Can you predict if the customer is going to honor the reservation or cancel it ?

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
```

```
df=pd.read_csv("Hotel_Reservations.csv")
```

Data Exploration

```
df.info()
## The data is cleaned
```

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

```
# Assuming df is your DataFrame containing the 'booking_status' column

# Use replace() to replace values in 'booking_status' column
df['booking_status'].replace({'Canceled': 1, 'Not_Canceled': 0}, inplace=True)
```

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
no_of_adults	36275.0	1.844962	0.518715	0.0	2.0	2.00	2.0	4.0
no_of_children	36275.0	0.105279	0.402648	0.0	0.0	0.00	0.0	10.0
no_of_weekend_nights	36275.0	0.810724	0.870644	0.0	0.0	1.00	2.0	7.0
no_of_week_nights	36275.0	2.204300	1.410905	0.0	1.0	2.00	3.0	17.0
required_car_parking_space	36275.0	0.030986	0.173281	0.0	0.0	0.00	0.0	1.0
lead_time	36275.0	85.232557	85.930817	0.0	17.0	57.00	126.0	443.0
arrival_year	36275.0	2017.820427	0.383836	2017.0	2018.0	2018.00	2018.0	2018.0
arrival_month	36275.0	7.423653	3.069894	1.0	5.0	8.00	10.0	12.0
arrival_date	36275.0	15.596995	8.740447	1.0	8.0	16.00	23.0	31.0
repeated_guest	36275.0	0.025637	0.158053	0.0	0.0	0.00	0.0	1.0
no_of_previous_cancellations	36275.0	0.023349	0.368331	0.0	0.0	0.00	0.0	13.0
no_of_previous_bookings_not_canceled	36275.0	0.153411	1.754171	0.0	0.0	0.00	0.0	58.0
avg_price_per_room	36275.0	103.423539	35.089424	0.0	80.3	99.45	120.0	540.0
no_of_special_requests	36275.0	0.619655	0.786236	0.0	0.0	0.00	1.0	5.0
booking_status	36275.0	0.327636	0.469358	0.0	0.0	0.00	1.0	1.0

```
df.isnull().sum()
# No Missing values
```

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

Data Visualisation

```
df.no_of_adults.value_counts()
```

```
2    26108
1     7695
3     2317
0      139
4       16
Name: no_of_adults, dtype: int64
```

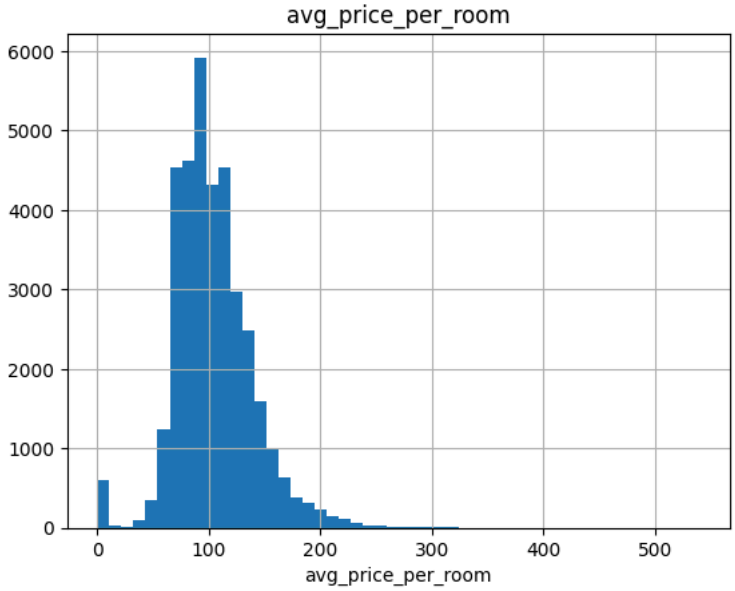
```
sns.countplot(data=df, x='no_of_adults', hue='booking_status', palette=sns.palettes.mpl_palette('Dark2'))
plt.xlabel('no_of_adults')
plt.ylabel('booking_status')
plt.title('No of Adults by booking status')
plt.gca().spines[['top', 'right']].set_visible(False)
plt.show()
```

```
df.repeated_guest.value_counts()
```

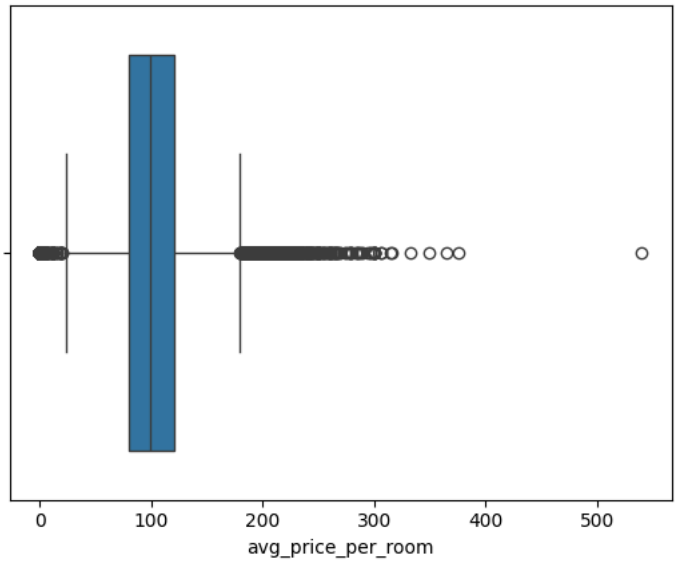
```
0    35345
1      930
Name: repeated_guest, dtype: int64
```

```
sns.countplot(data=df, x='repeated_guest', hue='booking_status', palette=sns.palettes.mpl_palette('Dark2'))
plt.xlabel('repeated_guest')
plt.ylabel('booking_status')
plt.title('No repeated_guest by booking status')
plt.gca().spines[['top', 'right']].set_visible(False)
plt.show()
```

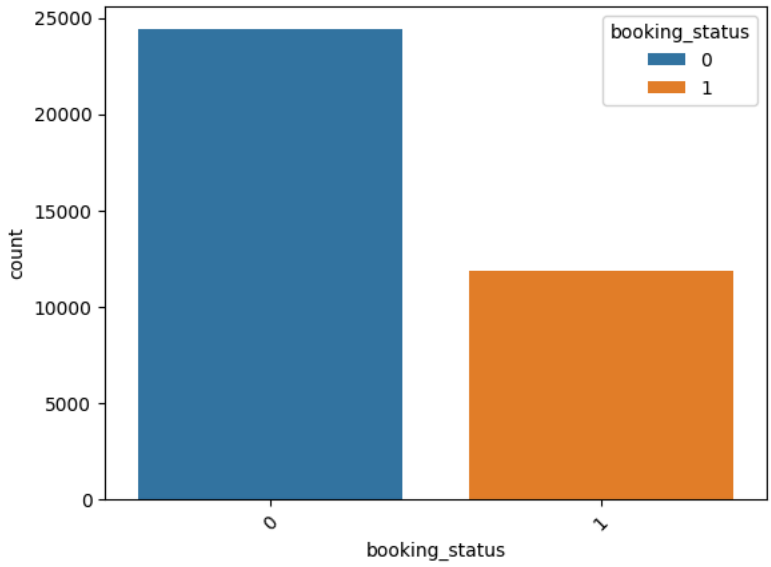
```
df['avg_price_per_room'].hist(bins=50)

plt.title('avg_price_per_room')
plt.xlabel('avg_price_per_room')
plt.show()
```



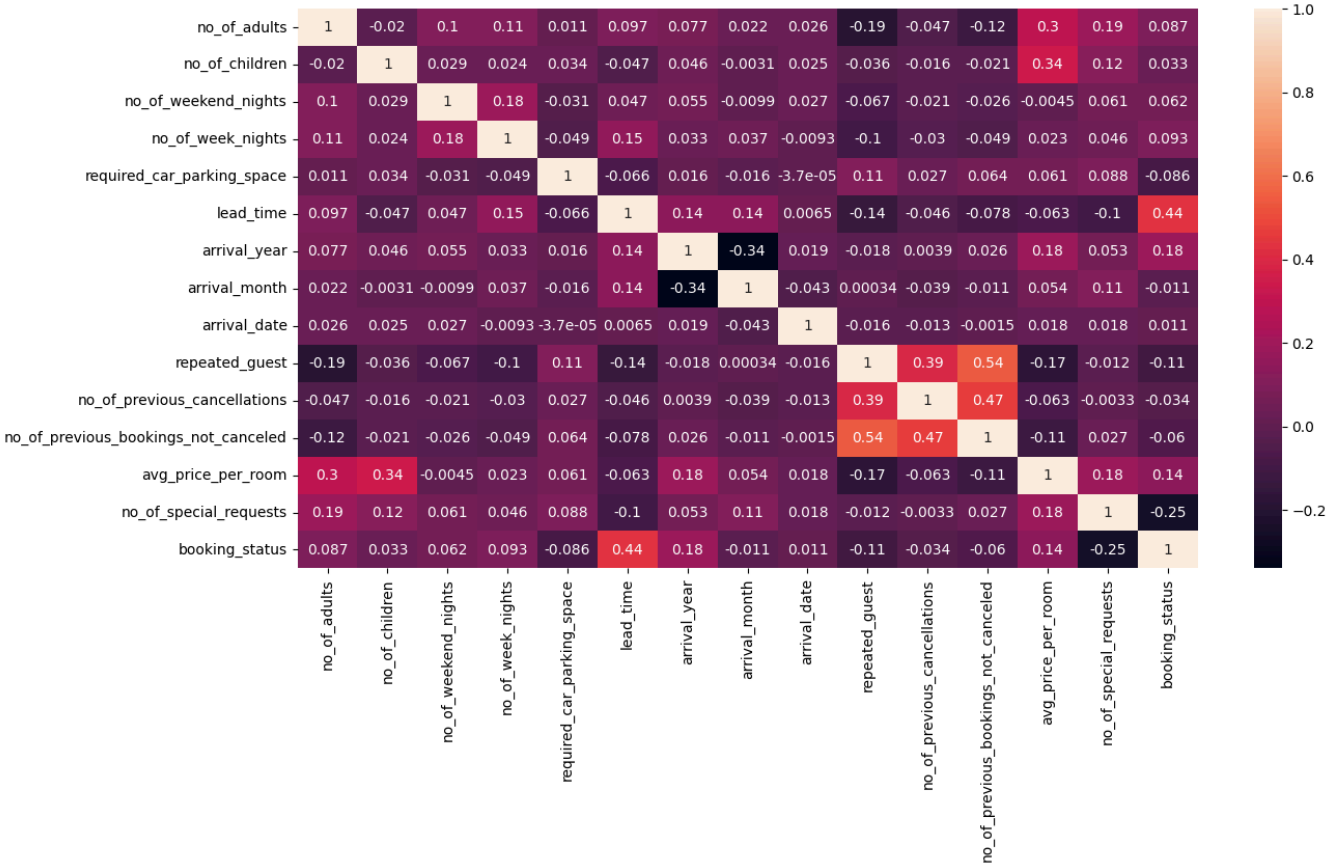

```
# Boxplot for checking outliers
sns.boxplot(x=df['avg_price_per_room'])
plt.show()
```




```
sns.countplot(x='booking_status', hue='booking_status', data=df)
plt.xticks(rotation=45)
plt.show()
```



```
# Assuming 'df' is your DataFrame
plt.figure(figsize=(14, 7))
# Explicitly specify numeric_only=True to avoid future issues
sns.heatmap(df.corr(numeric_only=True), annot=True)
plt.show()
```



Based on correlation table above

1) No of adults related with avg price per room

2) No of adults related with no of special requests

3) No of required car park is higher for repeated guest

4) No of previous booking **not cancelled** is higest for repeated guests

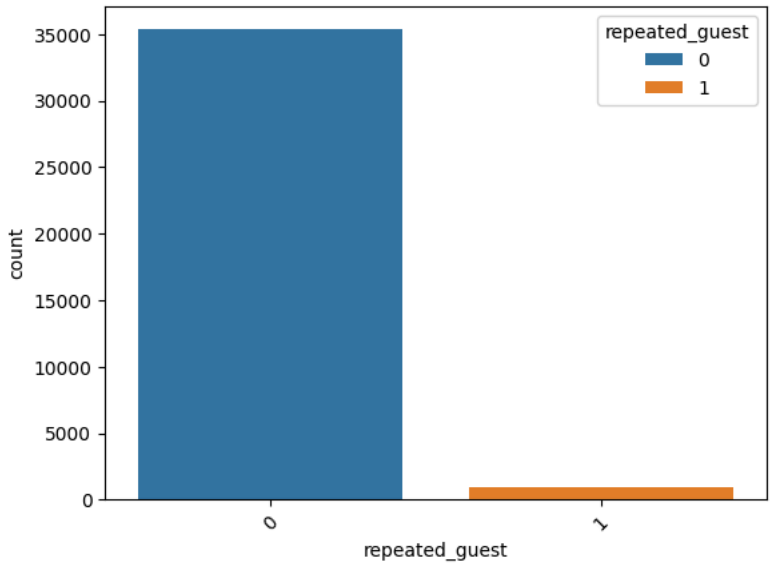
5) No of previous booking **cancelled** is higher for repeated guests

6) No of previous cancellations is higher for no of previous booking not canceled

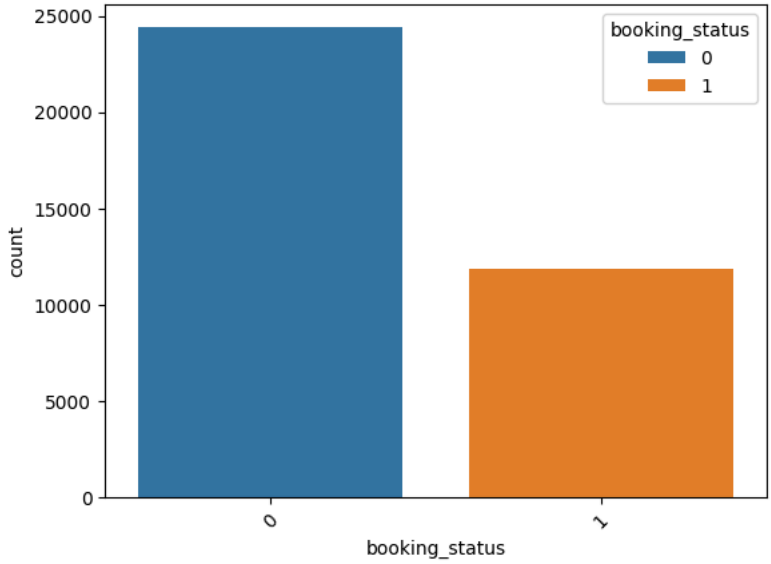
```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
no_of_adults	36275.0	1.844962	0.518715	0.0	2.0	2.00	2.0	4.0
no_of_children	36275.0	0.105279	0.402648	0.0	0.0	0.00	0.0	10.0
no_of_weekend_nights	36275.0	0.810724	0.870644	0.0	0.0	1.00	2.0	7.0
no_of_week_nights	36275.0	2.204300	1.410905	0.0	1.0	2.00	3.0	17.0
required_car_parking_space	36275.0	0.030986	0.173281	0.0	0.0	0.00	0.0	1.0
lead_time	36275.0	85.232557	85.930817	0.0	17.0	57.00	126.0	443.0
arrival_year	36275.0	2017.820427	0.383836	2017.0	2018.0	2018.00	2018.0	2018.0
arrival_month	36275.0	7.423653	3.069894	1.0	5.0	8.00	10.0	12.0
arrival_date	36275.0	15.596995	8.740447	1.0	8.0	16.00	23.0	31.0
repeated_guest	36275.0	0.025637	0.158053	0.0	0.0	0.00	0.0	1.0
no_of_previous_cancellations	36275.0	0.023349	0.368331	0.0	0.0	0.00	0.0	13.0
no_of_previous_bookings_not_canceled	36275.0	0.153411	1.754171	0.0	0.0	0.00	0.0	58.0
avg_price_per_room	36275.0	103.423539	35.089424	0.0	80.3	99.45	120.0	540.0
no_of_special_requests	36275.0	0.619655	0.786236	0.0	0.0	0.00	1.0	5.0
booking_status	36275.0	0.327636	0.469358	0.0	0.0	0.00	1.0	1.0

```
sns.countplot(x='repeated_guest', hue='repeated_guest', data=df)
plt.xticks(rotation=45)
plt.show()

## No of repeated guest is lower
```

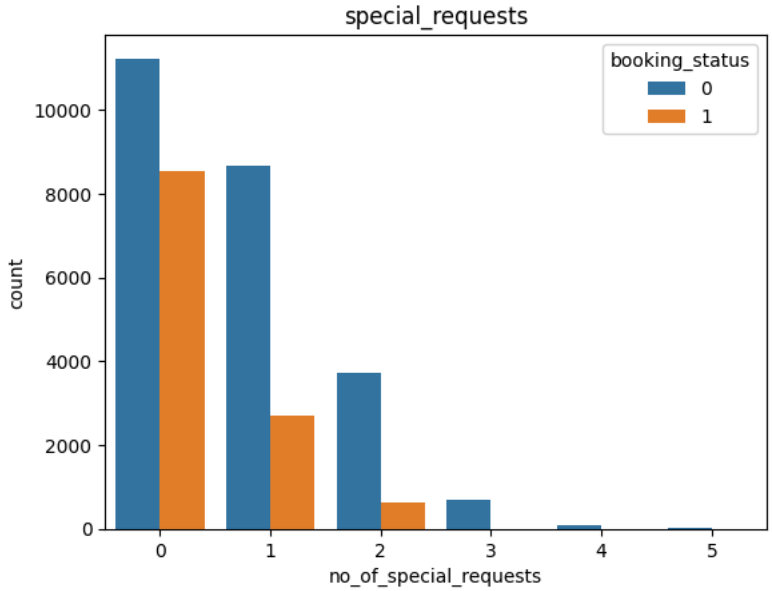



```
sns.countplot(x='booking_status',  hue='booking_status', data=df)
plt.xticks(rotation=45)
plt.show()
```



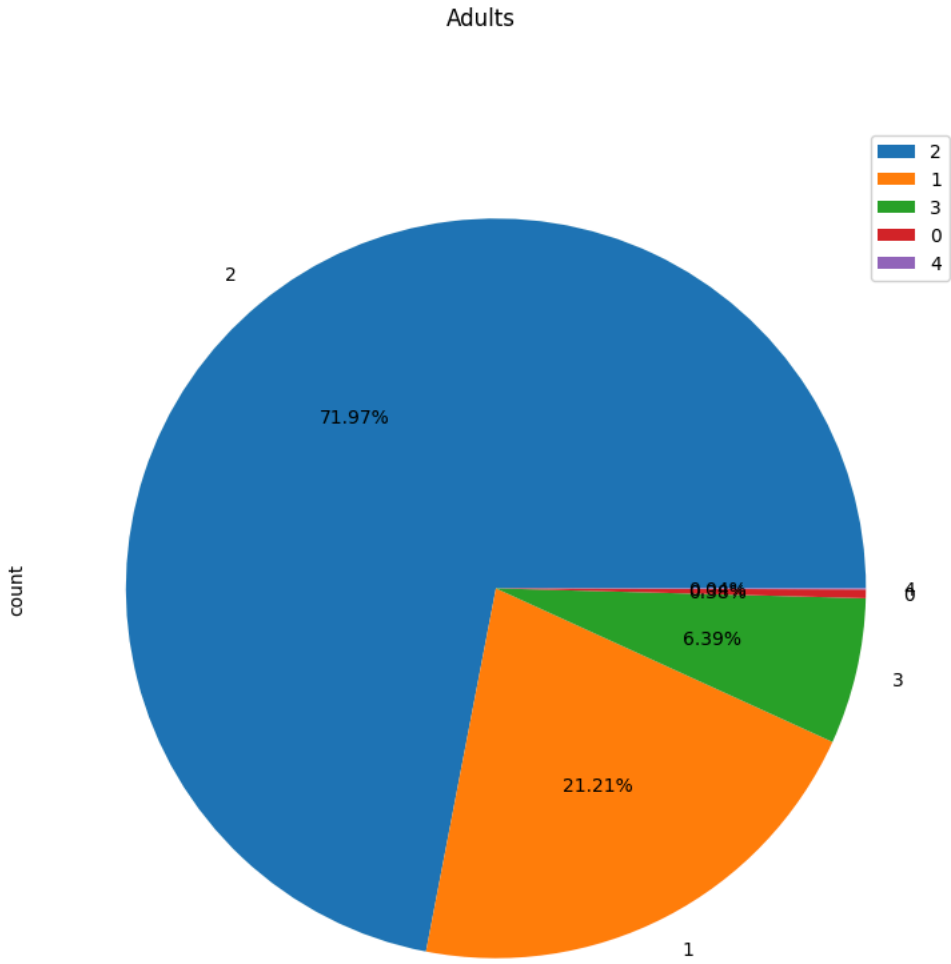

```
sns.countplot(x='no_of_special_requests', hue='booking_status',data=df)

plt.title('special_requests')
```

```
Text(0.5, 1.0, 'special_requests')
```




```
df.groupby('no_of_adults')['Booking_ID'].agg(['count']).sort_values(by='count',ascending=False).plot(kind='pie',autopct='%1.2f%%',subplot
```

```
array([<Axes: ylabel='count'>], dtype=object)
```



Mechine Learning

```
df
```

ival_year	arrival_month	arrival_date	market_segment_type	repeated_guest	no_of_previous_cancellations	no_of_previous_bookings_no
2017	10	2	Offline	0	0	
2018	11	6	Online	0	0	
2018	2	28	Online	0	0	
2018	5	20	Online	0	0	
2018	4	11	Online	0	0	
...	...	...	...	...	...	
2018	8	3	Online	0	0	
2018	10	17	Online	0	0	
2018	7	1	Online	0	0	
2018	4	21	Online	0	0	
2018	12	30	Offline	0	0	

Next steps: Generate code with df View recommended plots

```
df.type_of_meal_plan.value_counts()
```

```
Meal Plan 1     27835
Not Selected     5130
Meal Plan 2      3305
Meal Plan 3         5
```

```
df.room_type_reserved.value_counts()
```

```
Room_Type 1    28130
Room_Type 4     6057
Room_Type 6      966
Room_Type 2      692
Room_Type 5      265
Room_Type 7      158
Room_Type 3        7
Name: room_type_reserved, dtype: int64
```