Boston University
Questrom School of Business
Professor Tsoukalas
BA305 Spring 2023
Section A1

Hotel Reservations Cancellations
Analysis Report

Team 1:
Sahir Doshi
Sophia Stearn
Andrew Trapp
Daniela Wong

May 2023

Introduction

For hotels, reservation cancellations are an unwelcome occurrence that happen on a frequent basis. The rise of the digital age has only exacerbated this issue, as customers can cancel their bookings with the click of a button. Typically, a cancellation results from a customer experiencing a change in their plans and/or scheduling conflicts. Hotel chains have acknowledged this instance, sometimes providing free-of-charge or low-charge cancellation fees in hopes to retain customers for future bookings. However, while benefiting the customer, this may severely harm a hotel chain by causing a significant loss in revenue over time. It is important for a hotel to understand their reservation cancellation rate to maximize their profits while still maintaining customer loyalty and full capacity. As such, we are presented with the aim of our report: deducing the factors which impact a customer's reservation cancellation, each factor's respective weight in the customer's decision, and ultimately finding a model which predicts the likelihood of cancellation.

The hotel and tourism industry typically accounts for 10% of world GDP which truly emphasizes the importance and the repercussions that this project could account for. We wanted to address three main pain points that hotels face: minimize costs/losses, maintain customer loyalty and have as much full capacity as possible. Through our experimentation with multiple models, we aim to find a model that will be able to predict the likelihood of cancellation. Through this, hotels will be able to change their business models and have more flexibility in their policies.

Data Overview

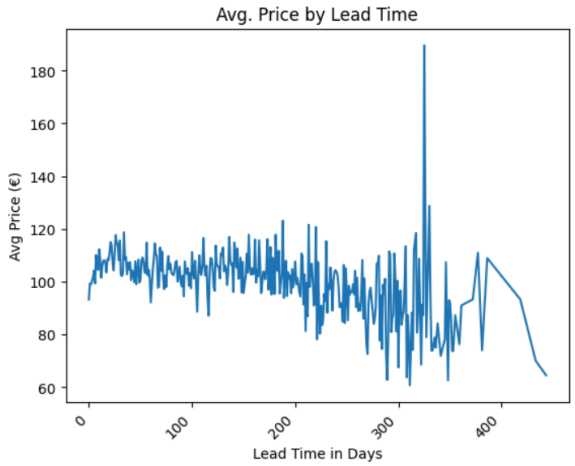
Our dataset comes from Kaggle, a respected and credible source for data among other features. The author of this dataset is a Kaggle user named Ahsan Raza. While we were unable to find any credentials for this person, Kaggle's credibility system ranked this dataset as 10.0 - a full score. This dataset presumably comes from Europe as price information is listed in euros. While we don't have an exact idea of what European country(ies) or region(s) it stems from, we can make calculated guesses based on information gathered from our exploratory data analysis.

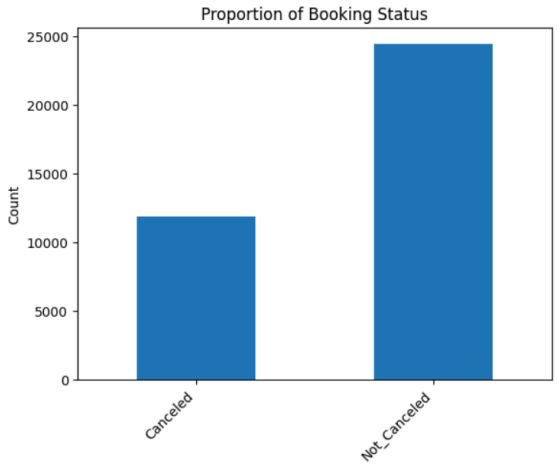
The raw data consists of 19 columns and 36,275 data points. There were no columns with missing values. Our response variable is Booking_Status, a binary variable with two outcomes: Not_Cancelled and Canceled. The details of each column, and our decision on each column's inclusion/exclusion are summarized in the Appendix, Figure 1.

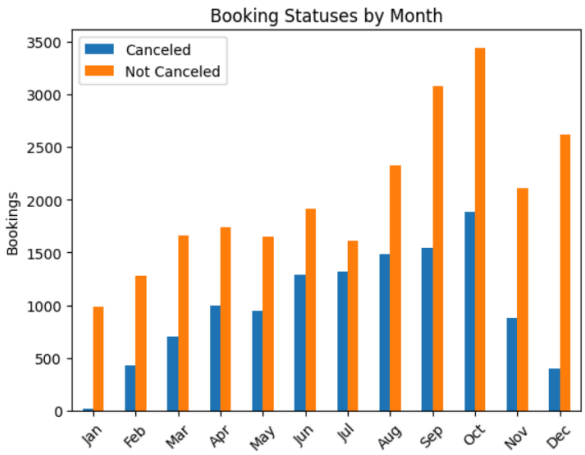
Exploratory Data Analysis

The data exploration we conducted served to discover whether predictors had any significant relationships. From the first table, we can see that Room Types 1 and 4 accounted for the majority of bookings, which we further discuss in Bucketing, Scaling, Adding Dummies. Interestingly, the average price by lead time shows that generally speaking the average price of the room gradually decreases as lead time extends, except for one anomaly near the 330-day mark, which may be due to a lack of samples. Our dataset had roughly half as many cancellations as non-cancellations, which suggests that ⅔ of the time bookings are not canceled. Finally, October had the most cancellations at 35.4%, which we suspect may be due to the German festival Oktoberfest, assuming this dataset comes from Germany. During this holiday period, plans may fluctuate significantly, so cancellations could be more common.

Type	Price/Night	Stay	Guests	Proportion
Room_Type 1	95.918532	2.904977	1.819694	0.775465
Room_Type 4	125.287317	3.474492	2.230642	0.166975







Data Pre-Processing

Bucketing, Scaling, Adding Dummies

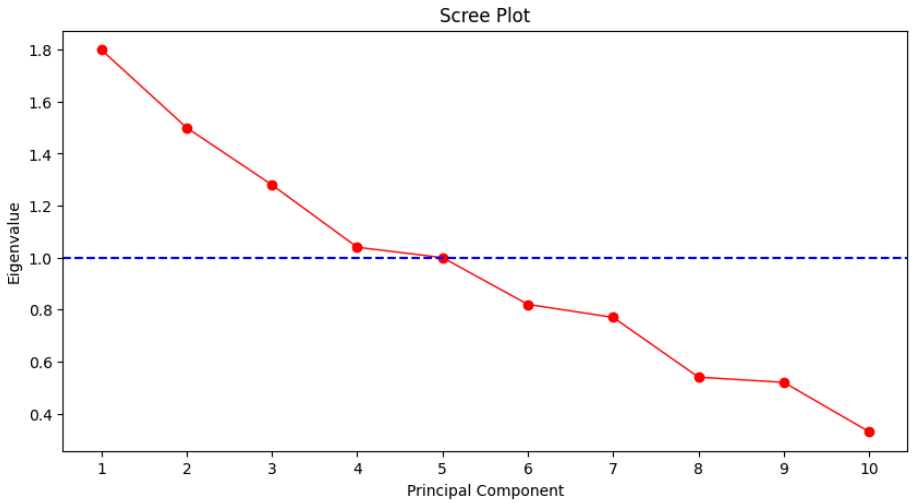
Two predictors, type of meal plan and type of room reserved, contained many options which were only numbered. Through the exploratory analysis, we discovered that 95% of rooms fell under room type 1 or 4. This predictor was then bucketed into "common" and "uncommon" room types. Based on average prices of 100 and 125 euros per night, 1.75 and 2.25 guests, and 3 and 3.5 days, respectively, we concluded that these were variations of standard hotel room types, e.g. one to two queen beds, one bath.

PCA. Our full PCA table can be found in the Appendix under Figure 1. We found that 5 variables accounted for 61% of the variance, specifically:

	% of Variance Explained	Cumulative % Explained
0	0.166	0.166
1	0.138	0.304
2	0.118	0.422
3	0.096	0.518
4	0.092	0.610
…		

(where the rows 0 - 4 correspond to the columns in the full PCA table)

Additionally, we used components above 1 eigenvalue according to the Latent Root Criterion. These 5 components showed no correlation between them, shown in the Appendix, Figure 3.



Correlation Matrix Heat Map

Refer Appendix, *Figure 2*. Our heat map shows that there is no significant correlation between any two variables, with the highest correlation value being 0.5 between *Number of Previous Cancellations* and *Number of Previous Bookings Not Canceled*, and *Repeated Guest* and *Number of Previous Bookings Not Canceled*. As such, we decided that, since there are no predictors showing interaction, we do not need to make any modifications to our data (i.e. creating an interaction term).

Train/Test Split (Stratification, etc.)

A training/testing split of 67/33 was chosen arbitrarily. Stratification on the output variable was selected in the split function and verified by evaluating the proportion of output 1 (canceled reservations) in the training and testing set.

Modeling

Baseline Accuracy

Selecting the majority vote, which is to always predict that a guest will not cancel, will lead to an accuracy of roughly 67%. This prediction method has no practical application in business, but it serves as a line by which we need to improve upon. The model produces a recall, precision, and F1 of 0%. As such, any model which can produce a recall and precision higher than 0% would have some practical business application as the goal of this project is to consider the costs associated with misclassifications, rather than simply focusing on accuracy. Confusion Matrix in Appendix, Figure 4.

Logistic Regression

Logistic regression is a method used to predict binary outputs for classification problems by assigning a probability that an instance will belong to class 0 or 1. Our first logistic model only uses a maximum iterations of 1000 to serve as a baseline for which we can improve a logistic regression model. The model was applied to both the scaled dataset as well as the PCA dataset, producing accuracy scores of 80.5% and 79%, respectively. We then chose to use the standardized dataset as computational costs were low for a logistic regression, leading to these performance metrics: precision = 0.73, recall = 0.64. F1 = 0.68. The next step was to loop over every possible threshold by which classification is decided. The best threshold was 0.54 and resulted in a modest increase in accuracy to 81%. Despite this increase in accuracy, it came at a tradeoff to recall as it slumped to 0.56, dragging the F1 to 0.63. The last changes we tried with logistic regression in an attempt to increase performance was by testing a variety of penalty types. These

included L1, L2, and Elastic Net. All three penalty types yielded similar results with accuracy = 0.81, precision = 0.73, recall = 0.64, and F1 = 0.68. Confusion Matrix in Appendix, Figure 5.

k-Nearest Neighbors

K-Nearest Neighbors is a prediction method, in our case used for classification, which aims to predict a class based on k-number of "similar" instances. Similarity is measured by the "distance" of predictors to other "nearby" predictors. Class is then determined by a majority vote. Similar to logistic regression, we ran a kNN on both the standardized data as well as the PCA data. A loop was run to test the best accuracy by stepping up from 1 neighbor to 99 neighbors. The best accuracy for the standardized kNN was 85.5% using 11 neighbors, while the best accuracy for the PCA kNN was 88%, using 1 neighbor. To prevent potential overfitting, we chose to use 11 neighbors which yielded an accuracy = 0.86, precision = 0.81, recall = 0.76, and F1 = 0.78. Confusion Matrix in Appendix, Figure 6.

Decision Trees - Full, Pruned using Penalty, Grid Search for Best Parameters

The first decision tree made had no penalty or parameter constraints assigned and created a full tree with 6383 nodes, 3192 leaves, and a maximum depth of 44. The performance metrics yielded - accuracy: 0.87, precision: 0.79, recall: 0.8, f1: 0.8. While the performance is impressive, this tree is prone to overfitting. Full trees can create leaves for every instance in the data. In our dataset there are 36,275 instances, meaning that each leaf, on average, classified roughly 12 instances. As such, the next step was to test a variety of penalties on the decision tree classifier to train a tree which more generally classified instances in order to save on computational costs and prevent overfitting.

To evaluate a variety of penalty values a cost complexity pruning path was run on the full tree to extract 1365 potential alpha values. These values were all tested by running a loop and outputting two graphs, one measuring the alphas against impurity, and the other measuring the former against accuracy. Each pointed to an alpha value of .003 as the optimal penalty to balance strong accuracy and impurity. The performance metrics for the pruned tree using an alpha penalty were - accuracy: 0.83, precision: 0.79, recall: 0.65, f1: 0.72. Slightly poorer performance than the full tree, but much better at generalizing data as the tree had 25 nodes, 10 leaves, and a max depth of 6.

The last tree tested utilized the RandomizedSearchCV and GridSearchCV function to test various specific parameters, rather than an all-encompassing penalty term. The best parameters for a tree were:

'max_depth': 10, 'min_impurity_decrease': 0, 'min_samples_leaf': 20, 'criterion': 'entropy'

The 5-fold CV score result was 86%, though it could still be prone to overfitting as number of nodes = 407, max depth = 10, and leaves = 204, and had low interpretability and high computational costs.

Adaptive Boosting Classifier and Random Forest

For ensemble methods, we first evaluated an Adaptive Boosting Classifier on the data and utilized a random search to find the best parameters. The best parameters were: 'algorithm': 'SAMME', 'base_estimator': DecisionTreeClassifier(max_depth=7), 'learning_rate': 1, 'n_estimators': 1735. These estimators yielded a CV score of 89.2%, the best performing method we applied. Adaptive Boosted classification is a strong learning algorithm because of its ability to learn from past mistakes and focus on the "hardest" instances to classify, while having no theoretical risk of overfitting. The downside is that the method sacrifices interpretability and can be thrown off by outliers or noise in the data.

Random Forest, similarly, utilizes many weak predictors and aggregates the results to make a strong prediction. Though it does not learn sequentially like AdaBoost, but in parallel. Yet again, a grid search was run to test best parameters, though computational costs can grow very large by including too many parameters, so only n_estimators and criterion were included. The best parameters were:

'criterion': 'entropy', 'n_estimators': 2281. These estimators yielded a CV score of 89.1%, nearly as accurate as AdaBoost. While random forest offers similar benefits to AdaBoost, it is much less interpretable. RF does output the feature importance, providing some interpretability, but a tree cannot be made. Confusion Matrix in Appendix, Figure 7.

Neural Network

The last model tested was a neural network which utilized a randomized search to find the best parameters of hidden_layer_sizes=(50,50), activation = 'relu', solver = 'lbfgs', learning_rate = 'adaptive', alpha = .1, max_iter = 2000. These parameters yielded a CV score of 86%, precision: 0.8 recall: 0.79 f1: 0.8. Surprisingly this performance was worse than the ensemble methods. One of the reasons could be due to the length of time required to search for optimal parameters. A neural network can take up to 5 minutes to train on its own, when this is tested in CV = 5, it now may take roughly 20-30 minutes, and if testing several sets of parameters, the time to evaluate may grow to many hours. The cost complexity of testing eventually becomes much too large to find the true optimal parameters, which may or may not far exceed the strength of the current parameters. Additionally, the method has very low interpretability, making it difficult to justify in a business setting where qualitative observations are important. Confusion Matrix in Appendix, Figure 8.

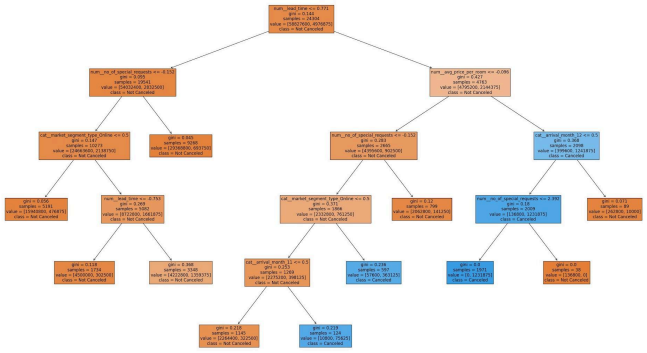
Results & Explanation

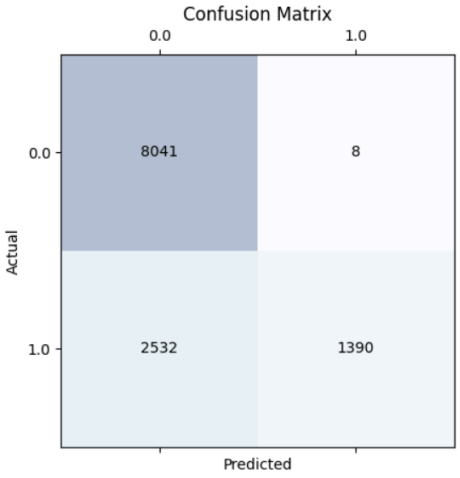
All models exceeded the baseline model's accuracy, indicating the usefulness of machine learning in this context. However, accuracy may not always reveal whether the users are being accurately identified in classification problems. Therefore, we assessed our models based on three other metrics: F1 score, precision, and recall. Precision denotes the ability to correctly identify canceled reservations, while recall refers to the ability to identify uncanceled reservations. These metrics have an implicit trade-off, wherein if the cost of false positives is high, precision is prioritized, and if the cost of false negatives is high, recall is prioritized. When the costs are either unknown or equal, the focus shifts to F1 score, which gives equal importance to both precision and recall. We aimed to provide a recommendation which addresses the precision, as the cost of false positives are higher than the costs of false negatives, while having high interpretability and easy scalability. As such, we chose to create a cost-weighted decision tree.

To create a model which adequately addresses the needs of a hotel management team, we first need to understand the nature of hotel costs and statistics. First we researched cancellation terms and costs. Generally, hotels do not charge a cancellation fee unless it occurs within 48-72 hours of the reservation, in those cases a fee of one night's stay and taxes is charged. While the fee may mitigate some loss, the hotel will miss out on the revenue related to the stay. In the case of a false negative, which would be to classify a stay as probable to not cancel but the reservation ultimately does cancel, we can start by taking the average cost of a stay in the most common rooms, 101 euros per night for an average stay of 3 nights totaling 304 euros. Next we can subtract out the fee for cancellation, which was noted as one night's stay and a fee, 101 euros and an assumption of a 50 euro fee. Speaking with the Courtyard by Marriott Boston Brookline, we learned that the majority of cancellations occur well in advance of the 3 days. Roughly one in ten cancellations occur during the penalty window as an assumption for the cost weight. As such, we can assign a penalty of 15 euros, (304-151)*(1/10), attributable to missed booking revenue, and add 10 euros as an assumption of missed amenities such as room service, for a total penalty of 25 euros. In the case of a false positive, when a reservation is classified as likely to cancel but ultimately shows up, the cost to the hotel is far greater because the hotel must now allocate an unexpected guest. We can create a cost weighting by using the Marriott info to assume that eight times out of ten, the guest can be accommodated in a vacant room at no cost. In one out of ten cases, the guest must be upgraded to a vacant room of higher quality. Room type 4 (R4) is one of the most common rooms, but is 152 euros more expensive per stay than room type 1 (R1) . A cost can then be calculated as 152 * .1 for a total penalty of 15.2. The last scenario is that one in ten times a reservation will need to be completely compensated as there is no vacancy for the guest. In this case, the reservation will have a cost of the full stay, which is 435

euros. The final cost weighting of a false positive is then calculated to be 15.2 + 43.5 = 58.7 and rounded up to 60 euros for ease.

With these weights inputted into the tree, a cost complexity pruning path is run on the full weighted tree, tested just like our original tree was, and utilized an alpha value of .001. The performance metrics for the pruned cost-weighted tree were: accuracy: 0.79, precision: 0.99, recall: 0.35, f1: 0.52. The tree has a marginal 12% improvement in accuracy over the baseline, but the precision is nearly perfect, achieving our goal of minimizing false positives and allocating most mistakes to the lower cost false negatives. Additionally, the tree has a max depth of 4, 10 leaves, and 19 total nodes, making it one of the most interpretable models we created. Feature importances also gave us insight into lead time, price per room, online booking type, and number of special requests as important indicators of whether a booking will be canceled or not.





Final Comparison

In identifying the model most appropriate to address the plight of the hotel industry, penultimate evaluations deduce precision to be the summary measure of greatest efficacy despite successful optimization of accuracy. Models utilizing robust ensemble methods such as decision trees with bagging, with boosting, and random forests with grid search yield amongst the highest observed accuracy improvements. Despite producing the most improved accuracy scores, these models' high complexities hinder interpretability. Prioritizing the precision metric enables the fitting of cost-weights, ultimately bearing financially apropos outcomes. Given the associated-cost variation of different outcomes, assigning monetary significance is crucial to constructing an efficacious model of hotel cancellations.

Furthermore, as explained in Results & Explanation, the cost of a false positive outweighs the cost of a false negative for a hotel. Therefore, we are looking for a model that emphasizes precision, keeps note of recall, and has a higher accuracy from the baseline.

We found that the model that best satisfies this criterion is Tuned Optimal Decision Tree with Costs Assigned. After first comparing the accuracy versus alpha, and then comparing the impurity versus alpha, we alpha = 0.001 to be most effective (refer Figure 9 and Figure 10 in Appendix). This model had the following results: accuracy = 78%, F1 = 52%, precision = 99%, recall = 35%, optimal threshold = 25%. The breakdown of each variable's importance is in the Appendix, Figure 11. Most importantly, the model only misclassified 8 data points in terms of false positives, shown in the confusion matrix above.

This confirms that an Optimal Threshold Decision Tree with Costs Assigned is the best model in a real-world scenario, while still accounting for all various criterion and errors.

Challenges

Through our project, we encountered three main problems: not knowing when cancellations occurred during the dataset, not knowing the exact location of the dataset, and the computational costs when doing parameter searches.

During our exploratory findings using the data, it would've been very useful to have insight as to what type of cancellations occurred, e.g., no show vs called ahead, as it would've been useful in both prediction and assigning costs.

As for location, we didn't have information regarding where the hotels were located which would've helped us understand where this model can be applied. Without knowing the location of the hotels, it is challenging to determine if the model is effective in all locations or just in specific regions. Moreover, the location can also impact the type of guests who stay at the hotels, which can influence their preferences and expectations.

Finally, we had many challenges with how much time tuning could take which would increase computational costs. In an ideal setting, we would have a supercomputer that would allow us to produce faster results or more machines to run in parallel.

Conclusion

In conclusion, our recommendation is for hotels to use the Optimal Threshold Decision Tree with Costs Assigned based on the parameters discussed in our Final Comparison. This model primarily focuses on

reducing false positives which would help hotels reduce their sunk costs. Despite our in-depth analysis, we still faced drawbacks, specifically with knowing the type of cancellation, where the data comes from, and the lack of a powerful enough computer to conduct parameter tuning for costly models. However, our model still provides a fairly-accurate and easily-interpretable result, with a near-perfection precision score and lowered computational costs. Hotels could use our model for their own businesses, albeit with some margin for error.

Appendix

	0	1	2	3	4
Number of Adults	-0.43	0.01	0.26	-0.16	-0.57
Number of Children	-0.32	0.29	-0.14	0.56	0.42
Number of Weekend Nights	-0.18	-0.07	0.45	-0.27	0.51
Number of Week Nights	-0.19	-0.17	0.52	0.01	0.31
Lead Time	-0.03	-0.41	0.40	0.46	-0.28
Number of Previous Cancellations	0.30	0.47	0.38	0.16	-0.16
Number of Previous Bookings Not Canceled	0.35	0.49	0.33	0.10	-0.05
Average Price Per Room	-0.51	0.25	-0.10	0.31	-0.16
Number of Special Requests	–0.34	0.36	0.07	-0.042	-0.06

Figure 1

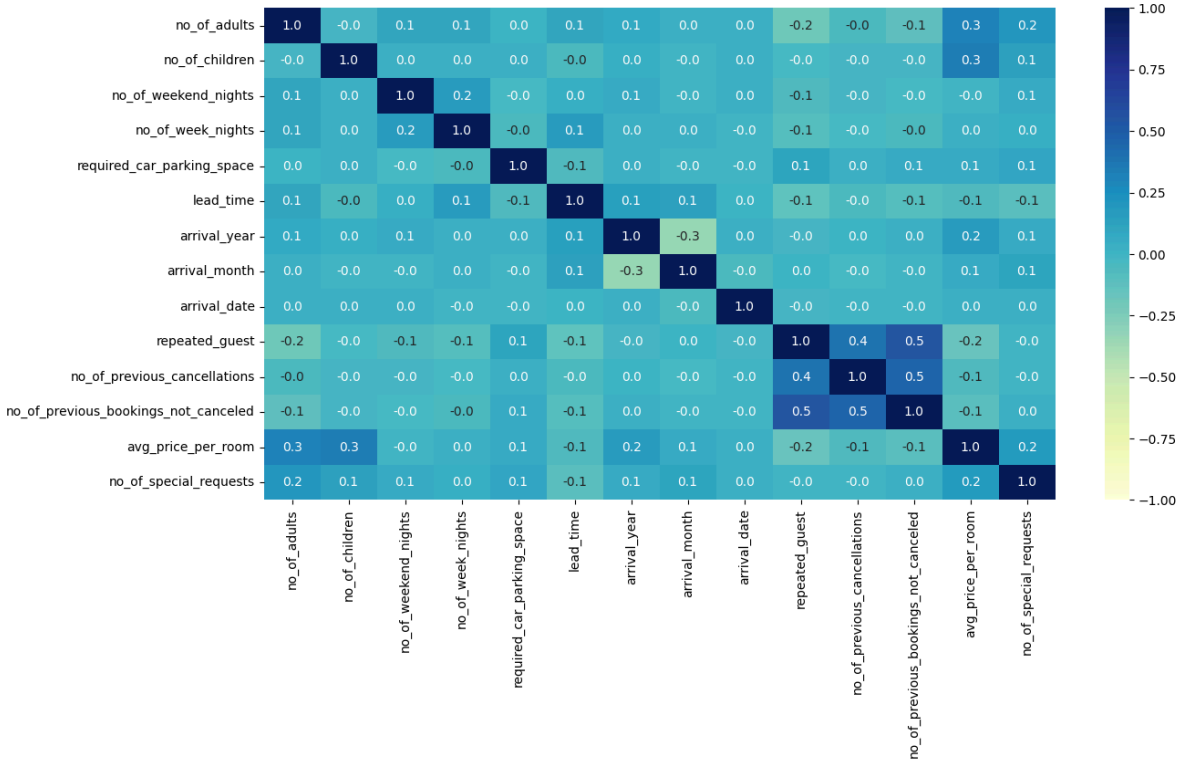


Figure 2

	0	1	2	3	4
0	1.0	-0.0	0.0	-0.0	0.0
1	-0.0	1.0	0.0	0.0	0.0
2	0.0	0.0	1.0	0.0	0.0
3	0.0	0.0	0.0	1.0	0.0
4	0.0	0.0	0.0	0.0	1.0

Figure 3

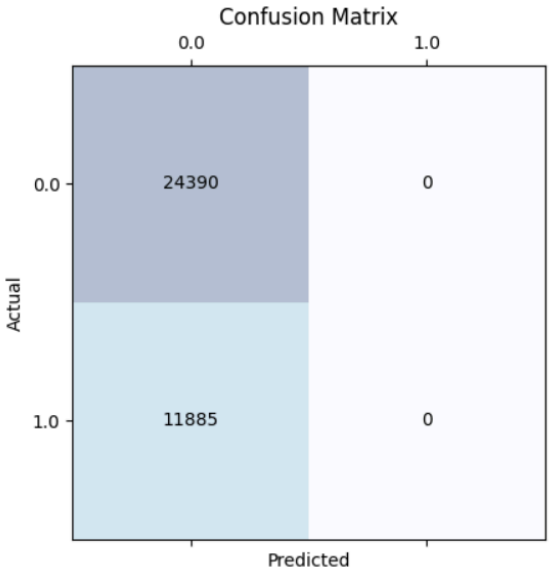


Figure 4

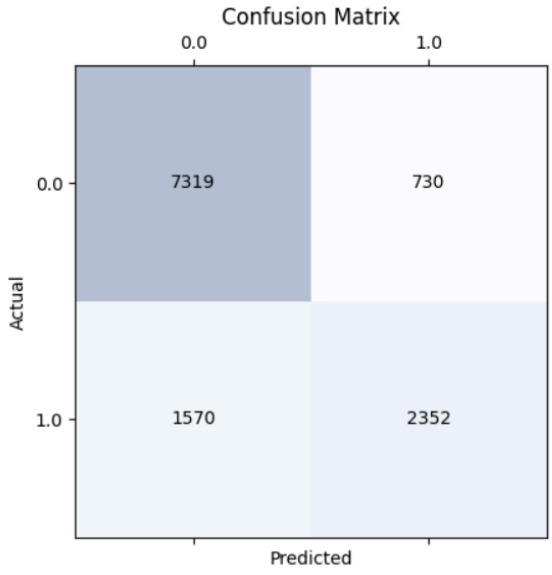


Figure 5

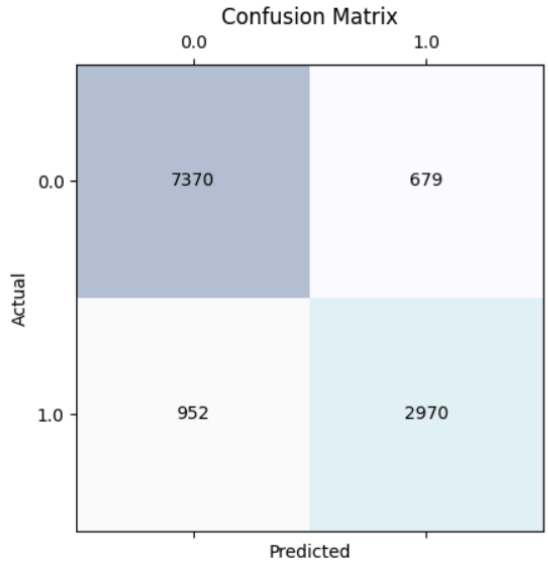


Figure 6

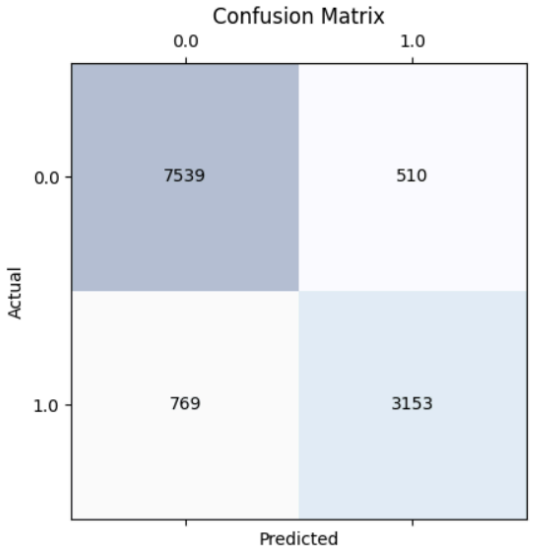


Figure 7

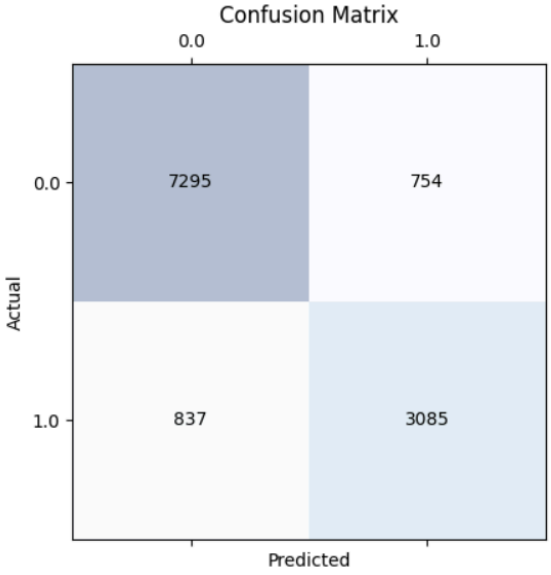


Figure 8

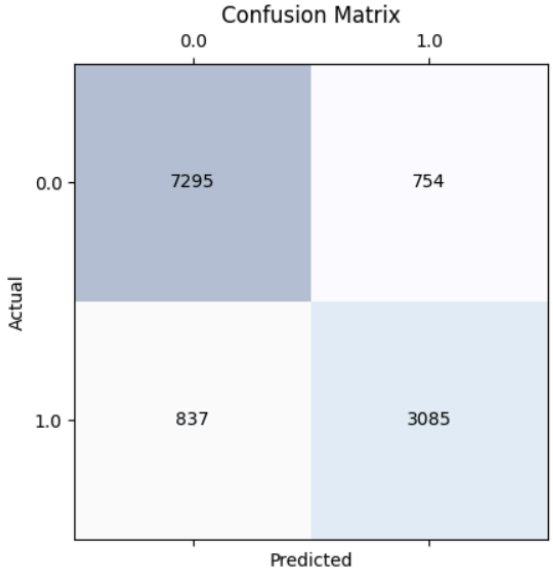


Figure 9

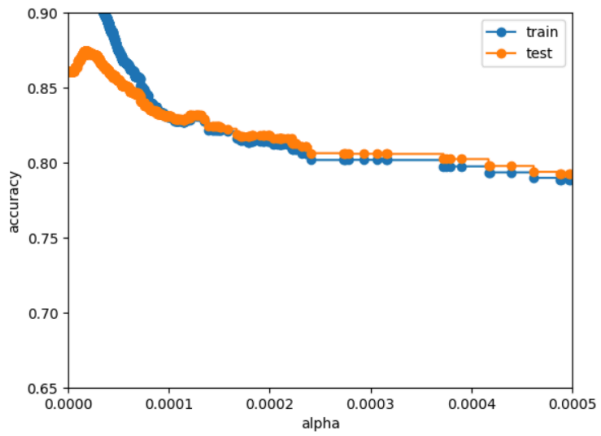


Figure 9

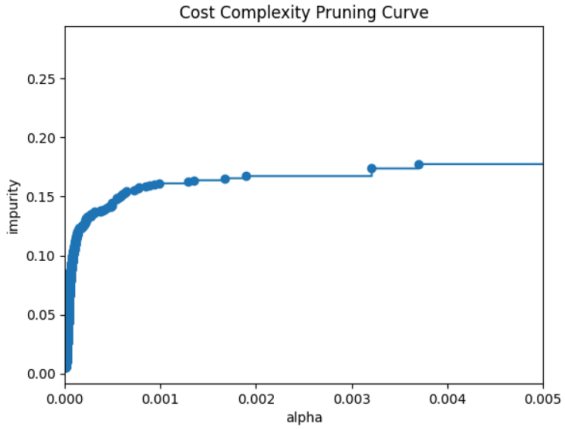


Figure 10

Feature	Importance
Lead Time	31%
Avg Price per Room	27%
Market Segment Type Online	18%
Number of Special Requests	13%
Arrived in December	10%
Others	1%

Figure 11