



# IMPORTING THE LIBRARIES

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

## looking at the dataset

```
In [6]: df.head()
```

```
Out[6]:
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_d
0	Resort Hotel	0	342	2015	July	
1	Resort Hotel	0	737	2015	July	
2	Resort Hotel	0	7	2015	July	
3	Resort Hotel	0	13	2015	July	
4	Resort Hotel	0	14	2015	July	

5 rows × 32 columns

```
In [7]: df.shape
```

```
Out[7]: (119390, 32)
```

```
In [8]: 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 [9]: df.describe().T
```

Out[9]:

	count	mean	std	min	
<b>is_canceled</b>	119390.0	0.370416	0.482918	0.00	
<b>lead_time</b>	119390.0	104.011416	106.863097	0.00	1
<b>arrival_date_year</b>	119390.0	2016.156554	0.707476	2015.00	201
<b>arrival_date_week_number</b>	119390.0	27.165173	13.605138	1.00	1
<b>arrival_date_day_of_month</b>	119390.0	15.798241	8.780829	1.00	
<b>stays_in_weekend_nights</b>	119390.0	0.927599	0.998613	0.00	
<b>stays_in_week_nights</b>	119390.0	2.500302	1.908286	0.00	
<b>adults</b>	119390.0	1.856403	0.579261	0.00	
<b>children</b>	119386.0	0.103890	0.398561	0.00	
<b>babies</b>	119390.0	0.007949	0.097436	0.00	
<b>is_repeated_guest</b>	119390.0	0.031912	0.175767	0.00	
<b>previous_cancellations</b>	119390.0	0.087118	0.844336	0.00	
<b>previous_bookings_not_canceled</b>	119390.0	0.137097	1.497437	0.00	
<b>booking_changes</b>	119390.0	0.221124	0.652306	0.00	
<b>agent</b>	103050.0	86.693382	110.774548	1.00	
<b>company</b>	6797.0	189.266735	131.655015	6.00	6
<b>days_in_waiting_list</b>	119390.0	2.321149	17.594721	0.00	
<b>adr</b>	119390.0	101.831122	50.535790	-6.38	6
<b>required_car_parking_spaces</b>	119390.0	0.062518	0.245291	0.00	
<b>total_of_special_requests</b>	119390.0	0.571363	0.792798	0.00	

## TREATING MISSING VALUES

- cleaning the dataset

In [11]: `df.isna().sum()`

```
Out[11]: 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 [12]: def data_clean(df):
         data = df.fillna(0,inplace=True)
         print(df.isna().sum())
```

```
In [13]: data_clean(df)
```

```

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
agent 0
company 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]: list_cols = ["children", "adults", "babies"]

for i in list_cols:
    print(f"{i} has unique values as {df[i].unique()}")

```

```

children has unique values as [ 0.  1.  2. 10.  3.]
adults has unique values as [ 2  1  3  4 40 26 50 27 55  0 20  6  5 10]
babies has unique values as [ 0  1  2 10  9]

```

```

In [15]: filtered_data = (df["children"] ==0) & (df["adults"]==0) & (df["babies"]==0)
final_data = df[~filtered_data]

```

```

In [16]: final_data.shape

```

```

Out[16]: (119210, 32)

```

# Data analysis

```
In [18]: ## Where do the guest come from?  
country_wise_data = final_data[final_data["is_canceled"] == 0][["country"]].value  
country_wise_data.columns = ["country", "No.of guests"]  
print(country_wise_data)
```

	country	No.of guests
0	PRT	20977
1	GBR	9668
2	FRA	8468
3	ESP	6383
4	DEU	6067
..	...	...
161	BHR	1
162	DJI	1
163	MLI	1
164	NPL	1
165	FR0	1

[166 rows x 2 columns]

```
In [19]: import plotly.express as px
```

```
In [20]: map_guests = px.choropleth(country_wise_data, locations = country_wise_data["c  
color = country_wise_data["No.of guests"],  
hover_name = country_wise_data["country"],  
title = "Home country of guests")  
map_guests.show()
```

How much do guests pay for a room per night?

```
In [22]: final_data["hotel"].unique()
```

```
Out[22]: array(['Resort Hotel', 'City Hotel'], dtype=object)
```

```
In [23]: final_data
```

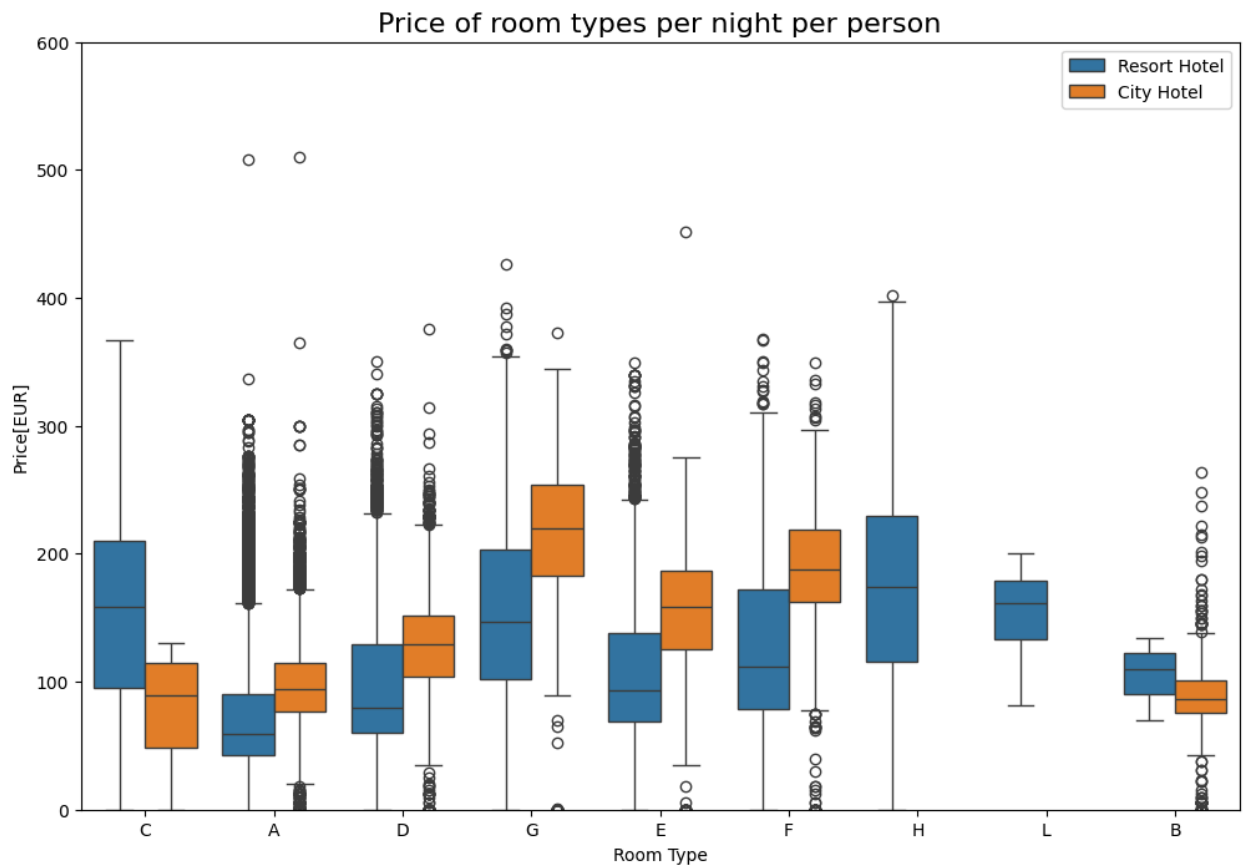
Out[23]:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	adr
<b>0</b>	Resort Hotel	0	342	2015	July	112.5
<b>1</b>	Resort Hotel	0	737	2015	July	112.5
<b>2</b>	Resort Hotel	0	7	2015	July	112.5
<b>3</b>	Resort Hotel	0	13	2015	July	112.5
<b>4</b>	Resort Hotel	0	14	2015	July	112.5
...	...	...	...	...	...	...
<b>119385</b>	City Hotel	0	23	2017	August	112.5
<b>119386</b>	City Hotel	0	102	2017	August	112.5
<b>119387</b>	City Hotel	0	34	2017	August	112.5
<b>119388</b>	City Hotel	0	109	2017	August	112.5
<b>119389</b>	City Hotel	0	205	2017	August	112.5

119210 rows × 32 columns

```
In [24]: data = final_data[final_data["is_canceled"] ==0]
```

```
In [25]: ## boxplot
plt.figure(figsize = (12,8))
sns.boxplot(x= "reserved_room_type", y = "adr", hue = "hotel", data=data)
plt.title("Price of room types per night per person", fontsize = 16)
plt.xlabel("Room Type")
plt.ylabel("Price[EUR]")
plt.legend(loc = "upper right")
plt.ylim(0,600)
plt.show()
```



How does the price per night (adr) vary over the year?

```
In [27]: data_resort = final_data[(final_data["hotel"] == "Resort Hotel") & (final_data["is"] == "Resort Hotel")]
data_city = final_data[(final_data["hotel"] == "City Hotel") & (final_data["is"] == "City Hotel")]
```

```
In [28]: # resort hotel: variable: data_resort
resort_hotel = data_resort.groupby(["arrival_date_month"])["adr"].mean().reset_index()
```

```
In [29]: # city hotel: variable: data_resort
city_hotel = data_city.groupby(["arrival_date_month"])["adr"].mean().reset_index()
```

```
In [30]: final = resort_hotel.merge(city_hotel, on = "arrival_date_month")
final.columns = ['month', 'price for resort hotel', 'price for city hotel']
print(final)
```

	month	price for resort hotel	price for city hotel
0	April	75.867816	111.962267
1	August	181.205892	118.674598
2	December	68.410104	88.401855
3	February	54.147478	86.520062
4	January	48.761125	82.330983
5	July	150.122528	115.818019
6	June	107.974850	117.874360
7	March	57.056838	90.658533
8	May	76.657558	120.669827
9	November	48.706289	86.946592
10	October	61.775449	102.004672
11	September	96.416860	112.776582

## sorting month

```
In [32]: from calendar import month_name
        for i, name in enumerate(month_name):
            print(i, name)
```

```
0
1 January
2 February
3 March
4 April
5 May
6 June
7 July
8 August
9 September
10 October
11 November
12 December
```

```
In [33]: from calendar import month_name
        def sort_month(df, colname):
            month_dict = {j:i for i, j in enumerate(month_name)}
            df["month_num"] = df[colname].apply(lambda x: month_dict[x])
            return df.sort_values(by = "month_num").reset_index().drop(['index', 'mont
```

```
In [34]: sort_month(final, "month")
```

Out[34]:

	month	price for resort hotel	price for city hotel
0	January	48.761125	82.330983
1	February	54.147478	86.520062
2	March	57.056838	90.658533
3	April	75.867816	111.962267
4	May	76.657558	120.669827
5	June	107.974850	117.874360
6	July	150.122528	115.818019
7	August	181.205892	118.674598
8	September	96.416860	112.776582
9	October	61.775449	102.004672
10	November	48.706289	86.946592
11	December	68.410104	88.401855

In [35]: `final.plot(kind = "line", x = "month", y = ['price for resort hotel', 'price f`

Out[35]: `<Axes: xlabel='month'>`



## Which are the most busy month?

```
In [37]: data_resort.head()
```

```
Out[37]:
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date
0	Resort Hotel	0	342	2015	July	2015-07-01
1	Resort Hotel	0	737	2015	July	2015-07-01
2	Resort Hotel	0	7	2015	July	2015-07-01
3	Resort Hotel	0	13	2015	July	2015-07-01
4	Resort Hotel	0	14	2015	July	2015-07-01

5 rows × 7 columns

```
In [38]: rush_resort = data_resort["arrival_date_month"].value_counts().reset_index()
rush_resort.columns = ["month", "No of guest in resort hotel"]
print(rush_resort)
```

	month	No of guest in resort hotel
0	August	3257
1	July	3137
2	October	2575
3	March	2571
4	April	2550
5	May	2535
6	February	2308
7	September	2102
8	June	2037
9	December	2014
10	November	1975
11	January	1866

```
In [39]: rush_city = data_city["arrival_date_month"].value_counts().reset_index()
rush_city.columns = ["month", "No of guest in city hotel"]
print(rush_city)
```

	month	No of guest in city hotel
0	August	5367
1	July	4770
2	May	4568
3	June	4358
4	October	4326
5	September	4283
6	March	4049
7	April	4010
8	February	3051
9	November	2676
10	December	2377
11	January	2249

```
In [40]: ## merging two dataframes
final_rush = rush_resort.merge(rush_city,on = "month")
print(final_rush)
```

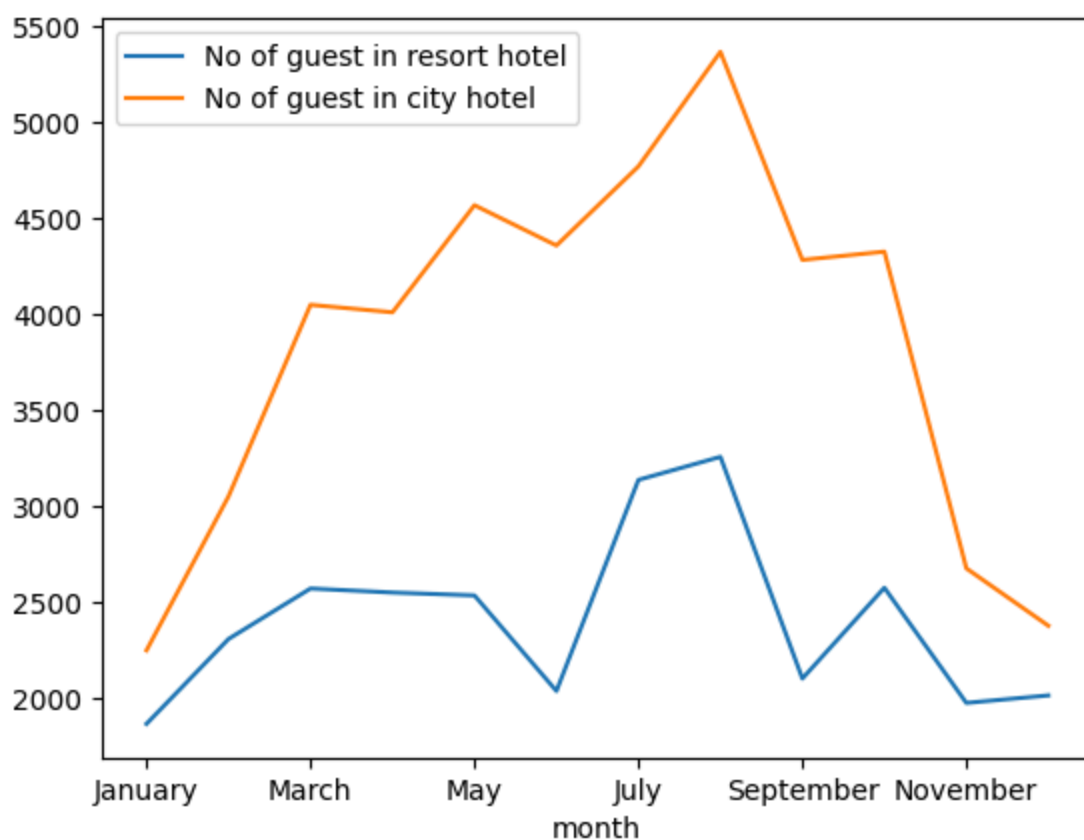
	month	No of guest in resort hotel	No of guest in city hotel
0	August	3257	5367
1	July	3137	4770
2	October	2575	4326
3	March	2571	4049
4	April	2550	4010
5	May	2535	4568
6	February	2308	3051
7	September	2102	4283
8	June	2037	4358
9	December	2014	2377
10	November	1975	2676
11	January	1866	2249

```
In [41]: final_month_sort = sort_month(final_rush,"month")
print(final_month_sort)
```

	month	No of guest in resort hotel	No of guest in city hotel
0	January	1866	2249
1	February	2308	3051
2	March	2571	4049
3	April	2550	4010
4	May	2535	4568
5	June	2037	4358
6	July	3137	4770
7	August	3257	5367
8	September	2102	4283
9	October	2575	4326
10	November	1975	2676
11	December	2014	2377

```
In [42]: final_month_sort.plot(kind = "line", x = "month",
                                y = ["No of guest in resort hotel", "No of guest in city
```

Out[42]: <Axes: xlabel='month'>



## How long do people stay on the hotel

```
In [44]: filter_condition = final_data["is_canceled"] == 0
clean_data = final_data[filter_condition]
```

```
In [45]: clean_data.head()
```

```
Out[45]:
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date
0	Resort Hotel	0	342	2015	July	
1	Resort Hotel	0	737	2015	July	
2	Resort Hotel	0	7	2015	July	
3	Resort Hotel	0	13	2015	July	
4	Resort Hotel	0	14	2015	July	

5 rows × 32 columns

```
In [46]: clean_data["total_nights"] = clean_data["stays_in_weekend_nights"] + clean_data["stays_in_week_nights"]
```

C:\Users\NYB\COMPUTER\AppData\Local\Temp\ipykernel\_4316\2600119523.py:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
In [47]: clean_data.head()
```

```
Out[47]:
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_day_of_month
--	-------	-------------	-----------	-------------------	--------------------	---------------------------

0	Resort Hotel	0	342	2015	July	1
1	Resort Hotel	0	737	2015	July	1
2	Resort Hotel	0	7	2015	July	1
3	Resort Hotel	0	13	2015	July	1
4	Resort Hotel	0	14	2015	July	1

5 rows x 33 columns

```
In [48]: stay = clean_data.groupby(["total_nights", "hotel"]).agg("count").reset_index()
stay = stay.iloc[:, 0:3]
print(stay)
```

	total_nights	hotel	is_canceled
0	0	City Hotel	251
1	0	Resort Hotel	371
2	1	City Hotel	9155
3	1	Resort Hotel	6579
4	2	City Hotel	10983
..	...	...	...
57	46	Resort Hotel	1
58	48	City Hotel	1
59	56	Resort Hotel	1
60	60	Resort Hotel	1
61	69	Resort Hotel	1

[62 rows x 3 columns]

```
In [49]: ## renaming a column
stay = stay.rename(columns = {"is_canceled": "No of stays"})
```

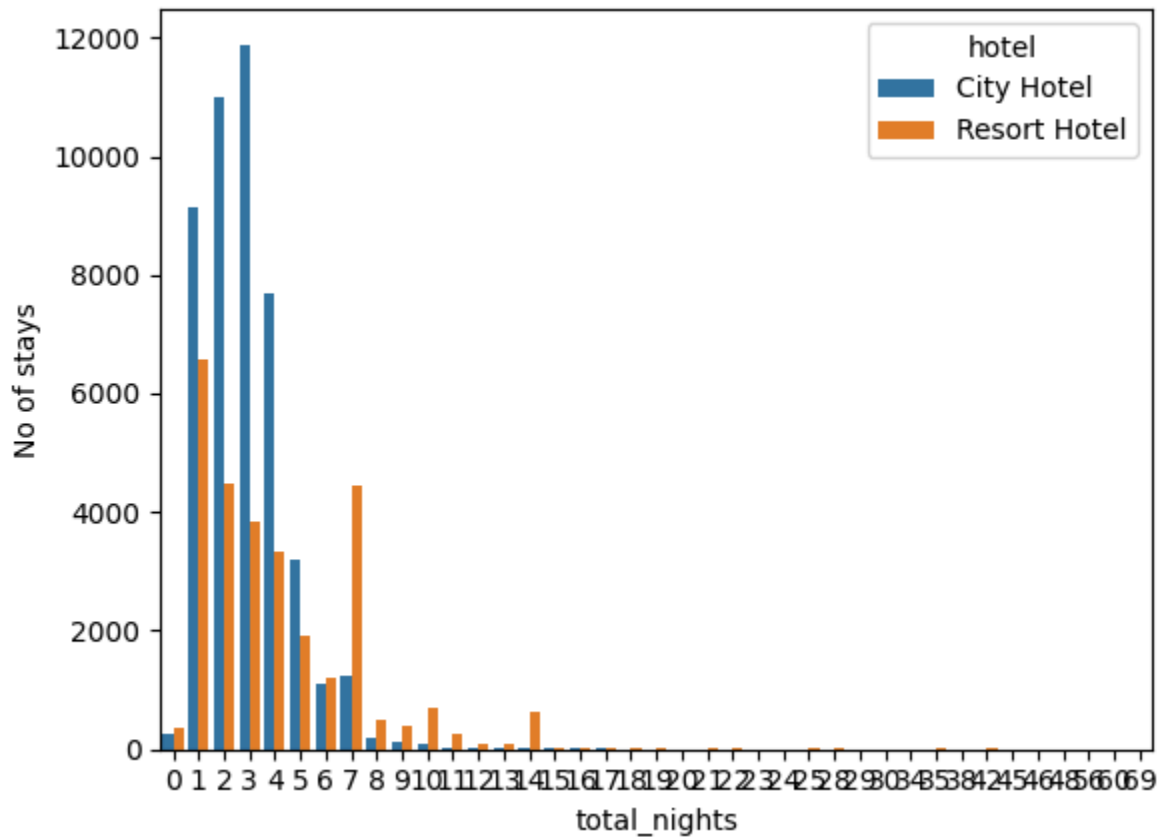
```
print(stay)
```

	total_nights	hotel	No of stays
0	0	City Hotel	251
1	0	Resort Hotel	371
2	1	City Hotel	9155
3	1	Resort Hotel	6579
4	2	City Hotel	10983
..	...	...	...
57	46	Resort Hotel	1
58	48	City Hotel	1
59	56	Resort Hotel	1
60	60	Resort Hotel	1
61	69	Resort Hotel	1

[62 rows x 3 columns]

```
In [50]: ## plotting
sns.barplot(x = "total_nights", y = "No of stays", hue = "hotel",
            hue_order = ["City Hotel", "Resort Hotel"], data = stay)
```

Out[50]: <Axes: xlabel='total\_nights', ylabel='No of stays'>



# Selecting Important numerical features using correlation

```
In [52]: correlation = final_data.corr()  
correlation
```

C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel\_4316\3741488391.py:1: FutureWarning:

The default value of numeric\_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric\_only to silence this warning.

```
Out[52]:
```

	<b>is_canceled</b>	<b>lead_time</b>	<b>arrival_date_year</b>	<b>arrival_date_week_number</b>
<b>is_canceled</b>	1.000000	0.292876	0.016622	0.008315
<b>lead_time</b>	0.292876	1.000000	0.040334	0.127046
<b>arrival_date_year</b>	0.016622	0.040334	1.000000	-0.540373
<b>arrival_date_week_number</b>	0.008315	0.127046	-0.540373	1.000000
<b>arrival_date_day_of_month</b>	-0.005948	0.002306	-0.000121	0.002306
<b>stays_in_weekend_nights</b>	-0.001323	0.085985	0.021694	0.085985
<b>stays_in_week_nights</b>	0.025542	0.166892	0.031203	0.166892
<b>adults</b>	0.058182	0.117575	0.030266	0.117575
<b>children</b>	0.004851	-0.037878	0.054710	-0.037878
<b>babies</b>	-0.032569	-0.021003	-0.013192	-0.021003
<b>is_repeated_guest</b>	-0.083745	-0.123209	0.010281	-0.123209
<b>previous_cancellations</b>	0.110139	0.086025	-0.119905	0.086025
<b>previous_bookings_not_canceled</b>	-0.057365	-0.073599	0.029234	-0.073599
<b>booking_changes</b>	-0.144832	0.002230	0.031416	0.002230
<b>agent</b>	-0.046770	-0.013114	0.056438	-0.013114
<b>company</b>	-0.083594	-0.085854	0.033682	-0.085854
<b>days_in_waiting_list</b>	0.054301	0.170008	-0.056348	0.170008
<b>adr</b>	0.046492	-0.065018	0.198429	-0.065018
<b>required_car_parking_spaces</b>	-0.195701	-0.116624	-0.013812	-0.116624
<b>total_of_special_requests</b>	-0.234877	-0.095949	0.108610	-0.095949

```
In [53]: correlation = correlation["is_canceled"][1:]
```

```
In [54]: correlation.abs().sort_values(ascending = False)
```

```
Out[54]: lead_time                0.292876
total_of_special_requests        0.234877
required_car_parking_spaces      0.195701
booking_changes                  0.144832
previous_cancellations           0.110139
is_repeated_guest                0.083745
company                          0.083594
adults                           0.058182
previous_bookings_not_canceled   0.057365
days_in_waiting_list            0.054301
agent                            0.046770
adr                              0.046492
babies                           0.032569
stays_in_week_nights             0.025542
arrival_date_year                0.016622
arrival_date_week_number         0.008315
arrival_date_day_of_month        0.005948
children                         0.004851
stays_in_weekend_nights          0.001323
Name: is_canceled, dtype: float64
```

```
In [55]: [col for col in final_data.columns if final_data[col].dtype != "0"] ## for num
```

```
Out[55]: ['is_canceled',
'lead_time',
'arrival_date_year',
'arrival_date_week_number',
'arrival_date_day_of_month',
'stays_in_weekend_nights',
'stays_in_week_nights',
'adults',
'children',
'babies',
'is_repeated_guest',
'previous_cancellations',
'previous_bookings_not_canceled',
'booking_changes',
'agent',
'company',
'days_in_waiting_list',
'adr',
'required_car_parking_spaces',
'total_of_special_requests']
```

```
In [56]: [col for col in final_data.columns if final_data[col].dtype == "0"] ## for cat
```

```
Out[56]: ['hotel',
          'arrival_date_month',
          'meal',
          'country',
          'market_segment',
          'distribution_channel',
          'reserved_room_type',
          'assigned_room_type',
          'deposit_type',
          'customer_type',
          'reservation_status',
          'reservation_status_date']
```

```
In [57]: list_not = ["days_in_waiting_list", "arrival_date_year"]
```

```
In [58]: num_features = [col for col in final_data.columns if final_data[col].dtype !=
print(num_features)
```

```
['is_canceled', 'lead_time', 'arrival_date_week_number', 'arrival_date_day_of_m
onth', 'stays_in_weekend_nights', 'stays_in_week_nights', 'adults', 'children',
'babies', 'is_repeated_guest', 'previous_cancellations', 'previous_bookings_no
t_canceled', 'booking_changes', 'agent', 'company', 'adr', 'required_car_parkin
g_spaces', 'total_of_special_requests']
```

## Selectiong important categorical features

```
In [60]: cat_not = ["country", "reservation_status", "booking_changes", "assigned_room_
```

```
In [61]: cat_features = [col for col in final_data.columns if final_data[col].dtype ==
print(cat_features)
```

```
['hotel', 'arrival_date_month', 'meal', 'market_segment', 'distribution_channe
l', 'reserved_room_type', 'deposit_type', 'customer_type', 'reservation_statu
s_date']
```

```
In [62]: final_data[cat_features]
```

Out[62]:

	hotel	arrival_date_month	meal	market_segment	distribution_channel
0	Resort Hotel	July	BB	Direct	Direct
1	Resort Hotel	July	BB	Direct	Direct
2	Resort Hotel	July	BB	Direct	Direct
3	Resort Hotel	July	BB	Corporate	Corporate
4	Resort Hotel	July	BB	Online TA	TA/TC
...	...	...	...	...	...
119385	City Hotel	August	BB	Offline TA/TO	TA/TC
119386	City Hotel	August	BB	Online TA	TA/TC
119387	City Hotel	August	BB	Online TA	TA/TC
119388	City Hotel	August	BB	Online TA	TA/TC
119389	City Hotel	August	HB	Online TA	TA/TC

119210 rows × 9 columns

```
In [63]: data_cat = final_data[cat_features]
print(data_cat.head())
```

```
      hotel arrival_date_month meal market_segment distribution_channel \
0  Resort Hotel          July   BB          Direct          Direct
1  Resort Hotel          July   BB          Direct          Direct
2  Resort Hotel          July   BB          Direct          Direct
3  Resort Hotel          July   BB      Corporate      Corporate
4  Resort Hotel          July   BB      Online TA          TA/TO

reserved_room_type deposit_type customer_type reservation_status_date
0                C   No Deposit    Transient          7/1/2015
1                C   No Deposit    Transient          7/1/2015
2                A   No Deposit    Transient          7/2/2015
3                A   No Deposit    Transient          7/2/2015
4                A   No Deposit    Transient          7/3/2015
```

```
In [64]: data_cat["reservation_status_date"] = pd.to_datetime(data_cat["reservation_status_date"])
print(data_cat["reservation_status_date"])
```

```
0      2015-07-01
1      2015-07-01
2      2015-07-02
3      2015-07-02
4      2015-07-03
```

```
...
119385 2017-09-06
119386 2017-09-07
119387 2017-09-07
119388 2017-09-07
119389 2017-09-07
```

```
Name: reservation_status_date, Length: 119210, dtype: datetime64[ns]
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\3316979727.py:1: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
In [65]: data_cat["year"] = data_cat["reservation_status_date"].dt.year
data_cat["month"] = data_cat["reservation_status_date"].dt.month
data_cat["day"] = data_cat["reservation_status_date"].dt.day
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\2015022175.py:1: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\2015022175.py:2: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\2015022175.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
In [66]: data_cat.head()
```

```
Out[66]:
```

	hotel	arrival_date_month	meal	market_segment	distribution_channel	rese
0	Resort Hotel	July	BB	Direct	Direct	
1	Resort Hotel	July	BB	Direct	Direct	
2	Resort Hotel	July	BB	Direct	Direct	
3	Resort Hotel	July	BB	Corporate	Corporate	
4	Resort Hotel	July	BB	Online TA	TA/TO	

```
In [67]: data_cat.drop("reservation_status_date", axis = 1, inplace = True)
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\3432505083.py:1: SettingWithCopyWarning:
```

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
In [148... data_cat.columns
```

```
Out[148... Index(['hotel', 'arrival_date_month', 'meal', 'market_segment',  
              'distribution_channel', 'reserved_room_type', 'deposit_type',  
              'customer_type', 'year', 'month', 'day'],  
              dtype='object')
```

```
In [150... data_cat["cancellation"] = final_data["is_canceled"]
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\2073349528.py:1: SettingWithCopyWarning:
```

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
In [152... print(data_cat)
```

	hotel	arrival_date_month	meal	market_segment	\
0	Resort Hotel	July	BB	Direct	
1	Resort Hotel	July	BB	Direct	
2	Resort Hotel	July	BB	Direct	
3	Resort Hotel	July	BB	Corporate	
4	Resort Hotel	July	BB	Online TA	
...	...	...	...	...	...
119385	City Hotel	August	BB	Offline TA/T0	
119386	City Hotel	August	BB	Online TA	
119387	City Hotel	August	BB	Online TA	
119388	City Hotel	August	BB	Online TA	
119389	City Hotel	August	HB	Online TA	

	distribution_channel	reserved_room_type	deposit_type	customer_type	\
0	Direct		C No Deposit	Transient	
1	Direct		C No Deposit	Transient	
2	Direct		A No Deposit	Transient	
3	Corporate		A No Deposit	Transient	
4	TA/T0		A No Deposit	Transient	
...	...	...	...	...	...
119385	TA/T0		A No Deposit	Transient	
119386	TA/T0		E No Deposit	Transient	
119387	TA/T0		D No Deposit	Transient	
119388	TA/T0		A No Deposit	Transient	
119389	TA/T0		A No Deposit	Transient	

	year	month	day	cancellation
0	2015	7	1	0
1	2015	7	1	0
2	2015	7	2	0
3	2015	7	2	0
4	2015	7	3	0
...	...	...	...	...
119385	2017	9	6	0
119386	2017	9	7	0
119387	2017	9	7	0
119388	2017	9	7	0
119389	2017	9	7	0

[119210 rows x 12 columns]

## Feature Encoding

### Mean Encoding Technique

In [154... data\_cat.columns

Out[154... Index(['hotel', 'arrival\_date\_month', 'meal', 'market\_segment',  
'distribution\_channel', 'reserved\_room\_type', 'deposit\_type',  
'customer\_type', 'year', 'month', 'day', 'cancellation'],  
dtype='object')

```
In [167... def mean_encode(df, col, mean_col):  
    df_dict = df.groupby([col])[mean_col].mean().to_dict()  
    df[col] = df[col].map(df_dict)  
    return df  
for col in data_cat.columns[0:8]:  
    data_cat = mean_encode(data_cat, col, "cancellation")
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user\_guide/indexing.html#returning-a-view-versus-a-copy
```

```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1126876799.py:3: SettingWithCopyWarning:
```

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel\_4316\1126876799.py:3: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel\_4316\1126876799.py:3: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.  
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

In [169... `print(data_cat)`

	hotel	arrival_date_month	meal	market_segment	\
0	0.277674	0.374644	0.374106	0.153712	
1	0.277674	0.374644	0.374106	0.153712	
2	0.277674	0.374644	0.374106	0.153712	
3	0.277674	0.374644	0.374106	0.187618	
4	0.277674	0.374644	0.374106	0.367590	
...	...	...	...	...	
119385	0.417859	0.377823	0.374106	0.343313	
119386	0.417859	0.377823	0.374106	0.367590	
119387	0.417859	0.377823	0.374106	0.367590	
119388	0.417859	0.377823	0.374106	0.367590	
119389	0.417859	0.377823	0.344653	0.367590	

	distribution_channel	reserved_room_type	deposit_type	customer_type
\				
0	0.174868	0.330827	0.28402	0.407864
1	0.174868	0.330827	0.28402	0.407864
2	0.174868	0.391567	0.28402	0.407864
3	0.220568	0.391567	0.28402	0.407864
4	0.410598	0.391567	0.28402	0.407864
...	...	...	...	...
119385	0.410598	0.391567	0.28402	0.407864
119386	0.410598	0.292683	0.28402	0.407864
119387	0.410598	0.318108	0.28402	0.407864
119388	0.410598	0.391567	0.28402	0.407864
119389	0.410598	0.391567	0.28402	0.407864

	year	month	day	cancellation
0	2015	7	1	0
1	2015	7	1	0
2	2015	7	2	0
3	2015	7	2	0
4	2015	7	3	0
...	...	...	...	...
119385	2017	9	6	0
119386	2017	9	7	0
119387	2017	9	7	0
119388	2017	9	7	0
119389	2017	9	7	0

[119210 rows x 12 columns]

```
In [171]: data_cat.drop(["cancellation"], axis = 1, inplace = True)
```

C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel\_4316\3718714894.py:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

## Preparing data

```
In [180... num_data = final_data[num_features]  
          cat_data = data_cat  
          dataframe = pd.concat([num_data, cat_data], axis = 1)
```

```
In [182... print(dataframe)
```

	is_canceled	lead_time	arrival_date_week_number	\
0	0	342	27	
1	0	737	27	
2	0	7	27	
3	0	13	27	
4	0	14	27	
...	...	...	...	
119385	0	23	35	
119386	0	102	35	
119387	0	34	35	
119388	0	109	35	
119389	0	205	35	

	arrival_date_day_of_month	stays_in_weekend_nights	\
0	1	0	
1	1	0	
2	1	0	
3	1	0	
4	1	0	
...	...	...	
119385	30	2	
119386	31	2	
119387	31	2	
119388	31	2	
119389	29	2	

	stays_in_week_nights	adults	children	babies	is_repeated_guest	\
0	0	2	0.0	0	0	
1	0	2	0.0	0	0	
2	1	1	0.0	0	0	
3	1	1	0.0	0	0	
4	2	2	0.0	0	0	
...	...	...	...	...	...	
119385	5	2	0.0	0	0	
119386	5	3	0.0	0	0	
119387	5	2	0.0	0	0	
119388	5	2	0.0	0	0	
119389	7	2	0.0	0	0	

	...	arrival_date_month	meal	market_segment	\
0	...	0.374644	0.374106	0.153712	
1	...	0.374644	0.374106	0.153712	
2	...	0.374644	0.374106	0.153712	
3	...	0.374644	0.374106	0.187618	
4	...	0.374644	0.374106	0.367590	
...	...	...	...	...	
119385	...	0.377823	0.374106	0.343313	
119386	...	0.377823	0.374106	0.367590	
119387	...	0.377823	0.374106	0.367590	
119388	...	0.377823	0.374106	0.367590	
119389	...	0.377823	0.344653	0.367590	

	distribution_channel	reserved_room_type	deposit_type	customer_type	\
--	----------------------	--------------------	--------------	---------------	---

0	0.174868	0.330827	0.28402	0.407864
1	0.174868	0.330827	0.28402	0.407864
2	0.174868	0.391567	0.28402	0.407864
3	0.220568	0.391567	0.28402	0.407864
4	0.410598	0.391567	0.28402	0.407864
...	...	...	...	...
119385	0.410598	0.391567	0.28402	0.407864
119386	0.410598	0.292683	0.28402	0.407864
119387	0.410598	0.318108	0.28402	0.407864
119388	0.410598	0.391567	0.28402	0.407864
119389	0.410598	0.391567	0.28402	0.407864

	year	month	day
0	2015	7	1
1	2015	7	1
2	2015	7	2
3	2015	7	2
4	2015	7	3
...	...	...	...
119385	2017	9	6
119386	2017	9	7
119387	2017	9	7
119388	2017	9	7
119389	2017	9	7

[119210 rows x 29 columns]

## handeling outliers

In [186... `dataframe.describe()`]

	is_canceled	lead_time	arrival_date_week_number	arrival_date_da
<b>count</b>	119210.000000	119210.000000	119210.000000	119210.000000
<b>mean</b>	0.370766	104.109227	27.163376	119210.000000
<b>std</b>	0.483012	106.875450	13.601107	119210.000000
<b>min</b>	0.000000	0.000000	1.000000	119210.000000
<b>25%</b>	0.000000	18.000000	16.000000	119210.000000
<b>50%</b>	0.000000	69.000000	28.000000	119210.000000
<b>75%</b>	1.000000	161.000000	38.000000	119210.000000
<b>max</b>	1.000000	737.000000	53.000000	119210.000000

8 rows x 29 columns

In [192... `## using seaborn librarie`  
`sns.distplot(dataframe["lead_time"])`

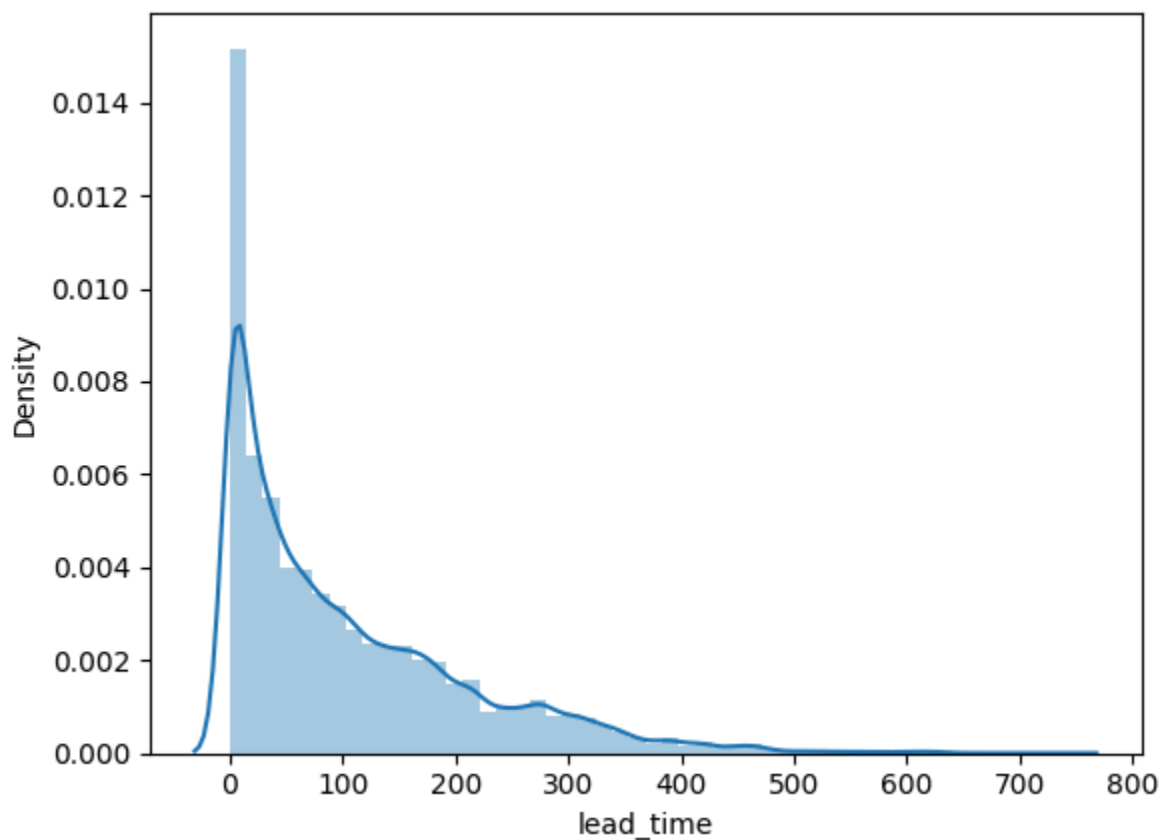
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel\_4316\1215022949.py:2: UserWarning:

`'distplot'` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `'displot'` (a figure-level function with similar flexibility) or `'histplot'` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

Out[192... <Axes: xlabel='lead\_time', ylabel='Density'>



```
In [194... def handle_outlier(col):  
    dataframe[col] = np.log1p(dataframe[col])
```

```
In [196... handle_outlier("lead_time")
```

```
In [198... ## using seaborn librarie  
sns.distplot(dataframe["lead_time"].dropna())
```

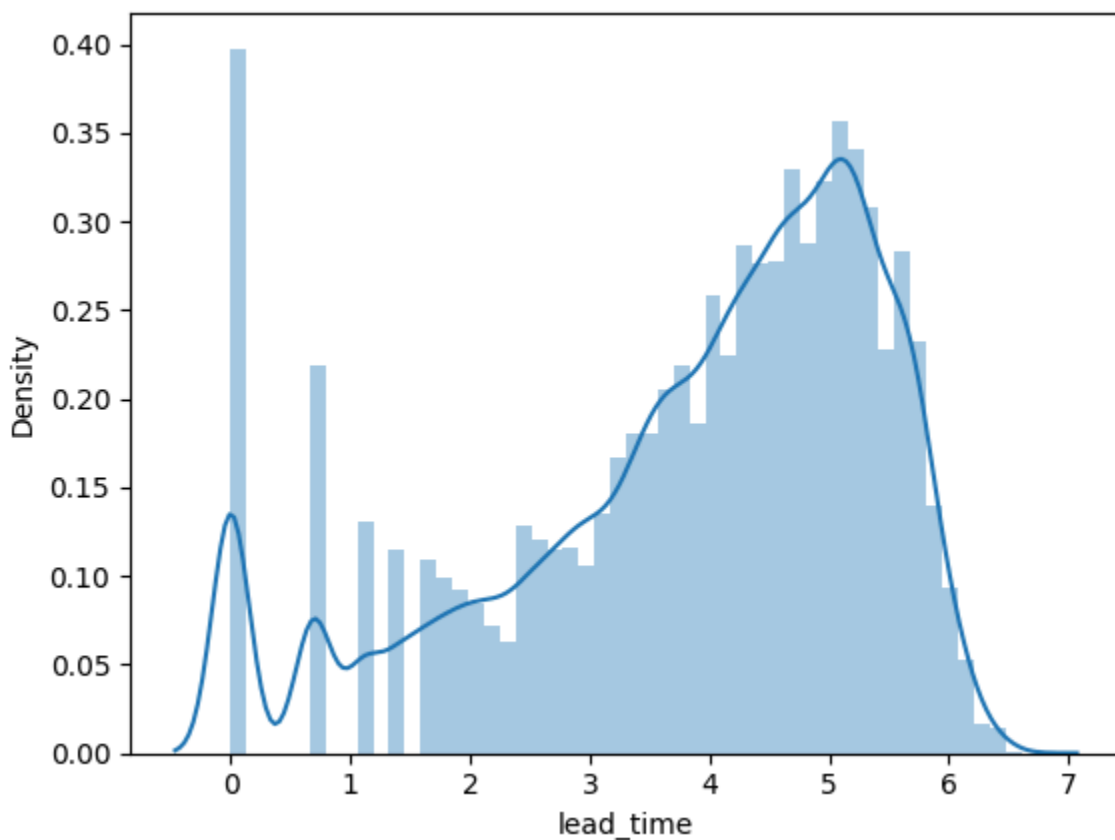
```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\4159535849.py:2: UserWarning:
```

``distplot`` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either ``displot`` (a figure-level function with similar flexibility) or ``histplot`` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
Out[198... <Axes: xlabel='lead_time', ylabel='Density'>
```



```
In [200... handle_outlier("adr")
```

```
C:\Users\NYB COMPUTER\AppData\Roaming\Python\Python311\site-packages\pandas\core\arraylike.py:402: RuntimeWarning:
```

invalid value encountered in log1p

```
In [202... sns.distplot(dataframe["adr"].dropna())
```

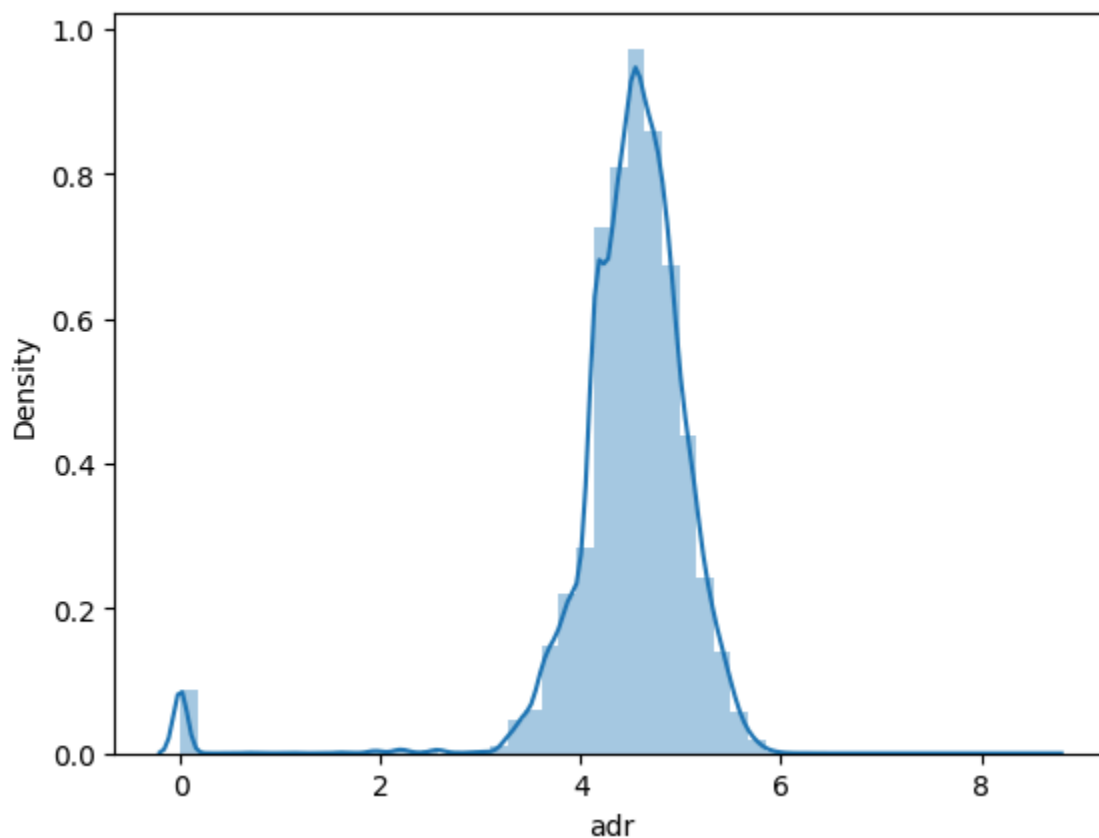
```
C:\Users\NYB COMPUTER\AppData\Local\Temp\ipykernel_4316\1296038743.py:1: UserWarning:
```

``distplot`` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either ``displot`` (a figure-level function with similar flexibility) or ``histplot`` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
Out[202... <Axes: xlabel='adr', ylabel='Density'>
```



```
In [208... dataframe.isnull().sum()
```

```
Out[208... is_canceled      0
            lead_time      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
            is_repeated_guest  0
            previous_cancellations  0
            previous_bookings_not_canceled  0
            booking_changes  0
            agent      0
            company     0
            adr         0
            required_car_parking_spaces  0
            total_of_special_requests  0
            hotel      0
            arrival_date_month  0
            meal        0
            market_segment  0
            distribution_channel  0
            reserved_room_type  0
            deposit_type  0
            customer_type  0
            year        0
            month       0
            day         0
            dtype: int64
```

```
In [206... dataframe.dropna(inplace =True)
```

## Feature Importance

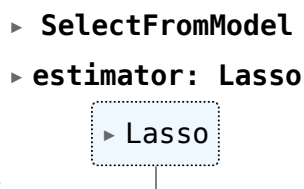
```
In [217... x = dataframe.drop("is_canceled", axis = 1)
            y = dataframe["is_canceled"]
```

```
In [219... from sklearn.linear_model import Lasso
            from sklearn.feature_selection import SelectFromModel
```

```
In [221... feature_sel_model = SelectFromModel(Lasso(alpha = 0.005, random_state = 0))
```

```
In [223... feature_sel_model.fit(x,y)
```

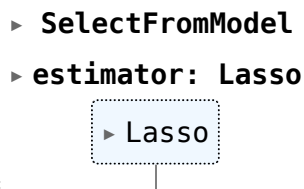
Out[223...



In [227...

```
SelectFromModel(estimator=Lasso(alpha = 0.005, random_state = 0))
```

Out[227...



In [233...

```
cols = x.columns
```

In [235...

```
feature_sel_model.get_support()
```

Out[235...

```
array([ True,  True, False, False, False,  True,  True, False, False,
        True, False,  True, False,  True,  True,  True,  True, False,
        False, False, False, False, False,  True, False,  True,  True,
        True])
```

In [237...

```
selected_feature = cols[(feature_sel_model.get_support())]
```

In [241...

```
print(selected_feature)
```

```
Index(['lead_time', 'arrival_date_week_number', 'adults', 'children',
       'previous_cancellations', 'booking_changes', 'company', 'adr',
       'required_car_parking_spaces', 'total_of_special_requests',
       'deposit_type', 'year', 'month', 'day'],
      dtype='object')
```

In [243...

```
print(f"Total features {x.shape[1]}")
```

Total features 28

In [245...

```
print(f"selected features {len(selected_feature)}")
```

selected features 14

In [247...

```
x = x[selected_feature]
```

In [249...

```
x.head()
```

	lead_time	arrival_date_week_number	adults	children	previous_cancellations
0	5.837730	27	2	0.0	(
1	6.603944	27	2	0.0	(
2	2.079442	27	1	0.0	(
3	2.639057	27	1	0.0	(
4	2.708050	27	2	0.0	(

In [ ]:

## Splitting the data and model building

```
In [254... from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, train_size = 0.75, ra
```

```
In [258... ## implementing logistic regression
from sklearn.linear_model import LogisticRegression
```

```
In [260... logistic_model = LogisticRegression()
logistic_model.fit(x_train,y_train)
```

F:\Anaconda\Lib\site-packages\sklearn\linear\_model\\_logistic.py:458: ConvergenceWarning:

lbfgs failed to converge (status=1):  
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:  
<https://scikit-learn.org/stable/modules/preprocessing.html>  
Please also refer to the documentation for alternative solver options:  
[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)

Out[260... ▼ LogisticRegression  
LogisticRegression()

```
In [266... y_pred = logistic_model.predict(x_test)
```

```
In [268... from sklearn.metrics import confusion_matrix
```

```
In [272... confusion_matrix(y_test, y_pred)
```

```
Out[272... array([[15723, 2959],
       [ 5459, 5662]], dtype=int64)
```

```
In [274... from sklearn.metrics import accuracy_score
```

```
In [276... accuracy_score(y_test,y_pred)
```

```
Out[276... 0.7175452135691038
```

## Implementation of different models

- logistic regression
- Naive Bayes
- Random Forest
- Decision Tree
- KNN

```
In [290... from sklearn.naive_bayes import GaussianNB  
from sklearn.neighbors import KNeighborsClassifier  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.tree import DecisionTreeClassifier
```

```
In [292... models = []  
models.append(("Naive Bayes", GaussianNB()))  
models.append(("RandomForest", RandomForestClassifier ()))  
models.append(("KNN", KNeighborsClassifier(n_neighbors = 5)))  
models.append(("Decision Tree", DecisionTreeClassifier()))
```

```
In [294... for name, model in models:  
    print(name)  
    model.fit(x_train,y_train)  
    ## making predictions  
    predictions = model.predict(x_test)  
    ## evaluating a model  
    from sklearn.metrics import confusion_matrix  
    print(confusion_matrix(predictions,y_test))  
  
    from sklearn.metrics import accuracy_score  
    print(accuracy_score(predictions,y_test))  
    print("\n")
```

```
Naive Bayes
[[8820 1289]
 [9862 9832]]
0.6258430359359796
```

```
RandomForest
[[18555 1207]
 [ 127 9914]]
0.9552394054289837
```

```
KNN
[[18522 1417]
 [ 160 9704]]
0.9470858638392108
```

```
Decision Tree
[[17864 813]
 [ 818 10308]]
0.9452739657081501
```

In [ ]: