BIA | BOSTON INSTITUTE OF ANALYTICS ®

SPINNAKER ANALYTICS

Predicting Hotel Reservation Cancellations

Predicting Hotel Reservation Cancellations

Objective :

To develop and compare machine learning models that can accurately predict hotel reservation cancellations, enabling hotels to take proactive measures for revenue optimization, efficient resource allocation, and improved customer retention.

Data Preprocessing :

Handling missing values (mode imputation for categorical features)
Label encoding for categorical variables
Feature scaling (if needed for some models)

Business Impact :

Revenue Protection ⟶ Predicting cancellations lets hotels overbook strategically or offer discounts to fill rooms, reducing revenue loss.Cost Optimization ⟶ Better staff scheduling and inventory planning saves operational costs.Customer Retention ⟶ Identify guests likely to cancel and offer personalized incentives to keep them.Strategic Decision-Making ⟶ Data-driven insights improve marketing strategies and pricing models.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Introduction

Hotel reservation cancellations are a major challenge in the hospitality industry, leading to significant revenue loss, resource wastage, and operational inefficiencies. Predicting cancellations in advance enables hotels to take proactive measures such as dynamic pricing, targeted promotions, or overbooking strategies to minimize the negative impact.This project, Predicting Hotel Reservation Cancellations, leverages data analytics and machine learning techniques to identify patterns and key factors influencing booking cancellations. By building and evaluating robust predictive models, the project aims to assist hotel management in making informed decisions, improving resource allocation, and enhancing overall customer satisfaction.

- Hotel booking cancellations can cause revenue loss and resource mismanagement.
- Predicting cancellations in advance allows for dynamic pricing and overbooking strategies.
- Aims to improve operational efficiency and customer satisfaction.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

THE WORK FLOW OF THE PROJECT IS GIVEN BELOW :

- Importing Important Libraries.
- Data Overview.
- EDA (Exploratory Data Analysis).
- Data Preprocessing.
- Modeling.
- Evaluation.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Project Goal

To build accurate and interpretable machine learning models that can predict hotel reservation cancellations, enabling hotels to take proactive measures for maximizing revenue, optimizing resources, and enhancing customer satisfaction.

Key Goals:

- Identify important factors influencing booking cancellations.
- Compare different classification models (Logistic Regression, Decision Tree, Random Forest).
- Provide actionable insights for hotel management to reduce cancellations.
- Deliver easy-to-understand visualizations and a well-documented solution.

Project Plan

Data Understanding

• Load dataset and check its structure, data types, and missing values.

Exploratory Data Analysis (EDA)

• Perform Exploratory Data Analysis (EDA) to understand patterns and correlations..

• Visualize relationships between features and the target

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Feature Engineering

- Create new features (e.g., weekend booking flag, total guests).
- Select important features based on correlation and domain knowledge.

Model Development

Train classification model :

- Logistic Regression
- Decision Tree
- Random Forest

Model Evaluation

Here's a complete **Model Evaluation code** for **classification problems** using Scikit-learn. This includes:

✅ Accuracy
✅ Confusion Matrix
✅ Classification Report

Importing Important Libraries

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import classification_report
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Data Overview

```
df.head()
```

	Booking_ID	no_of_adults	no_of_children	no_of_weekend_nights	no_of_week_nights	type_of_meal_plan	required_car_parking_space	room_type_reserved	lead_time	a
0	INN00001	2	0	1	2	Meal Plan 1	0	Room_Type 1	224	
1	INN00002	2	0	2	3	Not Selected	0	Room_Type 1	5	
2	INN00003	1	0	2	1	Meal Plan 1	0	Room_Type 1	1	
3	INN00004	2	0	0	2	Meal Plan 1	0	Room_Type 1	211	
4	INN00005	2	0	1	1	Not Selected	0	Room_Type 1	48	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36275 entries, 0 to 36274
Data columns (total 19 columns):
 #   Column                                Non-Null Count  Dtype  
---  ------                                --------------  -----  
 0   Booking_ID                            36275 non-null  object 
 1   no_of_adults                          36275 non-null  int64  
 2   no_of_children                        36275 non-null  int64  
 3   no_of_weekend_nights                  36275 non-null  int64  
 4   no_of_week_nights                     36275 non-null  int64  
 5   type_of_meal_plan                     36275 non-null  object 
 6   required_car_parking_space            36275 non-null  int64  
 7   room_type_reserved                    36275 non-null  object 
 8   lead_time                             36275 non-null  int64  
 9   arrival_year                          36275 non-null  int64  
 10  arrival_month                         36275 non-null  int64  
 11  arrival_date                          36275 non-null  int64  
 12  market_segment_type                   36275 non-null  object 
 13  repeated_guest                        36275 non-null  int64  
 14  no_of_previous_cancellations          36275 non-null  int64  
 15  no_of_previous_bookings_not_canceled  36275 non-null  int64  
 16  avg_price_per_room                    36275 non-null  float64
 17  no_of_special_requests                36275 non-null  int64  
 18  booking_status                        36275 non-null  object 
dtypes: float64(1), int64(13), object(5)
memory usage: 5.3+ MB
```

Data Summary

- Rows: 19,158 entries (records).
- Columns: 14 features in total.

Data Types:

- **2 numerical integer columns (int64)**
- **1 numerical float column (float64)**
- **10 categorical/text columns (object)**
- **1 target column (float64)**

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

```
df.isna().sum()        # Check null values in coulmns
```

```
Booking_ID                            0
no_of_adults                          0
no_of_children                        0
no_of_weekend_nights                  0
no_of_week_nights                     0
type_of_meal_plan                     0
required_car_parking_space            0
room_type_reserved                    0
lead_time                             0
arrival_year                          0
arrival_month                         0
arrival_date                          0
market_segment_type                   0
repeated_guest                        0
no_of_previous_cancellations          0
no_of_previous_bookings_not_canceled  0
avg_price_per_room                    0
no_of_special_requests                0
booking_status                        0
dtype: int64
```

FIND NULL VALUES .

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Exploratory Data Analysis (EDA)

```
df.describe()
```

	no_of_adults	no_of_children	no_of_weekend_nights	no_of_week_nights	required_car_parking_space	lead_time	arrival_year	arrival_month	arrival_date	rep
count	36275.000000	36275.000000	36275.000000	36275.000000	36275.000000	36275.000000	36275.000000	36275.000000	36275.000000	3
mean	1.844962	0.105279	0.810724	2.204300	0.030986	85.232557	2017.820427	7.423653	15.596995	
std	0.518715	0.402648	0.870644	1.410905	0.173281	85.930817	0.383836	3.069894	8.740447	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2017.000000	1.000000	1.000000	
25%	2.000000	0.000000	0.000000	1.000000	0.000000	17.000000	2018.000000	5.000000	8.000000	
50%	2.000000	0.000000	1.000000	2.000000	0.000000	57.000000	2018.000000	8.000000	16.000000	
75%	2.000000	0.000000	2.000000	3.000000	0.000000	126.000000	2018.000000	10.000000	23.000000	
max	4.000000	10.000000	7.000000	17.000000	1.000000	443.000000	2018.000000	12.000000	31.000000	

```
df.columns
```

```
Index(['Booking_ID', 'no_of_adults', 'no_of_children', 'no_of_weekend_nights',
       'no_of_week_nights', 'type_of_meal_plan', 'required_car_parking_space',
       'room_type_reserved', 'lead_time', 'arrival_year', 'arrival_month',
       'arrival_date', 'market_segment_type', 'repeated_guest',
       'no_of_previous_cancellations', 'no_of_previous_bookings_not_canceled',
       'avg_price_per_room', 'no_of_special_requests', 'booking_status'],
      dtype='object')
```

```
avg_price_perroom = df['avg_price_per_room'].mean()
```

```
avg_price_perroom
```

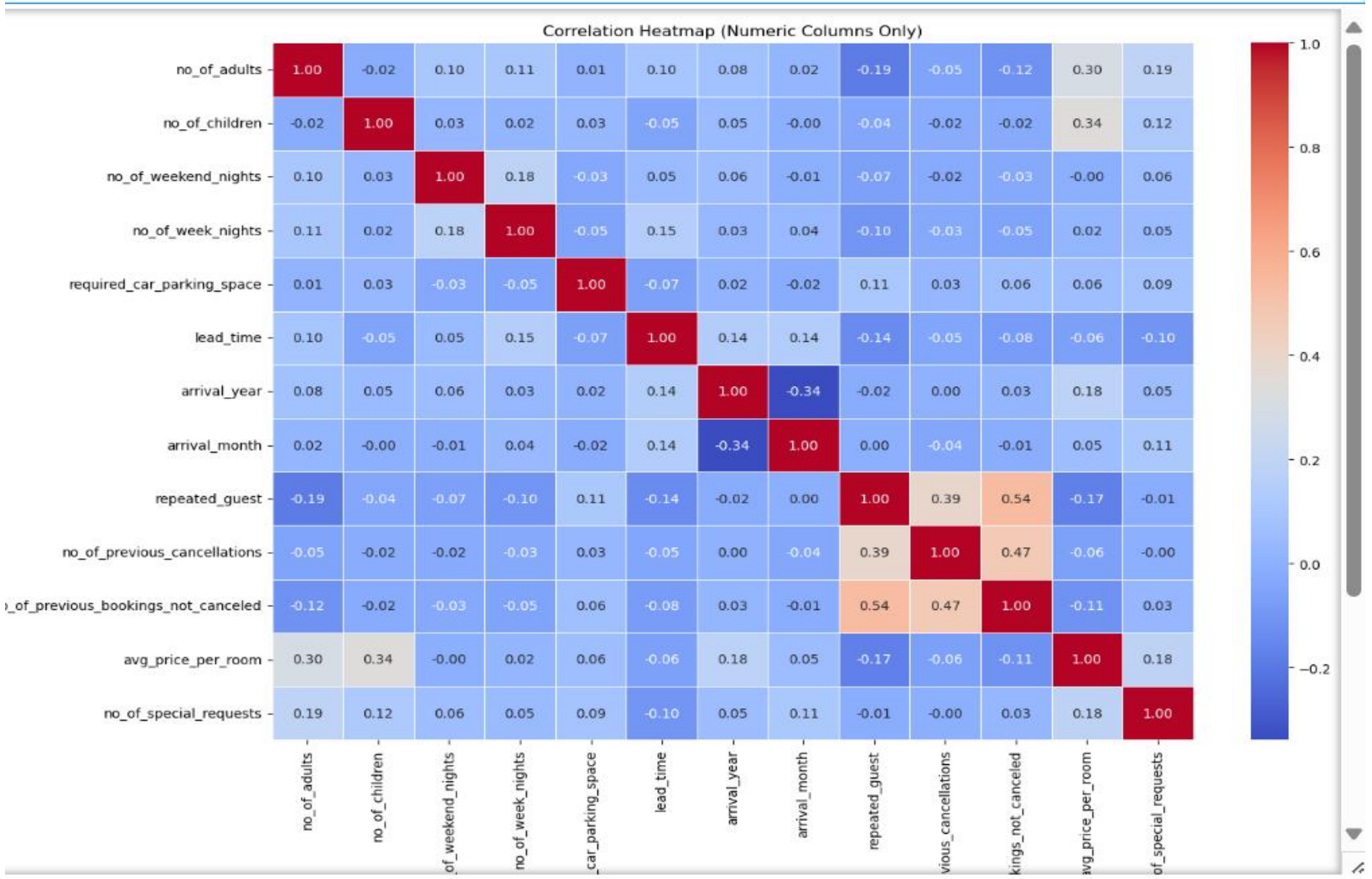
```
np.float64(103.42353907649897)
```

```
# केवल Numeric Columns से DataFrame बनाएं
numeric_df = df.select_dtypes(include=['int64', 'float64'])

# अब Correlation निकालें
correlation_matrix = numeric_df.corr()

# Heatmap बनाएँ


plt.figure(figsize=(15,10))
sns.heatmap(correlation_matrix, cmap='coolwarm', annot=True, fmt='.2f', linewidths=0.5)
plt.title("Correlation Heatmap (Numeric Columns Only)")
plt.show()
```

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Correlation Heatmap (All Numeric Columns only)



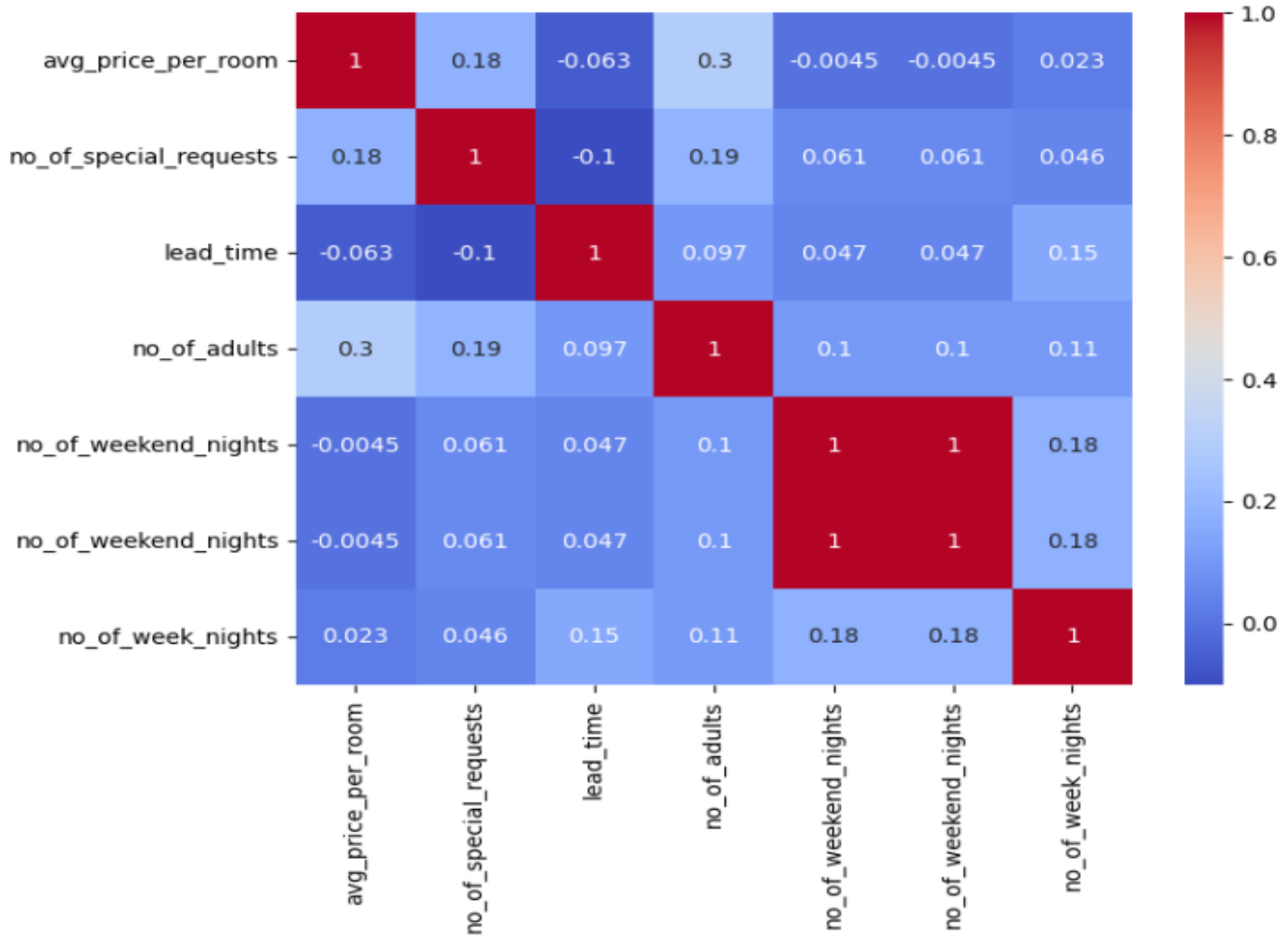
SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Specific Columns Heatmap

```
plt.figure(figsize=(8,6))
sns.heatmap(corr_data.corr(), annot=True, cmap='coolwarm')
```

```
<Axes: >
```



Seaborn Correlation Heatmap

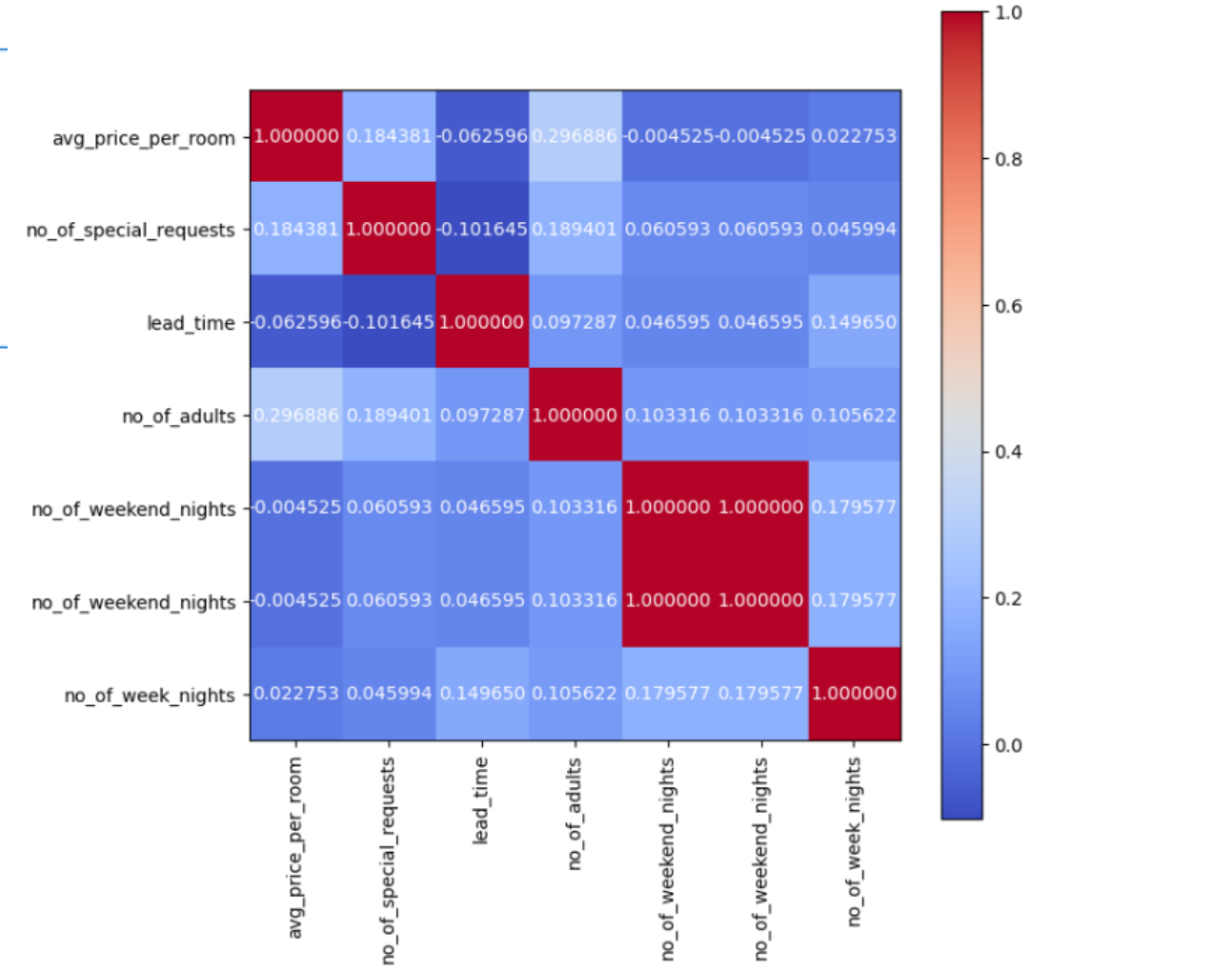
SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Matplotlib Correlation Heatmap

Matplotlib Correlation Heatmap

```
44]: plt.imshow(corr_data.corr(),cmap='coolwarm', interpolation='none')
plt.colorbar()
plt.xticks(range(len(corr_data.columns)), corr_data.columns, rotation=90)
plt.yticks(range(len(corr_data.columns)), corr_data.columns)
plt.gcf().set_size_inches(8,8)

labels = corr_data.corr().values
for y in range(labels.shape[0]):
    for x in range (labels.shape[1]):
        plt.text(x, y, '{:2f}'.format(labels[y, x]), ha='center', va='center', color='white')
```



SPINNAKER ANALYTICS

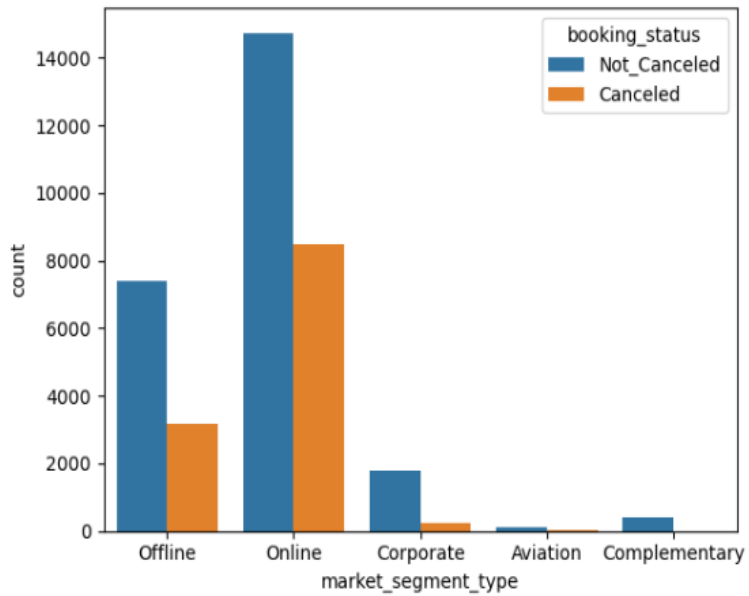
BIA BOSTON INSTITUTE OF ANALYTICS ®

Plots :

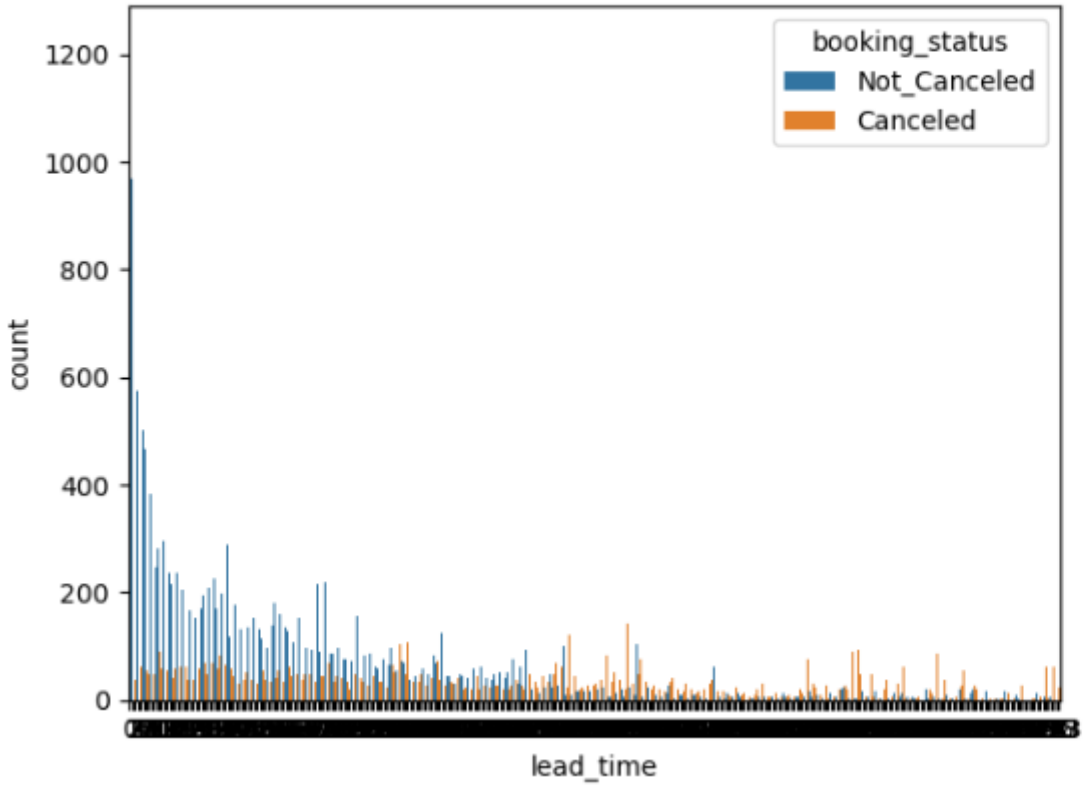
- **Count plots**

```
Name: count, dtype: int64
```

```
sns.countplot(x=df['market_segment_type'], hue=df['booking_status'])
plt.show()
```




```
sns.countplot(x=df['lead_time'], hue=df['booking_status'])
plt.show()
```



SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Market Segment Type vs Booking Status (Left Plot)

•Observation:

- **Online segment has the highest number of bookings, both canceled and not canceled.**
- **Offline bookings are fewer, but cancellations are lower compared to online.**
- **Corporate bookings have fewer cancellations.**
- **Aviation and Complementary bookings are rare.**

Online bookings tend to have a higher cancellation rate than offline or corporate bookings.This could indicate that online customers are more likely to change plans or have more flexible cancellation options.

Lead Time vs Booking Status (Right Plot)

Observation:

- **Most bookings have short lead times (closer to the check-in date).**
- **Cancellation rates are higher when the lead time is long (booked far in advance).**
- **As lead time increases, the total number of bookings decreases.**

X-axis: Different market segment types (Offline, Online, Corporate, Aviation, Complementary).

Y-axis: Number of bookings.Colors (hue):Blue ⟶ Not_Canceled bookings

Orange ⟶ Canceled bookings

Why This is Important:

Business Importance

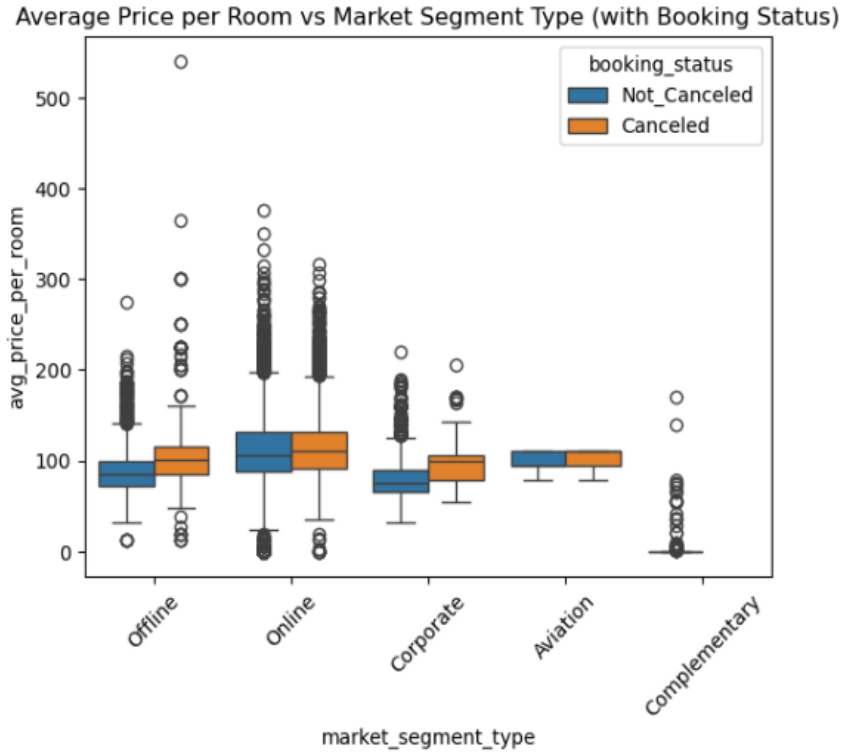
- **Targeted retention strategies: If online bookings have high cancellations, hotels can focus on stricter policies or better incentives for online customers.**
- **Revenue forecasting: Understanding which segments cancel more helps in more accurate occupancy predictions and pricing.**
- **Marketing focus: Low-cancellation segments (e.g., corporate) might deserve more sales efforts because they're reliable revenue sources.**

What You Can Gather from It:

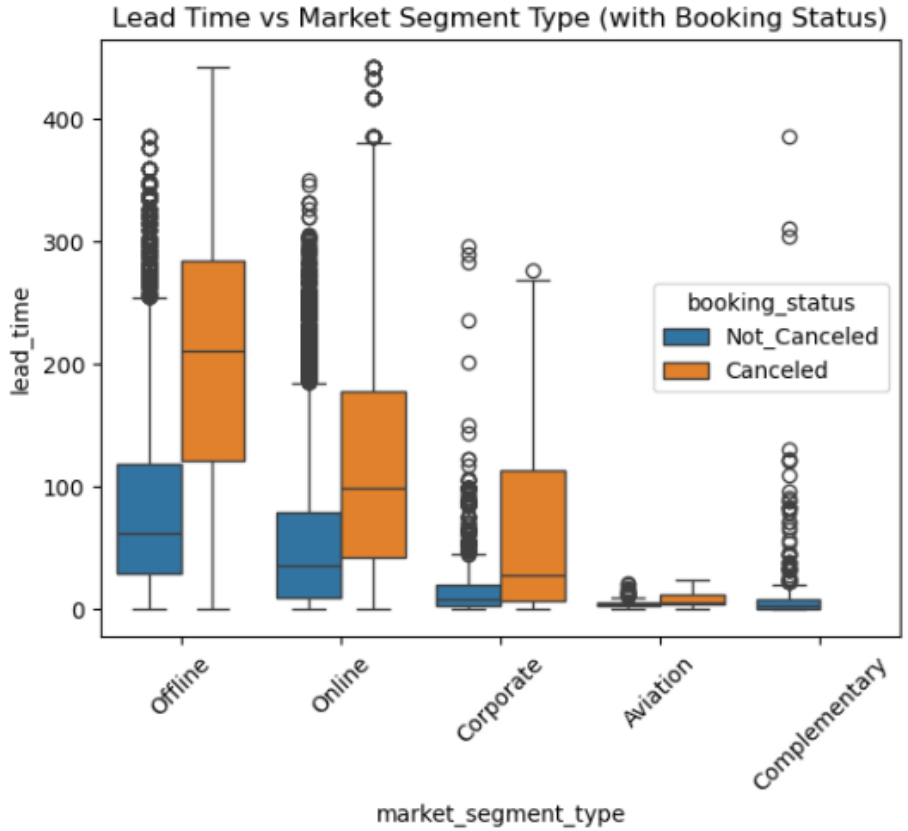
- Booking patterns vary by market segment – Online and Offline segments dominate bookings, while Corporate, Aviation, and Complementary are much smaller.
- Cancellation trends differ by segment –Online bookings: High volume, but also a significant cancellation rate.Offline bookings: Fewer cancellations compared to online.Corporate bookings: Very low cancellation rate.Aviation & Complementary: Small volumes, minimal cancellations.
- Why this is important –Helps identify which customer segments are most likely to cancel.Guides targeted strategies, like offering flexible policies for high-risk groups (e.g., online customers) or incentives for lower-risk ones.Useful for resource planning and revenue forecasting in the hotel industry.

- **Box Plots**

```
sns.boxplot(x='market_segment_type', y='avg_price_per_room', hue='booking_status', data=df)
plt.xticks(rotation=45)
plt.title("Average Price per Room vs Market Segment Type (with Booking Status)")
plt.show()
```




```
sns.boxplot(x='market_segment_type', y='lead_time', hue='booking_status', data=df)
plt.xticks(rotation=45)  # X-axis labels को घुमाना (optional)
plt.title("Lead Time vs Market Segment Type (with Booking Status)")
plt.show()
```



SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

What's happening in the plot:

1. **Market segments on X-axis**: Offline, Online, Corporate, Aviation, Complementary.
2. **Lead time on Y-axis**: Number of days before arrival when the booking was made.
3. **Color coding**:

- **Blue** = Not Canceled
- **Orange** = Canceled

Insights from this output:

Online & Offline bookings:

- Higher lead times are associated with more cancellations (orange boxes are taller for higher days).
- People booking months in advance (200–400 days) tend to cancel more often.

Corporate & Aviation bookings:

- Very low lead times (most under 30 days).
- Much lower cancellation rates compared to online/offline.

Complementary bookings:

- Low volume but surprisingly high cancellation variability for some bookings.

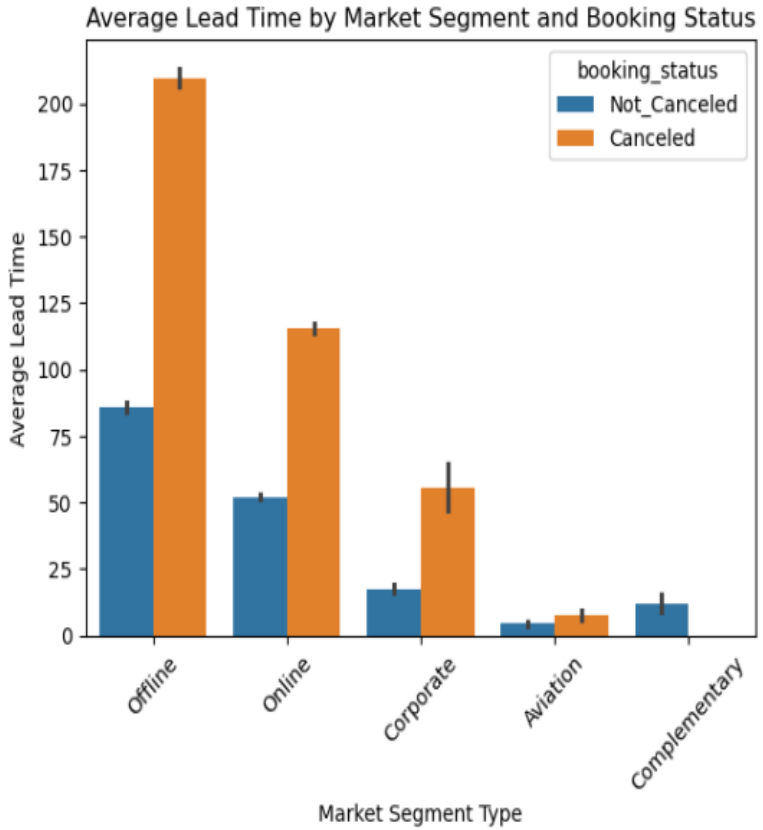
Key takeaway:

- The **longer the lead time**, the higher the cancellation probability—especially for **online bookings**.
- Shorter lead times (last-minute bookings) are more likely to be honored.

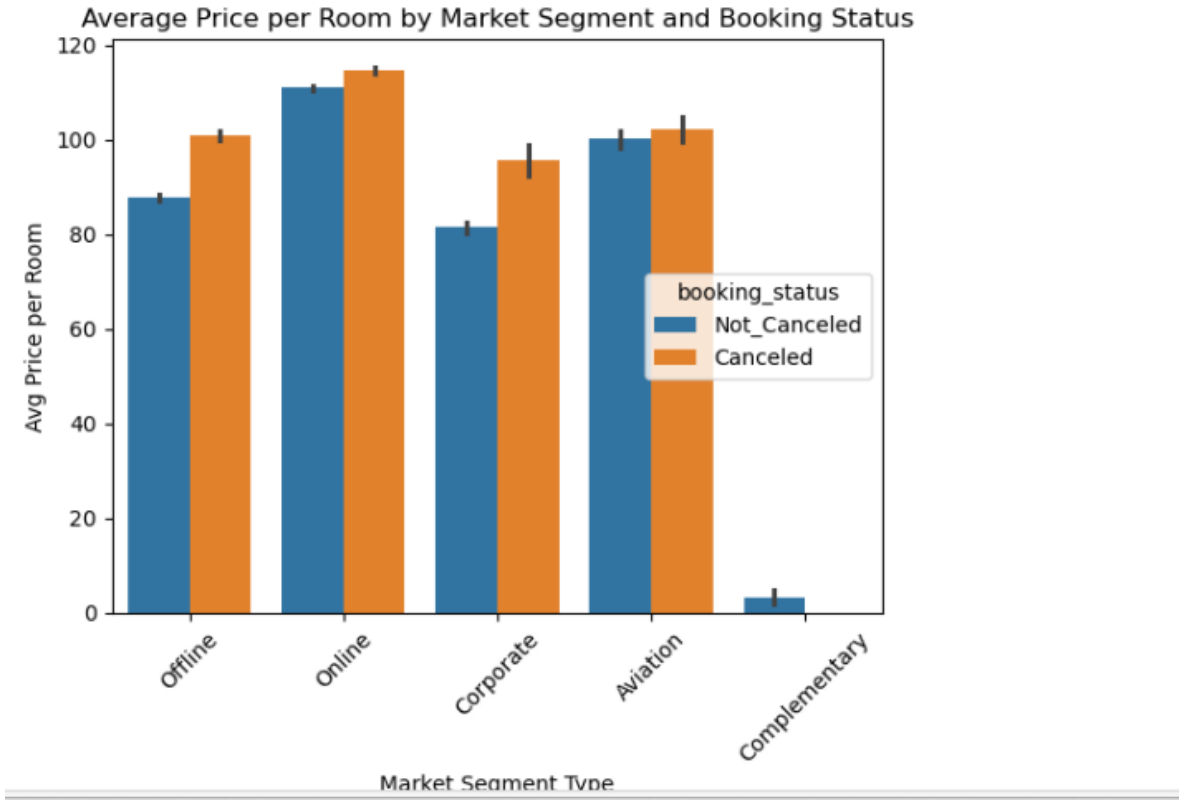
If you want, I can connect this insight to your **hotel cancellation prediction model** so it becomes a strong feature explanation in your project report.

- **Bar Plots**

```
sns.barplot(x='market_segment_type', y='lead_time', hue='booking_status', data=df, estimator='mean')
plt.xticks(rotation=45)
plt.title("Average Lead Time by Market Segment and Booking Status")
plt.ylabel("Average Lead Time")
plt.xlabel("Market Segment Type")
plt.show()
```




```
sns.barplot(x='market_segment_type', y='avg_price_per_room', hue='booking_status', data=df, estimator='me
plt.xticks(rotation=45)
plt.title("Average Price per Room by Market Segment and Booking Status")
plt.ylabel("Avg Price per Room")
plt.xlabel("Market Segment Type")
plt.show()
```



SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Why This is Important:

Average Lead Time by Market Segment and Booking Status

- Purpose: Shows how far in advance guests from different market segments make bookings, and how that relates to cancellations.
- Why important:

  Reveals that Offline and Online bookings with longer average lead times tend to have higher cancellation rates.

  Short lead times (e.g., Corporate, Aviation) are linked to fewer cancellations.

  This insight can help hotels target retention strategies for high lead-time segments.

Average Price per Room by Market Segment and Booking Status

- Purpose: Compares room prices between canceled and not canceled bookings for each segment.
- Why important:

  Highlights pricing trends — e.g., Online bookings have the highest average prices and also high cancellations.

  Helps identify if price sensitivity influences cancellations.

  Useful for dynamic pricing strategies to reduce cancellations while maximizing revenue.

SPINNAKER ANALYTICS

What You Can Gather from It:

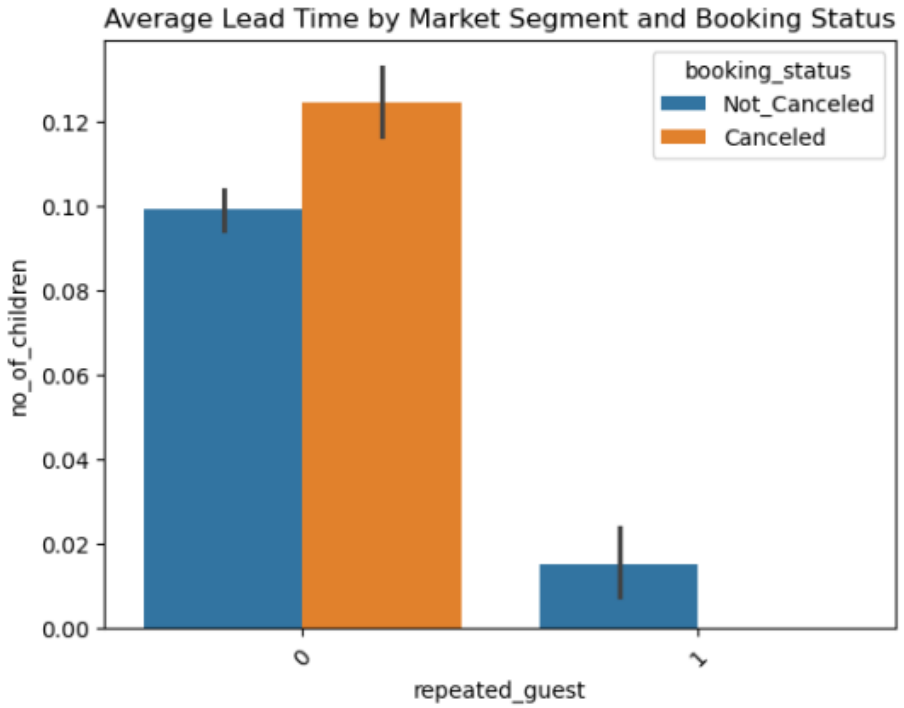
- Long lead times and high prices—especially in the **Online** and **Offline** segments—are strongly associated with higher cancellation rates.

  Short lead times and lower prices—especially in **Corporate** and **Aviation**—are associated with more confirmed stays.

BIA | BOSTON INSTITUTE OF ANALYTICS ®

Bar Plot

```
]: sns.barplot(x='repeated_guest', y='no_of_children', hue='booking_status', data=df, estimator='mean')
   plt.xticks(rotation=45)
   plt.title("Average Lead Time by Market Segment and Booking Status")
   plt.ylabel("no_of_children")
   plt.xlabel("repeated_guest")
   plt.show()
```



Plot Meaning:

- **X-axis ⟶ repeated_guest**
  - **0 = First-time guest**
  - **1 = Returning guest**
- **Y-axis ⟶ Average no_of_children per booking**

Color ⟶ Booking status (Not_Canceled = Blue, Canceled = Orange)

Insights:

1.First-time guests (0) have more children on average compared to returning guests.

2.Among first-time guests, canceled bookings show a slightly higher average number of children than non-canceled ones.

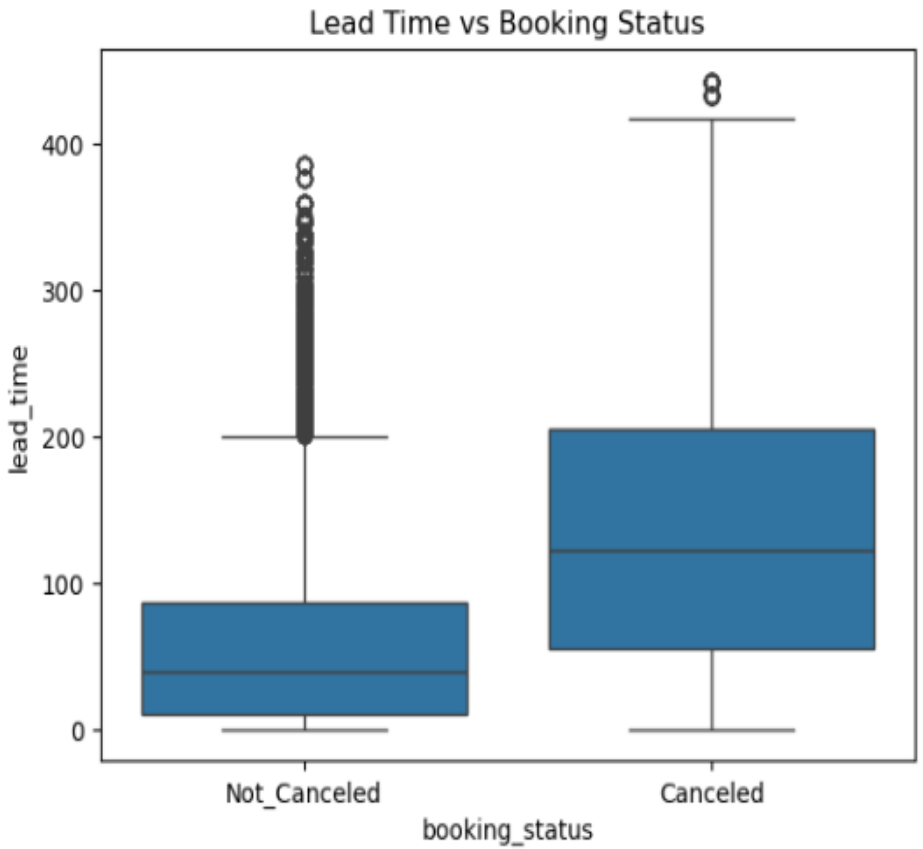
3.Returning guests (1) have a very low average number of children, and their cancellation rates appear low.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS

Box Plot

```
sns.boxplot(x='booking_status', y='lead_time', data=df)
plt.title("Lead Time vs Booking Status")
plt.show()
```



Plot Meaning:

- **X-axis** ⟶ Booking status (Not_Canceled vs Canceled)
- **Y-axis** ⟶ lead_time (days between booking and arrival date)
- Each box shows the distribution of lead times, with **median**, **quartiles**, and **outliers**.

Insights:

1. **Canceled bookings generally have much longer lead times than non-canceled ones.**
    1. **Median lead time for canceled bookings is over 100 days, whereas for non-canceled bookings it's below 50 days.**
2. **There is a wider spread of lead times for canceled bookings — some extend beyond 400 days.**
3. **Non-canceled bookings tend to be made closer to the stay date, with fewer extreme outliers.**
4. **The presence of many outliers in both groups suggests some guests book very far in advance, but this is more common among cancellations.**

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS

Model Training (Baseline + Advanced Models)

Logistic Regression Model Train and test

```
warnings.filterwarnings('ignore')
```

```
58]: def training_model(models, X_train, X_test, y_train, y_test):
    for modelparams in models:
        print('***************', modelparams['Name'], '***************')

        model = modelparams['model']
        model.fit(X_train, y_train)

        train_predict = model.predict(X_train)
        print('***************Training Accuracy***************')
        print(accuracy_score(y_train, train_predict))

        test_predict = model.predict(X_test)
        print('***************Testing Accuracy***************')
        print(accuracy_score(y_test, test_predict))

        print('***************Confusion Matrix***************')
        cn_matrix = confusion_matrix(y_test, test_predict)
        sns.heatmap(cn_matrix, annot=True)
        plt.show()

        print('***************Classification Report***************')
        print(classification_report(y_test, test_predict))
        print('-----------------------------------------------------')
        print('=====================================================')
        print()
```

```
59]: models = [
    {'Name': 'LogisticRegression', 'model': LogisticRegression()},
    {'Name': 'RandomForestClassifier', 'model': RandomForestClassifier()},
    {'Name': 'DecisionTreeClassifier', 'model': DecisionTreeClassifier()}]
```

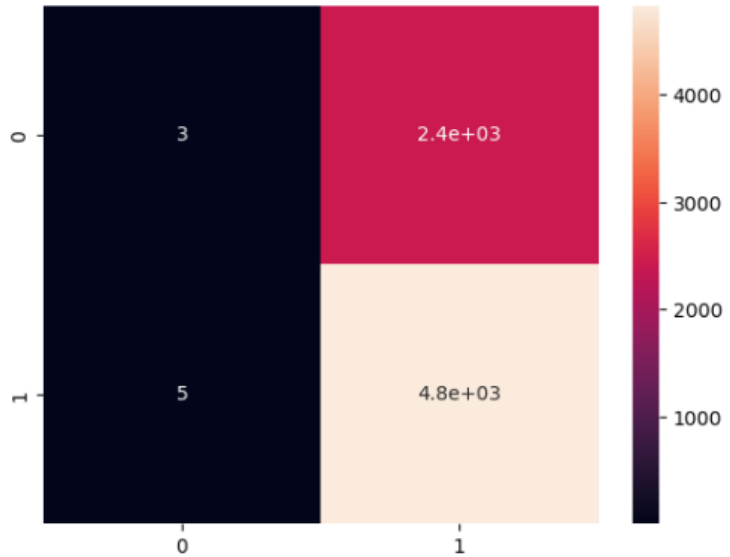
```
61]: X_train = X_train.copy().apply(lambda col: col.view('int64') if np.issubdtype(col.dtype, np.datetime64) else col)
X_test  = X_test.copy().apply(lambda col: col.view('int64') if np.issubdtype(col.dtype, np.datetime64) else col)
```

```
62]: training_model(models, X_train, X_test, y_train, y_test)
```

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

```
2]: training_model(models, X_train, X_test, y_train, y_test)
********** LogisticRegression **********
**********Training Accuracy**********
0.6729841488628532
**********Testing Accuracy**********
0.6667126119917298
**********Confusion Matrix**********
```




```
**********Classification Report**********
              precision    recall  f1-score   support

           0       0.38      0.00      0.00      2416
           1       0.67      1.00      0.80      4839

    accuracy                           0.67      7255
   macro avg       0.52      0.50      0.40      7255
weighted avg       0.57      0.67      0.53      7255
```

SPINNAKER ANALYTICS

Gathered Information

- Model Used: Logistic Regression
- Training and Testing Accuracy
  - Training Accuracy: 0.67298
  - Testing Accuracy: 0.66671
- Confusion Matrix: Provides detailed numbers of correct and incorrect predictions for each class.
  - True negatives (Class 0 predicted as 0): 3
  - False positives (Class 0 predicted as 1): 2,413
  - False negatives (Class 1 predicted as 0): 5
  - True positives (Class 1 predicted as 1): 4,834
- Classification Report: Includes precision, recall, f1-score, and support for both classes:
  - Class 0: Precision 0.38, Recall 0.00, F1-score 0.00, Support 2,416
  - Class 1: Precision 0.67, Recall 1.00, F1-score 0.80, Support 4,839
  - Accuracy: 0.67
  - Macro average and weighted average reported for precision, recall, and f1-score.

Output

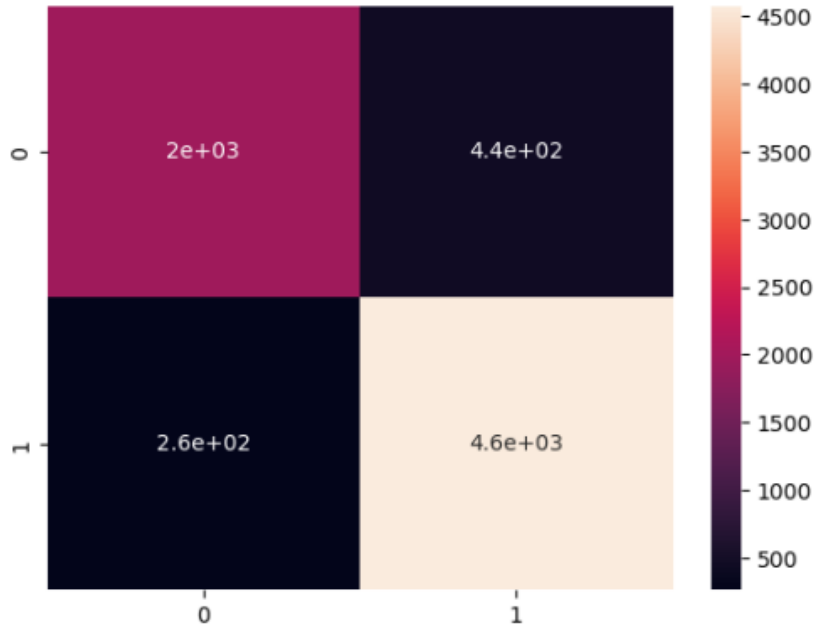
- The output consists of:
  - Printed model details and accuracy scores.
  - Confusion matrix visualized as a heatmap.
  - Printed classification report with evaluation metrics (precision, recall, f1-score, support) for each class.
  - Summary evaluation statistics: overall accuracy, macro average, and weighted average of metrics.

Interpretation

- The model performs well for class 1 (positive class) but fails almost entirely on class 0 (negative class), as reflected in zero recall and f1-score for class 0.
- The output gives a comprehensive performance summary, which can be used to analyze strengths and weaknesses of the trained logistic regression model.

BIA BOSTON INSTITUTE OF ANALYTICS ®

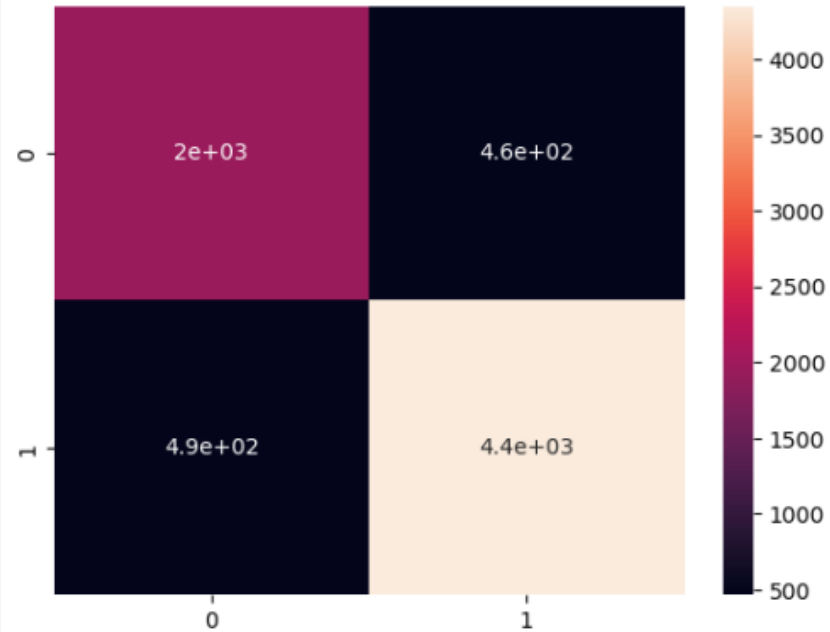
```
--------------------------------------------------
==================================================

********** RandomForestClassifier **********
**********Training Accuracy**********
0.9939696760854583
**********Testing Accuracy**********
0.9040661612680909
**********Confusion Matrix**********
```



**********Classification Report**********

	precision	recall	f1-score	support
0	0.88	0.82	0.85	2416
1	0.91	0.95	0.93	4839
accuracy			0.90	7255
macro avg	0.90	0.88	0.89	7255
weighted avg	0.90	0.90	0.90	7255

```
--------------------------------------------------
==================================================

********** DecisionTreeClassifier **********
**********Training Accuracy**********
0.9940041350792557
**********Testing Accuracy**********
0.8686423156443832
**********Confusion Matrix**********
```



**********Classification Report**********

	precision	recall	f1-score	support
0	0.80	0.81	0.80	2416
1	0.90	0.90	0.90	4839
accuracy			0.87	7255
macro avg	0.85	0.85	0.85	7255
weighted avg	0.87	0.87	0.87	7255

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

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

1. **RandomForestClassifier (Left)**

- **Training Accuracy:** 99.39%
- **Testing Accuracy:** 90.40%
- **Precision/Recall/F1-score:**
  - •Class 0 (Not Canceled) ⟶ Precision 0.88, Recall 0.82
  - •Class 1 (Canceled) ⟶ Precision 0.91, Recall 0.95
- **Observation:**
  - •Performs very well on predicting cancellations (Class 1), with high recall (0.95) — meaning it catches most cancellations.
  - •Slight overfitting since training accuracy is extremely high compared to testing accuracy, but still performs strongly on test data.

2. **DecisionTreeClassifier (Right)**

- **Training Accuracy:** 99.40%
- **Testing Accuracy:** 86.86%
- **Precision/Recall/F1-score:**
  - •Class 0 (Not Canceled) ⟶ Precision 0.80, Recall 0.81
  - •Class 1 (Canceled) ⟶ Precision 0.90, Recall 0.90
- **•Observation:**
  - •Slightly lower test performance than Random Forest.
  - •Shows higher overfitting — training accuracy is perfect but test accuracy drops more.
  - •Still captures cancellations well (0.90 recall) but slightly worse than Random Forest.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

```
]: from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Convert datetime columns to integers
X_train = X_train.copy().apply(lambda col: col.view('int64') if np.issubdtype(col.dtype, np
X_test  = X_test.copy().apply(lambda col: col.view('int64') if np.issubdtype(col.dtype, np.

# Train Random Forest
rf_model = RandomForestClassifier(n_estimators=200, max_depth=None, random_state=42)
rf_model.fit(X_train, y_train)
```

```
]: RandomForestClassifier
RandomForestClassifier(n_estimators=200, random_state=42)
```

```
]: y_pred_rf = rf_model.predict(X_test)

print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
print("\nClassification Report:\n", classification_report(y_test, y_pred_rf))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred_rf))
```

```
Random Forest Accuracy: 0.9050310130944177

Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.82      0.85      2416
           1       0.92      0.95      0.93      4839

    accuracy                           0.91      7255
   macro avg       0.90      0.88      0.89      7255
weighted avg       0.90      0.91      0.90      7255


Confusion Matrix:
 [[1992  424]
 [ 265 4574]]
```

What You Gathered

This is the information your code collected from the Random Forest model after training and testing:

1. Random Forest Model Performance

Accuracy:

Random Forest Accuracy: 0.905 (about 90.5%)

2. Classification Report

Precision, Recall, F1-score, Support for both classes (0 and 1):

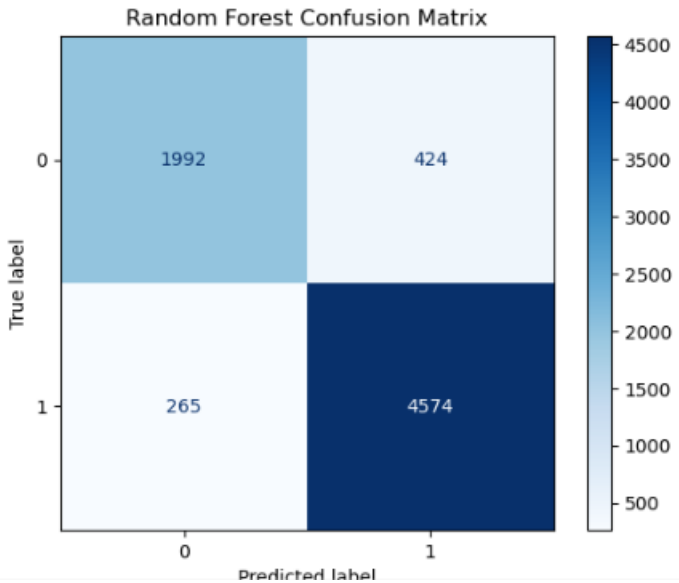
Accuracy: 0.91 (note: the classification report shows rounded value)

Macro avg (simple avg of metrics): Precision 0.89, Recall 0.88, F1-score 0.88

Weighted avg (accounts for support): Precision 0.90, Recall 0.91, F1-score 0.90

```
# Predictions
y_pred_rf = rf_model.predict(X_test)

# Accuracy
print("Random Forest Accuracy:", round(accuracy_score(y_test, y_pred_rf), 4))

# Classification Report
print("\nClassification Report:\n", classification_report(y_test, y_pred_rf))

# Confusion Matrix Plot
cm = confusion_matrix(y_test, y_pred_rf)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=rf_model.classes_)
disp.plot(cmap="Blues")
plt.title("Random Forest Confusion Matrix")
plt.show()
```

```
Random Forest Accuracy: 0.905

Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.82      0.85      2416
           1       0.92      0.95      0.93      4839

    accuracy                           0.91      7255
   macro avg       0.90      0.88      0.89      7255
weighted avg       0.90      0.91      0.90      7255
```



Summary:

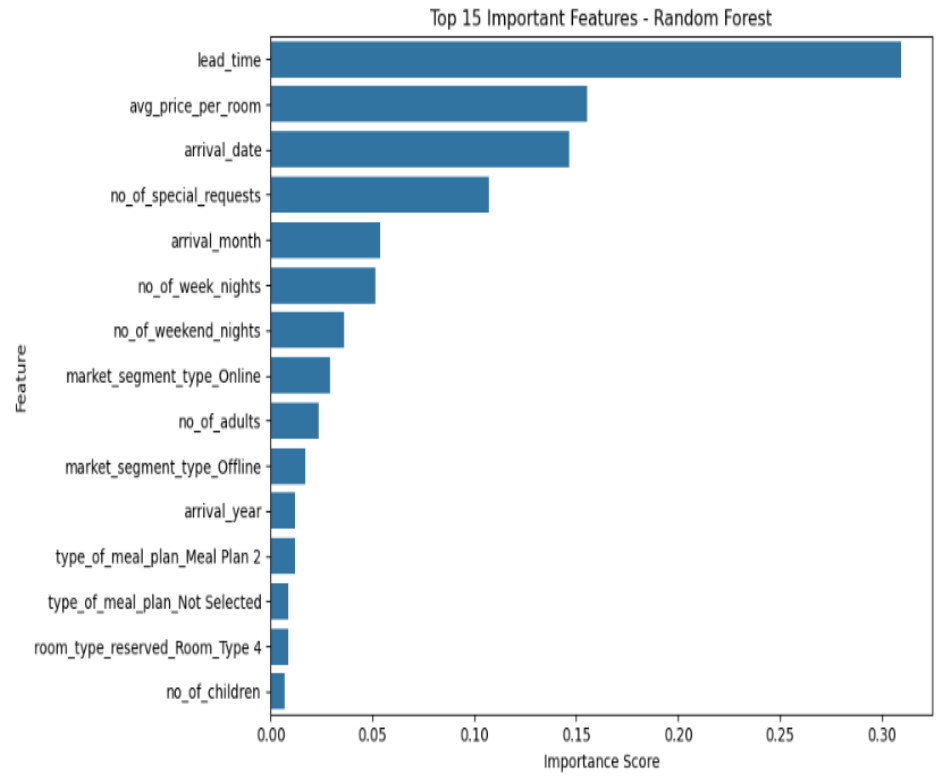
- You gathered: Random Forest model predictions, accuracy, classification report, and confusion matrix.
- You output: Accuracy score, detailed classification metrics (precision, recall, f1-score, support), and confusion matrix plot.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®

Predicted label

```
# Get feature importance
importances = pd.Series(rf_model.feature_importances_, index=X_train.columns)
importances = importances.sort_values(ascending=False)

# Plot top 15 features
plt.figure(figsize=(10, 6))
sns.barplot(x=importances[:15], y=importances.index[:15])
plt.title("Top 15 Important Features - Random Forest")
plt.xlabel("Importance Score")
plt.ylabel("Feature")
plt.tight_layout()
plt.show()
```



What You Gathered

Your model training and evaluation process using Random Forest produced a comprehensive set of outputs:

- Model Predictions: The predicted values for your test dataset, allowing further analysis of model accuracy and types of errors.
- Accuracy Score: Quantifies the proportion of correct predictions, showing the model's overall performance (90.5% accuracy).
- Classification Metrics:
  - Precision: Measures the accuracy of positive predictions for each class.
  - Recall: Measures how well the model identifies all relevant instances for each class.
  - F1-score: Harmonic mean of precision and recall, giving a balanced metric.
  - Support: Number of actual instances per class in the test data.

SPINNAKER ANALYTICS

BIA BOSTON INSTITUTE OF ANALYTICS ®


Thank You!