared error, mean absolute error, and R squared for regression tasks.

Sometimes, a single model might not be sufficient to achieve the desired predictive performance. In such cases, ensemble learning methods, which combine predictions from multiple models, can be applied to improve overall results. This process can offer more robust and stable predictions, mitigating the limitations of a single model.

Hyperparameter tuning is another crucial aspect of model development. Hyperparameters are parameters of the learning algorithm itself, which aren't learned from the data, but are set prior to training. Proper tuning of these parameters can significantly impact the model's performance.

Once a model is trained, evaluated, and fine-tuned, it can be deployed in various real-world applications to provide predictions on new, unseen data. These applications range widely across sectors and industries, from recommendation systems in e-commerce, to predictive maintenance in manufacturing, and disease diagnosis in healthcare.

Generally, the field of machine learning is highly dynamic, with continuous advancements and research contributing to the development of new and improved model architectures, techniques, and applications. From deep learning and reinforcement learning to transfer learning and beyond, machine learning continues to push the boundaries of what's possible in the world of data-driven decision making.

ML splits into three major types:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

3.1.1 Supervised Learning

Supervised Learning is a category of Machine Learning in which the algorithm learns from precisely tagged data. The Machine Learning process involves providing a subset of data for training, which includes the problem, the solution, and the type of data. The training, testing, and final datasets typically share similar features. Once trained, the algorithm identifies patterns in the provided data that correlate the input and the output. Finally, the educated model is applied to the final dataset and seeks to accurately predict the class labels for unseen instances. Supervised Learning is divided into two problem types: Classification and Regression (Wang et al. 2016; Saravanan & Sujatha 2018).

- Classification methods
  - Linear Classifiers
  - Support Vector Machines (SVM)
  - Decision Trees
  - K-Nearest Neighbor
  - Random Forest
  - Neural Networks
- Regression methods
  - Linear Regression
  - Logistic Regression

○ Polynomial Regression

○ Neural Networks

3.1.2 Unsupervised Learning

Unsupervised Learning, as opposed to its supervised counterpart, has the unique ability to operate with unlabeled data for both input and output. This inherent quality renders the need for human intervention obsolete, eliminating the necessity to manually annotate the data for machine interpretation. Given the absence of predefined labels, unsupervised learning algorithms induce hidden structures within the data. They discern patterns by identifying resemblances and disparities within the data corpus (Wang, 2021).

Common Unsupervised Learning approaches:

● Clustering

○ K-Means clustering (Exclusive and Overlapping Clustering)

○ Ward's linkage (Hierarchical Clustering)

○ Average linkage (Hierarchical Clustering)

○ Complete (or maximum) linkage (Hierarchical Clustering)

○ Single (or minimum) linkage (Hierarchical Clustering)

○ Gaussian Mixture Models (Probabilistic clustering)

● Association Rules

● Apriori Algorithms

● Dimensionality Reduction

● Principal Component Analysis (PCA)

● Singular Value Decomposition

● Autoencoders

3.1.3 Reinforcement Learning

Reinforcement Learning aligns itself with the principles of supervised learning, but it departs from the conventional methodology of training the model with subsets of data. Instead, it adopts a "trial and error" approach. The operational dynamics of the algorithm revolve around an adjudicator and an incentive structure.

In each iteration of the algorithm, the generated outcomes undergo assessment by the adjudicator, gauging their desirability. If the outputs align with the desirable criteria, the adjudicator promotes this solution through a reinforcement or reward mechanism. On the contrary, if the outputs fall short of the mark, the algorithm is run again, embarking on a new cycle of trial and error until it generates a more satisfactory result (Littman & Moore, 1996).

Some applications of Reinforcement Learning can be applied in the following fields:

● Resources management in computer clusters

● Traffic Light Control

● Robotics

● Web System Configuration

● Chemistry

● Personalized Recommendations

● Bidding and Advertising

● Games

● Deep Learning

3.2 Decision Tree Classifier

A Decision Tree Classifier is a simple, yet effective classification algorithm used in machine learning. It models decisions and their potential outcomes in a hierarchical tree structure, which closely resembles human decision-making processes. This algorithm works by dividing the dataset into smaller subsets based on various conditions, effectively creating a 'tree' of decisions. Each split in the tree represents a choice between possible alternatives, based on the features present in the data. As the tree grows, it forms branches that represent these decision paths, leading to leaf nodes that signify the final classification outcomes. The process of forming these decision rules is guided by the patterns and relationships discovered in the training data, where the algorithm learns which features most effectively split and classify the data. (Breiman et al.1984).

3.2.1 Mathematical Foundation

In a Decision Tree, each split is chosen to maximize class separability (Breiman et al.1984). This is quantified using Gini impurity, a metric that computes the probability of a randomly chosen element being incorrectly labeled if it was randomly labeled according to the distribution of labels in the set. For a set S, containing instances from K classes, Gini impurity $I_G(S)$.

$$I_G(S) = 1 - \sum_{k=1}^{K} p_k^2$$

where p_k is the relative frequency of class k within S (Raileanu & Stoffel, 2004). Lower Gini impurity values indicate better splits, leading to more homogeneous nodes.

3.2.2 Advantages

1. Simplicity: Decision trees are simple to understand and interpret, as they visually represent decision making.
2. Non-Parametric: They do not assume any distribution of the data, making them suitable for non-linearly separable data.
3. Variable Importance: Provides a clear indication of which fields are most important for prediction or classification.
4. Data Adaptability: Can handle both numerical and categorical data.

3.2.3 Limitations

1. Overfitting: Decision trees are prone to overfitting, especially if they are deep with many branches.
2. Instability: Small changes in the data can lead to a different split, causing a high variance in the model.

3.2.4 Practical Implications

Decision Tree Classifiers are widely used in operational settings, like strategic planning and customer relationship management. They are particularly beneficial in the banking sector for credit scoring, in the healthcare industry for diagnosing medical conditions, and in the e-commerce industry for predicting whether a customer will purchase a product.

3.3 Random Forest Classifier

The Random Forest Classifier, a versatile ensemble learning method, is primarily utilized for classification tasks, though it is also capable of performing regression. This technique operates by generating a large number of decision trees during the training phase. Each tree in the forest produces a prediction, and the final output of the classifier is determined by the mode of these predictions for classification tasks, effectively aggregating the decisions of individual trees to enhance the overall accuracy and robustness of the model. The power of Random Forests lies in this ensemble approach, as it significantly reduces the tendency of individual decision trees to overfit their training data. By combining the predictions of multiple trees, each trained on a random subset of the data, Random Forests are able to capture complex patterns while maintaining the ability to generalize well to new, unseen data. (Breiman, 2001).

3.3.1 Mathematical Foundation

Mathematically, the Random Forest algorithm improves upon the simplicity of decision trees by creating an ensemble of diverse trees, each contributing to the final prediction. The decision function for a Random Forest with N trees is given by:

$$\hat{y}(x) = \frac{1}{N}\sum_{n=1}^{N} T_n(x)$$

where $T_n(x)$ represents the prediction of the nth decision tree for an input vector x. During the training phase, each tree T_n is grown on a bootstrap sample (randomly drawn with replacement) of the training data, and at each node, a random subset of features is considered for splitting. This randomness injects diversity and decorrelates the trees, thereby enhancing the ensemble's predictive performance. The final prediction $\hat{y}(x)$ is typically obtained by majority voting for classification or averaging for regression.

By aggregating the predictions of individually weak learners (trees), the Random Forest becomes a powerful model that can capture complex structures in the data while maintaining generalizability. The model's ability to handle large datasets with high dimensionality, its intrinsic method for feature selection, and its robustness to noise make it a popular choice across various domains. Furthermore, the algorithm provides measures of feature importance, which can be derived from the sum of the decrease in Gini impurity or mean squared error (for regression) across all trees in the forest, thereby offering insights into the underlying structure of the data (James et al., 2013).

3.3.2 Advantages

1. Accuracy: Random forests often produce highly accurate classifiers by reducing the variance component of the model's error.

2. Robustness: It is less affected by noise in the data compared to individual decision trees.
3. Feature Importance: It can handle a large number of input variables and is able to determine the most significant variables.
4. Versatility: Can be used for both classification and regression task and has methods for balancing errors in unbalanced data sets.

3.3.3 Limitations

1. Complexity: They are more complex and computationally more expensive than individual decision trees.
2. Interpretability: As an ensemble method, it is more difficult to interpret than a single decision tree.

3.3.4 Practical Implications

Random Forest is a powerful tool in a variety of applications, from banking (for loan default prediction) to medicine (for disease identification). It is also commonly used in the field of bioinformatics for classifying different types of tissue or samples based on gene expression data.

3.4 Extra Tree Classifier

The Extra Trees Classifier, an abbreviation for Extremely Randomized Trees Classifier, is an ensemble learning method closely related to the Random Forest Classifier. It is distinguished by the increased randomness introduced during the tree construction process. Each tree in the Extra Trees ensemble is built using the entire dataset rather than a bootstrap sample, and splits are determined by random thresholds for each feature, instead of the optimal thresholds sought by traditional decision trees and Random Forests. This randomness helps in creating an even more diversified model that can be less prone to overfitting (Geurts et al., 2006).

3.4.1 Mathematical Foundation

The Extra Tree Classifier constructs a multitude of decision trees, each developed with a certain degree of randomness in the selection of features and thresholds for splitting.

by averaging the outputs from all N trees. Each tree T_n ontributes its prediction, influenced by the randomness parameter Θ_n , ensuring diversity among the trees:

$$\hat{y}(x) = \frac{1}{N}\sum_{n=1}^{N} T_n(x;\Theta_n)$$

By incorporating this randomness, the Extra Trees Classifier often yields an increase in model robustness and is particularly useful for preventing overfitting in scenarios where the dataset may have a large number of features.

The Extra Trees algorithm's capacity for handling complex datasets with potentially irrelevant features, its computational efficiency due to the simplicity of the random splits, and its ability to provide feature importance metrics, similar to Random Forests, make it a valuable tool for a wide range of predictive tasks in machine learning.

3.4.2 Advantages

1. Speed: Extra Trees can be faster to train than other ensemble classifiers like Random Forest due to the use of the whole dataset and random thresholds.
2. Reduced Overfitting: By randomizing the cut-points and using all data, they tend to overfit less compared to regular decision trees.
3. No Need for Bootstrap Sampling: This allows for a slight increase in speed since every sample is used to build the trees.
4. Feature Selection: Like Random Forests, Extra Trees can provide insights into feature importance, helping in dimensionality reduction.

3.4.3 Limitations

1. Complexity and Interpretability: Similar to Random Forests, an ensemble of many trees is more complex and harder to interpret than a single decision tree.
2. Computational Resources: It can still be computationally intensive, particularly with very large datasets and many trees.

3.4.4 Practical Implications

Extra Trees are applicable in various domains like bioinformatics for gene classification, finance for credit scoring, and computer vision for object recognition. They are particularly useful when the dataset is large and there's a need for speed in building a robust model.

3.5 Gradient Boosting Classifier

The Gradient Boosting Classifier is a powerful machine learning technique that builds on decision trees to create a predictive model. It sequentially adds predictors to an ensemble, each one correcting its predecessor. This method belongs to the family of boosting algorithms that convert weak learners into strong ones. Gradient boosting is particularly adept at dealing with heterogeneous data and capturing complex variable interactions. One of the key strengths of the Gradient Boosting Classifier is its flexibility. It allows for the optimization of different loss functions, making it applicable to a wide range of predictive modeling problems, from regression to classification (Friedman, 2001).

3.5.1 Mathematical Foundation

Gradient Boosting methodically builds an ensemble model in a stage-wise fashion. Each new stage introduces a weak learner $h_t(x)$ to amend the errors of the existing ensemble. The ensemble model at stage t , $F_t(x)$, is updated by adding the product of the learning rate ρ_t and the weak learner $h_t(x)$ to the previous stage's model $F_{t-1}(x)$:

$$F_t(x) = F_{t-1}(x) + \rho_t h_t(x)$$

Mathematically, the process can be described as an optimization problem where the goal is to find a set of basic predictors and their corresponding weights that, when combined, minimize the loss function, which measures the difference between the predicted and actual values.

The strength of Gradient Boosting lies in its ability to learn from the missteps of previous stages, allowing the ensemble to become increasingly accurate with each iteration. This characteristic makes it particularly effective for complex datasets with intricate relationships between variables. By iteratively adding these small improvements, Gradient Boosting can turn a collection of simple and initially less accurate models into a single, highly accurate predictive model.

3.5.2Advantages

1. Flexibility: Can be used with a variety of loss functions and hence can be adapted to various problems.
2. Predictive Power: Often provides highly accurate models and can capture complex non-linear patterns.

3. Feature Importance: Naturally performs feature selection, which can be beneficial with high-dimensional data.

3.5.3 Limitations

1. Computational Cost: Training can be slow because trees are built sequentially.
2. Risk of Overfitting: Especially with noisy data and without proper tuning of parameters.
3. Less Interpretability: More complex than simple decision trees, making the model harder to interpret.

3.5.4 Practical Implications

Gradient Boosting Classifiers have been successfully applied in numerous domains, such as financial risk assessment, biology for gene discovery, and web search engines. The method is known for its effectiveness in Kaggle competitions where predictive accuracy is the ultimate measure of success.

3.6 XgBoost Classifier

XGBoost, standing for eXtreme Gradient Boosting, represents a highly optimized, scalable, and fast version of gradient boosting, acclaimed in the machine learning community for its effectiveness and efficiency. This ensemble algorithm builds strong predictive models by combining numerous weak models, typically decision trees, into a single, more accurate and robust model. It has been engineered to maximize computational speed and use of resources, which is instrumental in handling large-scale and high-dimensional data that challenge many traditional algorithms. Moreover, its ability to perform both classification and regression tasks, handle missing data, and provide a platform for parallel processing makes it a comprehensive tool for data scientists.

3.6.1 Mathematical Foundation

XGBoost, a highly efficient implementation of gradient boosting, focuses on computational speed and model performance. It iteratively optimizes a comprehensive objective function at each step t . This function incorporates both the loss l (evaluated between the actual value y_i and the predictive score $\widehat{y_i^{(t-1)}} + f_t(x_i))$ and a regularization term $\Omega(f_t)$which is crucial for controlling model complexity and preventing overfitting.

he equation representing the objective function of XGBoost at iteration t can be expressed as:

$$\text{Obj}^{(t)} = \sum_{i=1}^{m} l\left(y_i, \widehat{y_i^{(t-1)}} + f_t(x_i)\right) + \Omega(f_t)$$

This reflects the two-fold goal of XGBoost: to find a series of functions ft(x)ft(x) that collectively produce the most accurate predictions on the training data, while simultaneously ensuring that the functions remain simple enough to maintain generalizability across different data sets. Such a comprehensive objective function, combining loss minimization with regularization, allows XGBoost to deliver models that are not only powerful in terms of predictive performance but also robust against the risk of overfitting, making it an exceptional tool for a wide range of machine learning tasks.

3.6.2 Advantages

1. Speed and Performance: XGBoost is designed to be highly efficient, scalable, and portable.
2. Regularization: It includes L1 (Lasso) and L2 (Ridge) regularization which prevents overfitting and improves model generalization.
3. Handling Sparse Data: XGBoost can handle sparse data (missing values or zero elements) by default.
4. Cross-validation: It has an in-built routine to perform cross-validation at each iteration of the boosting process.
5. Flexibility: XGBoost allows users to define custom optimization objectives and criteria.

3.6.3 Limitations

1. Computational Complexity: Although it's fast, XGBoost can be computationally intensive and may require hardware with higher specifications for large datasets.
2. Overfitting Risk: If not tuned properly, XGBoost models can overfit, which impacts their ability to generalize unseen data.

3. Steep Learning Curve: The large number of hyperparameters can be overwhelming and requires significant expertise to tune correctly.

3.6.4 Practical Implications

XGBoost has found immense popularity in areas requiring accurate predictions, such as finance for risk management and credit scoring, healthcare for medical diagnostics, and e-commerce for dynamic pricing strategies. Its robustness and versatility make it a go-to algorithm for both regression and classification problems.

3.7 LightGBM (Light Gradient Boosting Machine)

LightGBM, an acronym for Light Gradient Boosting Machine, is a sophisticated and efficient machine learning framework developed by Microsoft. It excels in handling large datasets, primarily through its innovative use of tree-based learning algorithms. The framework's design focuses on speed and efficiency, aiming to outperform similar models like XGBoost in these areas. What sets LightGBM apart is its unique approach to building trees and handling data. It employs techniques such as Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB), which enhance its capability to manage high-dimensional data effectively. This efficiency makes LightGBM a preferred choice in scenarios demanding rapid processing of extensive data. Notably, its distributed nature allows for parallel and GPU learning, further boosting its performance and making it a powerful tool in the realm of predictive analytics and classification tasks (Ke et al., 2017).

3.7.1 Mathematical Foundation:

LightGBM, an advanced implementation of gradient boosting, stands out for its efficiency with large datasets and high-dimensional features. It employs a refined objective function, balancing the loss l across all m instances and incorporating a regularization component Ω for each of the J features:

$$\text{Obj} = \sum_{i=1}^{m} l(y_i, \hat{y}_i) + \sum_{j=1}^{J} \Omega(f_j)$$

A key innovation in LightGBM is its unique handling of data splitting. Unlike traditional methods, LightGBM utilizes a histogram-based algorithm, dividing continuous feature values into discrete bins. This technique significantly reduces the computational cost of handling continuous features, thereby accelerating the training process without compromising on the model's accuracy. Furthermore, LightGBM adopts an exclusive feature bundling mechanism, which effectively reduces the feature space when dealing with sparse data, further enhancing the model's efficiency.

These technical innovations contribute to LightGBM's reduced memory usage and increased speed of model training, positioning it as an optimal solution for scenarios involving large volumes of data or when computational resources are a constraint. This efficiency makes LightGBM not just a practical choice for large-scale data analysis but also for real-time applications where rapid processing and decision-making are crucial. Its ability to deliver high performance with lower resource consumption meets the evolving demands for more effective and scalable machine learning tools in various fields, ranging from financial modeling to scientific research (Ke et al., 2017).

3.7.2 Advantages:

1. Speed & Efficiency: Historically faster training than other boosting algorithms, especially on large datasets.
2. Memory Optimization: Uses histogram-based techniques, which is less memory-consuming.
3. Highly Customizable: Offers a plethora of tunable parameters which can be customized for specific tasks.
4. Support for Parallel Learning: Can harness the power of multi-core machines.
5. Handling of Large Datasets: Designed for scale; can handle datasets with a large number of features or data points.

3.7.3 Limitations:

1. Overfitting on Small Data: Due to its complexity, there's a risk of overfitting, especially when the dataset is small.
2. Steep Learning Curve: The abundance of tunable parameters can be overwhelming and might require a deep understanding for effective tuning.
3. Less Interpretable: Like other boosting algorithms, results might not be as interpretable as simpler models.

3.7.4 Practical Implications:

LightGBM is popular in various applications where speed and efficiency are paramount, especially in situations with vast amounts of data or features. Industries ranging from e-commerce (for recommendation systems) to banking (fraud detection) harness its power. It's particularly favored in Kaggle competitions due to its performance capabilities. However, its complexity implies that domain expertise is often required to unlock its full potential and avoid pitfalls like overfitting.

3.8 Logistic Regression

Logistic Regression, often misunderstood as a regression algorithm due to its name, is actually a powerful tool for classification tasks. It excels in estimating the probability that a given instance, or input, falls within a specific category. Its most frequent use is in binary classification, where it predicts one of two possible outcomes, such as 'yes' or 'no', 'positive' or 'negative'. The algorithm works by applying a logistic function to transform linear combinations of input features into probabilities. This sigmoid-shaped function maps any real-valued number into a value between 0 and 1, making it ideal for calculating probabilities. Logistic Regression is highly valued for its simplicity and interpretability, allowing for a clear understanding of how input variables influence the probability of class membership. (Stoltzfus, 2011).

3.8.1 Mathematical Foundation

The core of logistic regression is the logistic function, often referred to as the sigmoid function:

$$p(x) = \frac{1}{1+1^{e^{-z}}}$$

Where z is a linear combination of the input features:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

The log-odds or the logit transformation, is the logarithm of the odds ratio:

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta 0 + \beta 1 x 1 + \beta 2 x 2 + \cdots + \beta n x n$$

To estimate the values of the coefficients ($\beta_0$values), logistic regression employs Maximum Likelihood Estimation (MLE). MLE is a method used in statistics to estimate the parameters of a statistical model. In the context of logistic regression, MLE seeks to find the set of coefficients that maximize the likelihood function, which measures how probable the observed data is, given a set of coefficients. This process involves using optimization algorithms to maximize the likelihood function over the parameter space.

MLE in logistic regression aims to produce a model that best explains the relationship between the feature variables and the probability of the outcome. The optimization process iteratively adjusts the coefficients, seeking the values that yield the highest likelihood for the observed data. This results in a model that can effectively discriminate between the two classes in the dataset, thereby providing a robust method for classification problems.

3.8.2 Advantages

1. Interpretability**:** One of the main strengths of logistic regression is the interpretability of the coefficients. They give insights into the relationship between the independent variables and the outcome, which is crucial for understanding the underlying mechanisms or for making policy recommendations.
2. Low Variance**:** It tends not to overfit on high-dimensional datasets as much as more complex models might.
3. Efficiency**:** It requires fewer computational resources than more complex models. It's faster and requires less memory to execute.
4. Flexibility**:** It can handle both continuous and categorical variables.
5. Regularization**:** Techniques like L1 (Lasso) and L2 (Ridge) regularization can be applied to logistic regression to prevent overfitting, allowing it to handle large sets of features and prevent multicollinearity.
6. Probability Estimation**:** Unlike some other classification algorithms that can only provide the final classification, logistic regression provides the probability of the outcome, which can be crucial for decision-making processes in practical scenarios.

3.8.3 Limitations

1. Linearity Assumption: It assumes a linear relationship between the log odds of the output and the input variables. This can be restrictive and may not capture complex relationships in the data.

2. Binary Outcome**:** Traditional logistic regression is meant for binary classification tasks. For multi-class problems, techniques like multinomial logistic regression or 'one vs. all' methods are required.
3. Not Fit for Complex Relationships**:** Logistic regression might not perform as well when there are complex, non-linear relationships in the data. More sophisticated algorithms might be required in such cases.
4. Sensitive to Noise**:** If the dataset has many irrelevant input features, then logistic regression can suffer in terms of performance. Regularization can help, but careful feature selection is essential.
5. Multicollinearity**:** While regularization can handle multicollinearity to some extent, severe multicollinearity can still be problematic for logistic regression, as it makes coefficients unstable and harder to interpret.

3.8.4 Practical Implications

Logistic regression is a valuable tool for decision-making, particularly because it offers probabilities, enabling businesses and medical professionals to set custom thresholds tailored to their specific needs, instead of relying on a standard 0.5 threshold. This adaptability is further enhanced by the ability to understand the weight or importance of different features, guiding decision-makers to concentrate on the most critical variables impacting outcomes. Due to its simplicity and efficiency, logistic regression has found widespread use across various fields. In finance, it's used for credit scoring; in medicine, for disease diagnosis; in marketing, for predicting customer churn, among many other applications, illustrating its versatility and effectiveness in a range of contexts.

3.9 KNeighborsClassifier

The KNeighborsClassifier is a type of instance-based learning, distinct in its approach of not constructing a general internal model but rather storing instances of the training data. Classification for a new query point is determined through a majority vote among its nearest neighbors, identified within the training dataset. The class most common among these neighbors is assigned to the query point. This method's effectiveness depends largely on the choice of the number of neighbors ('k') and the distance metric used to measure proximity. While it's adaptable to complex dataset structures, its performance can be affected by the dataset's dimensionality and the chosen 'k' value (Cover & Hart, 1967)

3.9.1 Mathematical Foundation

KNeighborsClassifier, at its core, operates on the premise that data points in a feature space tend to be clustered by their class labels. The methodology involves identifying 'k' training samples closest to a point and predicting the label based on the majority class of these neighbors.

Euclidean Distance: A popular metric used in the KNN algorithm is the Euclidean distance. For two points in an n-dimensional space, the distance is:

$$\mathrm{d}(\mathrm{x}_1, \mathrm{x}_2) = \sqrt{\left\{\sum_{\{i=1\}}^{n} \left(\mathrm{x}_{\{1i\}} - \mathrm{x}_{\{2i\}}\right)^2\right\}}$$

Alternative Distance Metrics:

While the Euclidean distance is commonly used, other distance measures can cater to specific dataset characteristics:

- *Manhattan Distance.* It sums up the absolute differences of their coordinates: (Han, et al., 2022).

$$\mathrm{d}(\mathrm{x}_1, \mathrm{x}_2) = \sum_{\{i=1\}}^{n} \left|\mathrm{x}_{\{1i\}} - \mathrm{x}_{\{2i\}}\right|$$

- *Minkowski Distance.* A generalization of both Euclidean and Manhattan distances, defined by the parameter p.

$$d(x_1, x_2) = \left(\sum_{\{i=1\}}^{n} \left|x_{\{1i\}} - x_{\{2i\}}\right|^p\right)^{\left\{\frac{1}{p}\right\}}$$

It provides flexibility, with p=2p=2 yielding Euclidean and p=1p=1 yielding Manhattan distance (Deza & Deza, 2009).

Choosing the right distance metric is pivotal in KNN's performance, as it directly affects how 'closeness' is quantified. Moreover, the scale and dimensionality of the data are critical considerations, as high-dimensional spaces can lead to the 'curse of dimensionality', where the concept of proximity becomes less meaningful.

Preprocessing steps like normalization or dimensionality reduction may be necessary for optimizing the KNN classifier's performance.

3.9.2 Advantages

1. Simplicity: KNN is straightforward to understand and easy to implement.
2. No Model Assumptions: It makes no prior assumptions about the form of the data distribution, which can be advantageous with real-world data.
3. Adaptability: KNN is a non-parametric model, which means it adapts to the data distribution and can handle versatile data types.
4. Versatility: It can be used for both classification and regression tasks.

3.9.3 Limitations

1. Scalability: KNN can be computationally expensive since it requires storing the entire dataset and calculating distances for each query.
2. Curse of Dimensionality: KNN's performance degrades with high-dimensional data due to the distance measure becoming less meaningful.
3. Sensitive to Irrelevant Features: Since all features contribute equally to the distance computation, irrelevant or noisy features can disrupt the classifier’s performance.
4. Requirement for Feature Scaling: KNN requires feature scaling before training because features on larger scales can disproportionately influence the distance metric.

3.9.4 Practical Implications

KNeighborsClassifier is widely used in applications where relationships between data points can be reasonably captured by a distance metric. This includes fields like finance for credit scoring and fraud detection, healthcare for patient classification, and retail for recommendation systems. It’s particularly useful when the decision boundary is very irregular.

3.10 Support Vector Machine (SVM) Classifier

The Support Vector Machine (SVM) Classifier stands as a robust and supervised machine learning method, renowned for its efficacy in both classification and regression tasks. Ideal for handling complex datasets, particularly those of smaller or medium scale, SVM excels by identifying an optimal separating hyperplane. This hyperplane acts as a boundary that discerns between different class labels within the feature space. The elegance of SVM lies in its use of kernel functions, which enable the algorithm to operate in a high-dimensional space, facilitating the classification of data that is not linearly separable in its original space. By maximizing the margin between different classes, SVM ensures a solution that is not only accurate but also generalizes well to new data, reducing the risk of overfitting which is a common pitfall in machine learning models (Cortes & Vapnik, 1995). This characteristic makes SVM a versatile tool in the predictive modeling domain, where it can be adapted to a wide range of applications from image recognition to bioinformatics.

3.10.1 Mathematical Foundation

SVM aims to find an optimal hyperplane that maximizes the margin between two classes. The equation of the hyperplane is defined as $\boldsymbol{w} \cdot \boldsymbol{x} + b = 0$, where W represents the weights assigned to the features, and b is the bias (Cortes & Vapnik, 1995). SVM ensures that each instance x_i satisfies the condition for correct classification:

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$$

where y_i denotes the class label of instance. This creates the widest possible margin between the classes, enhancing the model's generalization ability

The optimization problem at the heart of SVM is often solved using quadratic programming, which finds the values of W and b that not only separate the classes but also do so with the largest margin. This maximization of margin is pivotal as it contributes to the model's ability to generalize well to unseen data, which is a desirable property in machine learning (Cortes & Vapnik, 1995). Generalization is crucial because it implies that the model can accurately predict the class labels for new, unseen instances, beyond the examples it was trained on, thus reflecting its real-world applicability and robustness.

3.10.2 Advantages

1. Effectiveness: SVMs are known for their effectiveness in high-dimensional spaces.

2. Versatility: The use of different kernel functions enables them to adapt to different types of data.
3. Robustness: They are robust to overfitting, especially in high-dimensional space.

3.10.3 Limitations

1. Scalability: SVMs can be inefficient on large datasets due to their quadratic complexity in the number of samples.
2. Kernel Choice: The choice of the kernel and its parameters can have a large impact on the performance of the SVM.
3. Interpretability: SVM models, especially with non-linear kernels, are less interpretable compared to simpler classifiers like logistic regression.

3.10.4 Practical Implications

SVMs have been successfully applied in fields like bioinformatics for protein classification, image recognition, handwriting recognition, and natural language processing. Their ability to model complex, subtle structures in data makes them a powerful tool for classification tasks where accuracy is paramount.

4 Methods and Tools

4.1 Dataset description

This data presents information about two datasets that contain data related to hotel demand. The first dataset (H1) pertains to a resort hotel, while the second dataset (H2) pertains to a city hotel. Both datasets have a similar structure, with 31 variables that describe 40,060 observations for H1 and 79,330 observations for H2. Each observation represents a hotel booking. The data includes bookings that took place between July 1st, 2015, and August 31st, 2017, including both bookings that were completed and bookings that were cancelled. As this is real hotel data, any information that could identify the hotel or customer has been removed.

Variable	Description	Type

adr	Average Daily Rate	Numeric
agent	ID of the travel agency that made the booking	Integer
arrival_date_day_of_month	Day of the month of the arrival date	Integer
arrival_date_month	Month of arrival date with 12 categories: "January" to "December"	Categorical
arrival_date_week_number	Week number of the arrival date	Integer
arrival_date_year	Year of the arrival date	Integer
assigned_room_type	Code for the type of room assigned	Categorical
babies	Number of babies	Integer
booking_changes	Number of changes/amendments made to the booking from the moment the booking was entered on the PMS until the moment of check-in or cancellation	Integer
children	Number of children	Integer
company	ID of the company/entity that made the booking	Integer
country	Country of origin	Categorical
customer_type	Type of booking	Categorical
days_in_waiting_list	Number of days the booking was in the waiting list before it was confirmed to the customer	Integer
distribution_channel	Booking distribution channel used to make the booking	Categorical
hotel	Type of hotel	Categorical
is_canceled	Value indicating if the booking was canceled	Categorical
is_repeated_guest	Value indicating if the booking name was from a repeated guest	Categorical
lead_time	Lead time is the number of days that elapsed between the entering date of the booking into the PMS and the arrival date	Integer
meal	Type of meal booked	Categorical
previous_bookings_not_canceled	Number of previous bookings not canceled by the customer prior to the current booking	Integer
previous_cancellations	Number of previous bookings that were canceled by the customer prior to the current booking	Integer
required_car_parking_spaces	Number of car parking spaces required by the customer	Integer
reservation_status	Reservation last status	Categorical
reservation_status_date	Date at which the last status was set	Date
reserved_room_type	Code of room type reserved	Categorical
stays_in_week_nights	Number of week nights	Integer

stays_in_weekend_nights	Number of weekend nights	Integer
total_of_special_requests	Number of special requests	Integer

Figure 1: List of the dataset's columns and description

4.2 Programming language and environment

The next phase of the research involves data analysis and the implementation of machine learning models using the Python programming language. Python was first developed in 1991 by Guido van Rossum (Venners, 2013) and has since become one of the most widely used programming languages in data science projects. One of the advantages of using Python is its relatively simple and human-like syntax, which makes it relatively easy to learn and understand. Additionally, Python has a wide range of open-source libraries created by various contributors that provide pre-built functions and commands to perform tasks that would be otherwise time-consuming or difficult to implement. Two of the most popular libraries used for data analysis and machine learning are Pandas (Pandas, 2023) and scikit-learn (Scikit-learn,2023), and they will be used in this dissertation. latest Python libraries for data science projects.

The programming environment used is Anaconda (Anaconda,2023). Anaconda is a distribution of Python that comes with a collection of pre-installed packages and tools that are commonly used in data science and scientific computing. This distribution includes Python, Jupyter Notebook, and many other libraries and tools. Anaconda makes it easy to install, manage and update packages, making it a popular choice among data scientists and researchers.

Jupyter Notebook (Jupyter, 2023) is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It is often used for data science and machine learning projects
Together, Anaconda and Jupyter Notebook provide a powerful and flexible platform for data science and machine learning projects. The combination of Anaconda's package management and Jupyter Notebook's interactive and collaborative features make it easy to perform complex data analysis tasks.

4.2.1 Pycaret

PyCaret is an open-source, low-code machine learning library in Python that automates end-to-end machine learning workflows. It is designed to simplify the machine learning process by providing a consistent interface that abstracts away the complexity of the underlying algorithms, enabling users to train and deploy machine learning models with minimal coding and configuration. Including Classification, Regression, Clustering, Anomaly Detection, NLP, and Association Rules. It was chosen for a specific project to make separate predictions about booking cancellations for each hotel.

PyCaret offers a wide range of functionalities that include data preprocessing, feature engineering, model selection and tuning, ensemble modeling, and deployment. It supports a variety of machine learning algorithms such as regression, classification, clustering, anomaly detection, natural language processing, and time-series forecasting. One of the key advantages of PyCaret is its ability to automate most of the repetitive tasks involved in building and deploying machine learning models, such as data preprocessing and feature engineering. PyCaret also provides a range of visualizations and performance metrics to help users understand the behavior of their models and identify areas for improvement.

PyCaret is an excellent choice for training multiple models, as it optimizes results by selecting the best-performing model when the auto ML function is invoked. Furthermore, PyCaret allows for the analysis of the chosen model using the plot_model function, which generates a comprehensive report and visualizations to examine the selected model PyCaret (2023).

4.3 Predictive analytics to hotel Booking Cancelations

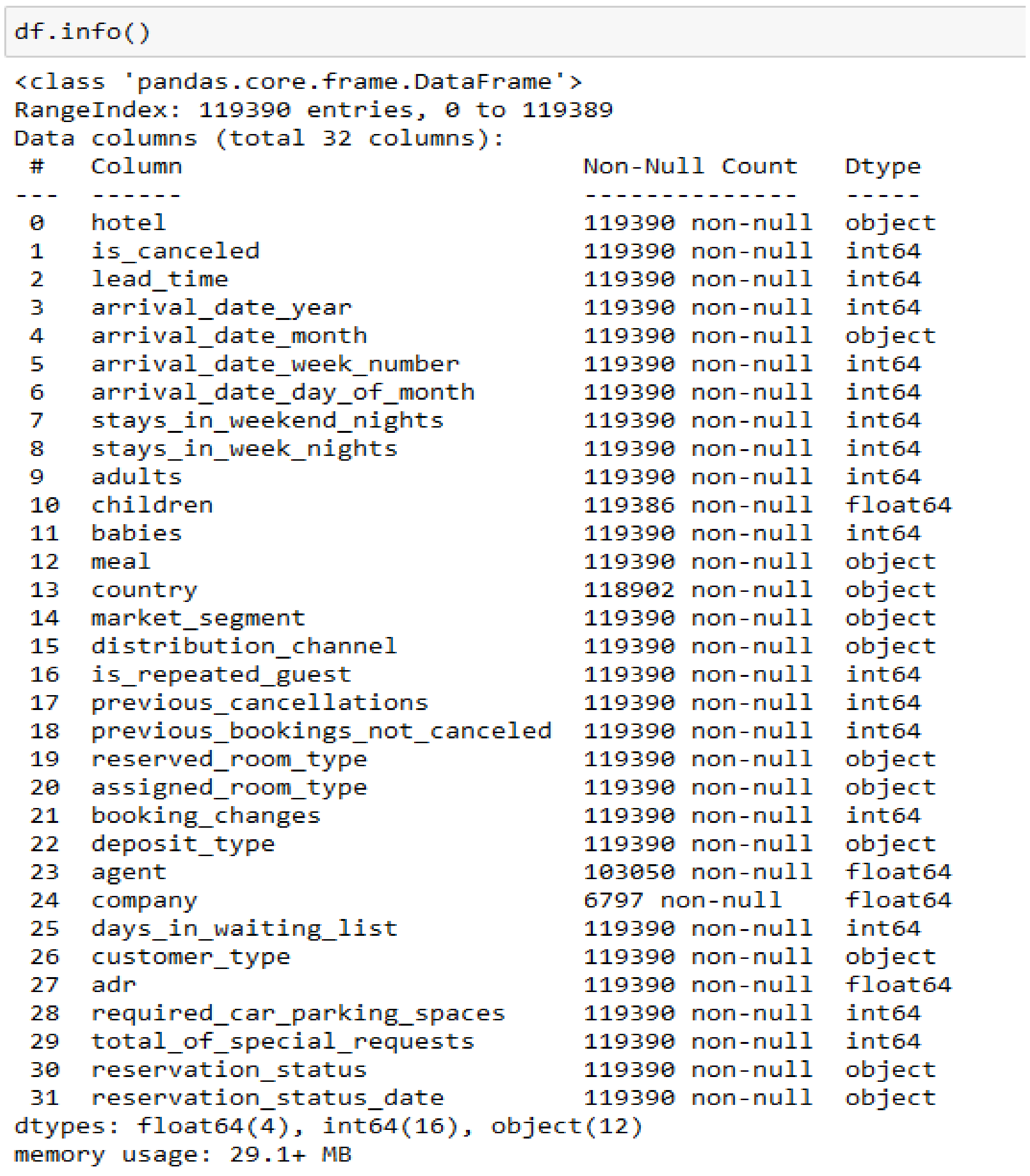
Predictive analytics is increasingly becoming a crucial tool in the hospitality industry, particularly for predicting hotel booking cancellations. It involves analyzing historical data and employing statistical algorithms to identify patterns and trends, which can then be used to forecast future events. This approach has proven to be highly effective in anticipating customer behavior, aiding hoteliers in managing bookings more efficiently.

Several predictive analytics methods have been employed to forecast hotel booking cancellations. Some of the most notable techniques include machine learning algorithms, big data analysis, and deep learning models. These methods have been explored and validated through various studies. For instance, researchers Sánchez, E. C., Sánchez-Medina, A. J., & Pellejero, M. (Sánchez-Medina & C-Sánchez 2020), utilized machine learning and big data to efficiently forecast cancellations in hotel bookings. In 2017, Antonio, N., De Almeida, A., & Nunes, L. (Antonio et al., 2017), focused on predicting hotel booking cancellations to minimize uncertainty and enhance revenue generation. Caicedo-Torres, W., & Payares, F. (Caicedo-Torres & Payares, 2016), in their 2016 research, examined a machine learning model for forecasting occupancy rates and demand in hotels. Another significant contribution was made in 2019 Ku, C. H., Chang, Y. C., Wang, Y., Chen, C. H., & Hsiao, S. H. (Ku et al., 2019), who used artificial intelligence and visual analytics in a deep-learning approach to analyze hotel reviews and responses. Additionally, in 2018, Al-Smadi, M., Qawasmeh, O., Al-Ayyoub, M., Jararweh, Y., & Gupta, B., compared a deep recurrent neural network with a support vector machine for aspect-based sentiment analysis of Arabic hotel reviews.

These methods can be applied to historical booking data, such as guest information, booking date, cancelation date, and room type to predict cancelation rates. By using these methods, hotels can anticipate changes in demand and take proactive steps to mitigate the impact of cancelations.

5 Data Analysis

In this chapter, we encompass the complete data analysis segment of the dissertation, a crucial part of our research journey. Our main goal is to concisely summarize and effectively present the gathered data, utilizing a range of visual aids such as charts, graphs, and tables for clarity and comprehensive understanding. This involves an in-depth examination of data patterns, trends, and relationships, which allows for the extraction of essential insights and the formulation of meaningful conclusions. The chapter initiates with a detailed overview of the data preparation and cleaning processes, ensuring data integrity and readiness for analysis. Subsequently, we employ various analytical techniques, from basic to advanced statistical methods, each carefully chosen to align with our specific research questions and objectives.

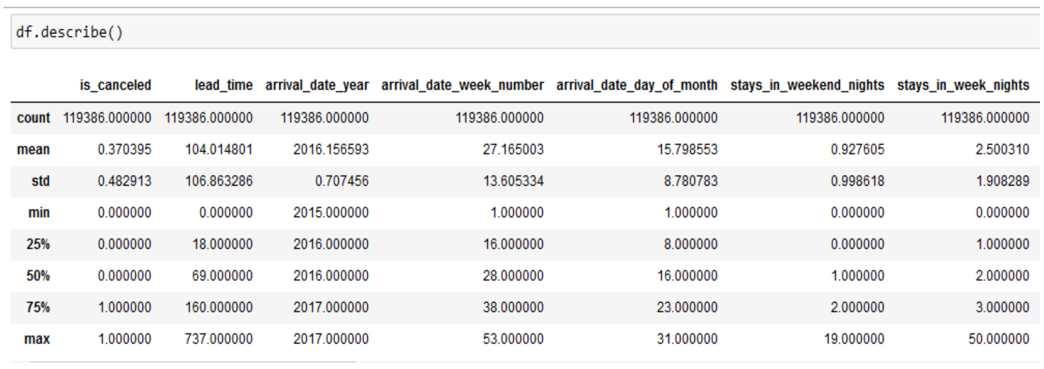


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119390 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119390 non-null  object
 1   is_canceled                     119390 non-null  int64
 2   lead_time                       119390 non-null  int64
 3   arrival_date_year               119390 non-null  int64
 4   arrival_date_month              119390 non-null  object
 5   arrival_date_week_number        119390 non-null  int64
 6   arrival_date_day_of_month       119390 non-null  int64
 7   stays_in_weekend_nights         119390 non-null  int64
 8   stays_in_week_nights            119390 non-null  int64
 9   adults                          119390 non-null  int64
 10  children                        119386 non-null  float64
 11  babies                          119390 non-null  int64
 12  meal                            119390 non-null  object
 13  country                         118902 non-null  object
 14  market_segment                  119390 non-null  object
 15  distribution_channel            119390 non-null  object
 16  is_repeated_guest               119390 non-null  int64
 17  previous_cancellations          119390 non-null  int64
 18  previous_bookings_not_canceled  119390 non-null  int64
 19  reserved_room_type              119390 non-null  object
 20  assigned_room_type              119390 non-null  object
 21  booking_changes                 119390 non-null  int64
 22  deposit_type                    119390 non-null  object
 23  agent                           103050 non-null  float64
 24  company                         6797 non-null    float64
 25  days_in_waiting_list            119390 non-null  int64
 26  customer_type                   119390 non-null  object
 27  adr                             119390 non-null  float64
 28  required_car_parking_spaces     119390 non-null  int64
 29  total_of_special_requests       119390 non-null  int64
 30  reservation_status              119390 non-null  object
 31  reservation_status_date         119390 non-null  object
dtypes: float64(4), int64(16), object(12)
memory usage: 29.1+ MB
```

Figure 2: Presentation of all the features

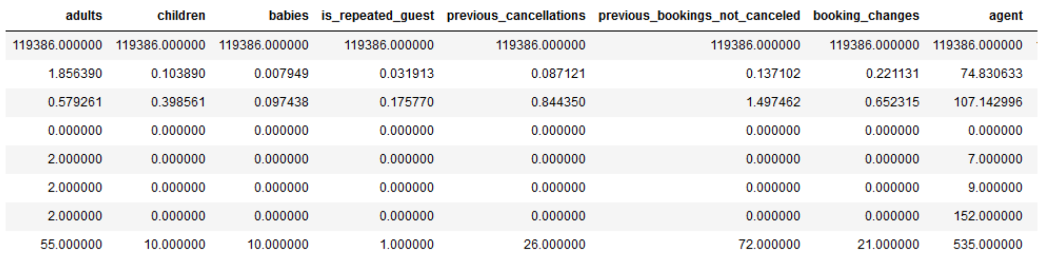
As can be seen, 'reservation_status_date' has an object data type, when it should has a date data type. Also, there are missing values in 'children', 'country', 'agent' and 'company' columns. This will be explored in the next section.



```
df.describe()
```

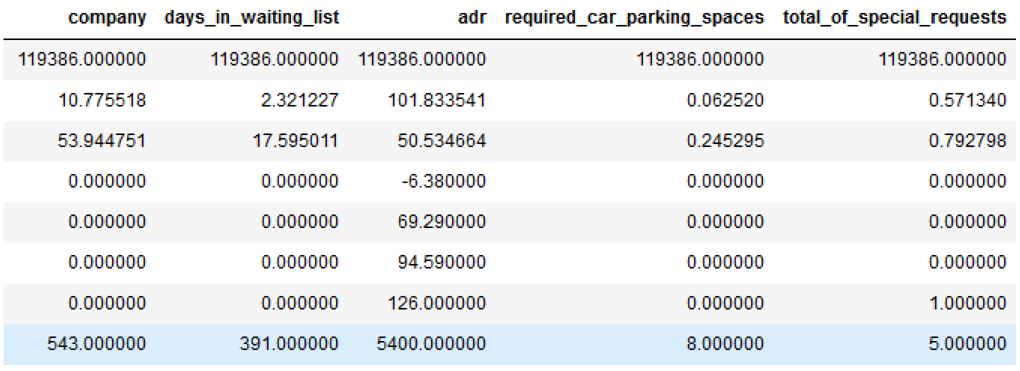
	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	stays_in_week_nights
count	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000
mean	0.370395	104.014801	2016.156593	27.165003	15.798553	0.927605	2.500310
std	0.482913	106.863286	0.707456	13.605334	8.780783	0.998618	1.908289
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.000000	0.000000
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.000000	1.000000
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.000000	2.000000
75%	1.000000	160.000000	2017.000000	38.000000	23.000000	2.000000	3.000000
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.000000	50.000000

Figure 3: Statistical description of numerical features / first part



adults	children	babies	is_repeated_guest	previous_cancellations	previous_bookings_not_canceled	booking_changes	agent
119386.000000	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000	119386.000000
1.856390	0.103890	0.007949	0.031913	0.087121	0.137102	0.221131	74.830633
0.579261	0.398561	0.097438	0.175770	0.844350	1.497462	0.652315	107.142996
0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	7.000000
2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	9.000000
2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	152.000000
55.000000	10.000000	10.000000	1.000000	26.000000	72.000000	21.000000	535.000000

Figure 4: Statistical description of numerical features / second part

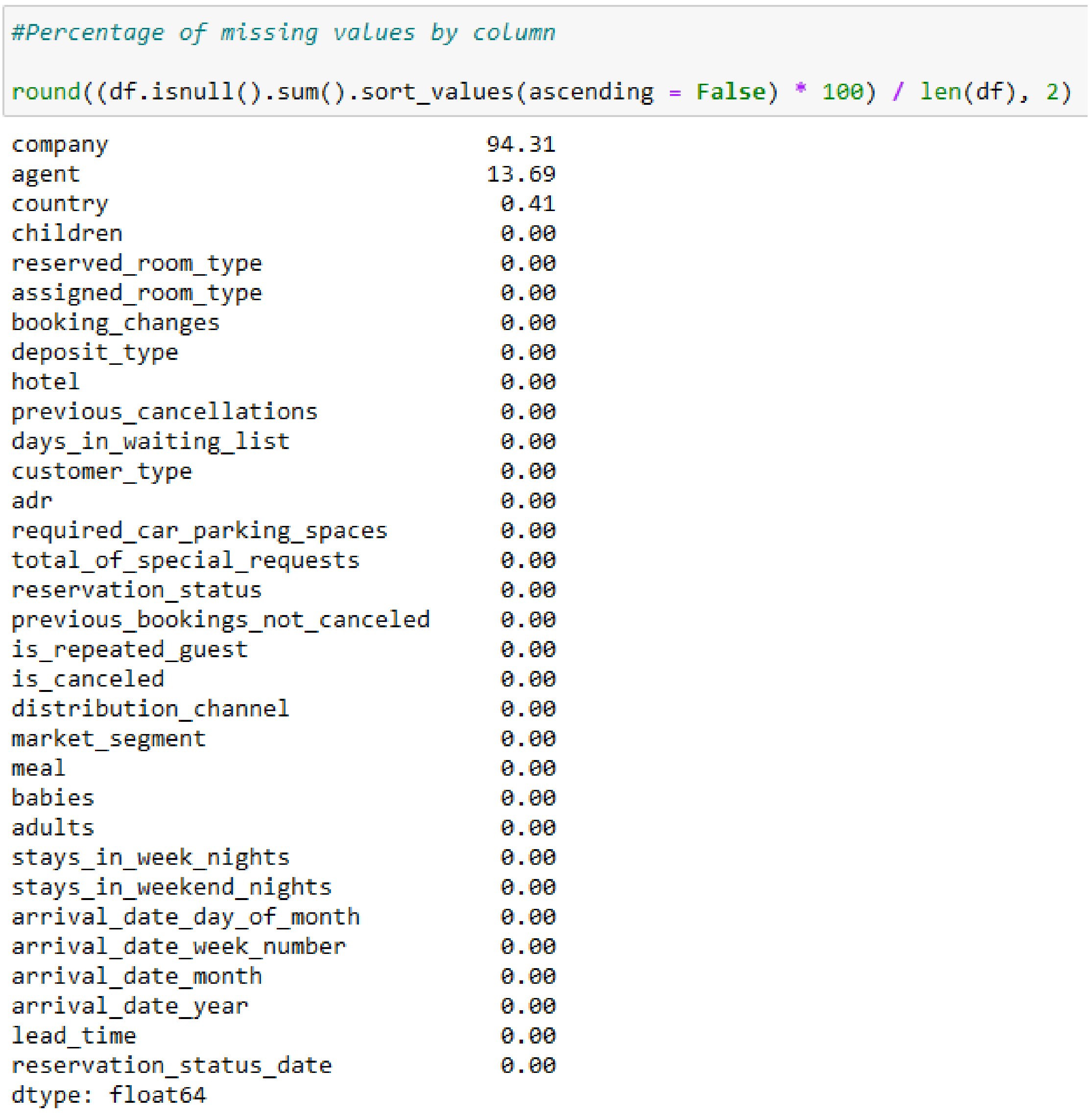


company	days_in_waiting_list	adr	required_car_parking_spaces	total_of_special_requests
119386.000000	119386.000000	119386.000000	119386.000000	119386.000000
10.775518	2.321227	101.833541	0.062520	0.571340
53.944751	17.595011	50.534664	0.245295	0.792798
0.000000	0.000000	-6.380000	0.000000	0.000000
0.000000	0.000000	69.290000	0.000000	0.000000
0.000000	0.000000	94.590000	0.000000	0.000000
0.000000	0.000000	126.000000	0.000000	1.000000
543.000000	391.000000	5400.000000	8.000000	5.000000

Figure 5: Statistical description of numerical features / third part

5.1 Handling missing values

In today's expansive big data environment, missing values remain a prevalent issue, continuously impairing the integrity and quality of the data collected. As we navigate through vast datasets, the absence of crucial data points creates significant challenges for analysts and data scientists alike. This persistent problem not only hinders the accurate analysis of data but also obstructs the extraction of valuable insights that drive informed decision-making. The presence of missing values often leads to skewed results and inaccurate models, thereby diminishing the reliability of data-driven applications and systems (Peng et al., 2023). Therefore, checking for and handling missing data before modeling is an important step.



```
#Percentage of missing values by column

round((df.isnull().sum().sort_values(ascending = False) * 100) / len(df), 2)
```

```
company                           94.31
agent                             13.69
country                            0.41
children                           0.00
reserved_room_type                 0.00
assigned_room_type                 0.00
booking_changes                    0.00
deposit_type                       0.00
hotel                              0.00
previous_cancellations             0.00
days_in_waiting_list               0.00
customer_type                      0.00
adr                                0.00
required_car_parking_spaces        0.00
total_of_special_requests          0.00
reservation_status                 0.00
previous_bookings_not_canceled     0.00
is_repeated_guest                  0.00
is_canceled                        0.00
distribution_channel               0.00
market_segment                     0.00
meal                               0.00
babies                             0.00
adults                             0.00
stays_in_week_nights               0.00
stays_in_weekend_nights            0.00
arrival_date_day_of_month          0.00
arrival_date_week_number           0.00
arrival_date_month                 0.00
arrival_date_year                  0.00
lead_time                          0.00
reservation_status_date            0.00
dtype: float64
```

Figure 6: Percentage of missing values by column

The columns labeled "Company" and "Agent" contain a significant amount of missing data, with 94.31% and 13.69% missing values, respectively. One solution could be to remove these columns entirely. However, the source material suggests otherwise. In some categorical variables like Agent or Company, “NULL” is presented as one of the categories. This should not be considered a missing value, but rather as “not applicable”. For example, if a booking “Agent” is defined as “NULL” it means that the booking did not came from a travel agent.

Another approach would be to replace the null values in these columns with the value of 0, since they are of the float64 data type, and the personal information has been removed. In contrast, the "Country" column has a relatively low number of missing values at 0.41% (affecting 488 rows). For this categorical variable, I have chosen to replace the null values with the mode, as it is a common technique for handling missing data in categorical variables. The "Children" column also has some missing values, but they represent such a small fraction of the dataset (less than 0.01%) that it would be appropriate to delete the affected rows (4 rows) rather than replacing the missing values with any estimated value.

5.2 Detecting and Treating Outliers

Outliers are data points that are significantly different from the other observations in a dataset. They can be caused by measurement errors, data entry errors, or legitimate phenomena. Detecting and treating outliers is important because they can have a significant impact on the analysis and interpretation of the data.

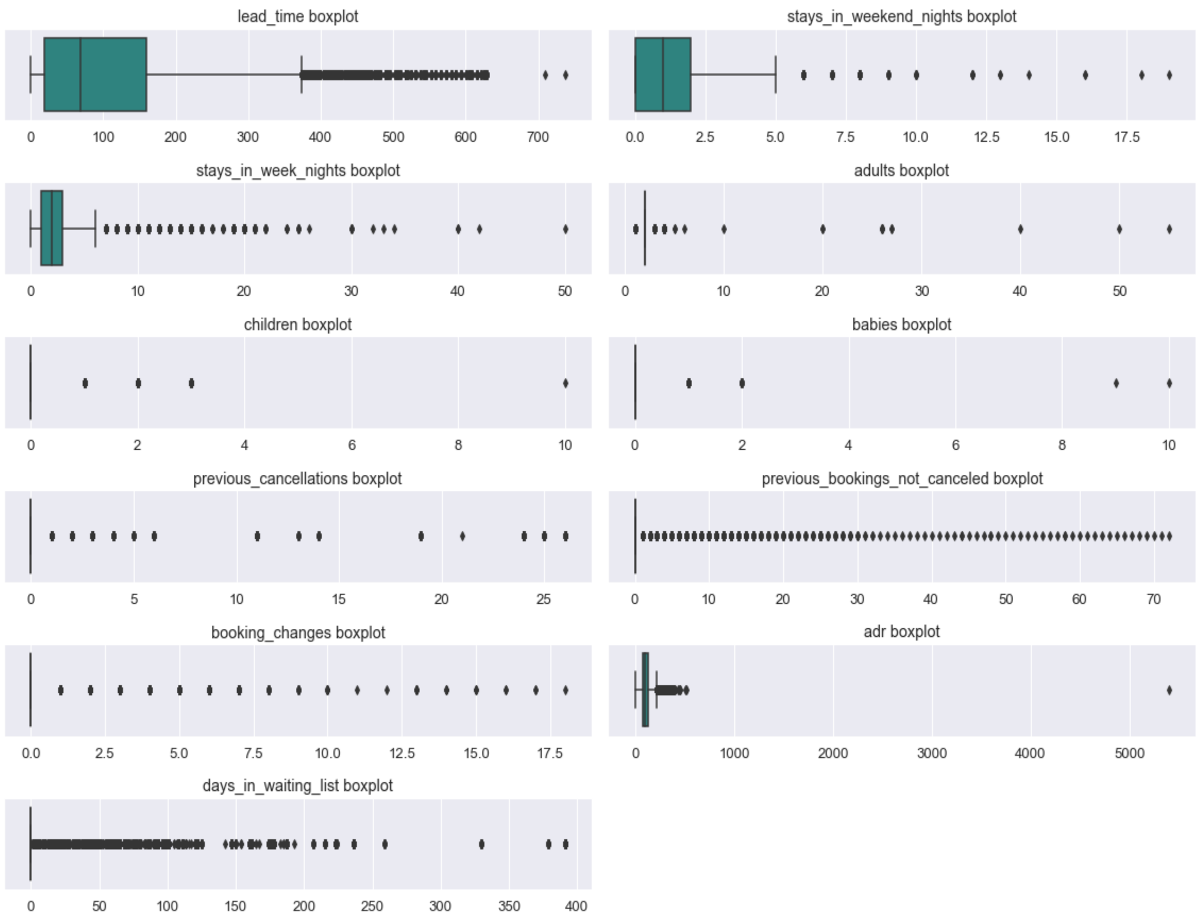


Figure 7: Boxplots Representing Distributions of Key Numerical Variables

Outliers that are significantly different from the rest of the observations in a dataset, can have a significant impact on the analysis and interpretation of the data. It is important to pay attention to both the quantity of outliers present and the degree to which they deviate from the other data points. For example, in the ‘adr’ column τhis outlier has an approximate value of 5400 and is far away from the other data points in the same column. This difference could indicate an error in data entry, meaning that it might not be a real value but rather a mistake made during the data entry process.

There are outliers with values that are like each other. This may happen because the dataset contains data from two different types of hotels: city hotels and resorts. City hotels are typically less expensive and can attract a wider range of customers, such as businessmen and airline workers, due to their central location. In contrast, resorts are typically more expensive and offer more amenities such as larger rooms or villas, which may account for outliers in the data with a high number of adults. It is not possible to determine the specific identity of the hotels in the dataset, which would be necessary to fully understand the characteristics and business model of each hotel.

5.3 Exploratory Data analysis

Exploratory Data Analysis (EDA) is an essential step in data science that involves scrutinizing data sets to summarize their main characteristics, often through visual methods like plots, to uncover trends and patterns. It focuses on understanding data distribution, and employs tools like histograms, box plots, and scatter plots to reveal the data's underlying structure.

The first step in the analysis is to create a visual representation of the data to gain a better understanding of any differences in the distribution of data for each hotel. This visual will help to identify any patterns or outliers in the data and thus, it will be a useful tool to understand the data better.

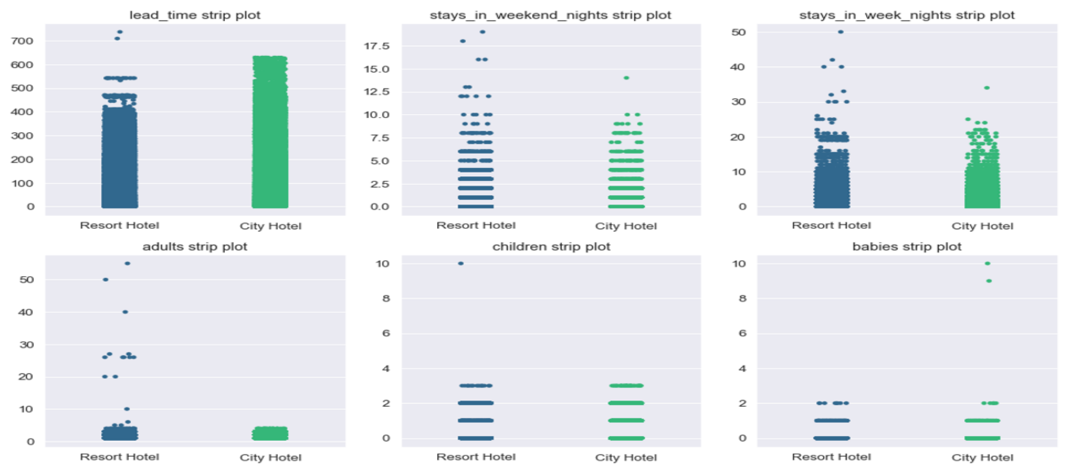


Figure 8: Distribution Analysis of Variables Between Resort and City Hotels / first part

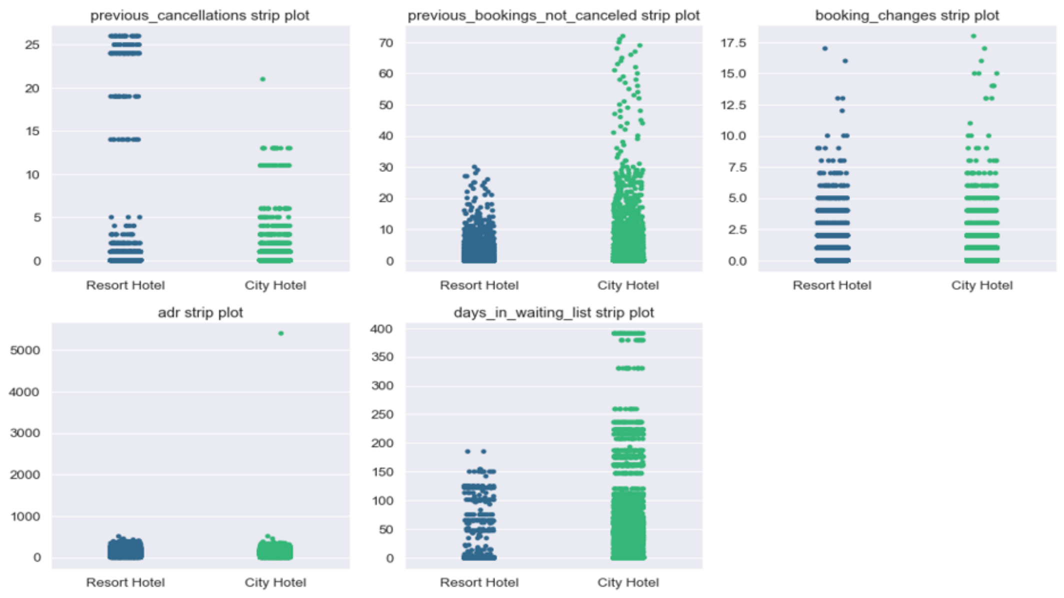


Figure 8: Distribution Analysis of Variables Between Resort and City Hotels / second part

There is valuable information or insights that can be obtained from the above visual.

1. In the **lead_time** column the values begin to spread out when the value reaches around 400. Resort Hotels show a broader range of lead times, including many bookings made well in advance.
2. In the **stays_in_weekend_nights** and **stays_in_week_nights** columns, there are more outliers present in the resort hotel data than in the city hotel data. However, both types of hotels display an increase in the spread of the data at 6 and 10 days, respectively, for each column.
3. In the **adults** column, the distribution of the city hotel data in this column is not as dispersed. This suggests that the resort hotel may have larger rooms or villas that can accommodate more adults, which would explain the presence of outliers. However, it is not possible to know the exact type of rooms offered by the hotel. The only thing that can be stated is that the data for both hotels is mostly concentrated between 1 and 4 adults per reservation, and any values beyond that are unusual.
4. In the columns for **children** and **babies**, there are 3 entries displaying unusually high figures, which may indicate a mistake in the inputting of the reservation information.
5. The distribution of the **booking_changes** variable is similar for both hotels, and it is highly unusual for a booking to require more than 5 modifications.

6. The **adr** (Average Daily Rate) variable for the city hotel has an unusually high value that appears to be an outlier. It would be advisable to remove this data point as it may be an error
7. The **days_in_waiting_list** column refers to the number of days that elapsed from the time the reservation was made until it was confirmed to the client. According to the strip plot, the data is very sparse, being unusual for a reservation to have a waiting period of more than 1 day, practically.
8. The **previous_cancellations** and **previous_bookings_not_canceled** columns show the number of reservations made by the customer prior to the current one that were either canceled or not. As per the graph, these values reach very high figures, such as in the case of the resort hotel where there are records of customers having made more than 20 previous cancellations before the current booking.

In addition, we explore a series of graphs that clarify various metrics and trends to travel and hospitality preferences.

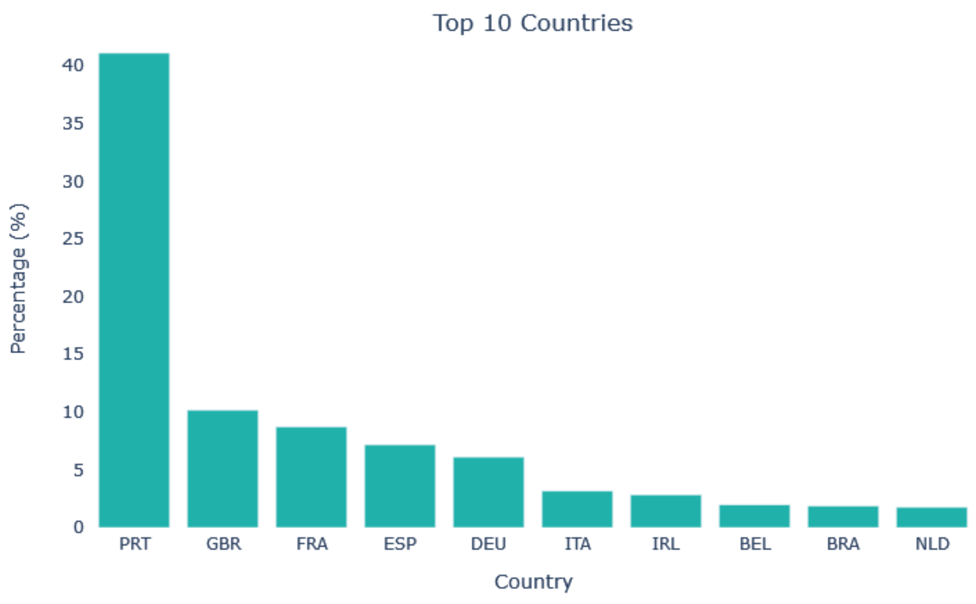


Figure 9: Top 10 Origin Countries of Customers

Figure 9 analyses the distribution of visitor requests by the percentage of bookings coming from different countries. Among the top 10 countries, Portugal (PRT) has the highest percentage, accounting for 41 % of visitor requests. This suggests a significant presence of visitors from Portugal. these percentages provide valuable information on the geographic representation of visitors and can help inform business decisions, such as marketing strategies or tailored services, to meet the specific needs and preferences of visitors from these countries.

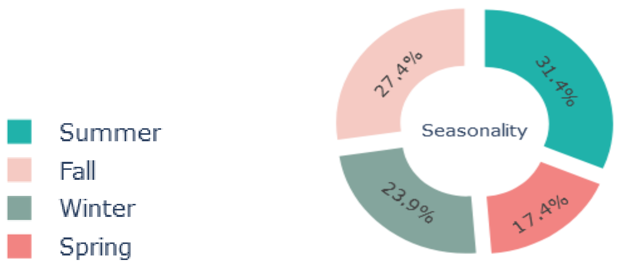


Figure 10: Distribution of Bookings by Seasonality

Figure 10 represents the distribution of "Seasonality" across four categories: Summer, Spring, Fall, and Winter. The "Percentage" illustrates the proportion each season represents in the overall dataset.

Summer holds the most substantial portion, making up about 31.4% of the total, suggesting it to be the most popular season for the dataset in question. Following Summer, Spring holds a significant fraction, with an estimated 27.4% representation, indicating a relatively high preference. Fall, while not as popular as the preceding seasons, accounts for a notable 23.9% of the dataset. Finally, Winter is the least represented season, comprising roughly 17.4% of the total data, pointing to a lesser preference or participation during this season.

The data is unequally distributed across the four seasons, with a greater emphasis on the Summer and Spring seasons. Generally, people tend to travel more during these seasons due to the favorable weather conditions, school holidays, and general inclination to engage in outdoor activities. These periods are often considered peak tourism season. Hence, city and resort hotels may experience higher bookings and occupancy rates during these times.

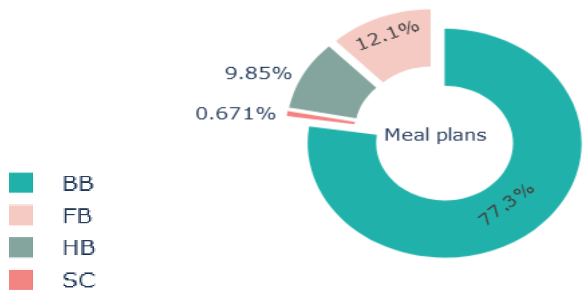


Figure 11: Percentage Distribution of Meal Plan Preferences

Figure 11 represents the distribution of different meal plans chosen by guests at a hotel. The meal plans are categorized into four types: Bed & Breakfast (BB), Half Board (HB), Self-Catering (SC), and Full Board (FB). Each meal plan type is accompanied by a corresponding percentage, indicating the proportion of guests that have selected each type of meal plan. Bed & Breakfast (BB) meal plan is the preferred option for a significant majority of guests, roughly 77.34%. This plan, which includes breakfast, likely offers the dual benefits of convenience and the freedom to explore local cuisine for the rest of the day's meals. The Half Board (HB) meal plan, offering two meals, is the second most favored, chosen by around 12.15% of guests. This selection might appeal to guests who appreciate a balance of assured meals and the flexibility to explore outside dining for at least one meal a day. The Self Catering (SC) option, without any included meals, is preferred by 9.85% of guests, suggesting a desire for flexibility or budget considerations. The Full Board (FB) plan, providing all three meals, is the least favored at only 0.67%, hinting at a general guest preference for meal flexibility and potential cost implications.

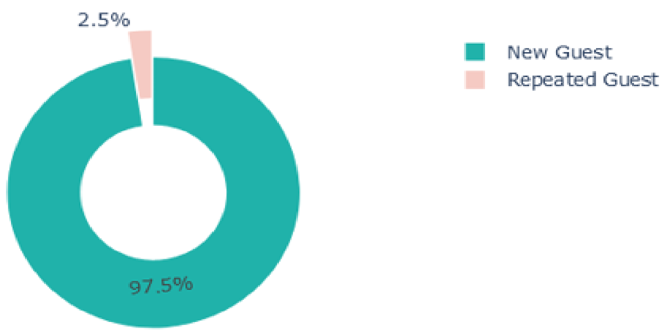


Figure 12: Proportion of New Versus Repeated Guests – City Hotel

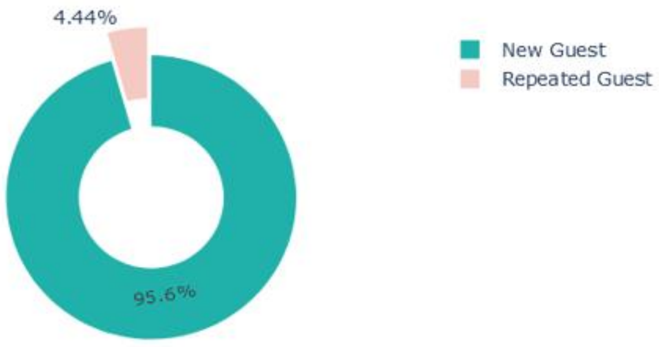


Figure 13: Proportion of New Versus Repeated Guests – Resort Hotel

Figures 12 and 13 provide insights into the proportion of new versus repeated guests at two types of hotels: City Hotel and Resort Hotel. For the City Hotel, an overwhelming majority of approximately 97.5% are new guests, with a minimal 2.5% being repeated guests. This suggests a high turnover of guests at this type of hotel, potentially indicating its appeal to business travelers or short-term visitors. On the other hand, Resort Hotel also has a high proportion of new guests at about 95.56%, but it exhibits a slightly higher percentage of repeated guests at around 4.44%. The slightly higher repeat rate may be indicative of the resort's ability to offer an experience that entices guests to return, possibly due to unique amenities, a leisurely environment, or exceptional customer service.

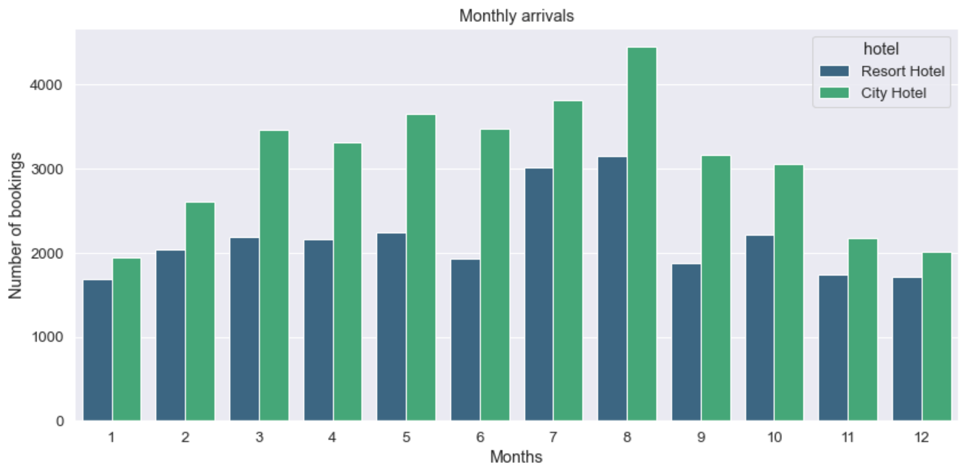


Figure 14: Monthly Guest Arrivals Comparison Between Resort and City Hotels

Figure 14 provides an understanding of the month of guest arrivals at both a City Hotel and a Resort Hotel. In Resort Hotel, the peak of guest arrivals seems to occur during the warmer months, notably in August and May, which likely corresponds to common vacation periods. In contrast, the City Hotel experiences fewer guest arrivals in the colder months of January and December. A similar trend is observed for the Resort Hotel, with the most arrivals occurring during the summer months of July and August, and the least in the colder months of January and November

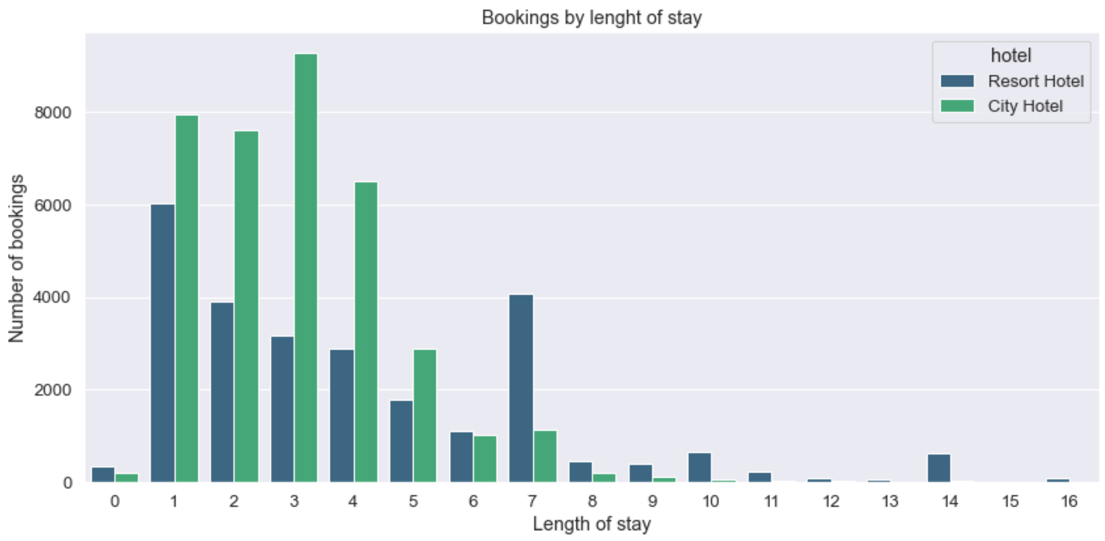


Figure 15: Length of Stay Distribution for City and Resort Hotel Bookings

In Figure 15 we can derive insights regarding the length of stay for guests at both City Hotels and Resort Hotels. Firstly, it's apparent that 1–4-night stays are far more popular at City Hotels, likely due to their urban locations and convenience for business travel or short city breaks. The numbers significantly drop as the length of stay increases, suggesting that City Hotels are often used for brief visits. Resort Hotels, on the other hand, see a relatively even distribution of stays from 1 to 7 nights, indicating they're popular for both short and week-long vacations. Interestingly, there's a substantial spike in seven-night stays, which corresponds to the typical one-week vacation duration. Overall, this analysis suggests that City Hotels are frequently used for shorter stays, whereas Resort Hotels are more versatile, catering to a variety of vacation lengths, but particularly popular for week-long getaways.

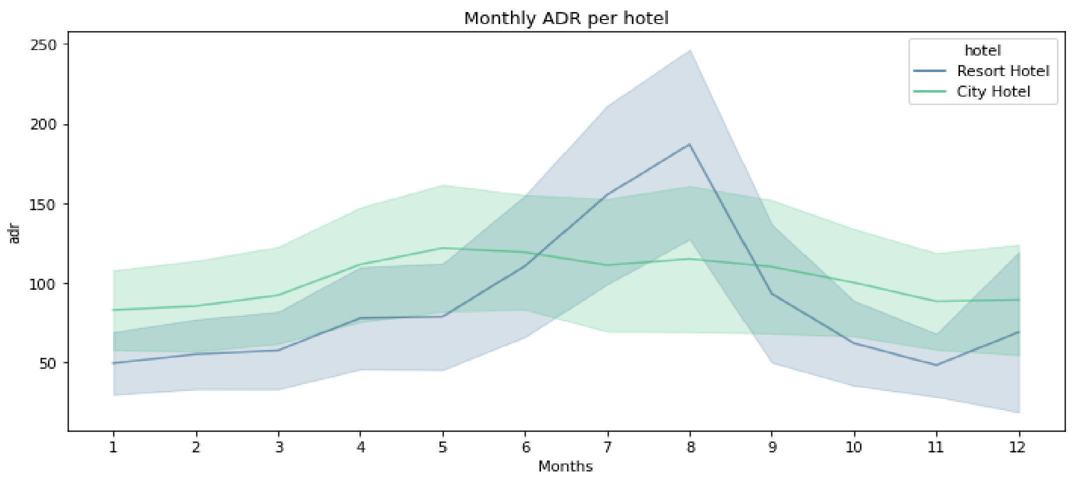


Figure 16: Monthly Trends in Average Daily Rate (ADR) for Resort and City Hotel

Figure 16 illustrates the fluctuations in the Average Daily Rate (ADR), a key metric in hospitality representing average room revenue per day, for both the City Hotel and the Resort Hotel throughout the year. For the City Hotel, the ADR reaches its peak in May and June, possibly reflecting a high-demand period, then slightly dips in July and August, and further reduces during autumn and winter. On the other hand, the Resort Hotel observes its highest ADR in August, likely due to an influx of vacationers in the summer. Interestingly, while the summer rates for the Resort Hotel substantially surpass those of the City Hotel, the opposite is true for the rest of the year, potentially due to the City Hotel's appeal to business travelers who maintain a consistent travel pattern throughout the year, contrasting with the seasonal leisure travelers attracted by the Resort Hotel.

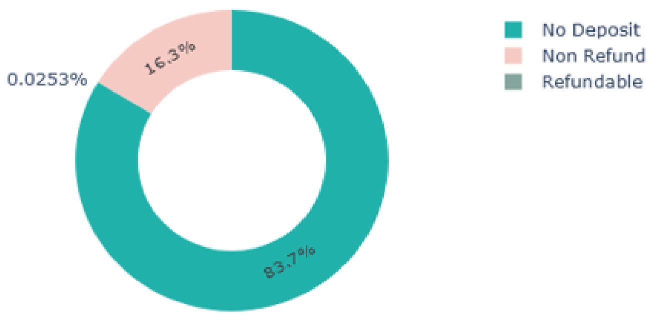


Figure 17: Distribution of Booking Deposit Types – City Hotel

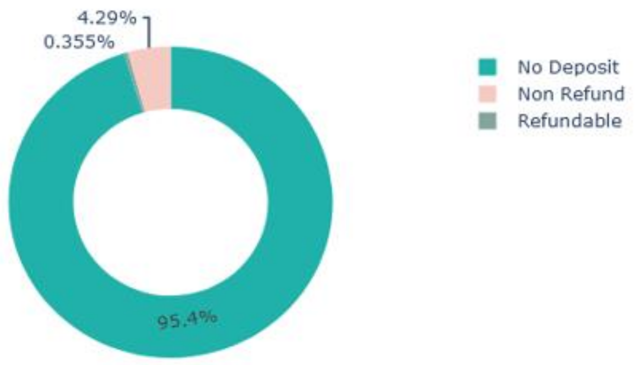


Figure 18: Distribution of Booking Deposit Types – Resort Hotel

The deposit types for bookings reveal a prevalent trend of flexibility in the hotel industry. City Hotels show 83.67% of bookings made without a deposit, likely to accommodate business or short-stay guests who might require frequent changes in their plans. Only 16.30% of bookings in these hotels are made with a non-refundable deposit, which is indicative of guests with confirmed travel plans. Meanwhile, bookings made with a refundable deposit are almost negligible. Resort Hotels show an even higher preference for bookings without a deposit, at 95.35%. This could be in response to the often-changing nature of vacation plans and might serve as a strategy to attract more guests by offering an easy booking policy. The proportion of non-refundable bookings in Resort Hotels is significantly lower at around 4.29%, while bookings made with a refundable deposit remain a minimal fraction, though slightly higher than in City Hotels.

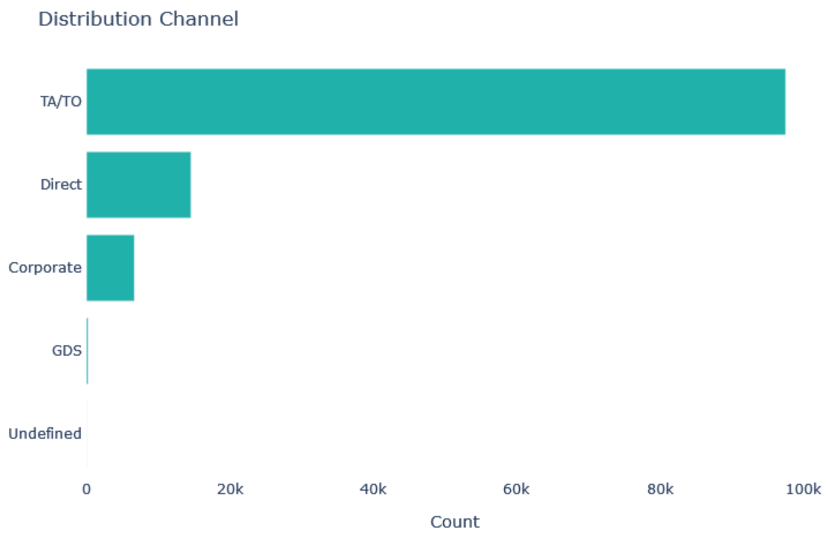


Figure 19: Customer Bookings by Distribution Channel

Figure 19 provides an overview of the booking distribution channels used by customers. Most bookings come through Travel Agents and Tour Operators (TA/TO), suggesting the hotel's strong partnerships with these entities due to their extensive reach and effective marketing strategies. The second largest category is Direct bookings, made without any intermediaries, typically via the hotel's own website, phone, or walk-ins, indicating that a significant portion of guests prefer direct interaction for a more personalized service or potentially better rates. The Corporate bookings category showcases the hotel's engagement with the corporate sector, likely involving reservations for employees travelling for work. The inclusion of the Global Distribution System (GDS), a digital network facilitating automated transactions between vendors and booking agents, reflects the hotel's adaptation to automated reservation processes. Lastly, the presence of an Undefined category implies that there are some bookings for which the distribution channel was not clearly recorded or categorized, potentially due to data entry errors or uncommon booking methods.

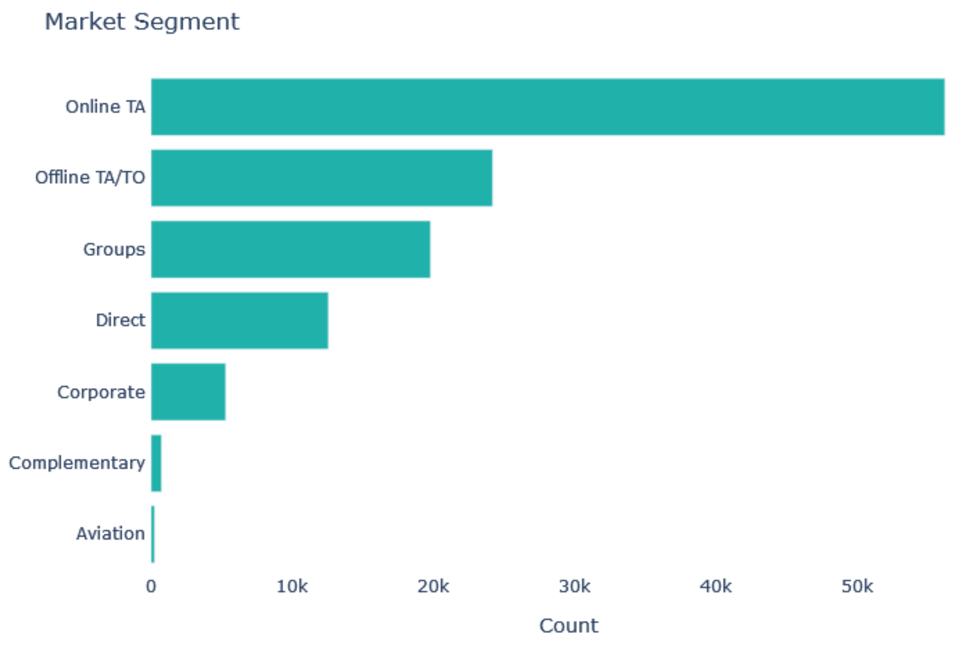


Figure 20: Bookings Distribution Across Various Market Segments

Figure 20 displays the distribution of bookings across different market segments. The "Market Segment" column represents various categories, including Aviation, Complementary, Corporate, Direct, Groups, Offline TA/TO (Travel Agents/Tour Operators), and Online TA (Travel Agents). These segments reflect the different channels through which bookings are made. It is notable that the largest segment in terms of count is Online TA, indicating a significant number of bookings are made through online travel agencies. Other prominent segments include Offline TA/TO and Groups, suggesting the importance of traditional travel agents and group bookings. Corporate and Direct segments also have a notable presence. This distribution highlights the diverse sources and channels through which bookings are generated, providing insights into the market dynamics and booking patterns.

5.4 Predictive modeling

5.4.1 Machine Learning Model Comparison

In order to construct proficient machine learning models aimed at forecasting hotel booking cancellations, it is critical to consider the specific attributes inherent to each hotel type. As such, the dataset shall be divided into two subsets, each representing a different hotel type: resort and city. Through the segregation of the dataset, it will be possible to optimize the models to comprehend and represent the distinctive cancellation behaviors and patterns associated with each hotel type. Furthermore, assessing the performance of the models on each subset will provide an opportunity to contrast the cancellation rates between resort and city hotels, thereby determining the existence of any significant discrepancies. This approach probably will engender the creation of more precise and informative machine learning models, thereby enhancing the accuracy of forecasts relating to hotel booking cancellations.

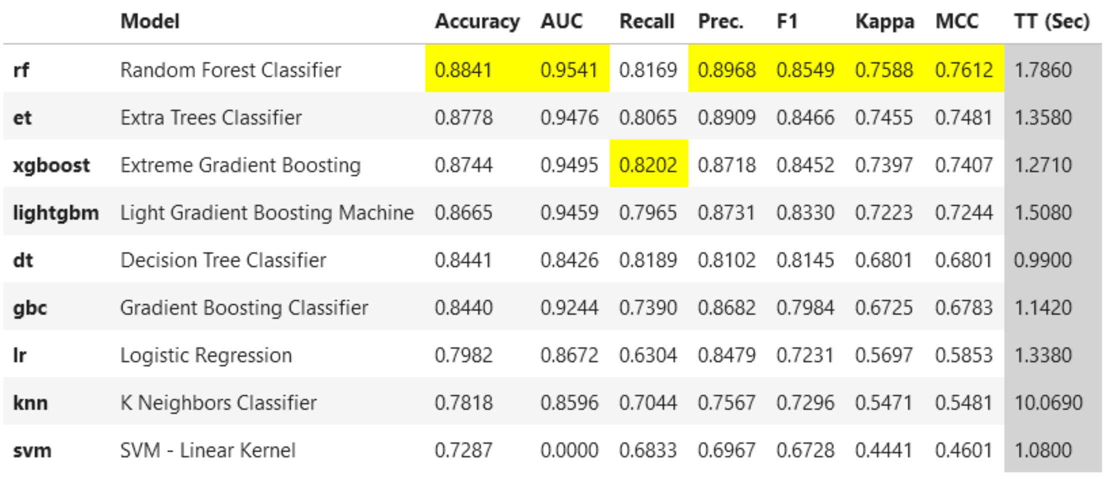
In this part we will use the “*compare_models”* function from Pycaret. Is a powerful tool for comparing the performance of different machine learning models. This function trains and evaluates the performance of all the estimators available in the model library using cross-validation": The “compare_models” function takes a set of machine learning algorithms (estimators) from PyCaret's model library and trains them on your dataset. It then evaluates the performance of each model using cross-validation, which involves splitting the data into multiple subsets, training the model on a subset, and evaluating its performance on the remaining subset. This helps to provide a more robust assessment of how well the models will perform on unseen data. The output of this function is a scoring grid with average cross-validated scores": After training and evaluating the models, “compare_models” generates a scoring grid that displays the performance metrics (scores) for each model.

The grid includes the following metrics:

1. Accuracy: Accuracy represents the proportion of correctly classified instances out of the total number of instances. It provides an overall measure of how well the model predicts the target variable.
2. AUC: AUC (Area Under the Curve) refers to the area under the Receiver Operating Characteristic (ROC) curve. It provides a measure of the model's ability to distinguish between positive and negative instances, with higher values indicating better performance.
3. Recall: Recall, also known as sensitivity or true positive rate, calculates the proportion of actual positive instances correctly identified by the model. It is especially useful when the goal is to minimize false negatives.
4. Precision: Precision quantifies the proportion of predicted positive instances that are actually true positive instances. It focuses on minimizing false positives and is useful when the cost of false positives is high.
5. F1 score: The F1 score is the harmonic mean of precision and recall. It combines the two metrics into a single value that represents the balance between precision and recall.
6. Kappa: Kappa is a statistic that measures the agreement between the predicted and actual classes, taking into account the possibility of agreement occurring by chance alone. It provides a measure of the model's performance beyond random chance.
7. MCC (Matthews Correlation Coefficient): MCC takes into account true and false positives and negatives and provides a measure of the quality of the binary classification predictions. It ranges from -1 to +1, where +1 indicates perfect predictions, 0 indicates random predictions, and -1 indicates complete disagreement between predictions and observations.
8. TT (Sec): TT stands for Time Taken and represents the time in seconds it took to compare the models.

   These scores are calculated based on the cross-validated evaluation results, providing an overview of how well each model performs on average. In our case will use it specifically for algorithms including Logistic Regression, Decision Tree Classifier, Random Forest Classifier, XgBoost Classifier, Decision Tree, Extra Trees Classifier, KNeighborsClassifier and LGBMClassifier, svm.

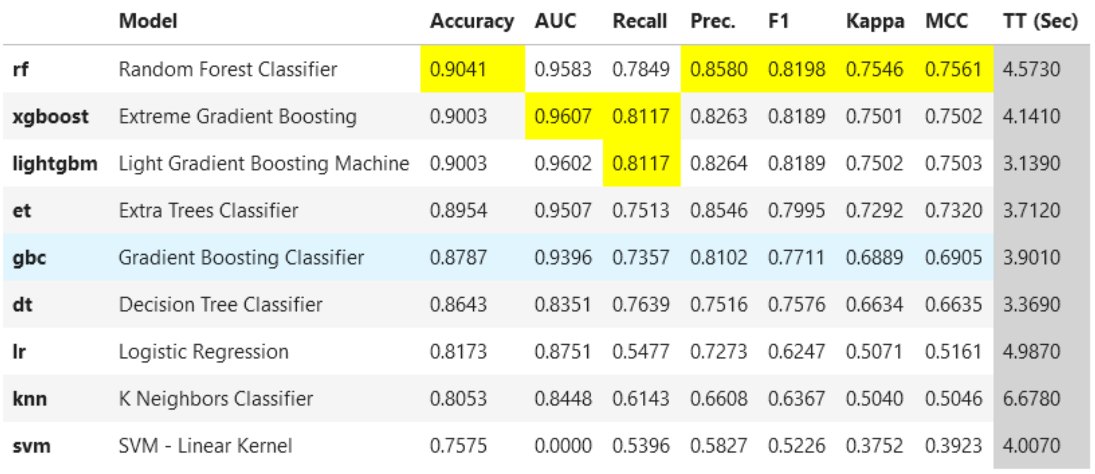
City Hotel results



	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
rf	Random Forest Classifier	0.8841	0.9541	0.8169	0.8968	0.8549	0.7588	0.7612	1.7860
et	Extra Trees Classifier	0.8778	0.9476	0.8065	0.8909	0.8466	0.7455	0.7481	1.3580
xgboost	Extreme Gradient Boosting	0.8744	0.9495	0.8202	0.8718	0.8452	0.7397	0.7407	1.2710
lightgbm	Light Gradient Boosting Machine	0.8665	0.9459	0.7965	0.8731	0.8330	0.7223	0.7244	1.5080
dt	Decision Tree Classifier	0.8441	0.8426	0.8189	0.8102	0.8145	0.6801	0.6801	0.9900
gbc	Gradient Boosting Classifier	0.8440	0.9244	0.7390	0.8682	0.7984	0.6725	0.6783	1.1420
lr	Logistic Regression	0.7982	0.8672	0.6304	0.8479	0.7231	0.5697	0.5853	1.3380
knn	K Neighbors Classifier	0.7818	0.8596	0.7044	0.7567	0.7296	0.5471	0.5481	10.0690
svm	SVM - Linear Kernel	0.7287	0.0000	0.6833	0.6967	0.6728	0.4441	0.4601	1.0800

Figure 20: Performance Metrics Comparison of Machine Learning Models – City Hotel

Resort Hotel results



	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
rf	Random Forest Classifier	0.9041	0.9583	0.7849	0.8580	0.8198	0.7546	0.7561	4.5730
xgboost	Extreme Gradient Boosting	0.9003	0.9607	0.8117	0.8263	0.8189	0.7501	0.7502	4.1410
lightgbm	Light Gradient Boosting Machine	0.9003	0.9602	0.8117	0.8264	0.8189	0.7502	0.7503	3.1390
et	Extra Trees Classifier	0.8954	0.9507	0.7513	0.8546	0.7995	0.7292	0.7320	3.7120
gbc	Gradient Boosting Classifier	0.8787	0.9396	0.7357	0.8102	0.7711	0.6889	0.6905	3.9010
dt	Decision Tree Classifier	0.8643	0.8351	0.7639	0.7516	0.7576	0.6634	0.6635	3.3690
lr	Logistic Regression	0.8173	0.8751	0.5477	0.7273	0.6247	0.5071	0.5161	4.9870
knn	K Neighbors Classifier	0.8053	0.8448	0.6143	0.6608	0.6367	0.5040	0.5046	6.6780
svm	SVM - Linear Kernel	0.7575	0.0000	0.5396	0.5827	0.5226	0.3752	0.3923	4.0070

Figure 21: Performance Metrics Comparison of Machine Learning Models – Resort Hotel

In analyzing machine learning model performance on the City and Resort Hotel datasets, a comparative approach reveals that certain patterns are consistent, while others vary based on the hotel type.

The Random Forest Classifier is the most effective model across both hotel types, displaying superior performance in terms of accuracy, AUC, and several other metrics. Conversely, the SVM with a Linear Kernel is least effective for both types, showing an AUC of 0.0, indicative of its limited ability to distinguish between classes.

The performance of other models such as Extreme Gradient Boosting, Light Gradient Boosting Machine, Extra Trees Classifier, and Gradient Boosting Classifier is generally good for both hotel types. However, models like the Decision Tree, Logistic Regression, K Neighbors Classifier, and SVM tend to underperform.

In terms of computational efficiency, the Light Gradient Boosting Machine and Decision Tree models are more expedient, while the K Neighbors Classifier and Extreme Gradient Boosting are more time-consuming across both datasets.

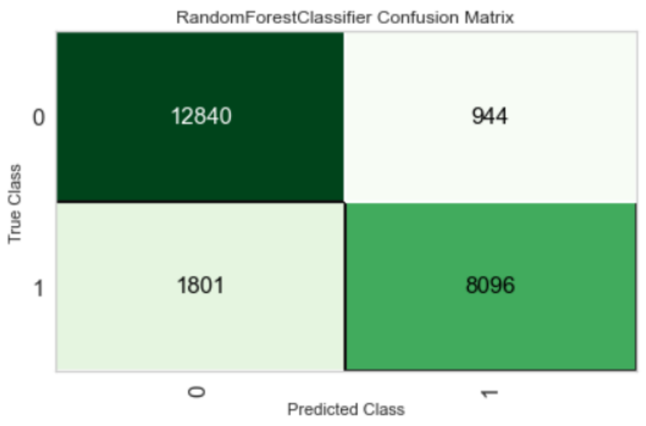
When comparing model performance between the two hotel types, some notable variances emerge. While the accuracy of all models is generally higher for the Resort Hotel, the precision metric, which reflects the correctness of positive predictions, is higher for the City Hotel. The F1 score, representing a balance between precision and recall, typically fares better for the City Hotel data as well, except in the case of the Extreme Gradient Boosting and Light Gradient Boosting Machine models.

Interestingly, the recall of certain models, like the Random Forest Classifier and Extreme Gradient Boosting, is higher for the Resort Hotel data. This indicates a superior ability of these models to correctly identify positive cases in the Resort Hotel data. Furthermore, some models, including the Random Forest Classifier and Extreme Gradient Boosting, take more time to process the City Hotel data compared to the Resort Hotel data.

In summary, the consistently high performance of tree-based algorithms across both datasets and metrics, they emerge as a strong choice. The Random Forest Classifier, with its balanced performance across metrics, could be considered as the best candidate model for both types of hotels.

5.4.2 Model performance

City Hotel



l

Figure 22: Random Forest Classifier Confusion Matrix – City Hotel

The values in the matrix represent:

- True Negatives (TN): The model correctly predicted class 0: 12.840 cases
- False Positives (FP): The model incorrectly predicted class 1 (while it was actually class 0): 944 cases.
- False Negatives (FN): The model incorrectly predicted class 0 (while it was actually class 1): 1.801 cases.
- True Positives (TP): The model correctly predicted class 1: 8.096 cases

  With these values, we can derive several important metrics:

1. **Accuracy:** (TP + TN) / (TP + FP + FN + TN) -> (8096 + 12840) / (8096 + 944 + 1801 + 12840) = 0.889 or 88.9%
2. **Precision (for class 1):** TP / (TP + FP) -> 8096 / (8096 + 944) = 0.895 or 89.5%
3. **Recall or Sensitivity (for class 1):** TP / (TP + FN) -> 8096 / (8096 + 1801) = 0.818 or 81.8%
4. **Specificity (for class 0):** TN / (TN + FP) -> 12840 / (12840 + 944) = 0.931 or 93.1%

The high accuracy of 88.9% suggests the model has performed quite well overall. The precision for class 1 (89.5%) indicates that the model is quite reliable when it predicts class 1; it's correct 89.5% of the time. The recall for class 1 (81.8%) is also high, meaning the model is able to find 81.8% of all actual class 1 cases. The model's specificity (93.1%) shows that it is also strong at predicting class 0; it correctly identifies 93.1% of all actual class 0 cases.

Resort Hotel

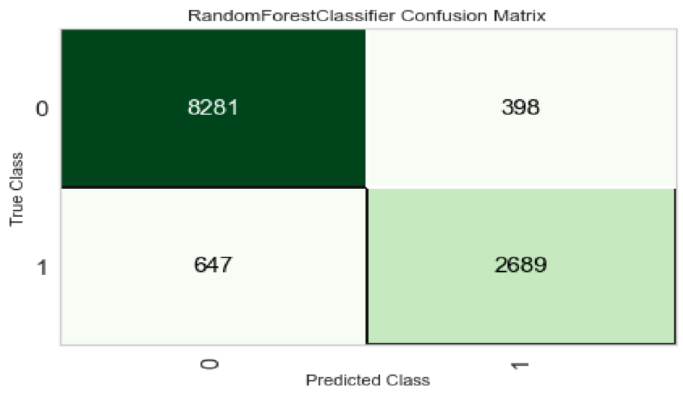


Figure 23: Random Forest Classifier Confusion Matrix – Resort Hotel

The values in the matrix represent:

- True Negatives (TN): The model correctly predicted class 0: 8281 cases
- False Positives (FP): The model incorrectly predicted class 1 (while it was actually class 0): 398 cases.
- False Negatives (FN): The model incorrectly predicted class 0 (while it was actually class 1): 647 cases.
- True Positives (TP): The model correctly predicted class 1: 2689 cases

With these values, we can derive several important metrics:

1. **Accuracy:** (TP + TN) / (TP + FP + FN + TN) -> (2689 + 8281) / (2689 + 398 + 647 + 8281) = 0.906 or 90.6%
2. **Precision (for class 1):** TP / (TP + FP) -> 2689 / (2689 + 398) = 0.871 or 87.1%
3. **Recall or Sensitivity (for class 1):** TP / (TP + FN) -> 2689 / (2689 + 647) = 0.806 or 80.6%
4. **Specificity (for class 0):** TN / (TN + FP) -> 8281 / (8281 + 398) = 0.954 or 95.4%

This model has an accuracy of 90.6%, suggesting strong overall performance. The precision for class 1 (87.1%) indicates that when the model predicts class 1, it's correct 87.1% of the time. The recall for class 1 (80.6%) is also relatively high, meaning the model is able to correctly identify 80.6% of all actual class 1 cases. The model's specificity (95.4%) shows that it is very good at predicting class 0; it correctly identifies 95.4% of all actual class 0 cases.

In conclusion, the Random Forest Classifier has demonstrated strong performance metrics for both the City and Resort Hotels. For the City Hotel, the model achieved an accuracy of 88.9%, with high precision and recall rates for class 1 predictions at 89.5% and 81.8% respectively, and an impressive specificity of 93.1% for class 0. This suggests that the model is not only reliable in its predictions but also effective in distinguishing between the classes, particularly in identifying true negatives. Similarly, for the Resort Hotel, the model exhibited an even higher accuracy of 90.6%, with a precision of 87.1% for class 1, and a recall of 80.6%. The specificity for class 0 predictions stands at 95.4%, indicating the model's strength in accurately identifying true negatives.

These metrics underscore the Random Forest Classifier's robustness in handling classifications for the hotel industry, showing a well-balanced capability in both identifying the correct class labels and minimizing errors. Such efficacy is critical in practical applications where the costs of false positives and false negatives carry significant business implications. For this reason, the model's performance, as evidenced by the confusion matrices for both hotel types, confirms its suitability for predictive tasks in this sector, offering reliable insights for strategic decision-making based on customer booking behaviors.

5.4.3 Feature Importance

City Hotel

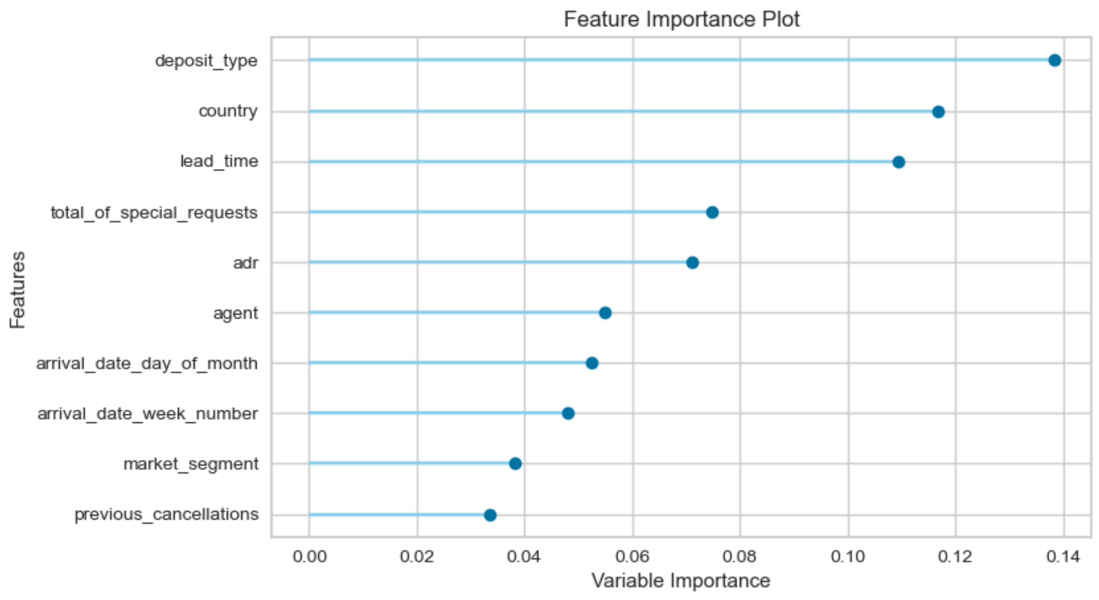


Figure 24: Feature Importance for City Hotel Bookings

This plot provides the relative importance of features, or variables, in predicting the target outcome for the City Hotel dataset using the random forest machine learning model. Starting with the highest importance, we see that deposit_type holds the most predictive power. This indicates that the type of deposit made by customers significantly influences the outcome, possibly indicating a relationship between the deposit type and the likelihood of booking cancellation. The country of the customer is the next most important feature. This suggests that the geographic location of customers, which often corresponds to different cultural norms and economic factors, has a substantial impact on their booking behaviors. Lead_time, or the duration between the booking and the actual stay, is also significant. A longer lead time could potentially increase the chances of cancellation due to changes in plans.

Interestingly, the total_of_special_requests made by customers also has substantial importance. This might indicate that customers with more specific needs or preferences are less likely to cancel their bookings. ADR (Average Daily Rate), which is a measure of the average realized room rental per day, also has a moderate level of importance. This might suggest that the price of the booking plays a role in the decision to cancel.
Other factors like the booking agent, arrival_date_day_of_month, arrival_date_week_number, market_segment, and previous_cancellations also contribute to the model's predictive power, but to a lesser degree.

Resort Hotel

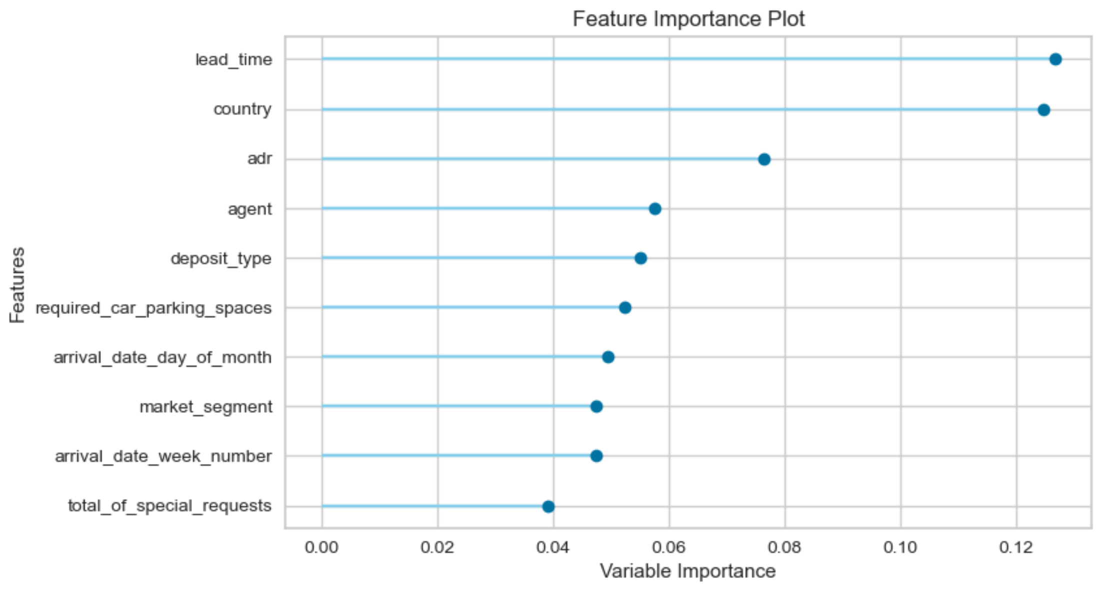


Figure 25: Feature Importance for Resort Hotel Bookings

In the Resort Hotel dataset, the variable lead_time holds the highest predictive importance. This suggests that the amount of time between when the booking is made, and the actual stay can significantly influence the likelihood of cancellation.
The country of the guest is the next most important feature. This implies that the geographic location or origin of guests can have a substantial impact on their booking behaviors. ADR (Average Daily Rate), a measure of the average realized room rental per day, is another significant factor. This indicates that the price or cost associated with the booking can sway the decision to cancel. The booking agent also appears to play an important role. This might mean that different agents or agencies have varying cancellation rates.

Interestingly, while deposit_type was the most important factor for the City Hotel, it has less predictive power in the Resort Hotel context. The variable required_car_parking_spaces is a notable factor unique to the Resort Hotel. This might suggest that guests who drive to resorts and require parking spaces are more committed to their bookings. Other variables such as arrival_date_day_of_month, market_segment, arrival_date_week_number, and total_of_special_requests also contribute to the predictive power of the model, but to a lesser extent.

In summary, both hotel types share similar influential factors such as lead time, country of origin, average daily rate, and booking agent. Yet, they also display unique characteristics, with the deposit type being more significant in City Hotels and required car parking spaces showing higher importance in Resort Hotels. These insights could potentially guide targeted strategies to mitigate booking cancellations in different hotel settings. City hotels often attract a diverse array of guests, including business travelers, tourists, and those visiting for events or short stays. The nature of these visits could make these bookings more susceptible to changes or cancellations due to varying reasons like changes in business plans, event cancellations, or alterations in travel plans. Therefore, the type of deposit made, which is a kind of financial commitment, could be a strong determinant of the likelihood of cancellation. For instance, guests who have made a non-refundable deposit might be less likely to cancel their booking, as they stand to lose their deposit amount. On the other hand, guests who have not made a deposit or made a refundable one might have a higher likelihood of cancellation, as they have less financial risk involved. This is not to say that 'deposit_type' is irrelevant for Resort Hotels. Still, it could be that other factors such as 'required_car_parking_spaces' might be more indicative of a guest's commitment to their stay in this specific setting.

6 Conclusions

This dissertation has offered a comprehensive examination of the application of machine learning algorithms in predicting hotel booking cancellations, with a focused analysis on different hotel types, City and Resort. The study initially highlighted the significant economic and operational impacts of booking cancellations on hotels, emphasizing the critical need for reliable predictive models in the hospitality industry.

In the comparative analysis of various machine learning algorithms, including decision trees, random forests, logistic regression etc., the research revealed diverse performances. Among these, the Random Forest Classifier stood out as a particularly effective model across both City and Resort Hotels. Its superiority in accuracy, AUC, and other metrics makes it a valuable tool for predicting cancellations. The Random Forest algorithm's ability to handle large datasets and produce stable and reliable predictions, even in the presence of noise and outliers, makes it especially suitable for the complex and variable-rich environment of hotel bookings.

In conclusion, the findings of this research are invaluable for the hospitality industry. They underscore the effectiveness of machine learning, particularly tree-based algorithms like Random Forest, in enhancing revenue management and customer satisfaction. These algorithms not only provide high accuracy in prediction but also offer insights into the factors driving cancellations, enabling hotels to devise targeted strategies.

7 References

Morales, D. R., & Wang, J. (2010). Forecasting cancellation rates for services booking revenue management using data mining. European Journal of Operational Research, 202(2), 554-562.

Protopapadakis 2013, p. 223

Kimes, 1989a, b · Smithetal, 1992· Hanks, Cross & Noland, 1992 · Tallury & vanRyzin, 2006

Bounatirou, M., & Lim, A. (2020). A case study on the impact of artificial intelligence on a hospitality company. In Sustainable Hospitality Management (Vol. 24, pp. 179-187). Emerald Publishing Limited

Marcus, B., & Anderson, C. K. (2008). Revenue management for low-cost providers. European Journal of Operational Research, 188(1), 258-272.

Badinelli, R. D. (2000). An optimal, dynamic policy for hotel yield management. European Journal of Operational Research, 121(3), 476-503.

Baker, T. K., & Collier, D. A. (1999). A comparative revenue analysis of hotel yield management heuristics. Decision Sciences, 30(1), 239-263.

Toh, R. S., & Dekay, F. (2002). Hotel room-inventory management: an overbooking model. Cornell Hotel and Restaurant Administration Quarterly, 43(4), 79-90.

Kimes, S. E., Chase, R. B., Choi, S., Lee, P. Y., & Ngonzi, E. N. (1998). Restaurant revenue management: Applying yield management to the restaurant industry. Cornell Hotel and Restaurant Administration Quarterly, 39(3), 32-39

Choi, S., & Mattila, A. S. (2004). Hotel revenue management and its impact on customers' perceptions of fairness. Journal of Revenue and pricing Management, 2, 303-314.
DeKay, F., Yates, B., & Toh, R. S. (2004). Non-performance penalties in the hotel industry. International Journal of Hospitality Management, 23(3), 273-286

Wangenheim, F. V., & Bayón, T. (2007). Behavioral consequences of overbooking service capacity. Journal of Marketing, 71(4), 36-47.

Phumchusri, N., & Maneesophon, P. (2014). Optimal overbooking decision for hotel rooms revenue management. Journal of Hospitality and Tourism Technology, 5(3), 261-277

Weatherford, L. R., & Kimes, S. E. (2003). A comparison of forecasting methods for hotel revenue management. International journal of forecasting, 19(3), 401-415

Pullman, M., & Rodgers, S. (2010). Capacity management for hospitality and tourism: A review of current approaches. International journal of hospitality management, 29(1), 177-187

Wirtz, J. (2003). Halo in customer satisfaction measures: The role of purpose of rating, number of attributes and customer involvement. International journal of service industry management, 14(1), 96-119

Hwang, J., & Wen, L. (2009). The effect of perceived fairness toward hotel overbooking and compensation practices on customer loyalty. International Journal of Contemporary Hospitality Management, 21(6), 659-675

Ivanov, S. H. (2007). Dynamic overbooking limits for guaranteed and nonguaranteed hotel reservations. Tourism today, 7, 100-108

Wang, H., Lei, Z., Zhang, X., Zhou, B., & Peng, J. (2016). Machine learning basics. Deep learning, 98-164

Saravanan, R., & Sujatha, P. (2018, June). A state of art techniques on machine learning algorithms: a perspective of supervised learning approaches in data classification. In 2018 Second international conference on intelligent computing and control systems (ICICCS) (pp. 945-949)

Wang, D. (2001). Unsupervised learning: foundations of neural computation. Ai Magazine, 22(2), 101-101

Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. Journal of artificial intelligence research, 4(1), 237-285.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). Classification and Regression Trees. Wadsworth & Brooks/Cole Advanced Books & Software.

Raileanu, L. E., & Stoffel, K. (2004). Theoretical comparison between the Gini Index and Information Gain criteria. Annals of Mathematics and Artificial Intelligence, 41(1), 77-93

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. Machine Learning, 63(1), 3-42

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30.

Stoltzfus, J. C. (2011). Logistic regression: a brief primer. Academic emergency medicine, 18(10), 1099-1104

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21-27.

Deza, M. M., & Deza, E. (2009). Encyclopedia of distances. In Encyclopedia of distances (pp. 1-583). Springer.

Han, J., Pei, J., & Tong, H. (2022). Data mining: concepts and techniques. Morgan kaufmann.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297.

Venners, Bill (13 January 2003). "The Making of Python". Artima Developer. Artima. Retrieved 22 March 2007

Pandas. (2023). Home. Retrieved from https://pandas.pydata.org/

Scikit-learn. (2023). Home. Retrieved from https://scikit-learn.org/stable/

Anaconda. (2023). Home. Retrieved from https://www.anaconda.com/

Jupyter. (2023). Home. Retrieved from https://jupyter.org/

PyCaret. (2023). Home. Retrieved from https://pycaret.org/

Sánchez, E. C., Sánchez-Medina, A. J., & Pellejero, M. (2020). Identifying critical hotel cancellations using artificial intelligence. Tourism Management Perspectives, 35, 100718.

Antonio, N., De Almeida, A., & Nunes, L. (2017). Predicting hotel booking cancellations to decrease uncertainty and increase revenue. Tourism & Management Studies, 13(2), 25-39.

Caicedo-Torres, W., & Payares, F. (2016). A machine learning model for occupancy rates and demand forecasting in the hospitality industry. In Advances in Artificial Intelligence-IBERAMIA 2016: 15th Ibero-American Conference on AI, San José, Costa Rica, November 23-25, 2016, Proceedings 15 (pp. 201-211). Springer International Publishing.

Ku, C. H., Chang, Y. C., Wang, Y., Chen, C. H., & Hsiao, S. H. (2019). Artificial intelligence and visual analytics: a deep-learning approach to analyze hotel reviews & responses. In 52nd Annual Hawaii International Conference on System Sciences (HICSS). Newcastle University.

Al-Smadi, M., Qawasmeh, O., Al-Ayyoub, M., Jararweh, Y., & Gupta, B. (2018). Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews. Journal of computational science, 27, 386-393.

Peng, J., Hahn, J., & Huang, K. W. (2023). Handling Missing Values in Information Systems Research: A Review of Methods and Assumptions. Information Systems Research, 34(1), 5-26.