Cwen Fernandes

Exploratory Data Analysis on Customer Bookings data for British Airways

We will explore the customer data first to get to know it better in depth.

In [3]:
```
#imports

import pandas as pd
import numpy as np
import os

import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
#get current working directory

cwd = os.getcwd()
#read the csv

df = pd.read_csv("customer_booking.csv",  encoding="ISO-8859-1")
```

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

Out[5]:

	num_passengers	sales_channel	trip_type	purchase_lead	length_of_stay	flight_hour	flight_day	route	booking_origin	wa
0	2	Internet	RoundTrip	262	19	7	Sat	AKLDEL	New Zealand	
1	1	Internet	RoundTrip	112	20	3	Sat	AKLDEL	New Zealand	
2	2	Internet	RoundTrip	243	22	17	Wed	AKLDEL	India	
3	1	Internet	RoundTrip	96	31	4	Sat	AKLDEL	New Zealand	
4	2	Internet	RoundTrip	68	22	15	Wed	AKLDEL	India	

In [6]:
```
df.shape
```

Out[6]:
```
(50000, 14)
```

In [7]:
```
df.describe()
```

Out[7]:

	num_passengers	purchase_lead	length_of_stay	flight_hour	wants_extra_baggage	wants_preferred_seat	wants_in_flight_
count	50000.000000	50000.000000	50000.00000	50000.00000	50000.000000	50000.000000	50000.0
mean	1.591240	84.940480	23.04456	9.06634	0.668780	0.296960	0.4
std	1.020165	90.451378	33.88767	5.41266	0.470657	0.456923	0.4
min	1.000000	0.000000	0.00000	0.00000	0.000000	0.000000	0.0
25%	1.000000	21.000000	5.00000	5.00000	0.000000	0.000000	0.0
50%	1.000000	51.000000	17.00000	9.00000	1.000000	0.000000	0.0
75%	2.000000	115.000000	28.00000	13.00000	1.000000	1.000000	1.0
max	9.000000	867.000000	778.00000	23.00000	1.000000	1.000000	1.0

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 14 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   num_passengers         50000 non-null  int64  
 1   sales_channel          50000 non-null  object 
 2   trip_type              50000 non-null  object 
 3   purchase_lead          50000 non-null  int64  
 4   length_of_stay         50000 non-null  int64  
 5   flight_hour            50000 non-null  int64  
 6   flight_day             50000 non-null  object 
 7   route                  50000 non-null  object 
 8   booking_origin         50000 non-null  object 
 9   wants_extra_baggage    50000 non-null  int64  
 10  wants_preferred_seat   50000 non-null  int64  
 11  wants_in_flight_meals  50000 non-null  int64  
 12  flight_duration        50000 non-null  float64
 13  booking_complete       50000 non-null  int64  
dtypes: float64(1), int64(8), object(5)
memory usage: 5.3+ MB
```

Sales Channel

In [10]:
```
per_internet = df.sales_channel.value_counts().values[0]  / df.sales_channel.count() *100
per_mobile = df.sales_channel.value_counts().values[1]  / df.sales_channel.count() *100
```

In [11]:
```
print(f"Number of bookings done through internet: {per_internet} %")
print(f"Number of bookings done through phone call: {per_mobile} %")
```

```
Number of bookings done through internet: 88.764 %
Number of bookings done through phone call: 11.236 %
```

Trip Type

In [12]:
```
per_round = df.trip_type.value_counts().values[0]/ df.trip_type.count() *100
per_oneway = df.trip_type.value_counts().values[1]/ df.trip_type.count() *100
per_circle = df.trip_type.value_counts().values[2]/ df.trip_type.count() *100
```

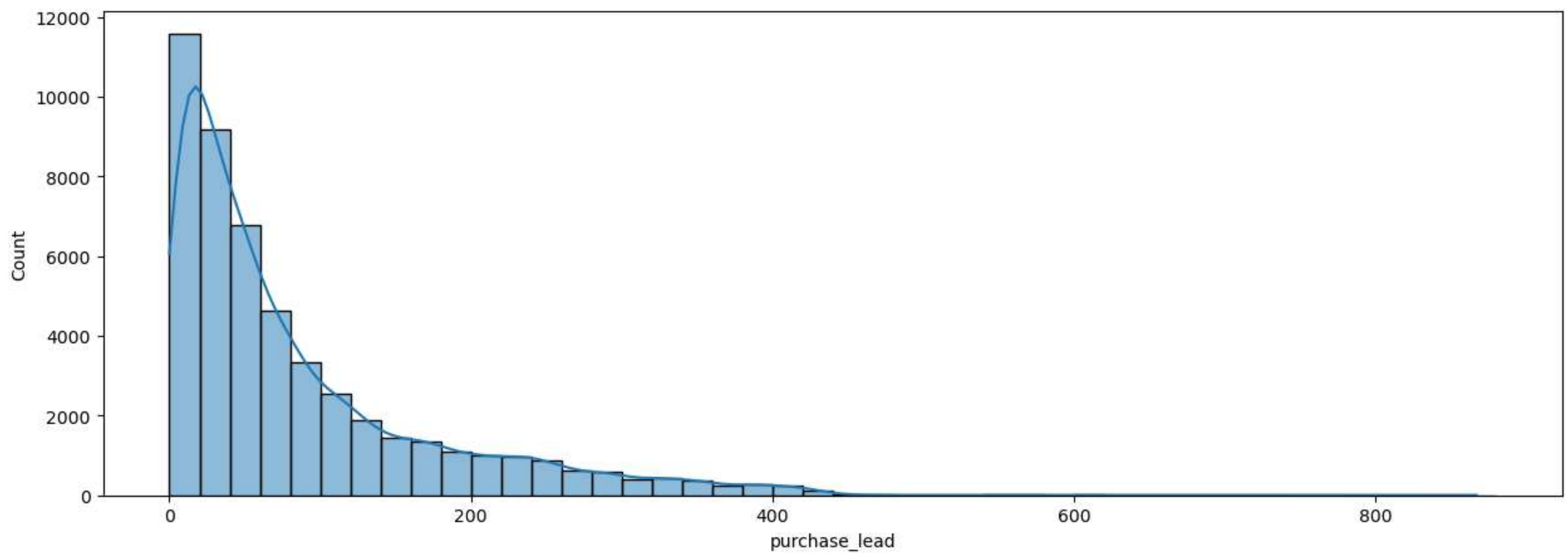
In [13]:
```
print(f"Percentage of round trips: {per_round} %")
print(f"Percentage of One way trips: {per_oneway} %")
print(f"Percentage of circle trips: {per_circle} %")
```

```
Percentage of round trips: 98.994 %
Percentage of One way trips: 0.774 %
Percentage of circle trips: 0.232 %
```

Purchase Lead

In [14]:
```
plt.figure(figsize=(15,5))
sns.histplot(data=df, x="purchase_lead", binwidth=20,kde=True)
```

Out[14]: <AxesSubplot:xlabel='purchase_lead', ylabel='Count'>



There are few bookings that were done more than 2 years before the travel date and it seems very unlikely that book that in advance. However, it might also be because of the cancellation and rebooking in a period of 6 months for twice. Generally airline keep the tickets for rebooking within a year. But at this point we will consider them as outliers which will effect the results of predictive model in a huge way.

In [15]:
```
(df.purchase_lead >600).value_counts()
```

Out[15]:
```
False    49992
True         8
Name: purchase_lead, dtype: int64
```

If we assume that no customer is booking in advance of more than 1 and half year we will remove all entries with purchase_lead more than 600 days.

In [16]:
```
df[df.purchase_lead > 600]
```

Out[16]:

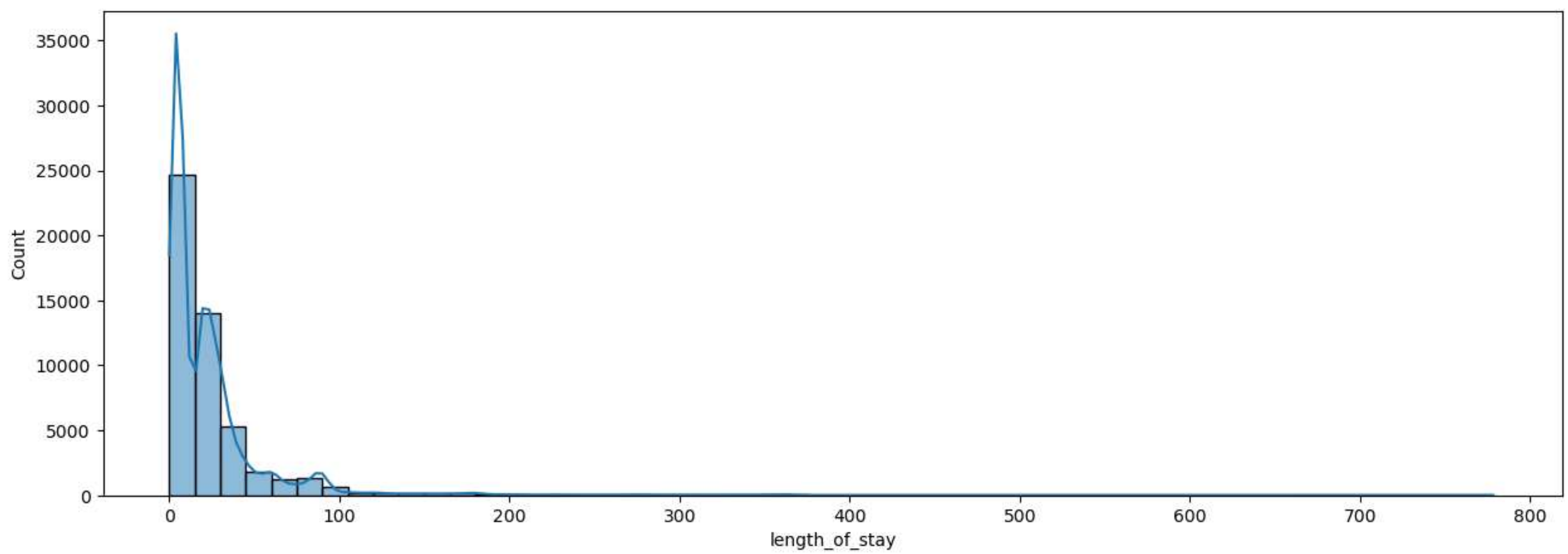
	num_passengers	sales_channel	trip_type	purchase_lead	length_of_stay	flight_hour	flight_day	route	booking_origir
835	3	Internet	RoundTrip	641	46	6	Sun	AKLKUL	Malaysia
6148	1	Internet	RoundTrip	614	19	11	Wed	COKMEL	Australia
24119	1	Internet	RoundTrip	704	23	8	Tue	PNHSYD	Australia
38356	2	Internet	RoundTrip	633	5	10	Sat	HKTOOL	Australia
39417	1	Mobile	RoundTrip	625	5	15	Fri	ICNRGN	Myanmar (Burma)
42916	1	Mobile	RoundTrip	605	6	18	Thu	BLRMEL	India
46716	2	Internet	RoundTrip	606	6	6	Fri	HKTTPE	United States
48259	3	Internet	RoundTrip	867	6	7	Mon	KIXMLE	Japar

In [17]:
```
#filtering the data to have only purchase lead days less than 600 days
df = df[df.purchase_lead <600 ]
```

Length Of Stay

In [18]:
```
plt.figure(figsize=(15,5))
sns.histplot(data=df, x="length_of_stay", binwidth=15,kde=True)
```

Out[18]:
```
<AxesSubplot:xlabel='length_of_stay', ylabel='Count'>
```



Let's see how many entries do we have that exceeds length of stay more than 100 days.

In [19]:
```
(df.length_of_stay> 200).value_counts()
```

Out[19]:
```
False    49713
True       279
Name: length_of_stay, dtype: int64
```

In [20]:
```
df[df.length_of_stay> 500].booking_complete.value_counts()
```

Out[20]:
```
0    9
1    1
Name: booking_complete, dtype: int64
```

We need to have more business knowledge to decide whether to remove these entries with more than 600 days of stay. There are could be many reasons for such bookings. But for now, we will just want to focus on bookings done for length of stay less than 500 days.

In [21]:
```
#filtering the data to have only length of stay days less than 500 days
df = df[df.purchase_lead <500 ]
```

Flight Day

We will map the flight day with a number of a week.

In [22]:
```
mapping = {
    "Mon" : 1,
    "Tue" : 2,
    "Wed" : 3,
    "Thu" : 4,
    "Fri" : 5,
    "Sat" : 6,
    "Sun" : 7
}

df.flight_day = df.flight_day.map(mapping)
```

In [23]:
```
df.flight_day.value_counts()
```

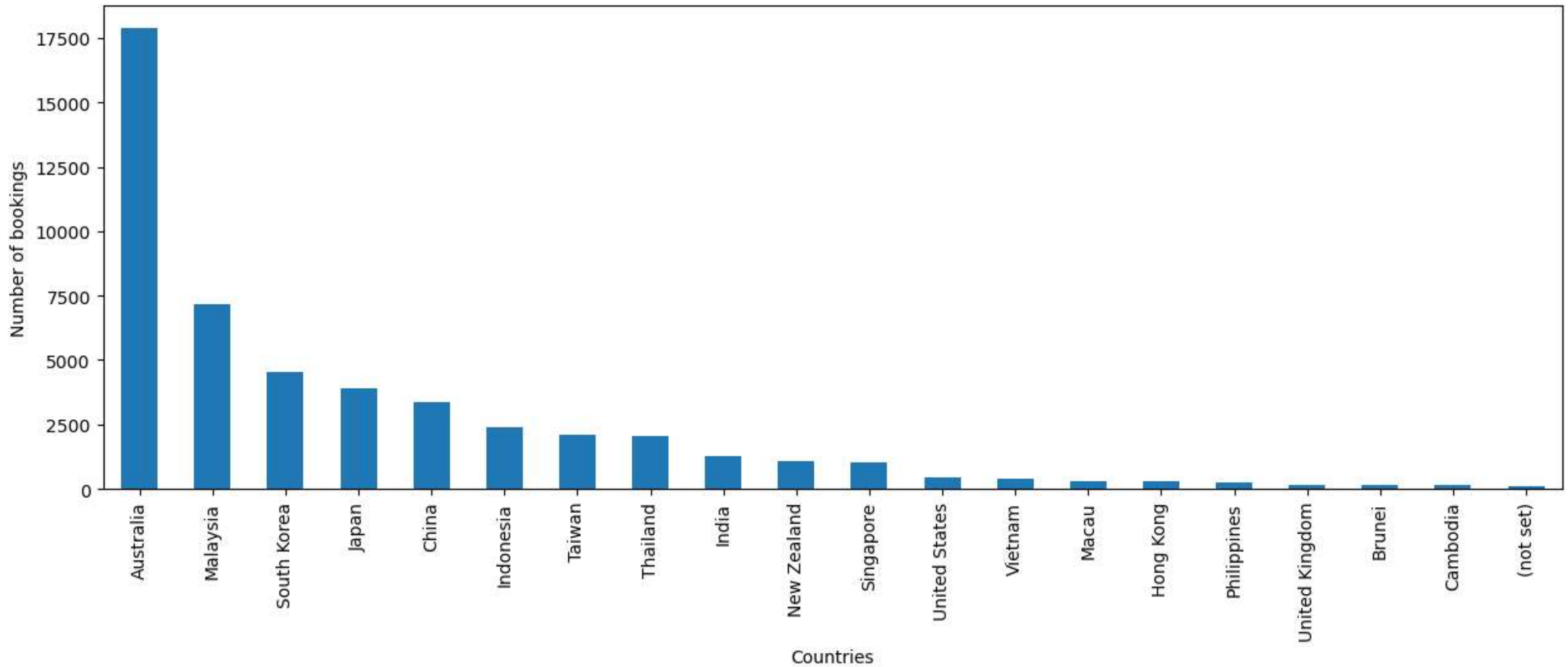
Out[23]:
```
1    8100
3    7671
2    7670
4    7423
5    6759
7    6550
6    5809
Name: flight_day, dtype: int64
```

Most of the customers want to travel on Monday and choose Saturday as least preffered day as flight day.

Booking Origin

In [24]:
```
plt.figure(figsize=(15,5))
ax = df.booking_origin.value_counts()[:20].plot(kind="bar")
ax.set_xlabel("Countries")
ax.set_ylabel("Number of bookings")
```

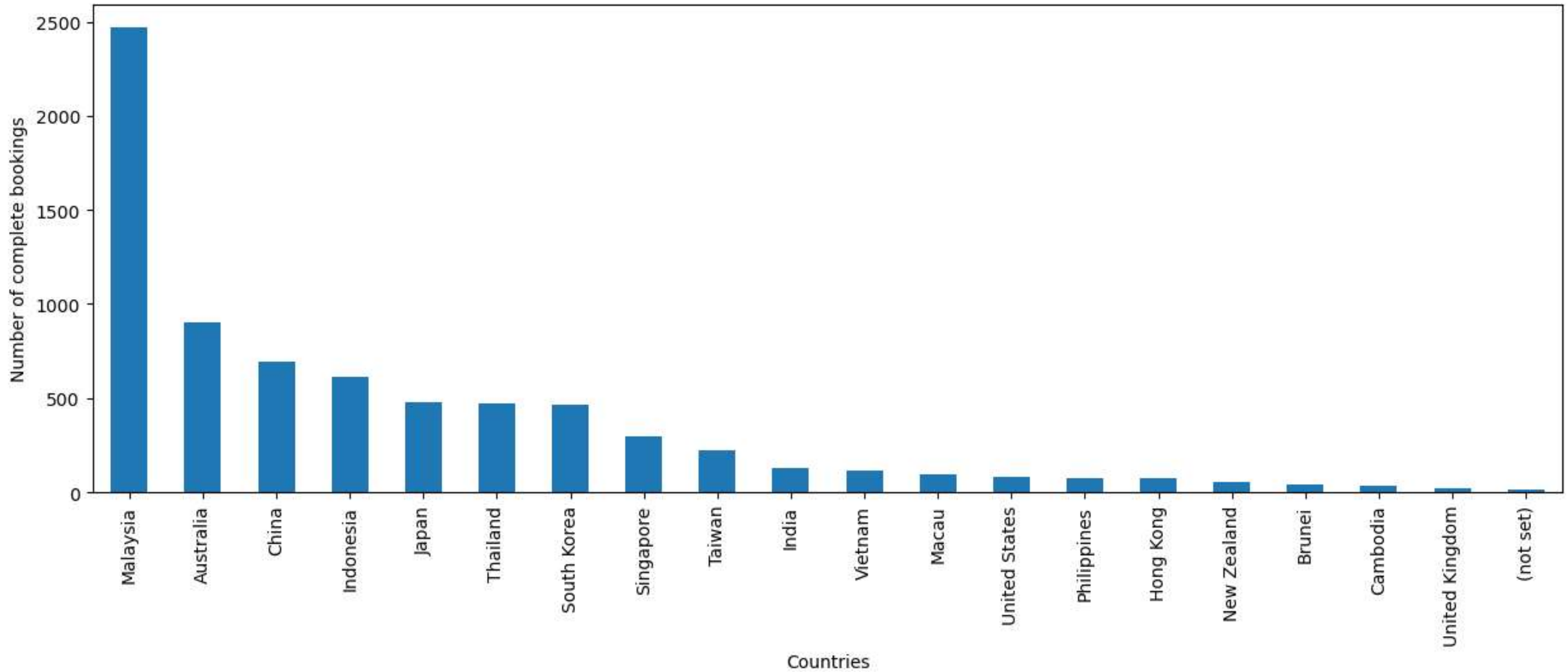
Out[24]:
```
Text(0, 0.5, 'Number of bookings')
```



Above chart shows travellers from which country had maximum booking applications.

In [25]:
```
plt.figure(figsize=(15,5))
ax = df[df.booking_complete ==1].booking_origin.value_counts()[:20].plot(kind="bar")
ax.set_xlabel("Countries")
ax.set_ylabel("Number of complete bookings")
```

Out[25]:
```
Text(0, 0.5, 'Number of complete bookings')
```



Above chart shows travellers from which country had their booking complete.

Booking complete

In [26]:
```
successful_booking_per = df.booking_complete.value_counts().values[0] / len(df) * 100
```

In [27]:
```
unsuccessful_booking_per = 100-successful_booking_per
```

In [28]:
```
print(f"Out of 50000 booking entries only {round(unsuccessful_booking_per,2)} % bookings were successfull
```

```
Out of 50000 booking entries only 14.96 % bookings were successfull or complete.
```

Export the dataset to csv

In [ ]:
```
df.to_csv("/kaggle/working/filtered_customer_airways_data.csv")
```

In [ ]:
```

```