Project Members

1. Ahmed Gomaa Nazief
2. Karim Ayman Mohammed
3. Mohamed Hassan El Qappaney
4. Mohamed Saeed Fathey

Dataset link: https://www.kaggle.com/datasets/jessemostipak/hotel-booking-demand/data

In [1]:
```
# Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1- Data Definition

hotel: (H1 = Resort Hotel or H2 = City Hotel)

is_canceled: Value indicating if the booking was canceled (1) or not (0)

lead_time: Number of days that elapsed between the entering date of the booking into the PMS and the arrival date.

arrival_date_year: Year of arrival date.

arrival_date_month: Month of arrival date.

arrival_date_week_number: Week number of year for arrival date

arrival_date_day_of_month: Day of arrival date.

stays_in_weekend_nights: Number of weekend nights (Saturday or Sunday) the guest stayed or booked to stay at the hotel.

stays_in_week_nights: Number of week nights (Monday to Friday) the guest stayed or booked to stay at the hotel.

adults: Number of adults.

children: Number of children.

babies: Number of babies.

meal: Type of meal booked. Categories are presented in standard hospitality meal packages: Undefined/SC – no meal.

country: Country of origin. Categories are represented in the ISO 3155–3:2013 format.

market_segment: Market segment designation. In categories, the term “TA” means “Travel Agents” and “TO” means “Tour".

distribution_channel: Booking distribution channel. The term “TA” means “Travel Agents” and “TO” means “Tour Operators”.

is_repeated_guest: Value indicating if the booking name was from a repeated guest (1) or not (0).

previous_cancellations: Number of previous bookings that were cancelled by the customer prior to the current booking.

previous_bookings_not_canceled: Number of previous bookings not cancelled by the customer prior to the current booking..

reserved_room_type: Code of room type reserved. Code is presented instead of designation for anonymity reasons.

assigned_room_type: Code for the type of room assigned to the booking. Sometimes the assigned room type differs from the reserved.

booking_changes: Number of changes/amendments made to the booking from the moment the booking was entered on.

deposit_type: Indication on if the customer made a deposit to guarantee the booking. This variable can assume three.

agent: ID of the travel agency that made the booking.

company: ID of the company/entity that made the booking or responsible for paying the booking.

days_in_waiting_list: Number of days the booking was in the waiting list before it was confirmed to the customer.

customer_type: Type of booking, (Transient، Contract، Group، أو Transient-Party).

adr: Average Daily Rate as defined by dividing the sum of all lodging transactions by the total number of staying nights.

required_car_parking_spaces: Number of car parking spaces required by the customer.

total_of_special_requests: Number of special requests made by the customer (e.g. twin bed or high floor).

reservation_status: Reservation last status, (Check-Out، Canceled، No-Show).

reservation_status_date: Date at which the last status was set.

Project Summary: Hotel Booking Analysis

This project focuses on analyzing hotel booking data to understand cancellation patterns and predict cancellation likelihood using machine learning models. Below is a concise overview of the key components:

Data Definition

- Dataset sourced from Kaggle, containing hotel booking details.
- Key features: hotel type, lead time, cancellation status, customer demographics, booking channels, and reservation status.
- Target variable: `is_canceled` (1 = canceled, 0 = not canceled).

Data Cleaning

- Handled missing values and inconsistencies in the dataset.
- Converted categorical variables to numerical using Label Encoding.
- Extracted new features from `reservation_status_date` (year, month, day) and dropped irrelevant columns.
- Removed unnecessary columns like `Unnamed: 0`.

Exploratory Data Analysis (EDA)

- Analyzed cancellation rates across lead time ranges, revealing higher cancellations for longer lead times.
- Visualized correlations using heatmaps to identify influential features.
- Identified key features correlated with cancellations, such as `reservation_status`, `lead_time`, and `deposit_type`.

Machine Learning

- Applied models: SVM, XGBoost, K-Nearest Neighbors, and Logistic Regression.
- Used SMOTE to address class imbalance in the dataset.
- Scaled features using RobustScaler and StandardScaler for model optimization.
- Evaluated models using Accuracy, Precision, Recall, F1-Score, and confusion matrices.
- Compared model performance to select the most effective predictor of cancellations.

Hotel Booking Analysis

1- EDA (Exploratory Data Analysis) & Cleaning

In [2]:
```
df = pd.read_csv('hotel_bookings.csv')
```

In [3]:
```
df
```

Out[3]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month
0	Resort Hotel	0	342	2015	July	27	1
1	Resort Hotel	0	737	2015	July	27	1
2	Resort Hotel	0	7	2015	July	27	1
3	Resort Hotel	0	13	2015	July	27	1
4	Resort Hotel	0	14	2015	July	27	1
...	...	...	...	...	...	...	...
119385	City Hotel	0	23	2017	August	35	30
119386	City Hotel	0	102	2017	August	35	31
119387	City Hotel	0	34	2017	August	35	31
119388	City Hotel	0	109	2017	August	35	31
119389	City Hotel	0	205	2017	August	35	29

119390 rows × 32 columns

In [4]:
```
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
```

In [5]:
```
df.describe()
```

Out[5]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_ni
count	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000	119390.00
mean	0.370416	104.011416	2016.156554	27.165173	15.798241	0.92
std	0.482918	106.863097	0.707476	13.605138	8.780829	0.99
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.00
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.00
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.00
75%	1.000000	160.000000	2017.000000	38.000000	23.000000	2.00
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.00

In [6]:
```
df.describe(include="object")
```

Out[6]:

	hotel	arrival_date_month	meal	country	market_segment	distribution_channel	reserved_room_type	assigned_room_ty
count	119390	119390	119390	118902	119390	119390	119390	119
unique	2	12	5	177	8	5	10	
top	City Hotel	August	BB	PRT	Online TA	TA/TO	A	
freq	79330	13877	92310	48590	56477	97870	85994	74

Check of Null Values

In [7]:
```
df.isna().sum()
```

Out[7]:

	0
hotel	0
is_canceled	0
lead_time	0
arrival_date_year	0
arrival_date_month	0
arrival_date_week_number	0
arrival_date_day_of_month	0
stays_in_weekend_nights	0
stays_in_week_nights	0
adults	0
children	4
babies	0
meal	0
country	488
market_segment	0
distribution_channel	0
is_repeated_guest	0
previous_cancellations	0
previous_bookings_not_canceled	0
reserved_room_type	0
assigned_room_type	0
booking_changes	0
deposit_type	0
agent	16340
company	112593
days_in_waiting_list	0
customer_type	0
adr	0
required_car_parking_spaces	0
total_of_special_requests	0
reservation_status	0
reservation_status_date	0

dtype: int64

In [8]:
```
#columns "agent", "company" have so many missing values, and columns "country", "children" have some missing va
#check if it's important

df['agent'].value_counts()
```

Out[8]:

agent	count
9.0	31961
240.0	13922
1.0	7191
14.0	3640
7.0	3539
...	...
197.0	1
144.0	1
388.0	1
453.0	1
480.0	1

333 rows × 1 columns

dtype: int64

In [9]:
```
#since "agent" & "company" columns are just Id and with high missing values, so we may drop them

df.drop(columns=['agent','company'],inplace=True)
```

In [10]:
```
df['children'].value_counts(), df['babies'].value_counts()
```

Out[10]:
```
(children
 0.0     110796
 1.0       4861
 2.0       3652
 3.0         76
 10.0         1
 Name: count, dtype: int64,
 babies
 0     118473
 1        900
 2         15
 10         1
 9          1
 Name: count, dtype: int64)
```

In [11]:
```
df['children'] = df['children'].fillna(0)
```

In [12]:
```
df['children'].isna().sum()
```

Out[12]:
```
np.int64(0)
```

In [13]:
```
df.isna().sum()
```

Out[13]:

	0
hotel	0
is_canceled	0
lead_time	0
arrival_date_year	0
arrival_date_month	0
arrival_date_week_number	0
arrival_date_day_of_month	0
stays_in_weekend_nights	0
stays_in_week_nights	0
adults	0
children	0
babies	0
meal	0
country	488
market_segment	0
distribution_channel	0
is_repeated_guest	0
previous_cancellations	0
previous_bookings_not_canceled	0
reserved_room_type	0
assigned_room_type	0
booking_changes	0
deposit_type	0
days_in_waiting_list	0
customer_type	0
adr	0
required_car_parking_spaces	0
total_of_special_requests	0
reservation_status	0
reservation_status_date	0

dtype: int64

In [14]:
```
df['country'].value_counts()
```

Out[14]:

country	count
PRT	48590
GBR	12129
FRA	10415
ESP	8568
DEU	7287
...	...
MRT	1
KIR	1
SDN	1
ATF	1
SLE	1

177 rows × 1 columns

dtype: int64

In [15]:
```
df['country'] = df['country'].fillna('PRT')
df
#Now there's no Null Values
```

Out[15]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month
0	Resort Hotel	0	342	2015	July	27	1
1	Resort Hotel	0	737	2015	July	27	1
2	Resort Hotel	0	7	2015	July	27	1
3	Resort Hotel	0	13	2015	July	27	1
4	Resort Hotel	0	14	2015	July	27	1
...	...	...	...	...	...	...	...
119385	City Hotel	0	23	2017	August	35	30
119386	City Hotel	0	102	2017	August	35	31
119387	City Hotel	0	34	2017	August	35	31
119388	City Hotel	0	109	2017	August	35	31
119389	City Hotel	0	205	2017	August	35	29

119390 rows × 30 columns

__Preprocessing__ (Handling Outliers and Dublicates and Logical Errors)

In [16]:
```
#it's illogical to have 0 adults and 0 children and 0 babies in the same record so we drop them

df = df[(df['adults'] + df['children'] + df['babies']) > 0].copy()
```

In [17]:
```
df
```

Out[17]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month
0	Resort Hotel	0	342	2015	July	27	1
1	Resort Hotel	0	737	2015	July	27	1
2	Resort Hotel	0	7	2015	July	27	1
3	Resort Hotel	0	13	2015	July	27	1
4	Resort Hotel	0	14	2015	July	27	1
...	...	...	...	...	...	...	...
119385	City Hotel	0	23	2017	August	35	30
119386	City Hotel	0	102	2017	August	35	31
119387	City Hotel	0	34	2017	August	35	31
119388	City Hotel	0	109	2017	August	35	31
119389	City Hotel	0	205	2017	August	35	29

119210 rows × 30 columns

In [18]:
```
df.describe()
```

Out[18]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_ni
count	119210.000000	119210.000000	119210.000000	119210.000000	119210.000000	119210.00
mean	0.370766	104.109227	2016.156472	27.163376	15.798717	0.92
std	0.483012	106.875450	0.707485	13.601107	8.781070	0.99
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.00
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.00
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.00
75%	1.000000	161.000000	2017.000000	38.000000	23.000000	2.00
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.00

In [19]:
```
#checking The Zero adults since no Hotel would accept from only kids

df['adults'].mean(), df['adults'].median()
```

Out[19]:
```
(np.float64(1.8592064424125492), 2.0)
```

In [20]:
```
#replacing 0 values of adults to the average value 2
df['adults'] = df['adults'].replace(0,2)
```

In [21]:
```
df['adults'].unique()
```

Out[21]:
```
array([ 2,  1,  3,  4, 40, 26, 50, 27, 55, 20,  6,  5, 10])
```

In [22]:
```
df['children'].value_counts(), df['babies'].value_counts()
```

Out[22]:
```
(children
 0.0     110620
 1.0       4861
 2.0       3652
 3.0         76
 10.0         1
 Name: count, dtype: int64,
 babies
 0     118293
 1        900
 2         15
 10         1
 9          1
 Name: count, dtype: int64)
```

In [23]:
```
#fixing outliers of cheldren and babies columns
df['children'] = df['children'].replace(10, 0)
df['babies'] = df['babies'].replace(10, 0)
df['babies'] = df['babies'].replace(9, 0)
```

In [24]:
```
df['children'].value_counts(), df['babies'].value_counts()
```

Out[24]:
```
(children
 0.0    110621
 1.0      4861
 2.0      3652
 3.0        76
 Name: count, dtype: int64,
 babies
 0    118295
 1       900
 2        15
 Name: count, dtype: int64)
```

In [25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 119210 entries, 0 to 119389
Data columns (total 30 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   hotel                           119210 non-null  object
 1   is_canceled                     119210 non-null  int64
 2   lead_time                       119210 non-null  int64
 3   arrival_date_year               119210 non-null  int64
 4   arrival_date_month              119210 non-null  object
 5   arrival_date_week_number        119210 non-null  int64
 6   arrival_date_day_of_month       119210 non-null  int64
 7   stays_in_weekend_nights         119210 non-null  int64
 8   stays_in_week_nights            119210 non-null  int64
 9   adults                          119210 non-null  int64
 10  children                        119210 non-null  float64
 11  babies                          119210 non-null  int64
 12  meal                            119210 non-null  object
 13  country                         119210 non-null  object
 14  market_segment                  119210 non-null  object
 15  distribution_channel            119210 non-null  object
 16  is_repeated_guest               119210 non-null  int64
 17  previous_cancellations          119210 non-null  int64
 18  previous_bookings_not_canceled  119210 non-null  int64
 19  reserved_room_type              119210 non-null  object
 20  assigned_room_type              119210 non-null  object
 21  booking_changes                 119210 non-null  int64
 22  deposit_type                    119210 non-null  object
 23  days_in_waiting_list            119210 non-null  int64
 24  customer_type                   119210 non-null  object
 25  adr                             119210 non-null  float64
 26  required_car_parking_spaces     119210 non-null  int64
 27  total_of_special_requests       119210 non-null  int64
 28  reservation_status              119210 non-null  object
 29  reservation_status_date         119210 non-null  object
dtypes: float64(2), int64(16), object(12)
memory usage: 28.2+ MB
```

In [26]:
```
#reservation_status_date was assigned as an object not datetime

df['reservation_status_date']= pd.to_datetime(df['reservation_status_date'])
```

In [27]:
```
#checking the Outliers of adults coulmn
df.adults.value_counts()
```

Out[27]:

adults	count
2	89903
1	23027
3	6202
4	62
26	5
5	2
27	2
20	2
40	1
55	1
50	1
6	1
10	1

dtype: int64

In [28]:
```
df11 = df
```

In [29]:
```
#setting any value more than 4 to 1

df11.loc[df['adults']>4,'adults'] = 1
df11.describe()
```

Out[29]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_ni
count	119210.000000	119210.000000	119210.000000	119210.000000	119210.000000	119210.00
mean	0.370766	104.109227	2016.156472	27.163376	15.798717	0.92
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.00
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.00
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.00
75%	1.000000	161.000000	2017.000000	38.000000	23.000000	2.00
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.00
std	0.483012	106.875450	0.707485	13.601107	8.781070	0.99

In [30]:
```
df11['days_in_waiting_list'].value_counts()
```

Out[30]:

days_in_waiting_list	count
0	115517
39	227
58	164
44	141
31	127
...	...
72	1
81	1
74	1
167	1
36	1

127 rows × 1 columns

dtype: int64

In [31]:
```
#simce almost any value other than 0 is outlier we set them all to 0

df11['days_in_waiting_list'] = [0 for x in df11['days_in_waiting_list']]
```

In [32]:
```
df11['days_in_waiting_list'].unique()
```

Out[32]:
```
array([0])
```

In [33]:
```
df11.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 119210 entries, 0 to 119389
Data columns (total 30 columns):
 #   Column                          Non-Null Count   Dtype         
---  ------                          --------------   -----         
 0   hotel                           119210 non-null  object        
 1   is_canceled                     119210 non-null  int64         
 2   lead_time                       119210 non-null  int64         
 3   arrival_date_year               119210 non-null  int64         
 4   arrival_date_month              119210 non-null  object        
 5   arrival_date_week_number        119210 non-null  int64         
 6   arrival_date_day_of_month       119210 non-null  int64         
 7   stays_in_weekend_nights         119210 non-null  int64         
 8   stays_in_week_nights            119210 non-null  int64         
 9   adults                          119210 non-null  int64         
 10  children                        119210 non-null  float64       
 11  babies                          119210 non-null  int64         
 12  meal                            119210 non-null  object        
 13  country                         119210 non-null  object        
 14  market_segment                  119210 non-null  object        
 15  distribution_channel            119210 non-null  object        
 16  is_repeated_guest               119210 non-null  int64         
 17  previous_cancellations          119210 non-null  int64         
 18  previous_bookings_not_canceled  119210 non-null  int64         
 19  reserved_room_type              119210 non-null  object        
 20  assigned_room_type              119210 non-null  object        
 21  booking_changes                 119210 non-null  int64         
 22  deposit_type                    119210 non-null  object        
 23  days_in_waiting_list            119210 non-null  int64         
 24  customer_type                   119210 non-null  object        
 25  adr                             119210 non-null  float64       
 26  required_car_parking_spaces     119210 non-null  int64         
 27  total_of_special_requests       119210 non-null  int64         
 28  reservation_status              119210 non-null  object        
 29  reservation_status_date         119210 non-null  datetime64[ns]
dtypes: datetime64[ns](1), float64(2), int64(16), object(11)
memory usage: 28.2+ MB
```

In [34]:
```
df11.describe()
```

Out[34]:

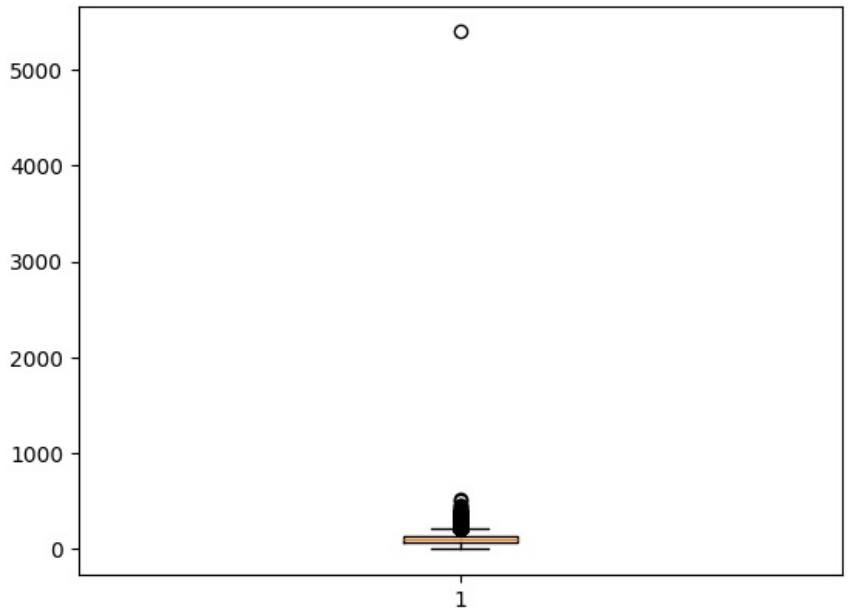
	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_ni
count	119210.000000	119210.000000	119210.000000	119210.000000	119210.000000	119210.00
mean	0.370766	104.109227	2016.156472	27.163376	15.798717	0.92
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.00
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.00
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.00
75%	1.000000	161.000000	2017.000000	38.000000	23.000000	2.00
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.00
std	0.483012	106.875450	0.707485	13.601107	8.781070	0.99

In [35]:
```
#for adr column there's a lot of wrong values and outliers

df11['adr'].median(), df11['adr'].mode()
```

Out[35]:
```
(94.95,
 0    62.0
 Name: adr, dtype: float64)
```

In [36]:
```
plt.boxplot(df11['adr'])
```

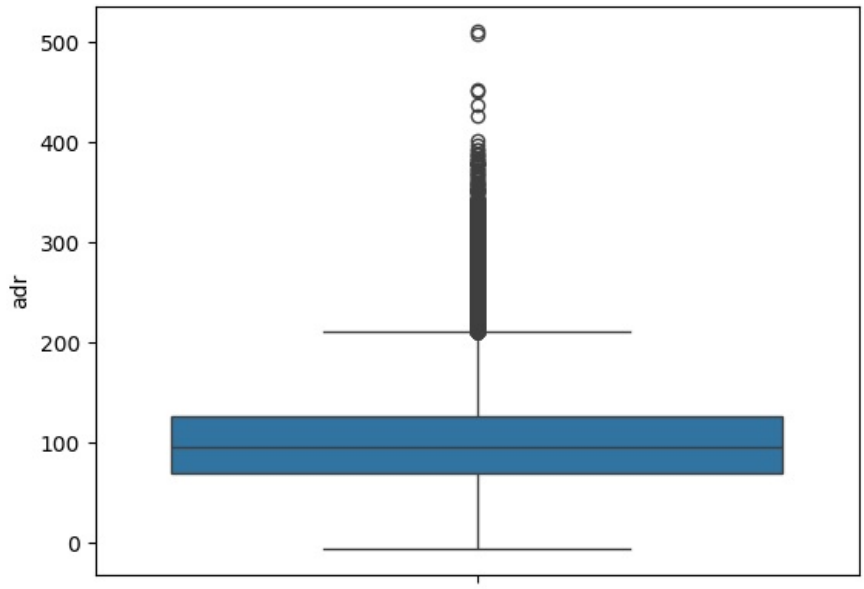
Out[36]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7fdad5e5b5d0>,
  <matplotlib.lines.Line2D at 0x7fdad5e64150>],
 'caps': [<matplotlib.lines.Line2D at 0x7fdad5e64c50>,
  <matplotlib.lines.Line2D at 0x7fdad5e65650>],
 'boxes': [<matplotlib.lines.Line2D at 0x7fdad63464d0>],
 'medians': [<matplotlib.lines.Line2D at 0x7fdad5e66010>],
 'fliers': [<matplotlib.lines.Line2D at 0x7fdad5e66950>],
 'means': []}
```



In [37]:
```
df11['adr'] = df11['adr'].loc[df11['adr']<5000]
```

In [38]:
```
sns.boxplot(df11.adr)
```

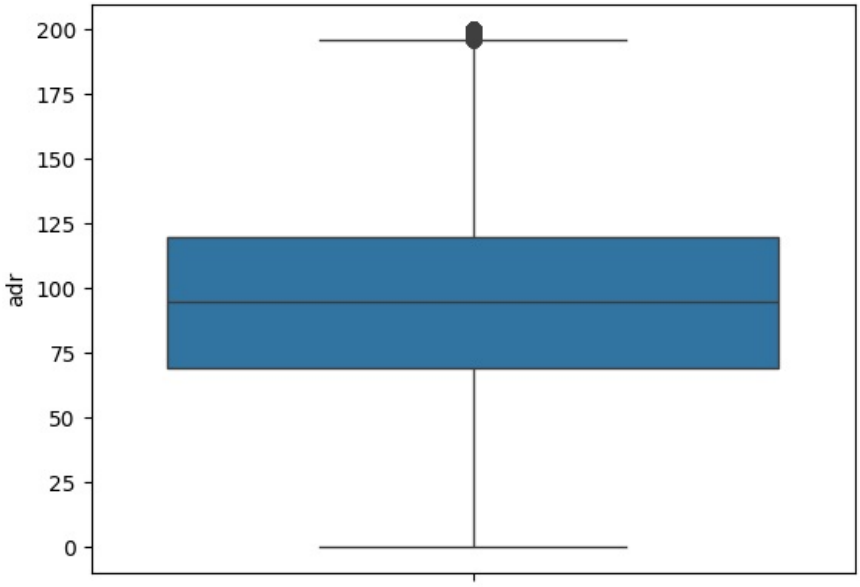
Out[38]:
```
<Axes: ylabel='adr'>
```



In [39]:
```
#replacing any value more than 200 with the average value
df11.loc[df11['adr']>200,'adr'] = df11['adr'].mean()
df11.loc[df11['adr']<0,'adr'] = df11['adr'].mean()
```

In [40]:
```
sns.boxplot(df11.adr)
```

Out[40]:
```
<Axes: ylabel='adr'>
```



Checking for duplicates

In [41]:
```
df11.duplicated().sum()
```

Out[41]:
```
np.int64(32052)
```

In [42]:
```
df11.drop_duplicates(inplace=True)
```

In [43]:
```
df11.shape
```

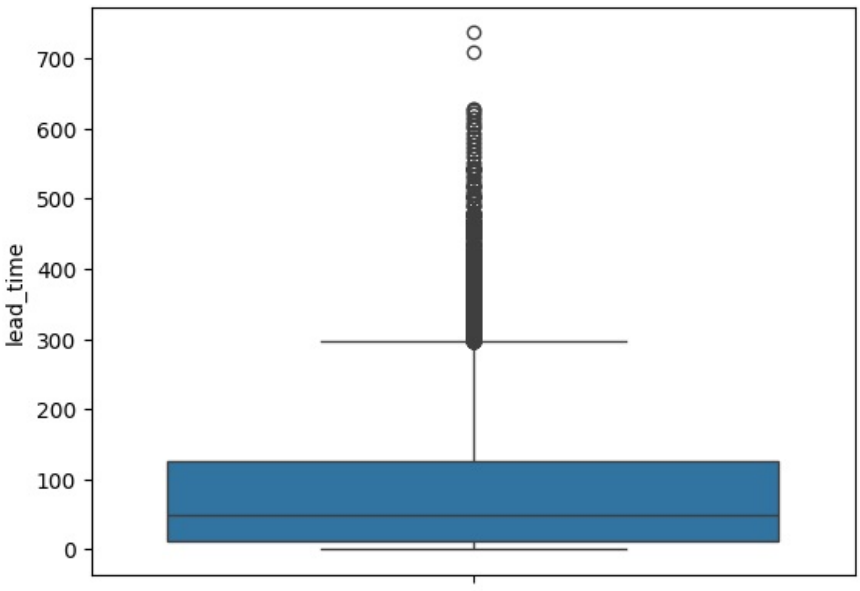
Out[43]:
```
(87158, 30)
```

In [44]:
```
df11['arrival_date_month'].unique()
```

Out[44]:
```
array(['July', 'August', 'September', 'October', 'November', 'December',
       'January', 'February', 'March', 'April', 'May', 'June'],
      dtype=object)
```

In [45]:
```
#checking for Lead_time Outliers
sns.boxplot(df11['lead_time'])
```

Out[45]:
```
<Axes: ylabel='lead_time'>
```



In [46]:
```
df11['lead_time'].median(), df11['lead_time'].mode()
```

Out[46]:
```
(49.0,
 0    0
 Name: lead_time, dtype: int64)
```

In [47]:
```
df11['lead_time'].value_counts()
```

Out[47]:

lead_time	count
0	5895
1	3198
2	1919
3	1702
4	1562
...	...
458	1
521	1
444	1
380	1
463	1

479 rows × 1 columns

dtype: int64

In [48]:
```
# Removing any value greater than 350 since it is not logical to book a hotel more than a year in advance.

max_reasonable_lead_time = 365
df11['lead_time'] = df11['lead_time'].clip(upper=max_reasonable_lead_time)
```

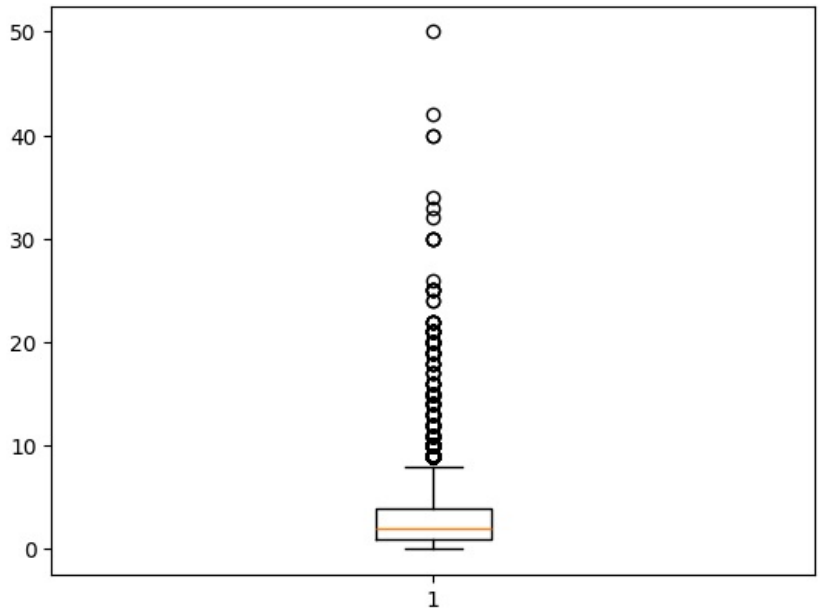
In [49]:
```
df11.describe()
```

Out[49]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nigh
count	87158.000000	87158.000000	87158.000000	87158.000000	87158.000000	87158.0000
mean	0.275316	79.540100	2016.210308	26.835242	15.816379	1.0048
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.0000
25%	0.000000	11.000000	2016.000000	16.000000	8.000000	0.0000
50%	0.000000	49.000000	2016.000000	27.000000	16.000000	1.0000
75%	1.000000	125.000000	2017.000000	37.000000	23.000000	2.0000
max	1.000000	365.000000	2017.000000	53.000000	31.000000	19.0000
std	0.446676	84.239732	0.686095	13.669256	8.835297	1.0274

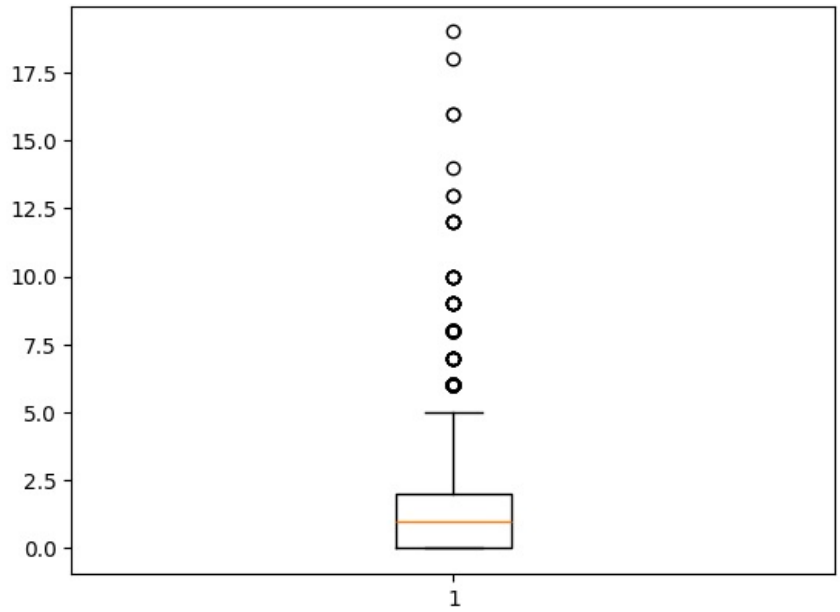
In [50]:
```
plt.boxplot(df11['stays_in_week_nights'])
```

Out[50]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7fdad3a49990>,
  <matplotlib.lines.Line2D at 0x7fdad3a4a1d0>],
 'caps': [<matplotlib.lines.Line2D at 0x7fdad3a4ab10>,
  <matplotlib.lines.Line2D at 0x7fdad3a4b450>],
 'boxes': [<matplotlib.lines.Line2D at 0x7fdad39f4050>],
 'medians': [<matplotlib.lines.Line2D at 0x7fdad3a4bd50>],
 'fliers': [<matplotlib.lines.Line2D at 0x7fdad3a54710>],
 'means': []}
```



In [51]:
```
plt.boxplot(df11['stays_in_weekend_nights'])
```

Out[51]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7fdad39a7050>,
  <matplotlib.lines.Line2D at 0x7fdad39a7ad0>],
 'caps': [<matplotlib.lines.Line2D at 0x7fdad39984d0>,
  <matplotlib.lines.Line2D at 0x7fdad3998e10>],
 'boxes': [<matplotlib.lines.Line2D at 0x7fdad39a6790>],
 'medians': [<matplotlib.lines.Line2D at 0x7fdad3a93290>],
 'fliers': [<matplotlib.lines.Line2D at 0x7fdad3999ed0>],
 'means': []}
```



In [52]:
```
#replacing any value greater than 5 with the average value since there's only 5 days for the weekdays

df11.loc[df11['stays_in_week_nights']>5,'stays_in_week_nights'] = int(df11['stays_in_week_nights'].mean())
```

In [53]:
```
df11.describe()
```

Out[53]:

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nigh
count	87158.000000	87158.000000	87158.000000	87158.000000	87158.000000	87158.0000
mean	0.275316	79.540100	2016.210308	26.835242	15.816379	1.0048
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.0000
25%	0.000000	11.000000	2016.000000	16.000000	8.000000	0.0000
50%	0.000000	49.000000	2016.000000	27.000000	16.000000	1.0000
75%	1.000000	125.000000	2017.000000	37.000000	23.000000	2.0000
max	1.000000	365.000000	2017.000000	53.000000	31.000000	19.0000
std	0.446676	84.239732	0.686095	13.669256	8.835297	1.0274

In [54]:
```
#replacing any value greater than 2 with the average value since there's only 2 days for the weekend
df11.loc[df11['stays_in_weekend_nights']>2,'stays_in_weekend_nights'] = int(df11['stays_in_weekend_nights'].mea
```

In [55]:
```
df11.dropna(inplace=True)
df11.drop_duplicates(inplace=True)
df11.shape
df
```

Out[55]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month
0	Resort Hotel	0	342	2015	July	27	1
1	Resort Hotel	0	365	2015	July	27	1
2	Resort Hotel	0	7	2015	July	27	1
3	Resort Hotel	0	13	2015	July	27	1
4	Resort Hotel	0	14	2015	July	27	1
...	...	...	...	...	...	...	...
119385	City Hotel	0	23	2017	August	35	30
119386	City Hotel	0	102	2017	August	35	31
119387	City Hotel	0	34	2017	August	35	31
119388	City Hotel	0	109	2017	August	35	31
119389	City Hotel	0	205	2017	August	35	29

87154 rows × 30 columns

Analysis Part

In [56]:
```
df11 = pd.read_csv('hotel_bookings_cleaned.csv')
df11.info()
```

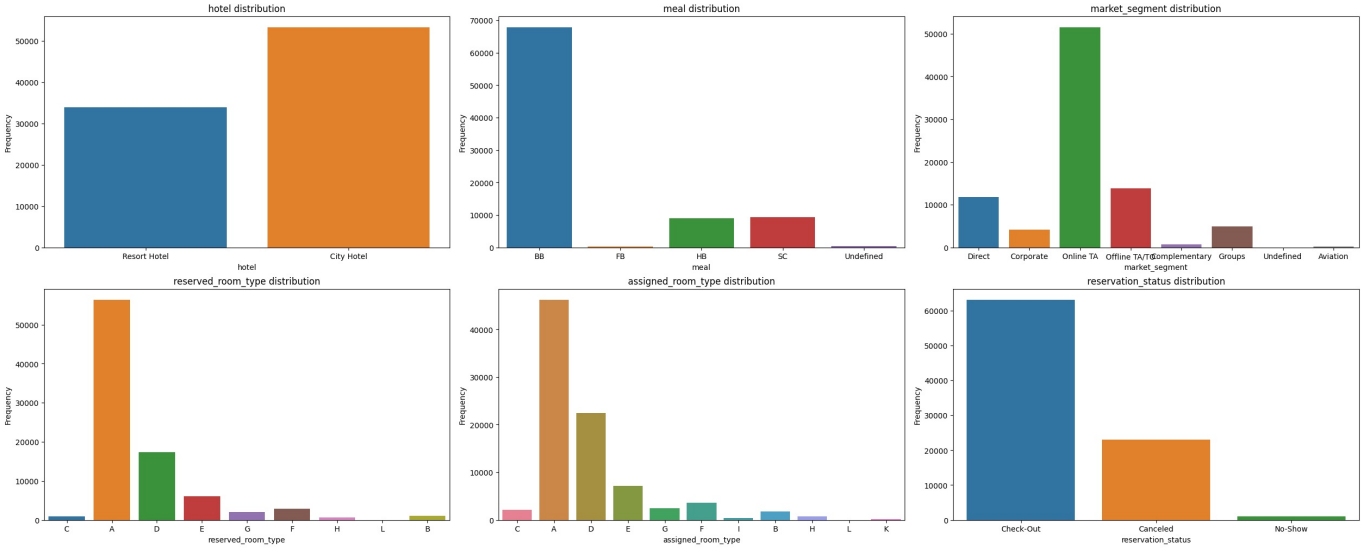
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87154 entries, 0 to 87153
Data columns (total 31 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Unnamed: 0                      87154 non-null  int64
 1   hotel                           87154 non-null  object
 2   is_canceled                     87154 non-null  int64
 3   lead_time                       87154 non-null  float64
 4   arrival_date_year               87154 non-null  int64
 5   arrival_date_month              87154 non-null  object
 6   arrival_date_week_number        87154 non-null  int64
 7   arrival_date_day_of_month       87154 non-null  int64
 8   stays_in_weekend_nights         87154 non-null  int64
 9   stays_in_week_nights            87154 non-null  int64
 10  adults                          87154 non-null  int64
 11  children                        87154 non-null  float64
 12  babies                          87154 non-null  int64
 13  meal                            87154 non-null  object
 14  country                         87154 non-null  object
 15  market_segment                  87154 non-null  object
 16  distribution_channel            87154 non-null  object
 17  is_repeated_guest               87154 non-null  int64
 18  previous_cancellations          87154 non-null  int64
 19  previous_bookings_not_canceled  87154 non-null  int64
 20  reserved_room_type              87154 non-null  object
 21  assigned_room_type              87154 non-null  object
 22  booking_changes                 87154 non-null  int64
 23  deposit_type                    87154 non-null  object
 24  days_in_waiting_list            87154 non-null  int64
 25  customer_type                   87154 non-null  object
 26  adr                             87154 non-null  float64
 27  required_car_parking_spaces     87154 non-null  int64
 28  total_of_special_requests       87154 non-null  int64
 29  reservation_status              87154 non-null  object
 30  reservation_status_date         87154 non-null  object
dtypes: float64(3), int64(16), object(12)
memory usage: 20.6+ MB
```

In [57]:
```
df11= pd.read_csv("hotel_bookings_cleaned.csv")
df11.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87154 entries, 0 to 87153
Data columns (total 31 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Unnamed: 0                      87154 non-null  int64
 1   hotel                           87154 non-null  object
 2   is_canceled                     87154 non-null  int64
 3   lead_time                       87154 non-null  float64
 4   arrival_date_year               87154 non-null  int64
 5   arrival_date_month              87154 non-null  object
 6   arrival_date_week_number        87154 non-null  int64
 7   arrival_date_day_of_month       87154 non-null  int64
 8   stays_in_weekend_nights         87154 non-null  int64
 9   stays_in_week_nights            87154 non-null  int64
 10  adults                          87154 non-null  int64
 11  children                        87154 non-null  float64
 12  babies                          87154 non-null  int64
 13  meal                            87154 non-null  object
 14  country                         87154 non-null  object
 15  market_segment                  87154 non-null  object
 16  distribution_channel            87154 non-null  object
 17  is_repeated_guest               87154 non-null  int64
 18  previous_cancellations          87154 non-null  int64
 19  previous_bookings_not_canceled  87154 non-null  int64
 20  reserved_room_type              87154 non-null  object
 21  assigned_room_type              87154 non-null  object
 22  booking_changes                 87154 non-null  int64
 23  deposit_type                    87154 non-null  object
 24  days_in_waiting_list            87154 non-null  int64
 25  customer_type                   87154 non-null  object
 26  adr                             87154 non-null  float64
 27  required_car_parking_spaces     87154 non-null  int64
 28  total_of_special_requests       87154 non-null  int64
 29  reservation_status              87154 non-null  object
 30  reservation_status_date         87154 non-null  object
dtypes: float64(3), int64(16), object(12)
memory usage: 20.6+ MB
```

In [58]:
```
plot_info = [
    {'col': 'hotel', 'title': 'hotel distribution', 'xlabel': 'hotel'},
    {'col': 'meal', 'title': 'meal distribution', 'xlabel': 'meal'},
```

```
    {'col': 'market_segment', 'title': 'market_segment distribution', 'xlabel': 'market_segment'},
    {'col': 'reserved_room_type', 'title': 'reserved_room_type distribution', 'xlabel': 'reserved_room_type'},
    {'col': 'assigned_room_type', 'title': 'assigned_room_type distribution', 'xlabel': 'assigned_room_type'},
    {'col': 'reservation_status', 'title': 'reservation_status distribution', 'xlabel': 'reservation_status'},
]

fig, axes = plt.subplots(2, 3, figsize=(25, 10), constrained_layout=True)

for ax, info in zip(axes.flatten(), plot_info):
    sns.countplot(x=info['col'], data=df11, ax=ax, hue=info['col'])
    ax.set_title(info['title'])
    ax.set_xlabel(info['xlabel'])
    ax.set_ylabel('Frequency')

plt.show()
```



Based on the analysis of categorical variable distribution:

- Since both hotels are located in Portugal, most customers are from Portugal (PRT).
- The most common way customers first learned about the hotel is through Online TA, and the most common booking method is through TA/TO.
- City hotels have more customers than resort hotels.
- The ratio of successful bookings to cancellations is approximately 2:1.

Data visualization dimensions:

1. Time
2. Hotel
3. Location
4. Customer

What are the trends in reservations and cancellations over time?

In [59]:
```
# Monthly Number of Customers at Resort Hotel
resort_not_canceled = df11[(df11['hotel'] == 'Resort Hotel')].copy()
resort_guests = resort_not_canceled['arrival_date_month'].value_counts().reset_index()
resort_guests.columns=['month','num of Resort hotel guests']
resort_guests
```

Out[59]:

	month	num of Resort hotel guests
0	August	4646
1	July	4307
2	May	2941
3	April	2825
4	June	2759
5	October	2722
6	March	2653
7	February	2484
8	September	2449
9	December	2110
10	November	2048
11	January	1957

In [60]:
```
# Monthly Number of Customers at City Hotel
resort_not_canceled =  df11[(df11['hotel'] == 'City Hotel')].copy()
city_guests = resort_not_canceled['arrival_date_month'].value_counts().reset_index()
city_guests.columns=['month','num of City hotel guests']
city_guests
```

Out[60]:

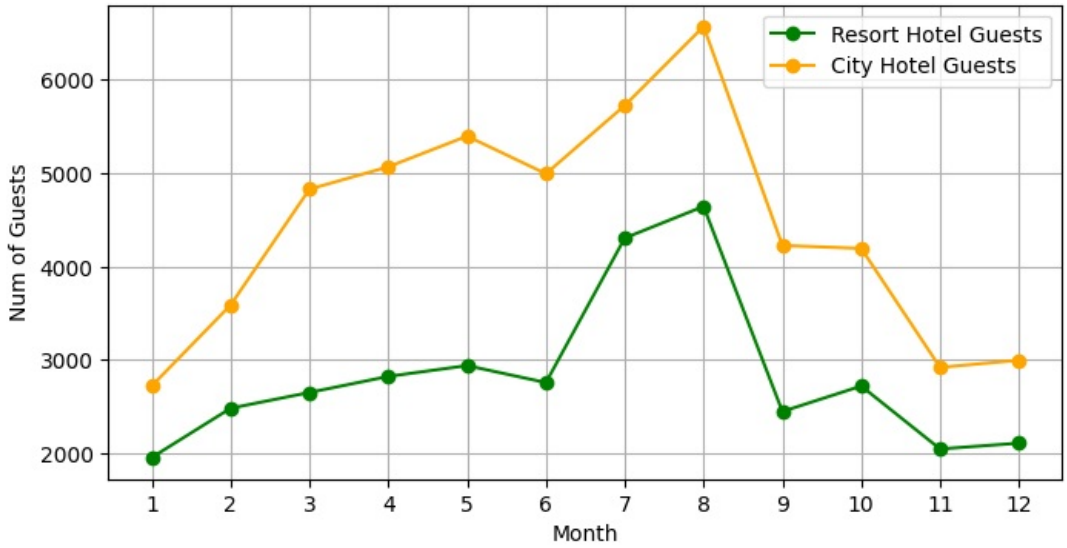
	month	num of City hotel guests
0	August	6573
1	July	5727
2	May	5399
3	April	5070
4	June	4996
5	March	4830
6	September	4228
7	October	4196
8	February	3590
9	December	2998
10	November	2922
11	January	2724

In [61]:
```
# Create a mapping of month names to numbers
month_map = {
    'January': 1, 'February': 2, 'March': 3, 'April': 4, 'May': 5, 'June': 6,
    'July': 7, 'August': 8, 'September': 9, 'October': 10, 'November': 11, 'December': 12
}
final_guests = city_guests.merge(resort_guests, on='month')

# Replace month names with numbers
final_guests['month'] = final_guests['month'].map(month_map)
final_guests['month'] = final_guests['month'].astype(int)
final_guests_sorted = final_guests.sort_values(by='month').reset_index(drop=True)
print(final_guests_sorted)
```

```
    month  num of City hotel guests  num of Resort hotel guests
0       1                      2724                        1957
1       2                      3590                        2484
2       3                      4830                        2653
3       4                      5070                        2825
4       5                      5399                        2941
5       6                      4996                        2759
6       7                      5727                        4307
7       8                      6573                        4646
8       9                      4228                        2449
9      10                      4196                        2722
10     11                      2922                        2048
11     12                      2998                        2110
```

In [62]:
```
plt.figure(figsize=(8, 4))

# Resort Hotel Customers
plt.plot(final_guests_sorted['month'], final_guests_sorted['num of Resort hotel guests'], marker='o', color='gr
```

```
# City Hotel Customers
plt.plot(final_guests_sorted['month'], final_guests_sorted['num of City hotel guests'], marker='o', color='orang

plt.xlabel('Month')
plt.ylabel('Num of Guests')
plt.xticks(final_guests_sorted['month'])
plt.legend()
plt.grid(True)
plt.show()
```



Both City Hotel and Resort Hotel experience an increase in customers during the summer and a decrease during the winter.

In [63]:
```
monthly_counts = df11.groupby('arrival_date_month').size()
monthly_cancellations = df11[df11['is_canceled'] == 1].groupby('arrival_date_month').size()


plt.figure(figsize=(8, 4))


plt.plot(final_guests_sorted['month'], monthly_counts, label='Total Reservations', marker='o', color='orange')

plt.plot(final_guests_sorted['month'], monthly_cancellations, label='Cancellations', marker='o', color='green')


plt.title('Monthly Reservations and Cancellations')
plt.xlabel('Month')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend()


plt.tight_layout()
plt.show()
```

Seasonal Impact : Both reservation and cancellation numbers show clear seasonal variations, with summer and year-end being peak

times, and winter and early year being low times.

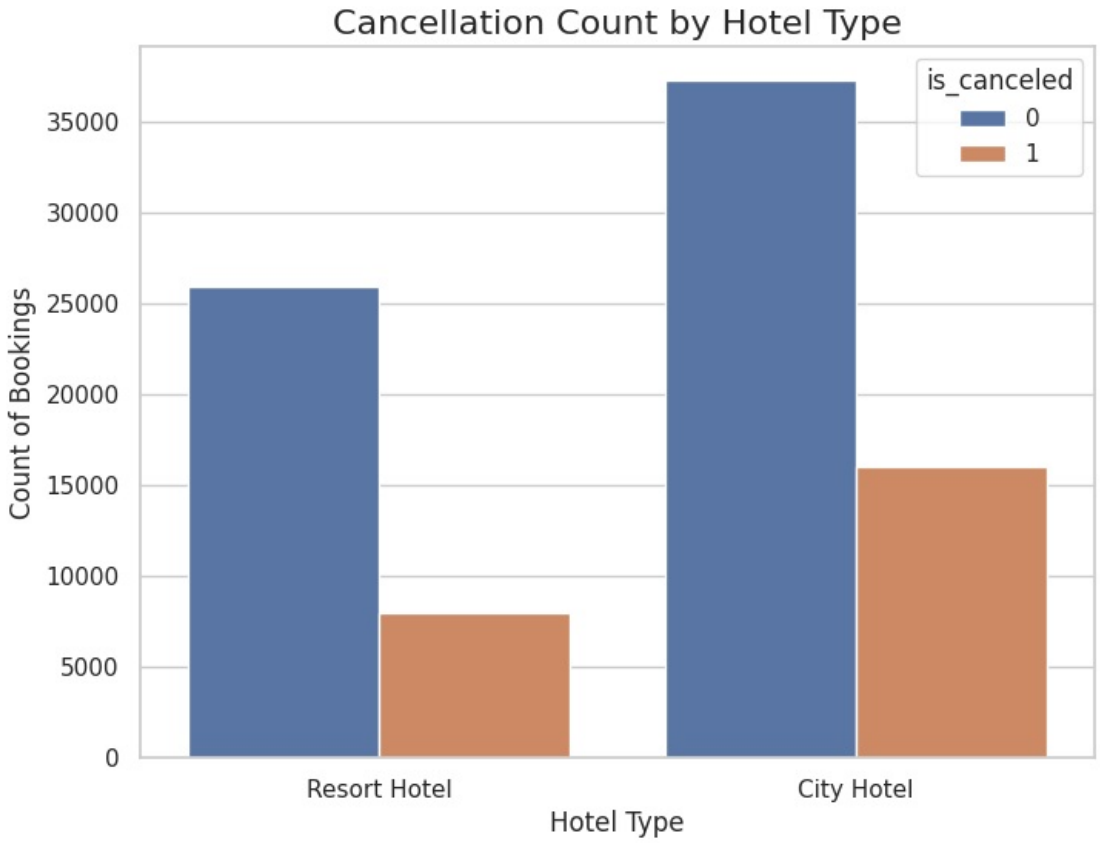
Cancellation Rate Changes : Higher cancellation rates in May and December may be related to specific seasonal factors such as weather changes or holiday arrangements. Similar Trends: The similarity in trends between reservations and cancellations suggests that both are likely influenced by similar external factors

---

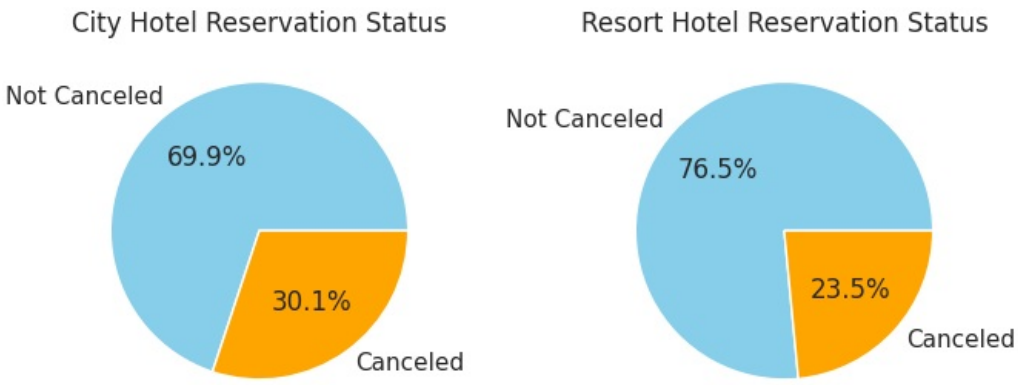
2- **Hotel**

What's the differences between the number of cancellation concerning types of hotel ?

In [64]:
```
sns.set(style="whitegrid")
plt.figure(figsize=(8, 6))
sns.countplot(x='hotel', hue='is_canceled', data=df11)

# Set the title and labels
plt.title('Cancellation Count by Hotel Type', fontsize=16)
plt.xlabel('Hotel Type', fontsize=12)
plt.ylabel('Count of Bookings', fontsize=12)
plt.show()
```



In [65]:
```
hotel_status = df11.groupby(['hotel', 'is_canceled']).size().reset_index(name='num_of_reservations')
hotel_status['status'] = hotel_status['is_canceled'].replace({0: 'Not Canceled', 1: 'Canceled'})
city_data = hotel_status[hotel_status['hotel'] == 'City Hotel']
resort_data = hotel_status[hotel_status['hotel'] == 'Resort Hotel']

# City Hotel Pie Chart
plt.figure(figsize=(7, 3))
plt.subplot(1, 2, 1)
plt.pie(city_data['num_of_reservations'], labels=city_data['status'], autopct='%1.1f%%', colors=['skyblue', 'ora
plt.title('City Hotel Reservation Status')

# Resort Hotel Pie Chart
plt.subplot(1, 2, 2)
plt.pie(resort_data['num_of_reservations'], labels=resort_data['status'], autopct='%1.1f%%', colors=['skyblue',
plt.title('Resort Hotel Reservation Status')
plt.tight_layout()
plt.show()
```

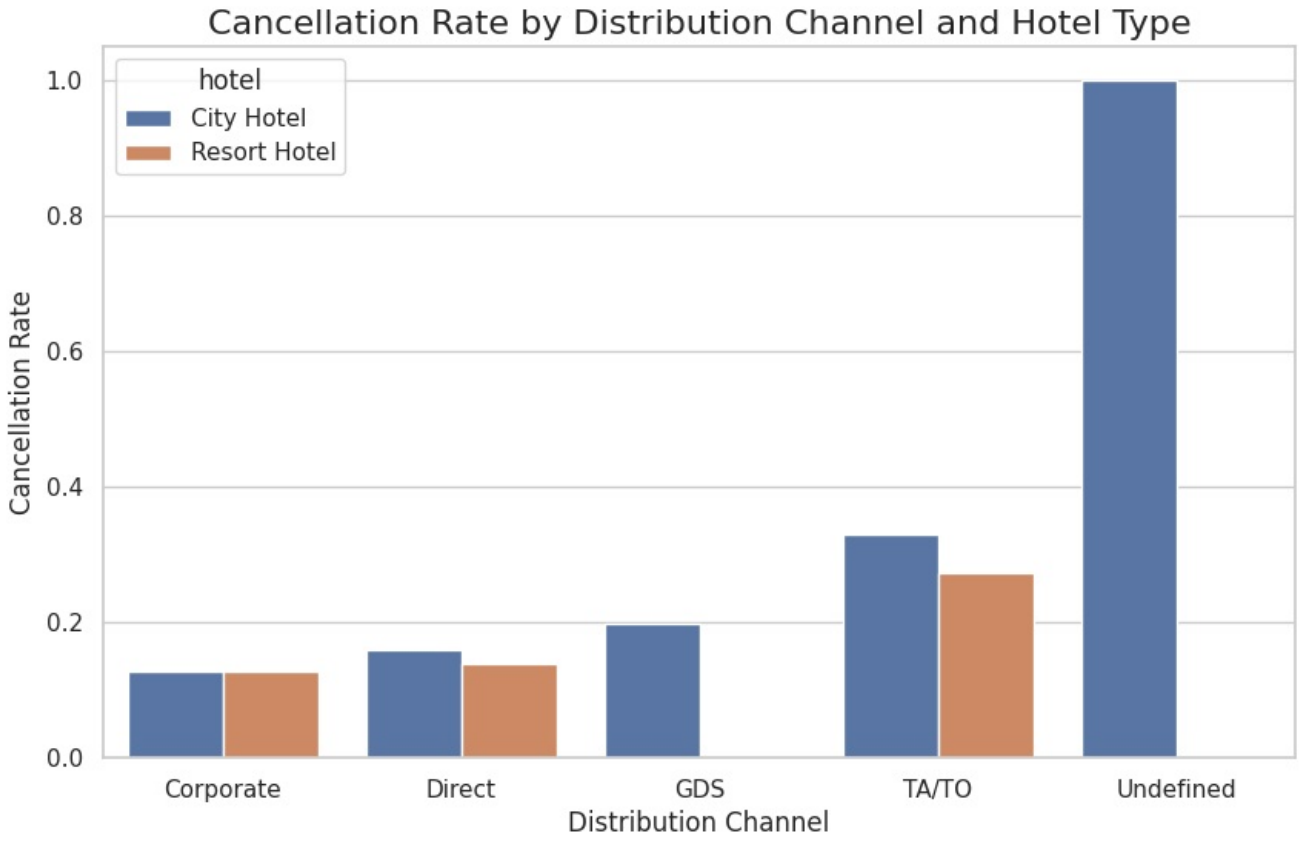


Cancellation Rate Comparison: City Hotels have a higher cancellation rate (30%) compared to Resort Hotels (24%). This suggests that City Hotels are more affected by cancellations, which could be due to various factors such as business travel fluctuations, easier access to alternative accommodations, or more competitive markets.

Volume of Bookings: Both hotel types have a higher volume of non-canceled bookings, indicating that the majority of customers do not cancel their reservations. However, the absolute number of canceled bookings in City Hotels is higher due to a larger total booking volume.

What's the impact of booking means on cancellation rate ?

In [66]:
```
# Set the plot style
sns.set(style="whitegrid")


channel_cancellation_rate = df11.groupby(['hotel', 'distribution_channel']).agg({'is_canceled': 'mean'}).reset_


plt.figure(figsize=(10, 6))


sns.barplot(x='distribution_channel', y='is_canceled', hue='hotel', data=channel_cancellation_rate)


plt.title('Cancellation Rate by Distribution Channel and Hotel Type', fontsize=16)
plt.xlabel('Distribution Channel', fontsize=12)
plt.ylabel('Cancellation Rate', fontsize=12)

plt.show()
```



Excluding the "Undefined" channel, the cancellation rates from other channels provide a clearer perspective, indicating that bookings through travel agencies/online travel agencies have a higher cancellation rate. Bookings made by corporate clients and through direct channels have lower cancellation rates, suggesting these customers are more likely to complete their reservations.

3- **location**

Where are most canceled orders from ?

In [67]:
```
cancel_by_country = df11[df11['is_canceled'] == 1]['country'].value_counts().reset_index()
cancel_by_country.columns = ['country', 'cancellation_num']
cancel_by_country
```

Out[67]:

	country	cancellation_num
0	PRT	9801
1	GBR	1982
2	ESP	1860
3	FRA	1732
4	ITA	1075
...	...	...
122	GLP	1
123	LIE	1
124	UMI	1
125	GNB	1
126	ETH	1

127 rows × 2 columns

In [68]:
```
total_canceled = cancel_by_country['cancellation_num'].sum()
top_10_canceled = cancel_by_country.head(10).copy()
top_10_canceled['percentage'] = (top_10_canceled['cancellation_num'] / total_canceled) * 100
top_10_canceled
```

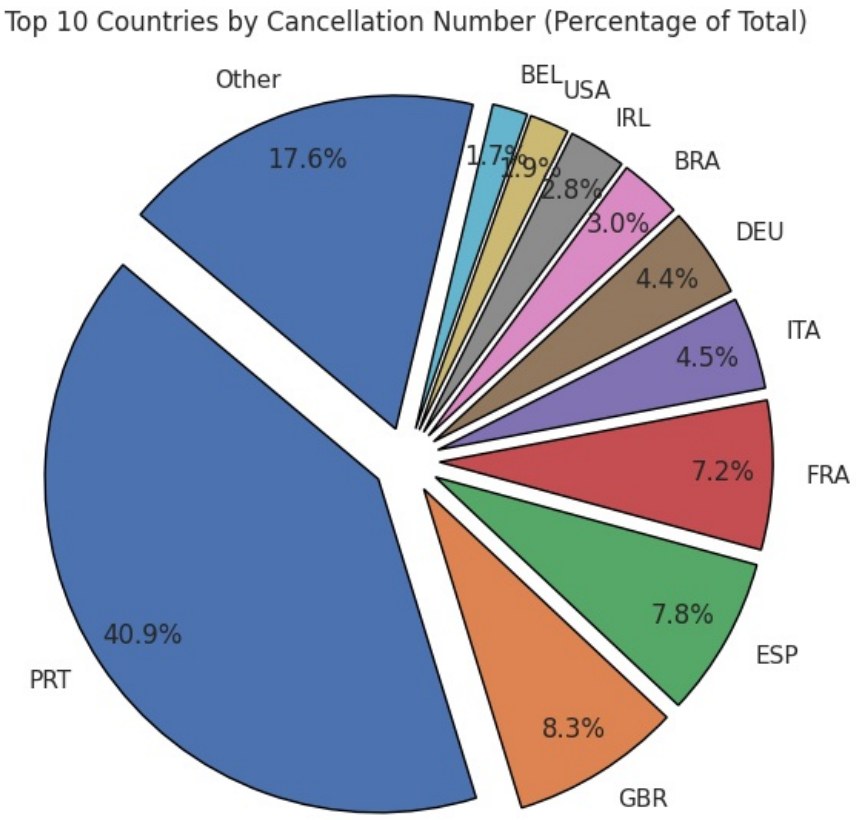
Out[68]:

	country	cancellation_num	percentage
0	PRT	9801	40.851117
1	GBR	1982	8.261087
2	ESP	1860	7.752584
3	FRA	1732	7.219073
4	ITA	1075	4.480660
5	DEU	1053	4.388963
6	BRA	726	3.026009
7	IRL	668	2.784261
8	USA	458	1.908970
9	BEL	411	1.713071

In [69]:
```
other_sum = cancel_by_country.loc[~cancel_by_country['country'].isin(top_10_canceled['country'])]['cancellation_

# Create a new DataFrame for the 'Other' category
other_data = pd.DataFrame([{'country': 'Other', 'cancellation_num': other_sum, 'percentage': other_sum / total_

# Use pd.concat to append the new row
top_10_canceled = pd.concat([top_10_canceled, other_data], ignore_index=True)
```

In [70]:
```
plt.figure(figsize=(6, 6))
explode = [0.1] * len(top_10_canceled)
plt.pie(
    top_10_canceled['percentage'],
    labels=top_10_canceled['country'],
    autopct='%1.1f%%',
    startangle=140,
    labeldistance=1.1,
    pctdistance=0.85,
    wedgeprops={'edgecolor': 'black'},
    explode=explode,
```

```
)

plt.title('Top 10 Countries by Cancellation Number (Percentage of Total)')
plt.tight_layout()
plt.show()
```



the top ten countries with highest cancellation rate are: Portugal, Great Britain, Spain, France, Italy, Germany, Ireland, Brazil, United States, Belgium the highest One is : Portugal with 40.9%

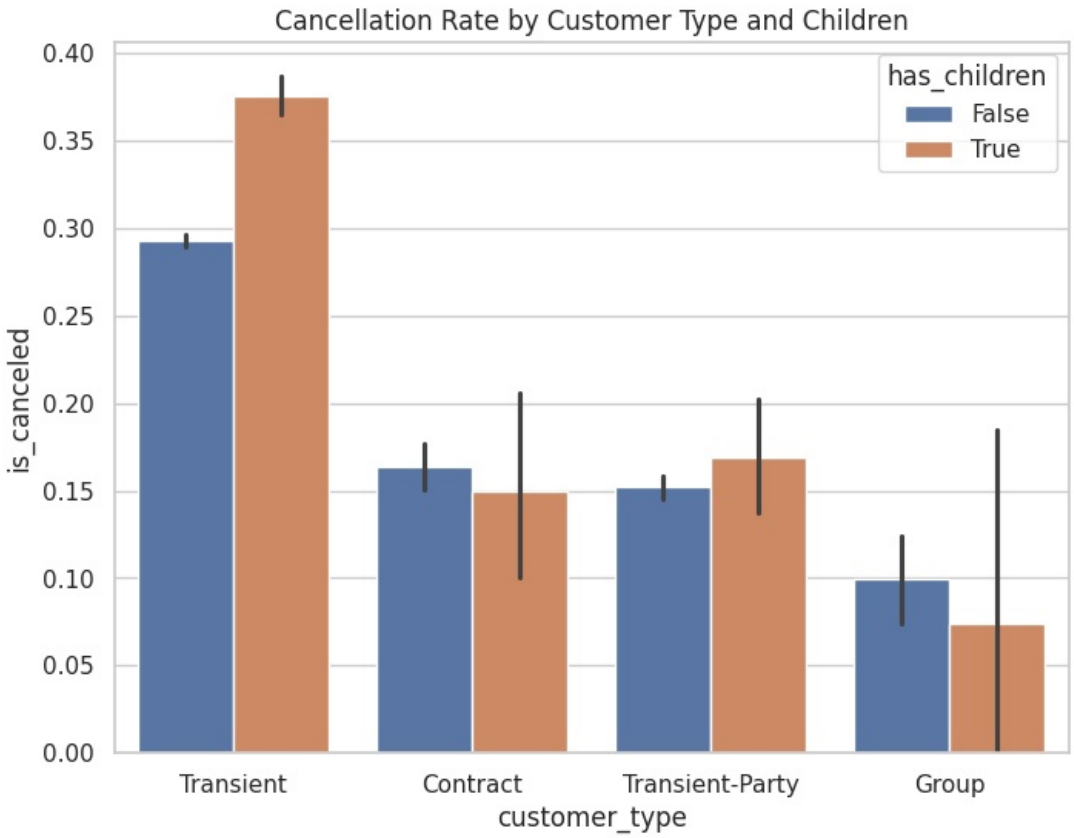
4- **Customer**

What's the relationship of customer type with cancellation rate ?

In [71]:
```
# Create a new column to determine if children are present
df11['has_children'] = df11['children'] > 0
cancel_rate = df11.groupby(['customer_type', 'has_children'])['is_canceled'].mean().reset_index()
print(cancel_rate)
```

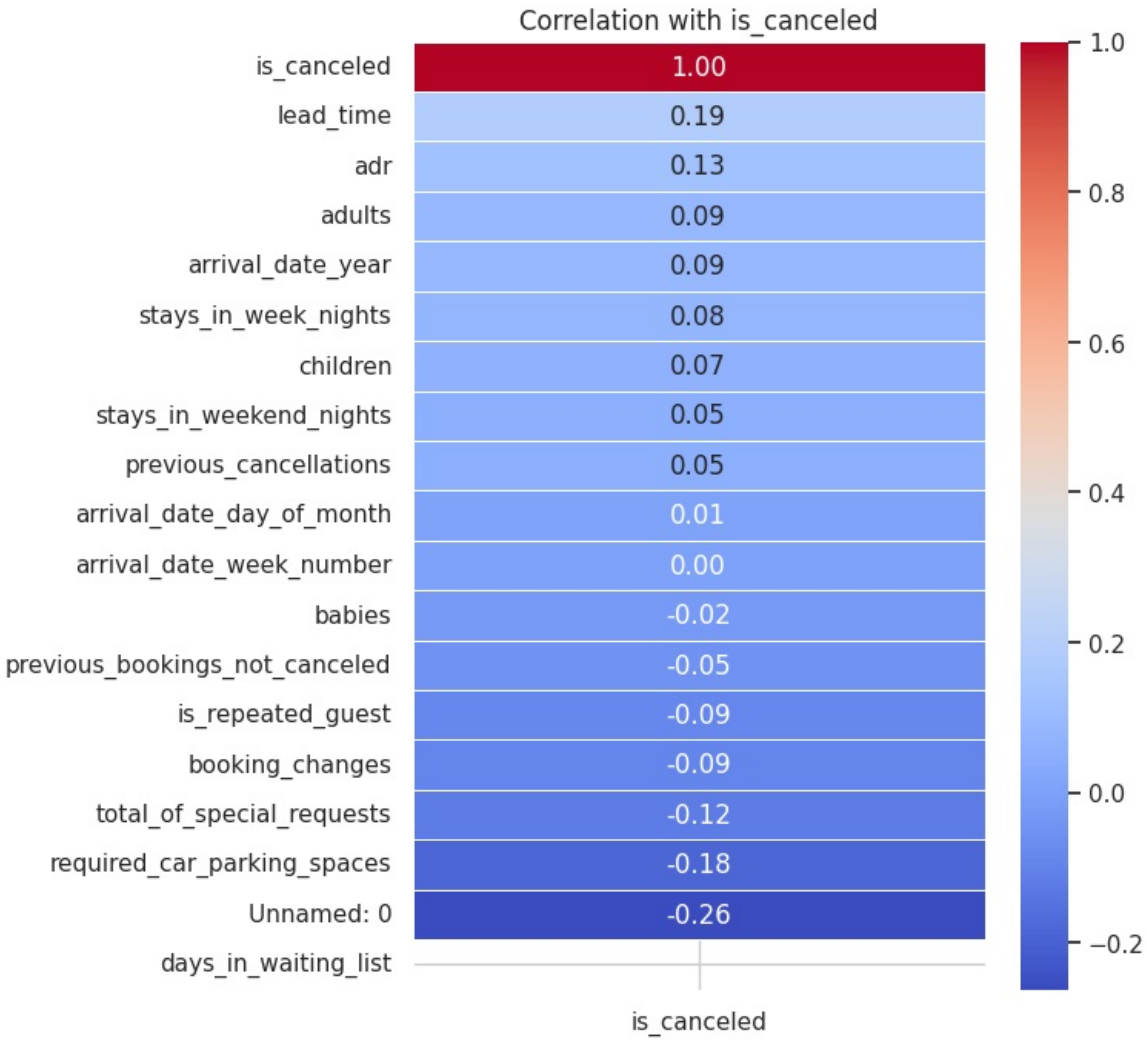
```
    customer_type  has_children  is_canceled
0        Contract         False     0.164034
1        Contract          True     0.150000
2           Group         False     0.099222
3           Group          True     0.074074
4       Transient         False     0.292615
5       Transient          True     0.375278
6  Transient-Party         False     0.152043
7  Transient-Party          True     0.168856
```

In [72]:
```
plt.figure(figsize=(8, 6))
sns.barplot(data=df11, x='customer_type', y='is_canceled', hue='has_children')
plt.title('Cancellation Rate by Customer Type and Children')
plt.show()
```



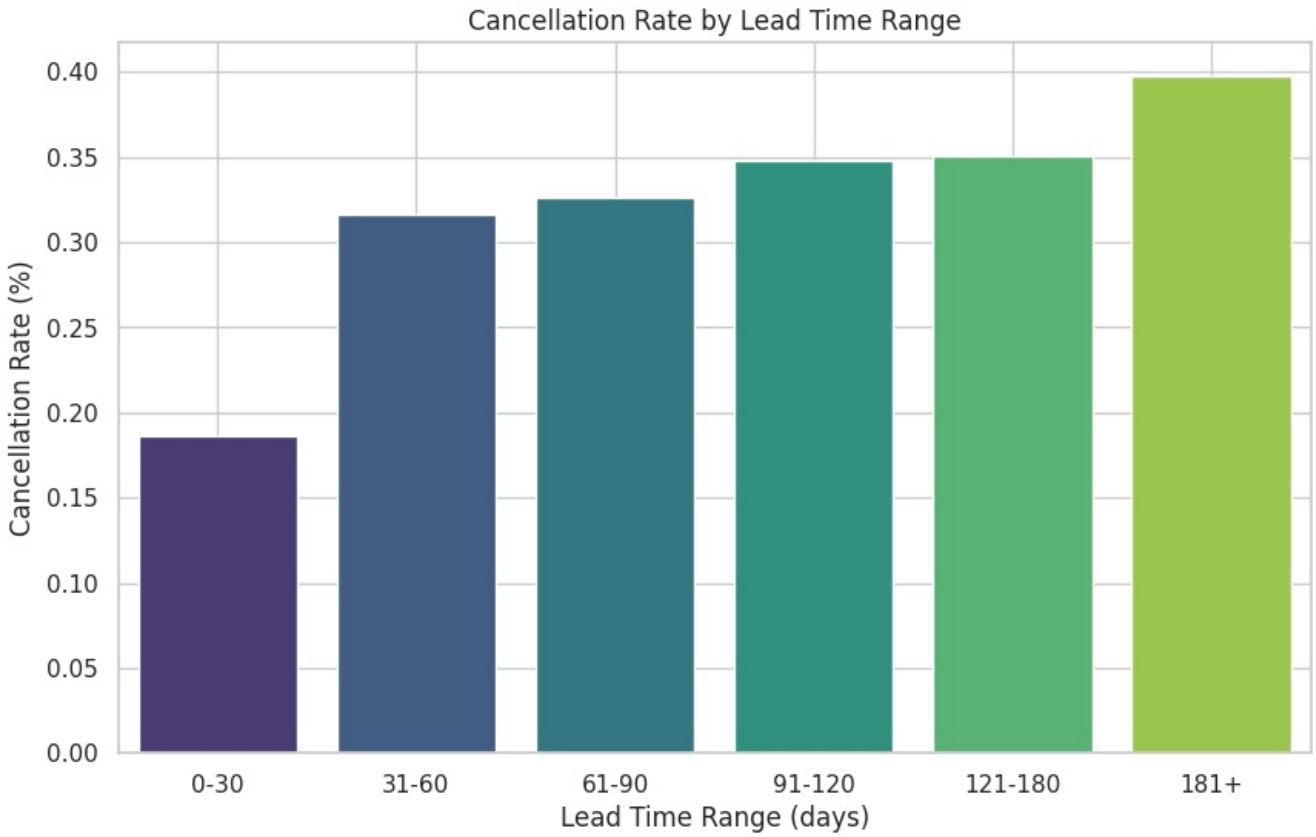
Customers with children (True) tend to have a slightly lower cancellation rate across all customer types compared to those without children (False), except for the 'Transient-Party' type where the cancellation rates are similar. Implication: This suggests that having children might be associated with a more committed booking behavior, possibly due to advanced planning needs for family trips.

In [73]:
```
# Select only numerical columns from the dataframe
numeric_df = df11.select_dtypes(include=['number'])

# Calculate the correlation matrix for numerical data
correlation_matrix = numeric_df.corr()

# Extract the correlation values
target_feature = 'is_canceled'
target_correlation = correlation_matrix[[target_feature]].sort_values(by=target_feature, ascending=False)
plt.figure(figsize=(6, 8))  #
sns.heatmap(target_correlation, annot=True, cmap='coolwarm', fmt='.2f', cbar=True, linewidths=0.5)
plt.title(f'Correlation with {target_feature}')
plt.show()
```



In [74]:

```
df['lead_time_range'] = pd.cut(df['lead_time'], bins=[0, 30, 60, 90, 120, 180, 365],
labels=['0-30', '31-60', '61-90', '91-120', '121-180', '181+'])

# Cancellation Rate by Range
lead_time_range_cancellation = df.groupby('lead_time_range', observed=False)['is_canceled'].agg(['count', 'mean
lead_time_range_cancellation.columns = ['lead_time_range', 'total_reservations', 'cancellation_rate']

plt.figure(figsize=(10, 6))
sns.barplot(data=lead_time_range_cancellation,
            x='lead_time_range',
            y='cancellation_rate',
            hue='lead_time_range',
            palette='viridis',
            legend=False)
plt.xlabel('Lead Time Range (days)')
plt.ylabel('Cancellation Rate (%)')
plt.title('Cancellation Rate by Lead Time Range')
plt.grid(True)
plt.show()
```



the longer the lead_time, the higher the cancellation rate

Modeling Part

In [75]:
```
df = pd.read_csv('hotel_bookings_cleaned.csv')
```

In [76]:
```
df = pd.read_csv('hotel_bookings_cleaned.csv')
```

In [77]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87154 entries, 0 to 87153
Data columns (total 31 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Unnamed: 0                      87154 non-null  int64
 1   hotel                           87154 non-null  object
 2   is_canceled                     87154 non-null  int64
 3   lead_time                       87154 non-null  float64
 4   arrival_date_year               87154 non-null  int64
 5   arrival_date_month              87154 non-null  object
 6   arrival_date_week_number        87154 non-null  int64
 7   arrival_date_day_of_month       87154 non-null  int64
 8   stays_in_weekend_nights         87154 non-null  int64
 9   stays_in_week_nights            87154 non-null  int64
 10  adults                          87154 non-null  int64
 11  children                        87154 non-null  float64
 12  babies                          87154 non-null  int64
 13  meal                            87154 non-null  object
 14  country                         87154 non-null  object
 15  market_segment                  87154 non-null  object
 16  distribution_channel            87154 non-null  object
 17  is_repeated_guest               87154 non-null  int64
 18  previous_cancellations          87154 non-null  int64
 19  previous_bookings_not_canceled  87154 non-null  int64
 20  reserved_room_type              87154 non-null  object
 21  assigned_room_type              87154 non-null  object
 22  booking_changes                 87154 non-null  int64
 23  deposit_type                    87154 non-null  object
 24  days_in_waiting_list            87154 non-null  int64
 25  customer_type                   87154 non-null  object
 26  adr                             87154 non-null  float64
 27  required_car_parking_spaces     87154 non-null  int64
 28  total_of_special_requests       87154 non-null  int64
 29  reservation_status              87154 non-null  object
 30  reservation_status_date         87154 non-null  object
dtypes: float64(3), int64(16), object(12)
memory usage: 20.6+ MB
```

In [78]:
```
df["is_canceled"].value_counts()
```

Out[78]:

is_canceled	count
0	63162
1	23992

dtype: int64

In [79]:
```
df.isnull().sum()
```

Out[79]:

	0
Unnamed: 0	0
hotel	0
is_canceled	0
lead_time	0
arrival_date_year	0
arrival_date_month	0
arrival_date_week_number	0
arrival_date_day_of_month	0
stays_in_weekend_nights	0
stays_in_week_nights	0
adults	0
children	0
babies	0
meal	0
country	0
market_segment	0
distribution_channel	0
is_repeated_guest	0
previous_cancellations	0
previous_bookings_not_canceled	0
reserved_room_type	0
assigned_room_type	0
booking_changes	0
deposit_type	0
days_in_waiting_list	0
customer_type	0
adr	0
required_car_parking_spaces	0
total_of_special_requests	0
reservation_status	0
reservation_status_date	0

dtype: int64

In [80]:
```
df['reservation_status_date'] = pd.to_datetime(df['reservation_status_date'])

df['reservation_status_date_year'] = df['reservation_status_date'].dt.year
df['reservation_status_date_month'] = df['reservation_status_date'].dt.month
df['reservation_status_date_day'] = df['reservation_status_date'].dt.day
```

In [81]:
```
df.drop(columns=['reservation_status_date'],inplace=True)
```

In [82]:
```
df.drop(columns=['Unnamed: 0'],inplace=True)
```

In [83]:
```
df.head()
```

Out[83]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	sta
0	Resort Hotel	0	342.00000	2015	July	27	1	
1	Resort Hotel	0	79.99159	2015	July	27	1	
2	Resort Hotel	0	7.00000	2015	July	27	1	
3	Resort Hotel	0	13.00000	2015	July	27	1	
4	Resort Hotel	0	14.00000	2015	July	27	1	

5 rows × 32 columns

In [84]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87154 entries, 0 to 87153
Data columns (total 32 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   hotel                           87154 non-null  object 
 1   is_canceled                     87154 non-null  int64  
 2   lead_time                       87154 non-null  float64
 3   arrival_date_year               87154 non-null  int64  
 4   arrival_date_month              87154 non-null  object 
 5   arrival_date_week_number        87154 non-null  int64  
 6   arrival_date_day_of_month       87154 non-null  int64  
 7   stays_in_weekend_nights         87154 non-null  int64  
 8   stays_in_week_nights            87154 non-null  int64  
 9   adults                          87154 non-null  int64  
 10  children                        87154 non-null  float64
 11  babies                          87154 non-null  int64  
 12  meal                            87154 non-null  object 
 13  country                         87154 non-null  object 
 14  market_segment                  87154 non-null  object 
 15  distribution_channel            87154 non-null  object 
 16  is_repeated_guest               87154 non-null  int64  
 17  previous_cancellations          87154 non-null  int64  
 18  previous_bookings_not_canceled  87154 non-null  int64  
 19  reserved_room_type              87154 non-null  object 
 20  assigned_room_type              87154 non-null  object 
 21  booking_changes                 87154 non-null  int64  
 22  deposit_type                    87154 non-null  object 
 23  days_in_waiting_list            87154 non-null  int64  
 24  customer_type                   87154 non-null  object 
 25  adr                             87154 non-null  float64
 26  required_car_parking_spaces     87154 non-null  int64  
 27  total_of_special_requests       87154 non-null  int64  
 28  reservation_status              87154 non-null  object 
 29  reservation_status_date_year    87154 non-null  int32  
 30  reservation_status_date_month   87154 non-null  int32  
 31  reservation_status_date_day     87154 non-null  int32  
dtypes: float64(3), int32(3), int64(15), object(11)
memory usage: 20.3+ MB
```

In [85]:
```
from sklearn.preprocessing import LabelEncoder

object_columns = df.select_dtypes(include=['object']).columns
label_encoders = {}
for column in object_columns:
    le = LabelEncoder()

    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
```

In [86]:
```
#df_pandas_encoded = pd.get_dummies(df, drop_first=True)
#df_pandas_encoded
```

In [87]:
```
df.head ()
```

Out[87]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays
0	1	0	342.00000	2015	5	27	1	
1	1	0	79.99159	2015	5	27	1	
2	1	0	7.00000	2015	5	27	1	
3	1	0	13.00000	2015	5	27	1	
4	1	0	14.00000	2015	5	27	1	

5 rows × 32 columns

In [88]:
```
plt.figure(figsize = (24, 12))

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

In [89]:
```
correlation = df.corr()['is_canceled'].abs().sort_values(ascending = False)
correlation
```

Out[89]:

	is_canceled
is_canceled	1.000000
reservation_status	0.888661
lead_time	0.186576
required_car_parking_spaces	0.184551
market_segment	0.182000
distribution_channel	0.150198
deposit_type	0.137287
adr	0.129073
total_of_special_requests	0.120648
reservation_status_date_month	0.098577
country	0.094347
booking_changes	0.093287
is_repeated_guest	0.088820
adults	0.088428
arrival_date_year	0.088187
stays_in_week_nights	0.082238
hotel	0.072260
children	0.066930
assigned_room_type	0.060867
stays_in_weekend_nights	0.054933
previous_bookings_not_canceled	0.052209
previous_cancellations	0.051512
reserved_room_type	0.047504
meal	0.046098
arrival_date_month	0.044297
customer_type	0.030588
babies	0.021618
reservation_status_date_day	0.012162
reservation_status_date_year	0.009921
arrival_date_day_of_month	0.005405
arrival_date_week_number	0.001591
days_in_waiting_list	NaN

dtype: float64

In [90]:
```
x = df[['reservation_status', 'lead_time', 'required_car_parking_spaces',
              'market_segment', 'distribution_channel', 'deposit_type', 'adr',
              'total_of_special_requests', 'reservation_status_date_month',
              'country', 'booking_changes', 'is_repeated_guest', 'adults',
              'arrival_date_year']].copy().values
y=df['is_canceled'].copy().values
```

In [91]:
```
from sklearn.model_selection import train_test_split, cross_val_score
x_train , x_test , y_train , y_test = train_test_split(x,y , test_size= 0.25 ,random_state=42)
```

In [92]:
```
from imblearn.over_sampling import SMOTE
smote = SMOTE(random_state=42)
x_train, y_train = smote.fit_resample(x_train, y_train)
```

In [93]:
```
from sklearn.preprocessing import RobustScaler
ro_scaler = RobustScaler()
x_train = ro_scaler.fit_transform(x_train)
x_test = ro_scaler.fit_transform(x_test)
```

In [94]:
```
from sklearn.preprocessing import StandardScaler
ros_scaler = StandardScaler()
x_train = ros_scaler.fit_transform(x_train)
x_test = ros_scaler.fit_transform(x_test)
```

In [95]:
```
from sklearn.svm import SVC

svc_linear=SVC(kernel='linear',C=1,gamma=0.01)
svc_clf_linear = svc_linear.fit(x_train,y_train)
svc_clf_linear.score(x_train,y_train)
```

Out[95]: 0.9758700941500053

In [96]:
```
svc_clf_linear.score(x_test,y_test)
```

Out[96]: 0.9880673734453165

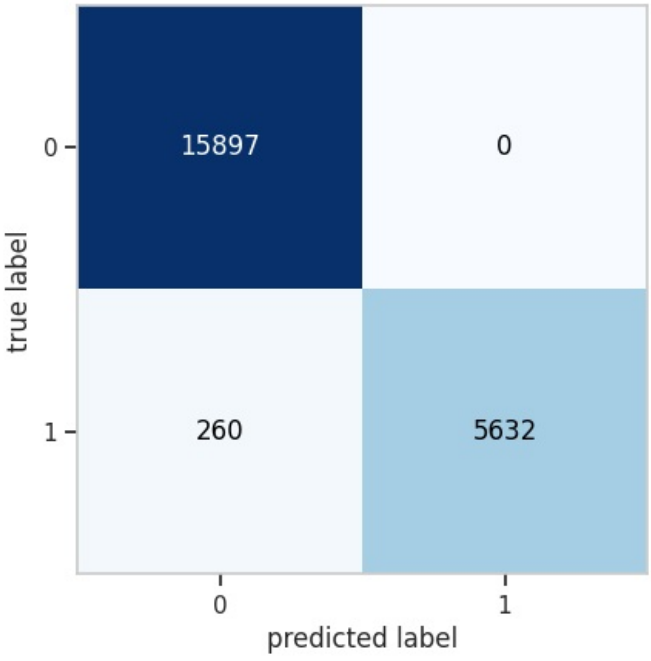
In [97]:
```
from sklearn import metrics
y_pred = svc_clf_linear.predict(x_test)

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

print("precition:", metrics.precision_score(y_test, y_pred))

print("f1 score:", metrics.f1_score(y_test, y_pred))

print("recall:", metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.9880673734453165
precition: 1.0
f1 score: 0.9774383894481083
recall: 0.9558723693143245
```

In [98]:
```
from sklearn.metrics import confusion_matrix
from mlxtend.plotting import plot_confusion_matrix
cm = confusion_matrix(y_test, svc_clf_linear.predict(x_test))
print(svc_clf_linear.score(x_test, y_test))
plot_confusion_matrix(cm)
```

```
0.9880673734453165
```

Out[98]: (<Figure size 640x480 with 1 Axes>,
<Axes: xlabel='predicted label', ylabel='true label'>)



In [99]:
```
from xgboost import XGBClassifier
xgb = XGBClassifier()
xgbmodel=xgb.fit(x_train, y_train)
xgbmodel.score(x_train,y_train)
```

Out[99]: 0.9999894213477203

In [100…
```
xgbmodel.score(x_test , y_test)
```

Out[100… 0.2704116756161366

In [101…
```
from sklearn import metrics
y_pred = xgbmodel.predict(x_test)

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

print("precition:", metrics.precision_score(y_test, y_pred))
```

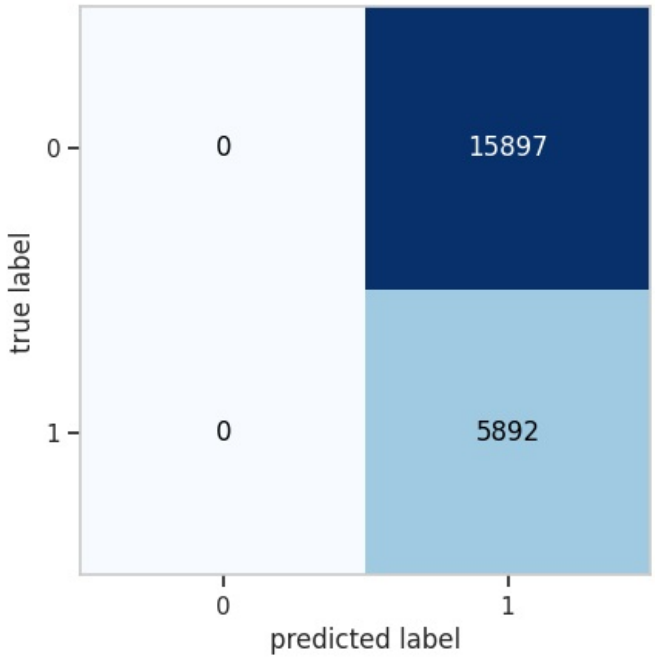
```
print("f1 score:", metrics.f1_score(y_test, y_pred))

print("recall:", metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.2704116756161366
precition: 0.2704116756161366
f1 score: 0.42570716375853473
recall: 1.0
```

In [102…
```
from sklearn.metrics import confusion_matrix
from mlxtend.plotting import plot_confusion_matrix
cm = confusion_matrix(y_test, xgbmodel.predict(x_test))
print(xgbmodel.score(x_test, y_test))
plot_confusion_matrix(cm)
```

```
0.2704116756161366
```

Out[102… (<Figure size 640x480 with 1 Axes>,
<Axes: xlabel='predicted label', ylabel='true label'>)



In [103…
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knnmodel=knn.fit(x_train, y_train)
knnmodel.score(x_train,y_train)
```

Out[103… 0.9944991008145563

In [104…
```
knnmodel.score(x_test , y_test)
```

Out[104… 0.9983477901693515

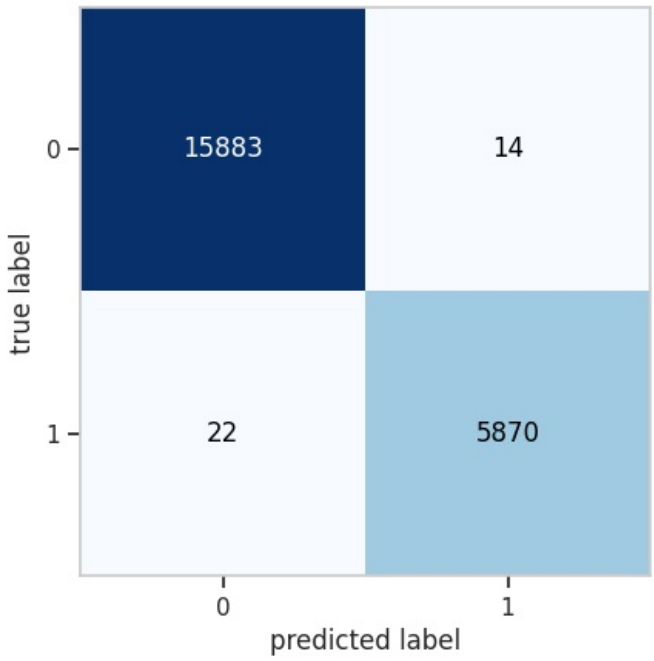
In [105…
```
from sklearn import metrics
y_pred = knnmodel.predict(x_test)

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

print("precition:", metrics.precision_score(y_test, y_pred))

print("f1 score:", metrics.f1_score(y_test, y_pred))

print("recall:", metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.9983477901693515
precition: 0.9976206662134602
f1 score: 0.9969429347826086
recall: 0.9962661235573659
```

In [106…
```
from sklearn.metrics import confusion_matrix
from mlxtend.plotting import plot_confusion_matrix
cm = confusion_matrix(y_test, knnmodel.predict(x_test))
print(knnmodel.score(x_test, y_test))
plot_confusion_matrix(cm)
```

```
0.9983477901693515
```

Out[106… (<Figure size 640x480 with 1 Axes>,
<Axes: xlabel='predicted label', ylabel='true label'>)



In [107…
```
from sklearn.linear_model import LogisticRegression
Lg = LogisticRegression()
Logisticmodel=Lg.fit(x_train, y_train)
Logisticmodel.score(x_train,y_train)
```

Out[107… 0.9769068020734158

In [108…
```
Logisticmodel.score(x_test , y_test)
```

Out[108… 0.9838450594336592

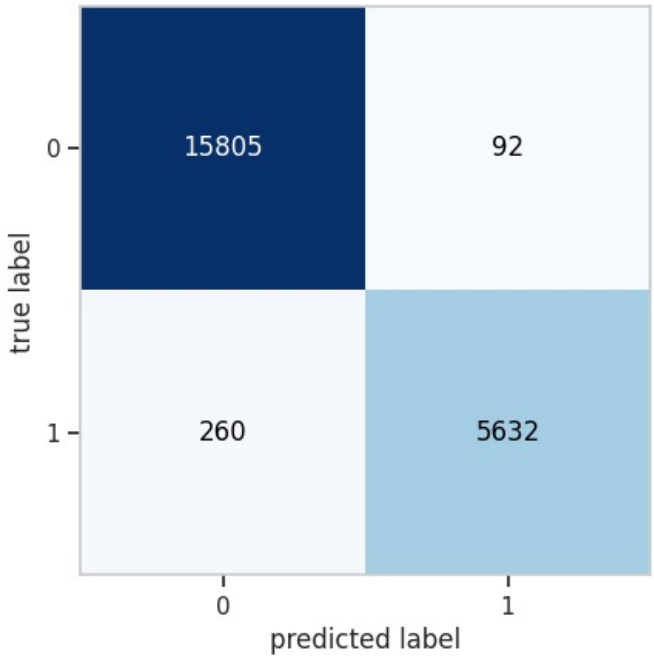
In [109…
```
from sklearn import metrics
y_pred = Logisticmodel.predict(x_test)

print("Accuracy:", metrics.accuracy_score(y_test, y_pred))

print("precition:", metrics.precision_score(y_test, y_pred))

print("f1 score:", metrics.f1_score(y_test, y_pred))

print("recall:", metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.9838450594336592
precition: 0.983927323549965
f1 score: 0.9696969696969697
recall: 0.9558723693143245
```

In [110…
```
from sklearn.metrics import confusion_matrix
from mlxtend.plotting import plot_confusion_matrix
cm = confusion_matrix(y_test, Logisticmodel.predict(x_test))
print(Logisticmodel.score(x_test, y_test))
plot_confusion_matrix(cm)
```

```
0.9838450594336592
```

Out[110… (<Figure size 640x480 with 1 Axes>,
<Axes: xlabel='predicted label', ylabel='true label'>)



THANK YOU