Data Mining and Visualization Project

Almog Sharoni 208611764

Netanel Daniel 209234392

GitHub Link

Introduction

***Problem* Statement**

Hotels frequently encounter challenges with managing reservations due to unexpected cancellations. These cancellations can lead to significant revenue losses and inefficient resource allocation. For instance, when guests cancel their reservations at the last minute, hotels may struggle to fill those rooms, resulting in lost income. Furthermore, inaccurate predictions of cancellations can cause overbooking, leading to customer dissatisfaction and reputational damage. Therefore, accurately predicting which reservations are likely to be canceled is crucial for hotels to optimize their occupancy rates and maximize profitability.

Objective

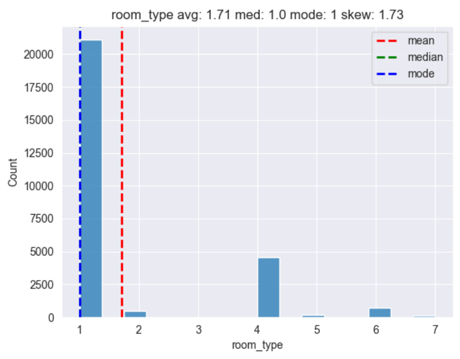
The primary objective of this project is to develop a robust predictive model to forecast hotel reservation cancellations using the provided dataset. By leveraging data mining techniques and classification algorithms, we aim to identify patterns and factors that influence cancellations. This predictive model will help hotels implement more effective reservation strategies, improve operational efficiency, and enhance customer satisfaction. Specifically, we will:

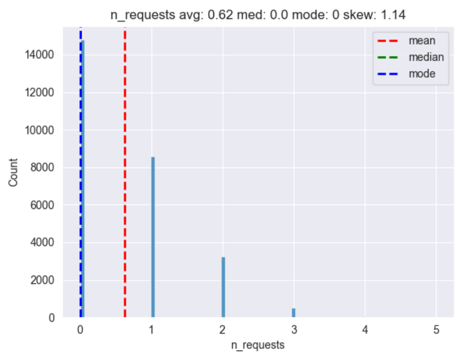
- Analyze and preprocess the dataset to extract meaningful insights.
- Train multiple classification models and compare their performance.
- Select the best-performing model to predict cancellations on a test dataset.

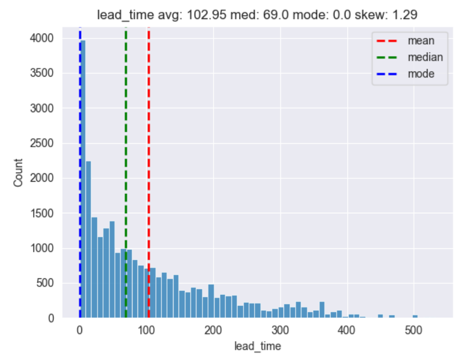
Data Information

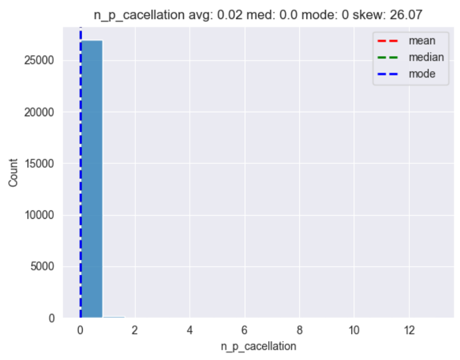
Train shape: 27213 rows with 18 features

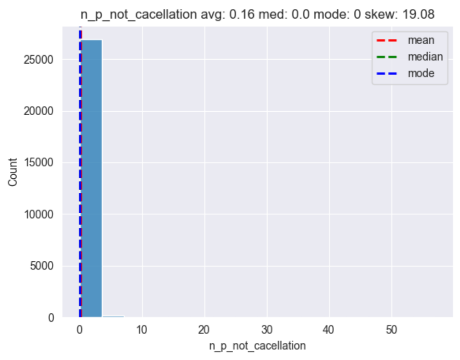
Column	Description	Non-Null Count	Dtype
ID	Reservation ID	27213	object
weekend_nights	Number of weekend nights	27213	int64
week_nights	Number of weeknights	27213	int64
room_type	Type of room booked	27213	object
board_type	Type of board (e.g., meal plan)	19045	object
n_adults	Number of adults	27213	int64
n_less_12	Number of children aged less than 12	27213	int64
n_more_12	Number of children aged more than 12	27213	int64
booked_tour	Indicates whether a tour was included in the reservation	27213	int64
n_requests	Number of special requests made by the guest	27213	int64
lead_time	Number of days between the reservation date and the arrival date	26794	float64
purchase_type	Type of purchase (e.g., direct, online)	22366	object
n_p_cacellation	Number of previous reservations that were canceled by the customer	27213	int64
n_p_not_cacellation	Number of previous reservations not canceled by the customer	27213	int64
repeated	Indicates whether the reservation is a repeat reservation	27213	int64
price	Price of the reservation	23808	float64
date	Date of the reservation	27213	object
is_canceled	Target value, 0 – not canceled, 1 – canceled	27213	int64

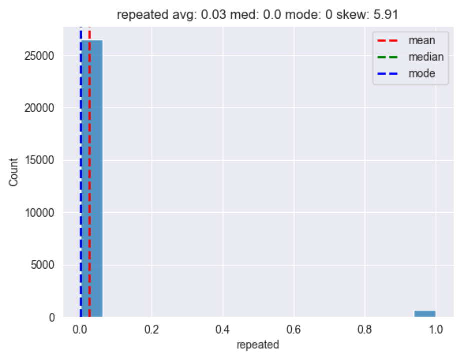

room_type avg: 1.71 med: 1.0 mode: 1 skew: 1.73
mean
median
mode
Count
room_type

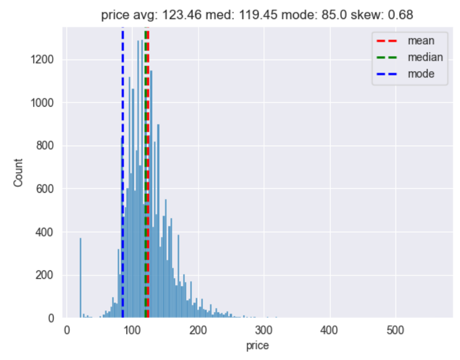

n_requests avg: 0.62 med: 0.0 mode: 0 skew: 1.14
mean
median
mode
Count
n_requests

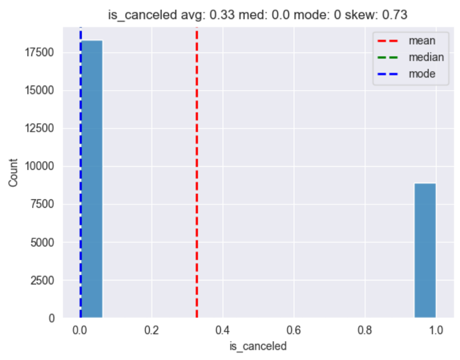

lead_time avg: 102.95 med: 69.0 mode: 0.0 skew: 1.29
mean
median
mode
Count
lead_time

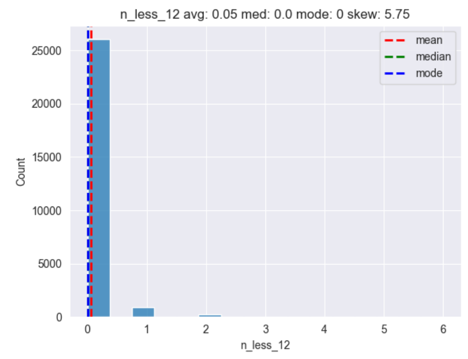

n_p_cacellation avg: 0.02 med: 0.0 mode: 0 skew: 26.07
mean
median
mode
Count
n_p_cacellation

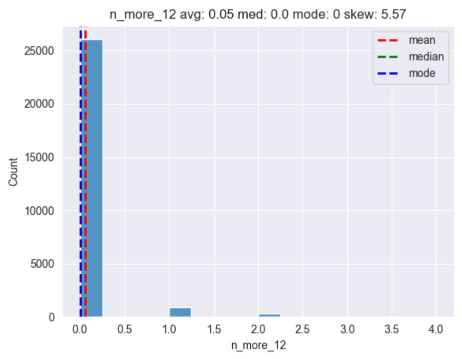

n_p_not_cacellation avg: 0.16 med: 0.0 mode: 0 skew: 19.08
mean
median
mode
Count
n_p_not_cacellation

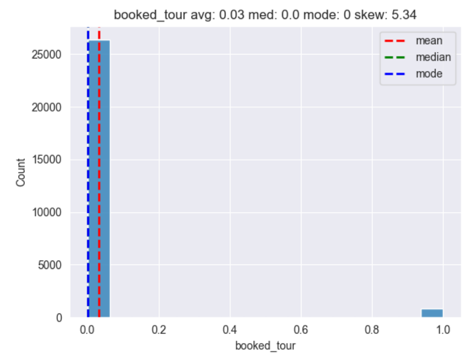

repeated avg: 0.03 med: 0.0 mode: 0 skew: 5.91
mean
median
mode
Count
repeated

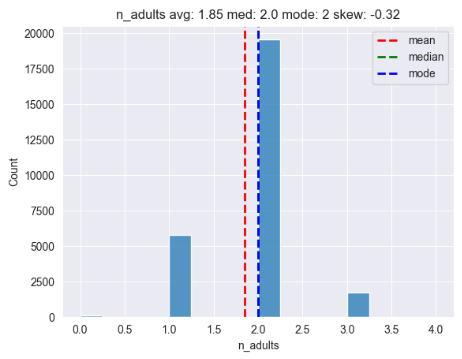

price avg: 123.46 med: 119.45 mode: 85.0 skew: 0.68
mean
median
mode
Count
price

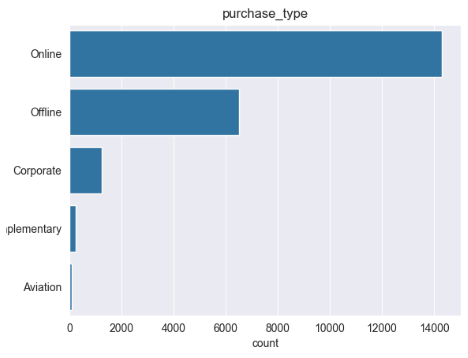

is_canceled avg: 0.33 med: 0.0 mode: 0 skew: 0.73
mean
median
mode
Count
is_canceled

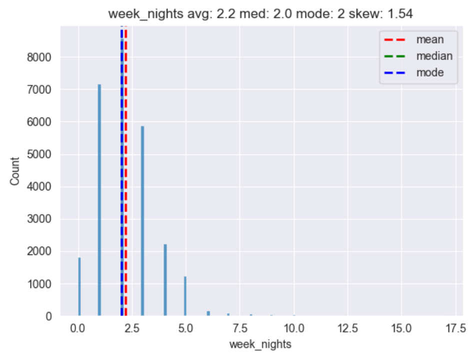

n_less_12 avg: 0.05 med: 0.0 mode: 0 skew: 5.75
mean
median
mode
Count
n_less_12

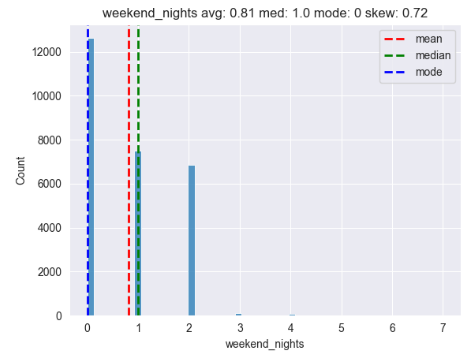

n_more_12 avg: 0.05 med: 0.0 mode: 0 skew: 5.57
mean
median
mode
Count
n_more_12

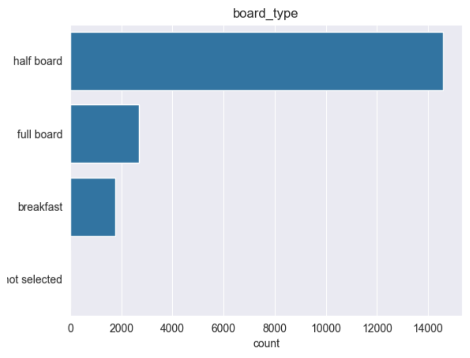

booked_tour avg: 0.03 med: 0.0 mode: 0 skew: 5.34
mean
median
mode
Count
booked_tour


n_adults avg: 1.85 med: 2.0 mode: 2 skew: -0.32
mean
median
mode
Count
n_adults


purchase_type
Online
Offline
Corporate
ıplementary
Aviation
count

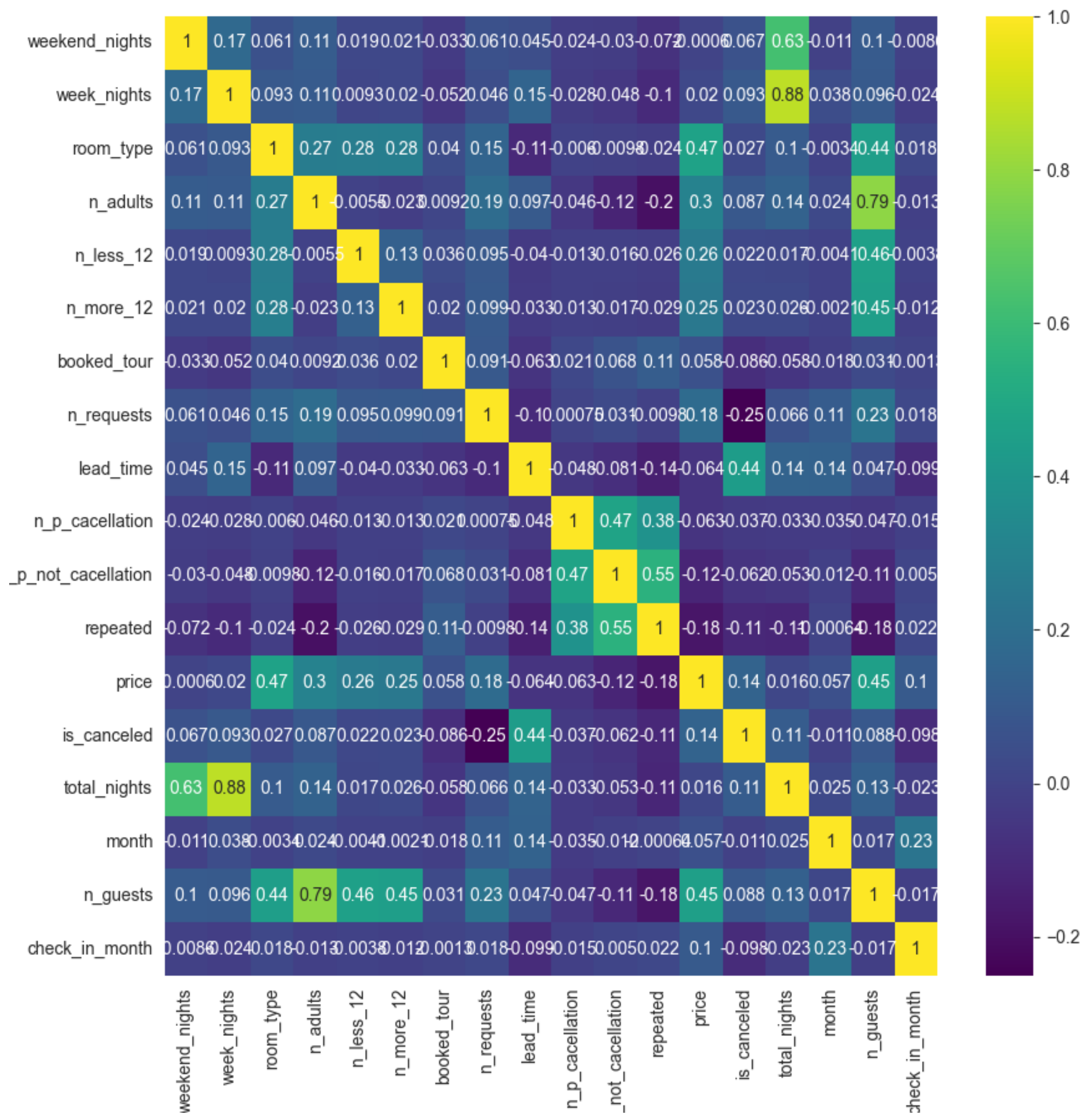

week_nights avg: 2.2 med: 2.0 mode: 2 skew: 1.54
mean
median
mode
Count
week_nights


weekend_nights avg: 0.81 med: 1.0 mode: 0 skew: 0.72
mean
median
mode
Count
weekend_nights


board_type
half board
full board
breakfast
ıot selected
count

Data Statistic

Attributes Correlation



Main insights:

Cancellation:

- **Lead Time (Positive Correlation):** Cancellations are positively correlated with lead time (0.44). Longer lead times increase the chance of unexpected events or changes in plans leading to cancellations.
- **Number of Requests (Negative Correlation):** Cancellations are negatively correlated with the number of requests (-0.25). Customers who make more requests are likely more invested in their stay and less likely to cancel.

- **Number of Repeated Stays (Negative Correlation):** There is a negative correlation (-0.11) between cancellations and the number of repeated stays. Repeat customers are typically more satisfied and less likely to cancel.
- **Price (Positive Correlation):** Cancellations are positively correlated with price (0.14), as higher prices may lead customers to seek better deals elsewhere, increasing the likelihood of cancellation.

Price:

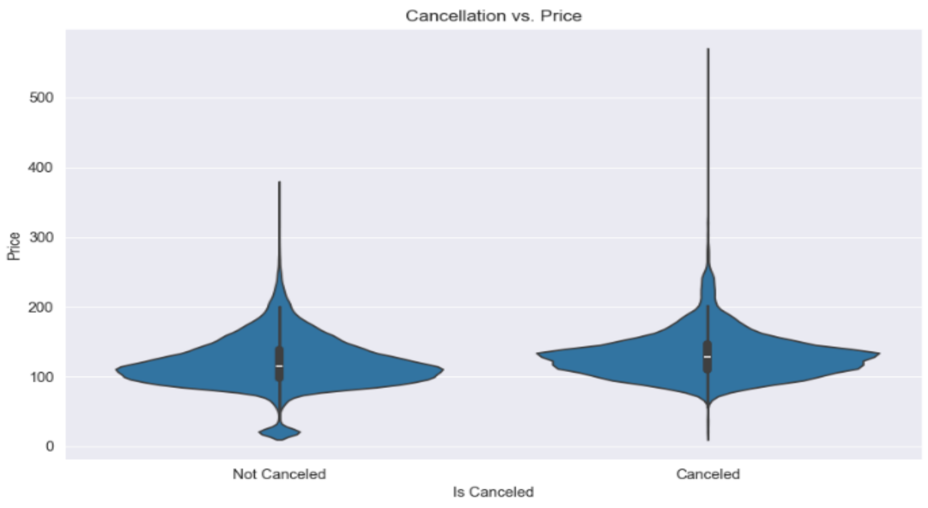
- **Room Type (Positive Correlation):** Price is positively correlated with room type (0.47). Higher quality rooms typically command higher prices.
- **Number of Guests (Positive Correlation):** There is a positive correlation (0.45) between price and the number of guests, as accommodating more guests generally incurs higher costs.
- **Number of Requests (Positive Correlation):** Price is positively correlated with the number of requests (0.18). Fulfilling more requests usually increases the overall cost.
- **Number of Repeated Stays (Negative Correlation):** Price is negatively correlated with the number of repeated stays (-0.18), likely because loyal customers often receive discounts.

Room Type:

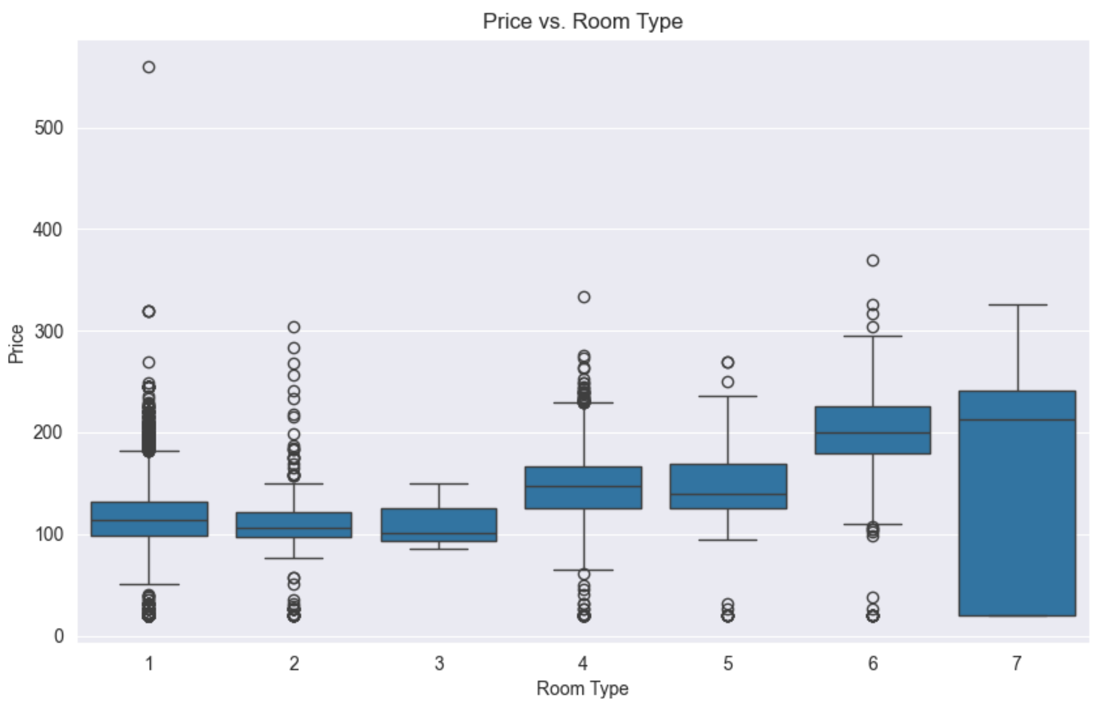
- **Number of Guests (Positive Correlation):** Room type is positively correlated with the number of guests (0.44), as larger rooms are needed to accommodate more guests.

Additional Insights:

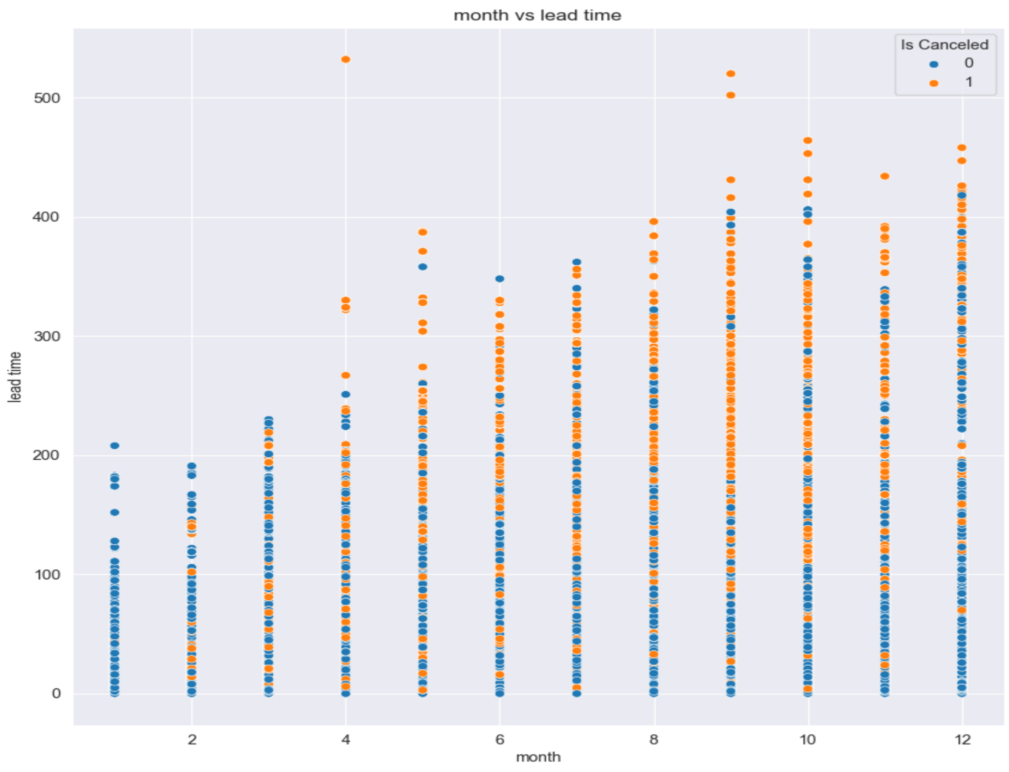
1. **Cancellation vs. Price:** We can see that higher prices are associated with a higher cancellation rate, while lower prices are associated with a lower cancellation rate.



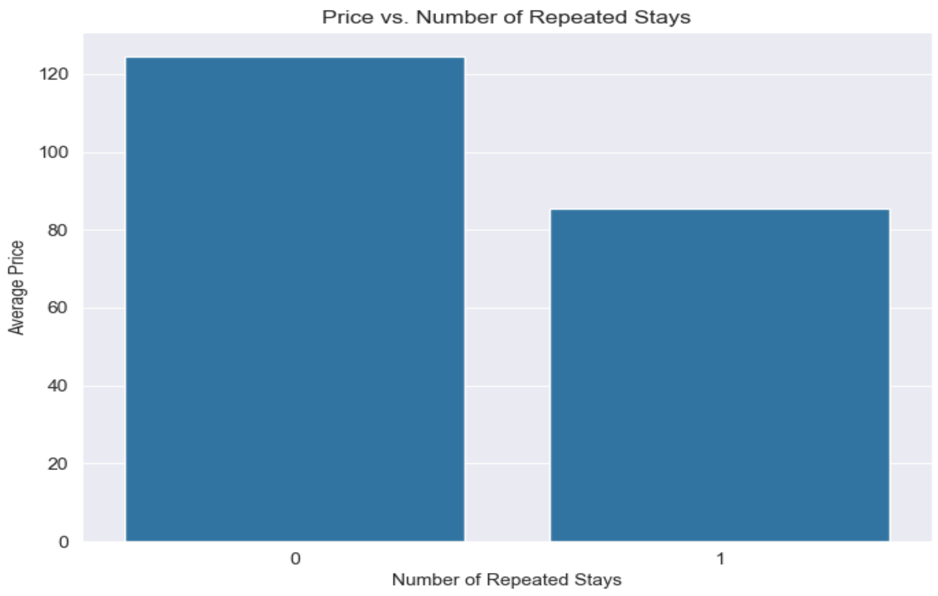
2. **Price vs. Room Type:** It seems that there is positive trend between the room type and the price.



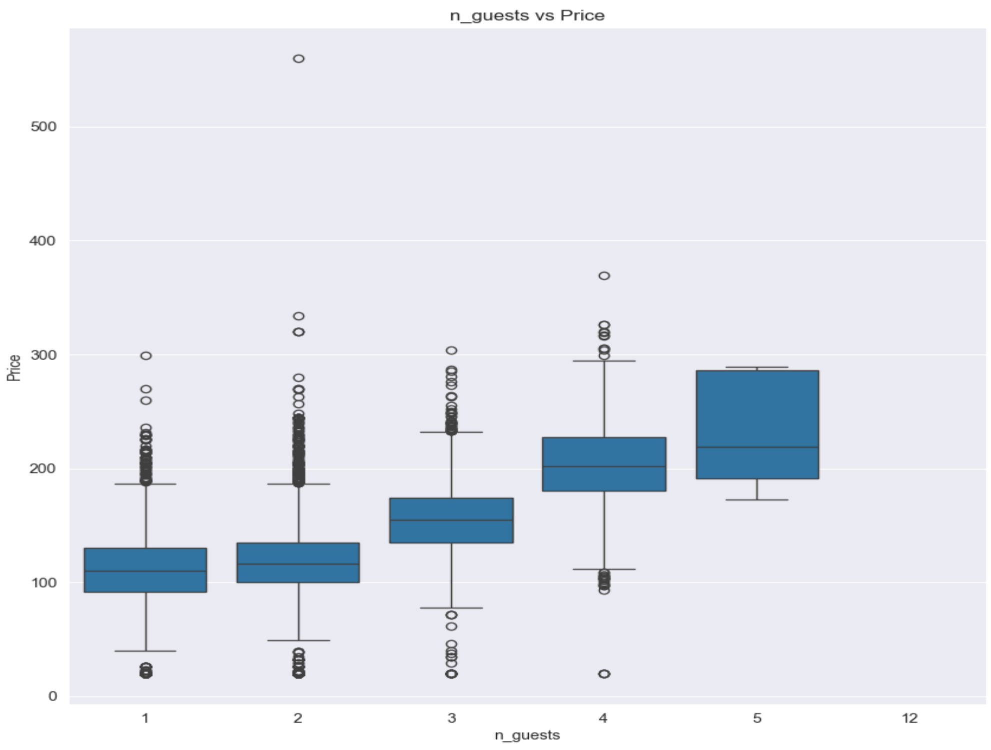
3. **Month vs. Lead Time with Cancellation:** We can observe that during periods of shorter lead times, particularly during holiday seasons, there is a lower tendency for cancellations.



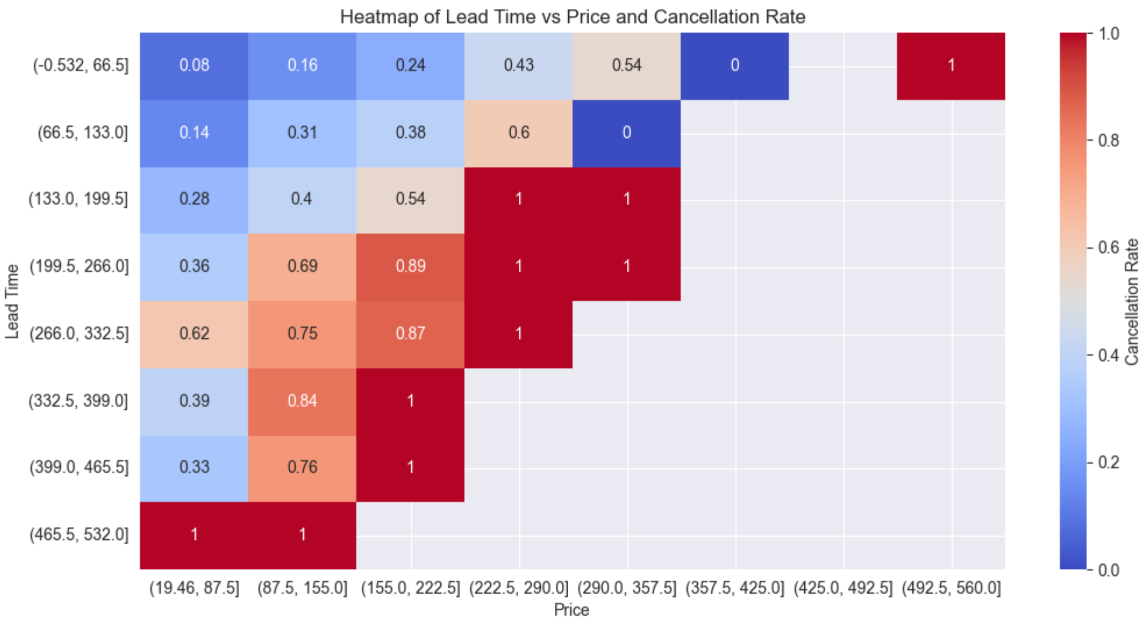
4. **Price vs. Repeated Stays:** The downward trend in the bar plot indicates that customers with more repeated stays tend to pay lower prices, likely due to loyalty discounts.



5. **Price vs. Number of Guests:** This plot illustrates how hotel reservation prices vary based on the number of guests. It shows if pricing changes with the size of the group, providing insights into accommodation costs relative to guest count.

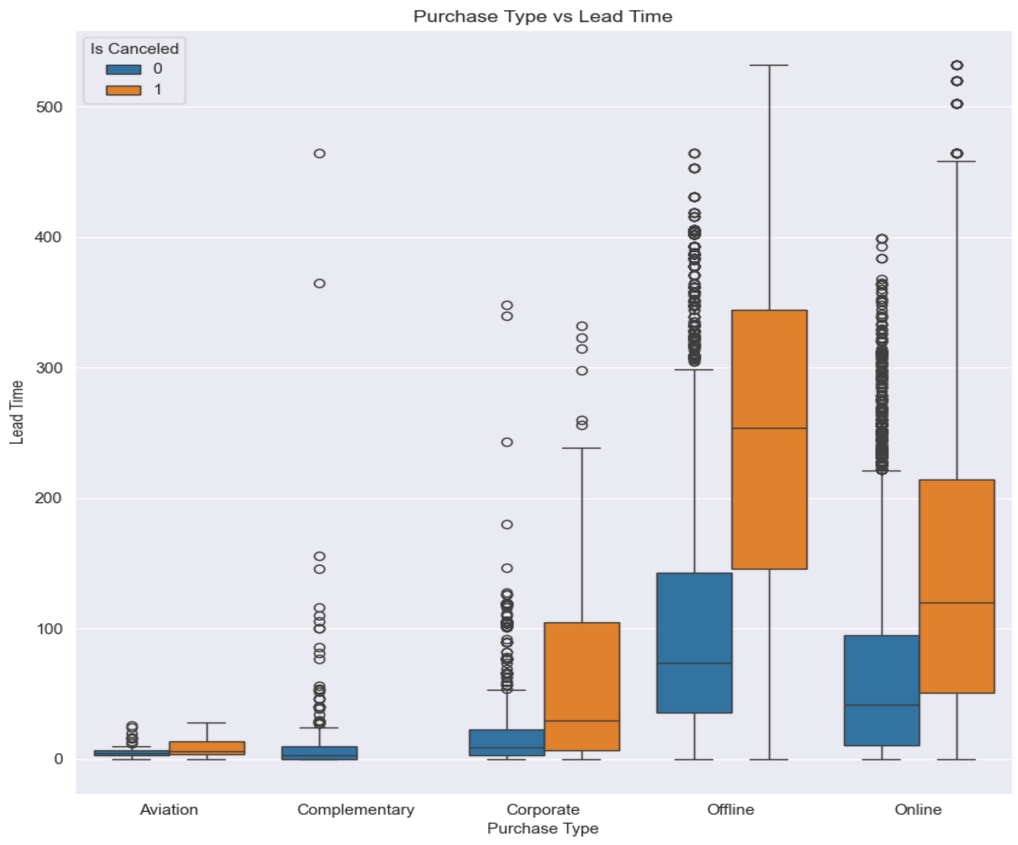


6. **Price Lead Time Trend:** as both the price and lead time increase, the cancellation rate also increases.

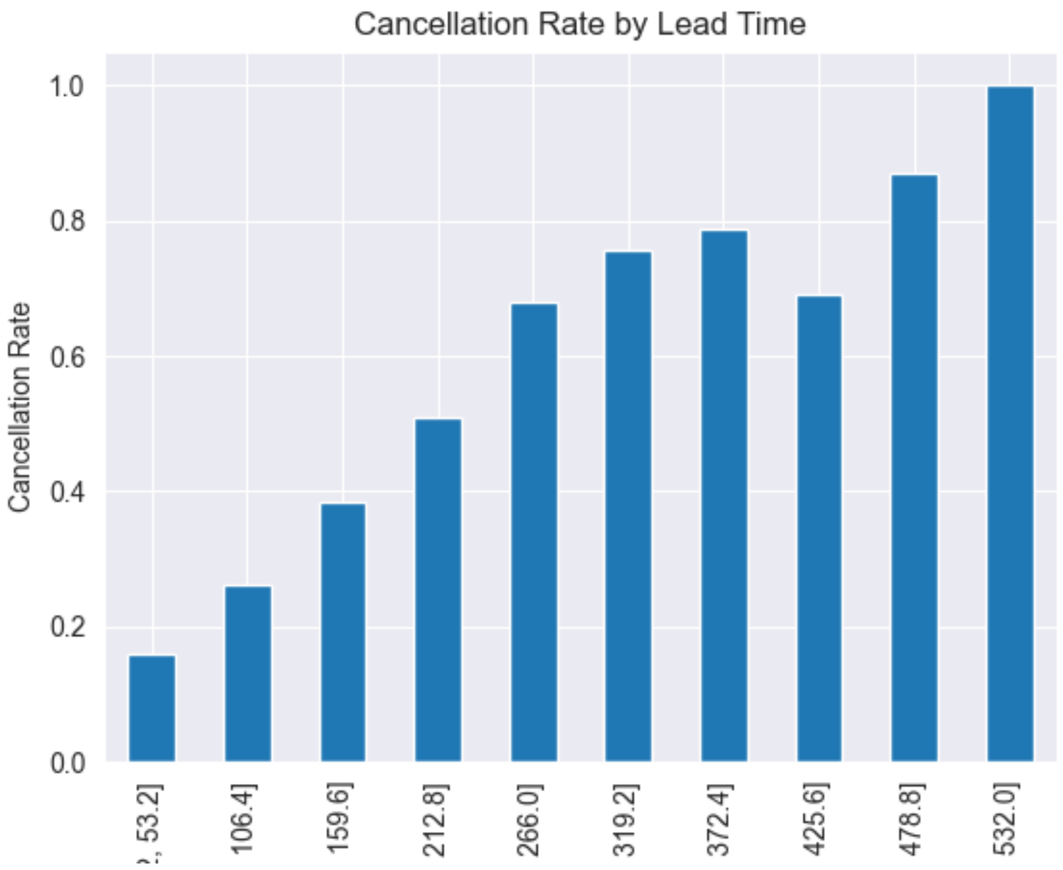


Lead Time	(19.46, 87.5]	(87.5, 155.0]	(155.0, 222.5]	(222.5, 290.0]	(290.0, 357.5]	(357.5, 425.0]	(425.0, 492.5]	(492.5, 560.0]
(-0.532, 66.5]	0.08	0.16	0.24	0.43	0.54	0		1
(66.5, 133.0]	0.14	0.31	0.38	0.6	0			
(133.0, 199.5]	0.28	0.4	0.54	1	1			
(199.5, 266.0]	0.36	0.69	0.89	1	1			
(266.0, 332.5]	0.62	0.75	0.87	1				
(332.5, 399.0]	0.39	0.84	1					
(399.0, 465.5]	0.33	0.76	1					
(465.5, 532.0]	1	1						

7. **Purchase type against Lead time:**
we can see that for the offline users, the lead time is more dominant feature on the cancelation.
we can see that for aviation, the lead time is very low – its means that they uses to order a reservation in the last minute.



8. **Cancelation rate vs lead time:** We can see strong trend.



Data cleaning:

- Reservation Date: Upon reviewing the dataset, we identified inconsistencies in the dates. Specifically, there were entries marked as 29.2.18, which is incorrect as February 2018 does not have a 29th day. Therefore, we corrected these entries by replacing them with 1.3.18.
  Additionally, we observed variations in date formats across the dataset. To standardize the date representation, we utilized pd.to_datetime with the format parameter set to 'mixed'. This ensured uniformity in how dates are interpreted and processed for further analysis.

- Board Type: During our analysis, we discovered that 8168 entries were missing values for the "Board Type" feature. Additionally, there were only 4 rows where this feature was marked as "not selected". To address this, we filled the missing values (NaNs) with "not selected" to ensure completeness in the dataset and maintain consistency in our analysis.

- Purchase Type: Upon inspection, we found that there were 4847 missing values (NaNs) in the "Purchase Type" feature. Analysis revealed that the cancellation rate distribution for these missing values mirrored that of the entire dataset. Therefore, we opted to categorize these missing values under a new category labeled "Missing" to maintain dataset integrity and avoid potential biases in subsequent analyses.

- Price: After observing correlations between the number of guests, room type, and price, we calculated the median price for each category. Initially, we used this information to fill in missing values based on categorical medians. However, some values were still missing. To address this, we proceeded to fill these remaining gaps with the overall median price, ensuring robustness in our dataset for subsequent analyses.

- Lead Time: We encountered 419 missing values for the "Lead Time" feature. Given the relatively small number of missing entries and after examining the data, we decided to fill these missing values with the median lead time. This approach ensures that the dataset remains robust for analysis purposes. Additionally, we considered filling missing values with the "Purchase Type" feature. However, upon inspection, we found that rows lacking lead time values also lacked purchase type values, making this option unfeasible in this context. Therefore, filling with the median lead time was deemed the most appropriate solution.

Added Features

- **n_guests**: Calculated as the sum of n_adults, n_less_12, and n_more_12.
- **total_nights**: Sum of weekend_nights and week_nights.
- **has_kids**: Binary indicator (0 or 1) based on whether n_less_12 or n_more_12 is greater than zero.
- **previous_cancellation_rate**: Ratio of n_p_cacellation to the total of n_p_cacellation and n_p_not_cacellation, rounded to two decimal places.

1. **check_in_date:** dervided from the date and the lead time.

- **check_in_day**: Day of the week derived from check_in_date.
- **check_in_month**: Month of the year derived from check_in_date.
- **Month_order**: Month of the year derived from date

- **repeated**, **board_type**, **purchase_type**, **room_type**, **month_order**, **booked_tour**: Converted to categorical data types for categorical variables.

Removed Features:

- **ID**: Assuming it's a unique identifier.
- **date**: After converting to check_in_date and kept the lead time.
- **check_in_date**: After extracting day and month information.
- **n_p_cacellation**, **n_p_not_cacellation**: After calculating previous_cancellation_rate.

Data Transformation:

In our analysis, we applied both min-max scaling and standardization to normalize the data. We observed that the differences between the two normalization techniques were minimal.

- **Min-Max Scaling:** This method scales features to a specified range (usually 0 to 1). It preserves the distribution of the data but is sensitive to outliers.
- **Standardization:** This technique transforms features to have a mean of 0 and a standard deviation of 1. It is less sensitive to outliers and can handle data with varying ranges effectively.

Despite applying both transformations, the impact on our data analysis did not show significant differences between them. This suggests that for our dataset, both methods provided similar results in terms of scaling the features appropriately for our modeling and analysis tasks.

For the categorial features, we converted them to Boolean representations using one-hot encoding.

Test Prerocess

When implementing the preprocessing steps on the attached test dataset, we ensured consistency with the steps applied to the training dataset. Here's a summary of the adjustments made during the preprocessing phase.

By implementing these preprocessing adjustments consistently across both the training and test datasets, we ensured that our models would operate effectively and provide reliable predictions based on standardized and properly prepared data.

Evaluation Metrics

For predicting hotel reservation cancellations, we think that the most appropriate evaluation metric would be:

F1 Score

The F1 Score is the harmonic means of precision and recall. It provides a balanced measure, which is particularly useful when there is an uneven class distribution between cancellations and non-cancellations.

$$F_1 score = 2\ \frac{Precision *\ Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Explanation: The F1 Score ensures that both false positives and false negatives are considered. This is critical in the context of hotel reservations, as both predicting too many cancellations (leading to unnecessary interventions) and missing actual cancellations (leading to unexpected vacancies) can be costly for the hotel.
In practice, to develop a more precise approach, we need detailed information on how cancellations impact the hotel's revenue based on the timing and price of the booking. Additionally, understanding how the hotel can act differently with this knowledge is crucial. With this information, we can theoretically build a cost-loss matrix.

In the Exercise we chose Accuracy as this is the metric that we are examined on.

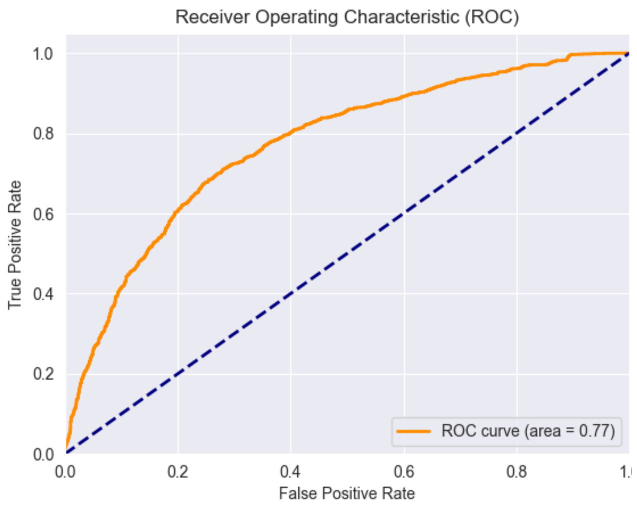
Training Classifier

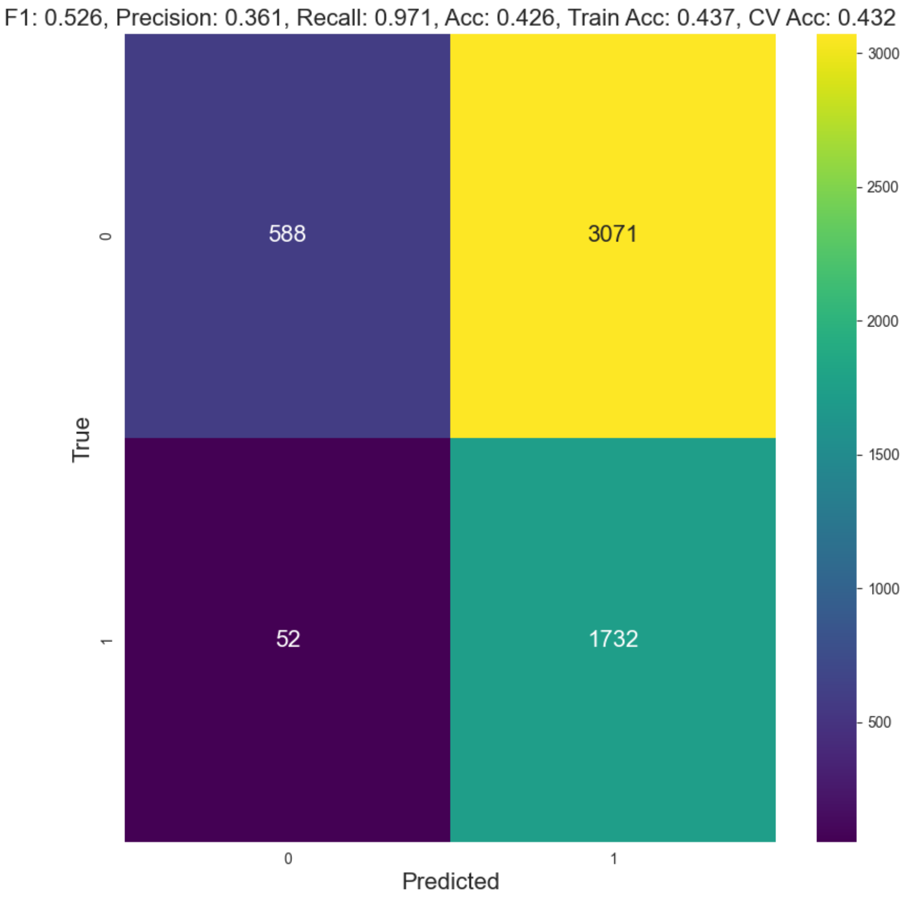
We split the data to 80% Train and 20% Validation with stratify.

Naïve Bayes

We trained GaussianNB and MultinominalNB, with cross validation of 5 folds.

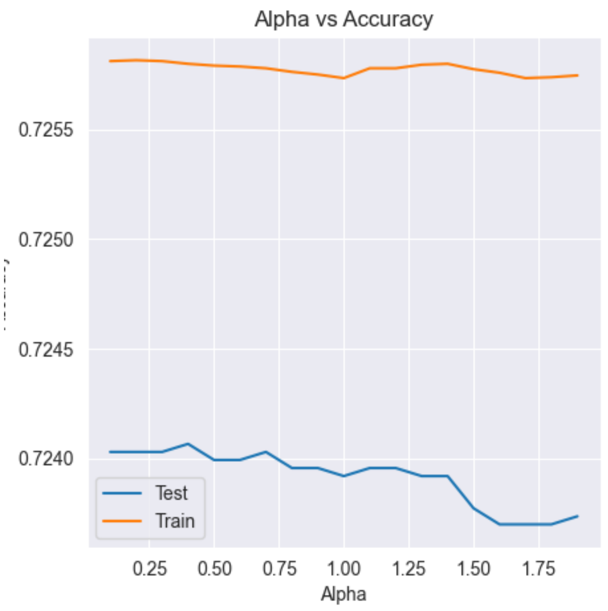
GaussianNB is a variant of the Naive Bayes algorithm tailored for continuous data. It assumes that the features follow a Gaussian distribution. This means that for each class, the distribution of the feature values is modeled using a normal distribution. It is particularly useful for problems where the input features are continuous and can be approximated by a Gaussian distribution.

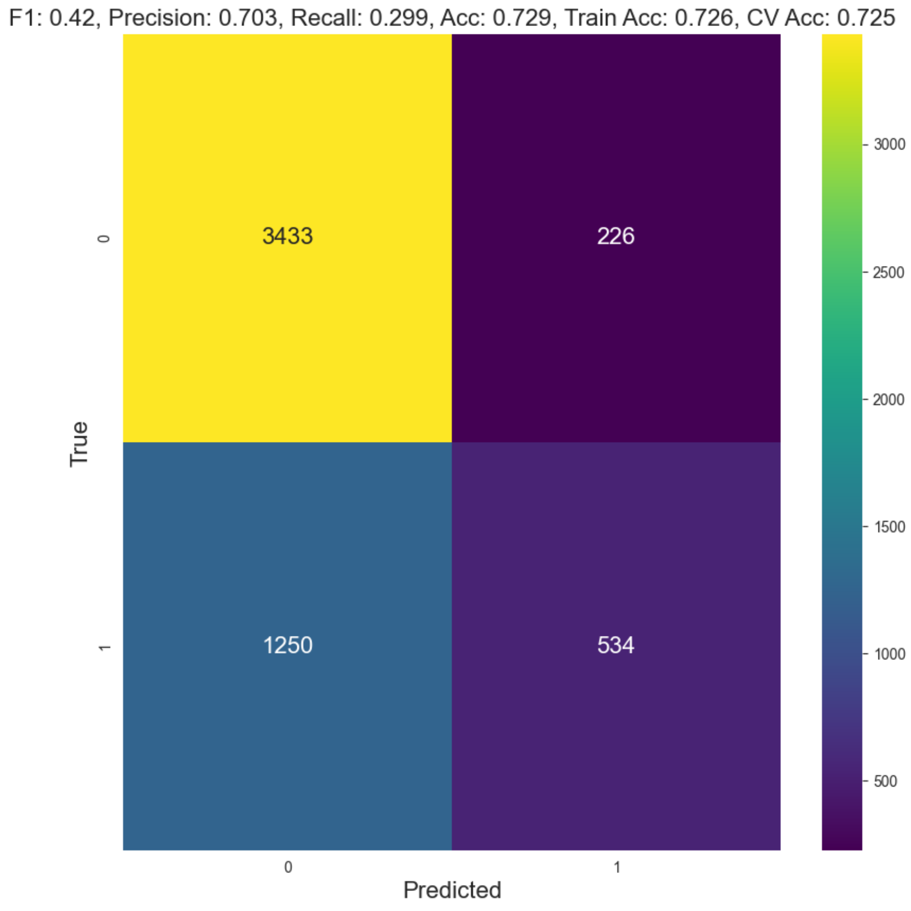




We observed that the model achieved high recall but low precision. This indicates that the model tends to classify most samples as 'cancel', suggesting that it did not effectively capture the underlying patterns in the data.

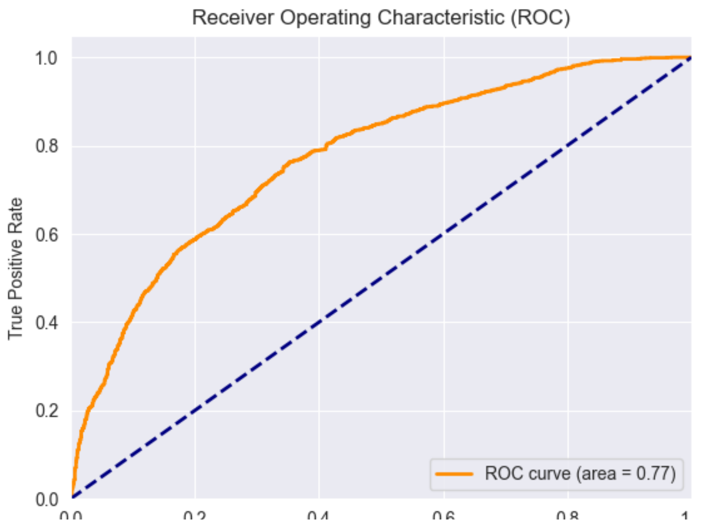
MultinomialNB is a variant of the Naive Bayes algorithm designed for discrete data, especially for text classification problems where word frequencies are used as features. It assumes that the features are drawn from a multinomial distribution, which is appropriate for count data. Moreover, it uses Laplacian





Smoothing. By adjusting the alpha parameter, we can play with the algorithm and find the optimal value.

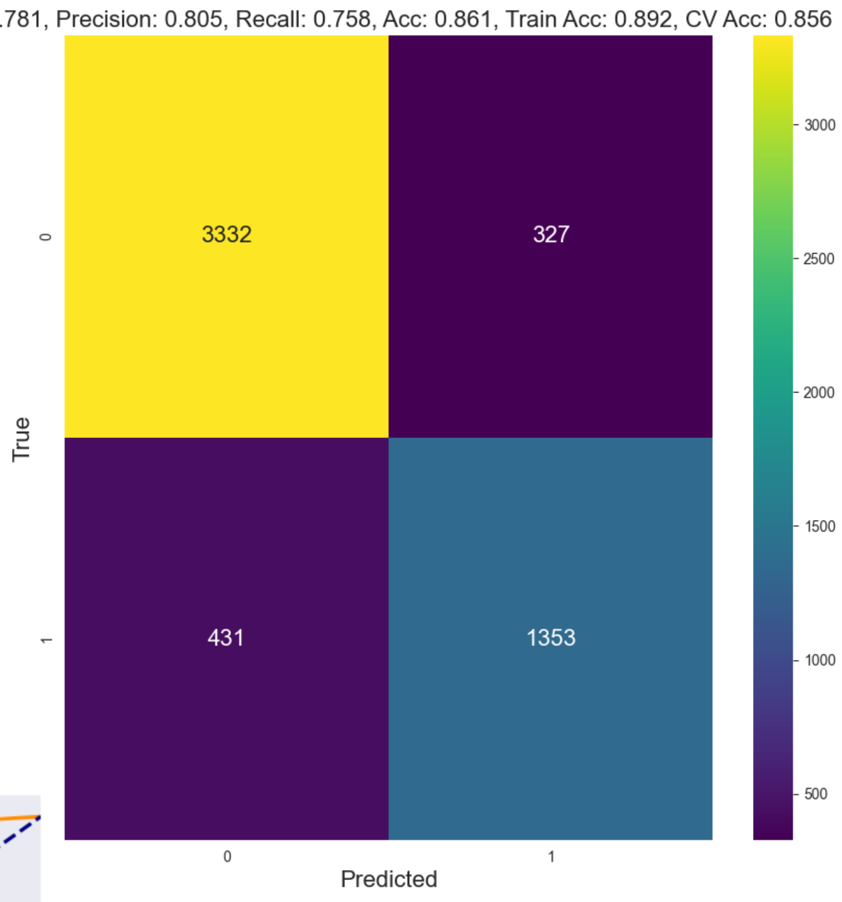
It seems that the MultinominalNB takes the other direction from the GaussianNB: its tend to classify more as not cancelled. Better accuracy but less f1 score and Recall. Same ROC.

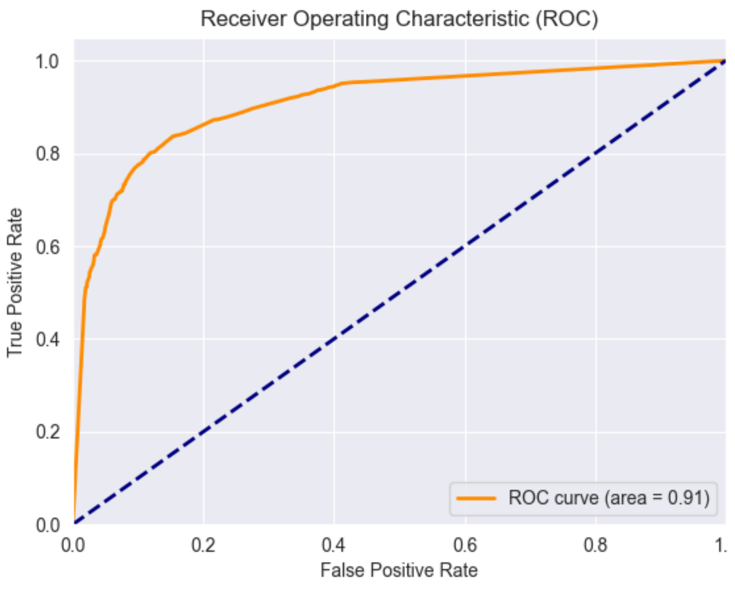


Decision Tree

We performed grid search with several hyperparameters on the decision tree model, (with cross validation) including criterion (gini or entropy) max depth, min samples split and min samples leaf, and the best model is with
{'criterion': 'entropy',
'max_depth': 15,
'min_samples_leaf': 5,
'min_samples_split': 15}

The Decision Tree outperformed the NB.





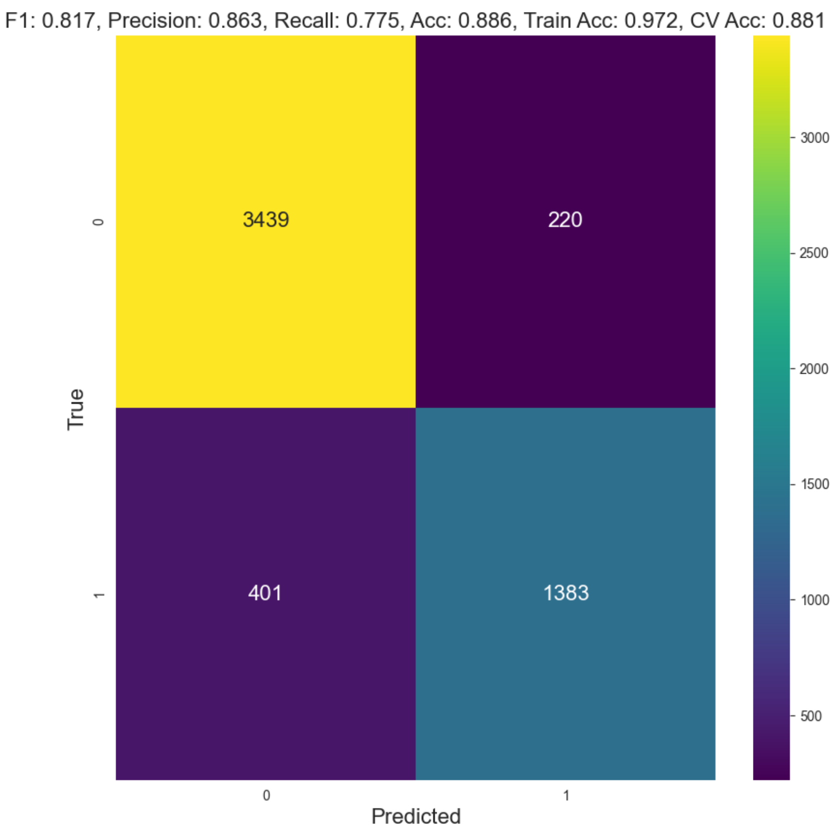
XGBoost

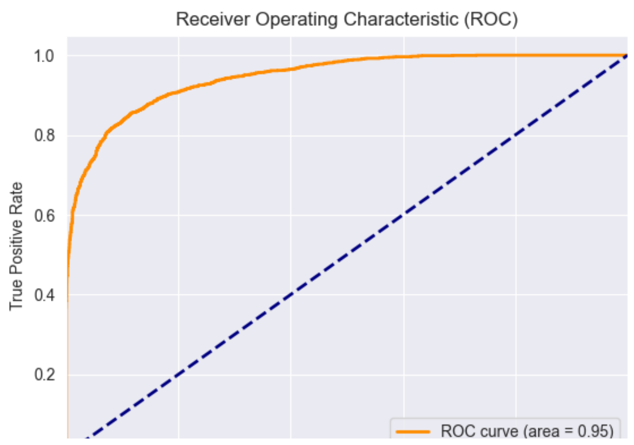
XGBoost (Extreme Gradient Boosting) is an advanced implementation of gradient boosting designed for speed and performance. It builds an ensemble of decision trees sequentially, where each new tree corrects errors from the previous ones. XGBoost includes features like parallel processing, regularization (to prevent overfitting), and efficient handling of missing data. It's widely used for its accuracy and scalability.

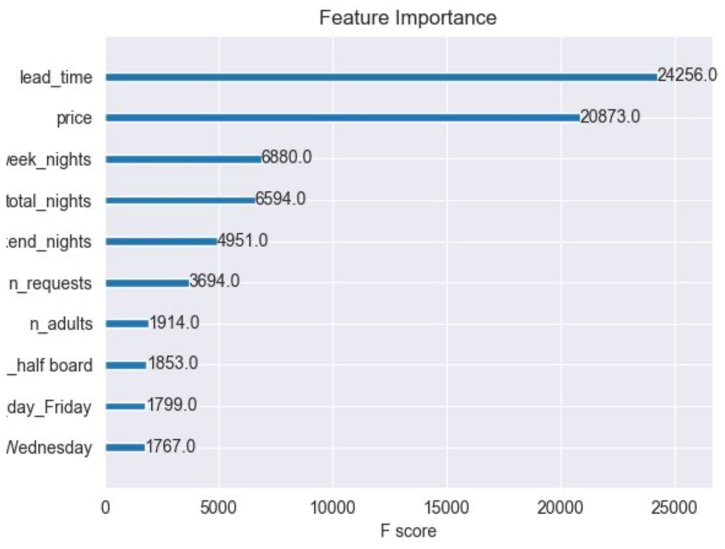
We run random search over 100 different hyperparameters, with cv of 5:

Hyperparameters:

- `max_depth`: 11
- `eta`: 0.025
- `n_estimators`: 722
- `gamma`: 0.29
- `min_child_weight`: 1
- `subsample`: 0.9178
- `colsample_bytree`: 0.984
- `lambda`: 0.505
- `alpha`: 0.46







The XGB is better than Decision Tree in all the Evaluation Metrics.

The Feature Importance are very reasonable, as we observed in the data statistic stage, the lead and the price are the most significant features.

XGB with SMOTEENN:

We identified that the classes are imbalanced, the majority is no cancelation.

SMOTEENN (SMOTE-Tomek) is a hybrid method for dealing with class imbalance in datasets, particularly in classification problems. It combines two techniques: SMOTE (Synthetic Minority Over-sampling Technique) and Tomek links (a method for undersampling).

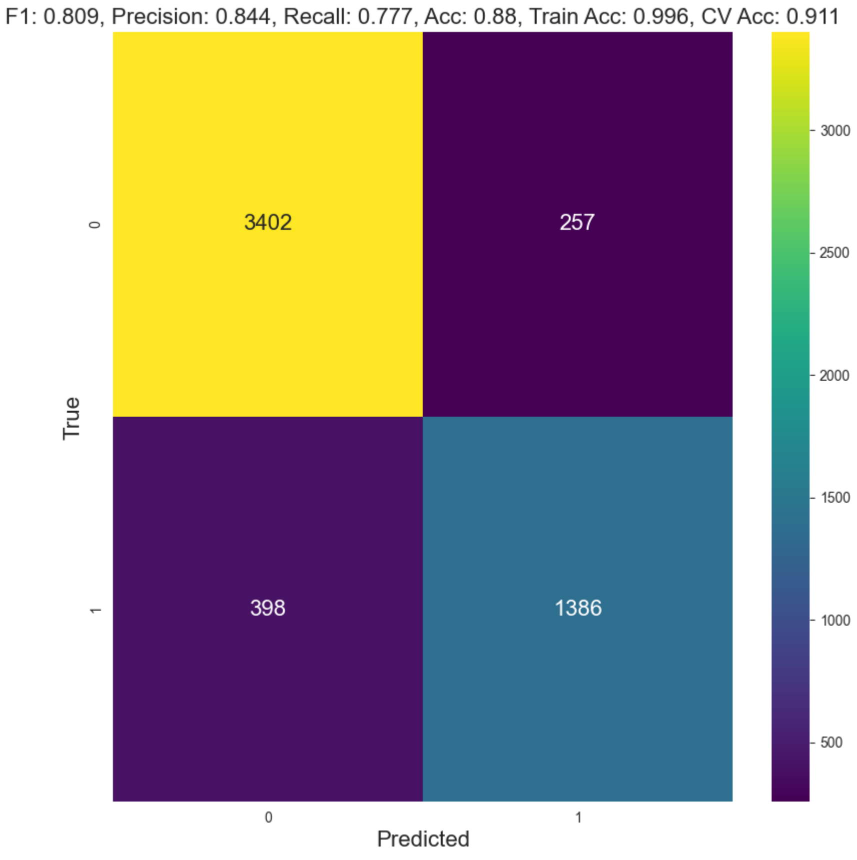
SMOTE - An over-sampling technique that generates synthetic samples for the minority class.

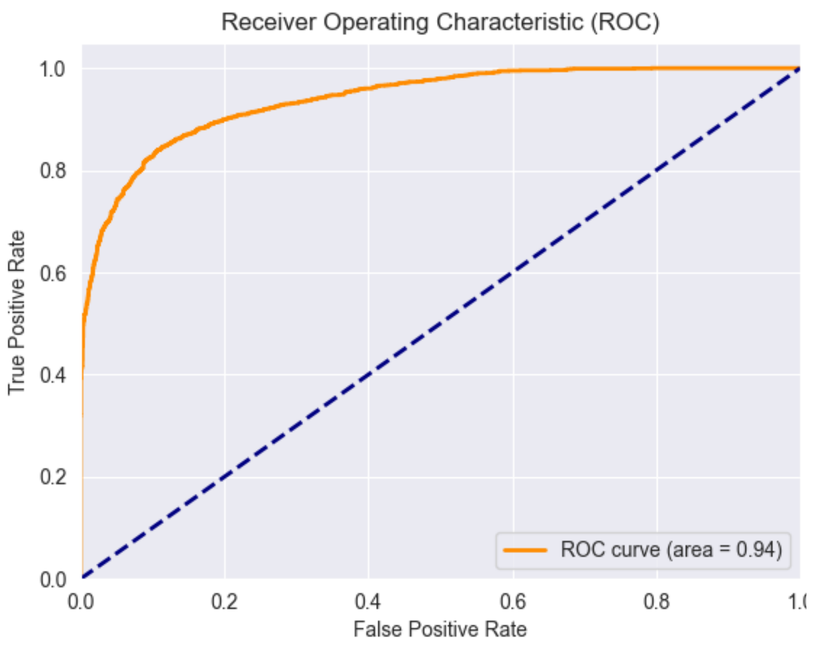
Tomek Links - A method for cleaning the dataset by identifying and removing borderline examples that are likely to be noise.

SMOTEENN - combines SMOTE and Tomek links to perform both over-sampling of the minority class and under-sampling of the majority class.

By combining these techniques, SMOTEENN can improve the quality of the synthetic samples and reduce the noise in the dataset and should leading to better performance for classification models.

Applying SMOTETomek to the dataset aimed to address class imbalance by combining SMOTE's oversampling with Tomek links' undersampling. However, the subsequent XGBoost model did not show an





improvement in results, indicating that this approach may not have effectively enhanced the model's performance.

statistical significance

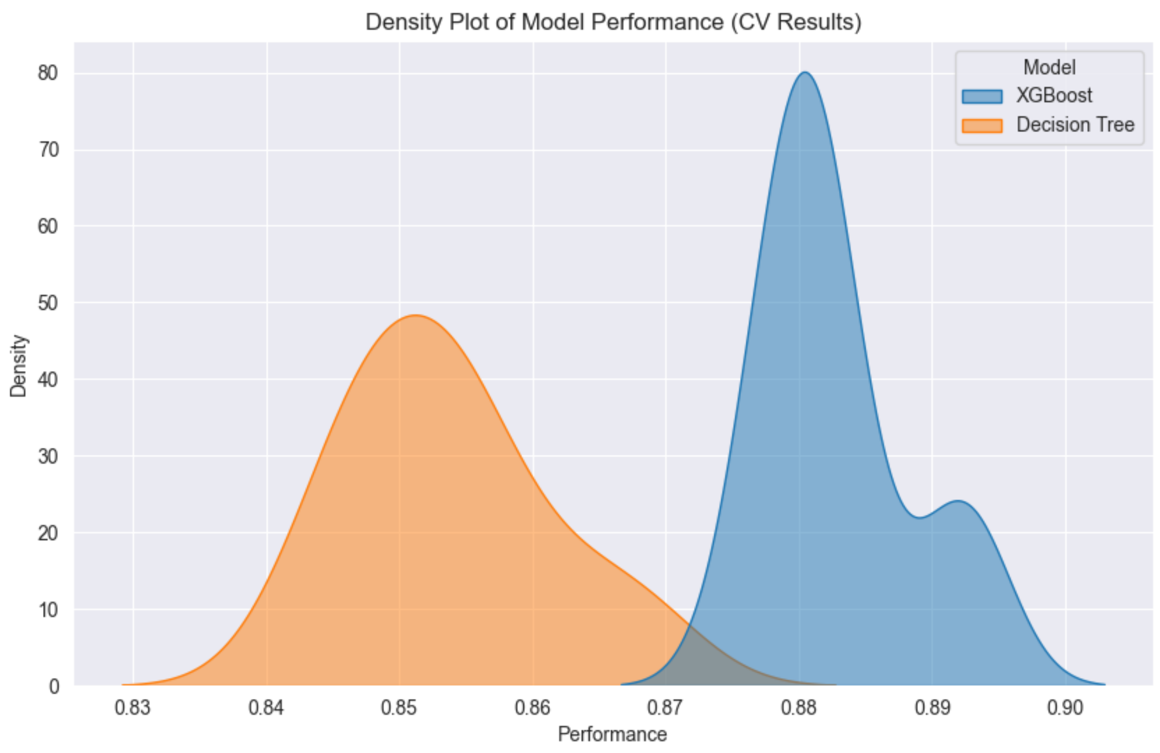
Two-sided test for the null hypothesis that two related or repeated samples have identical average (expected) values.

T-Statistic: 20.5708

P-Value: 0.000000007083

Therefore, we reject the null hypothesis.

We compared the best XGBoost model (without imbalance fixing) against the best decision tree model using 10-fold cross-validation. We calculated the P-value and found it to be extremely low, indicating a significant difference between the performances of the two models. This suggests that the XGBoost model performs significantly better than the decision tree in this scenario.



Finally, we trained again the xgb model on all the data and predicted the test and we saved to csv file.(called hotel_tests_predictions.csv)