

AIRLINE EDA

Workflow

Importing required Modules

1. Importing Required Modules
2. importing numpy for mathematical operation on arrays and dataframe.
3. importing pandas for reading data and data manipulation.
4. importing matplotlib and seaborn to show the insights and visualization from the dataset.
5. importing warnings for Warning messages that are typically issued in dataframe where it is useful to alert the user of some condition in a program, where that condition (normally) doesn't warrant raising an exception and terminating the program.

- Importing required libraries
- loading the data set
- Basic understanding of data

-- checking shape

-- checking info

-- fetching columns names

--- checking unique values

Importing Libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Loading the Data

In [2]:

```
df = pd.read_excel("Data_Train.xlsx")
df
```

Out[2]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Dura
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10 22 Mar	2h
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25 10 Jun	
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h
...	
10678	Air Asia	9/04/2019	Kolkata	Banglore	CCU → BLR	19:55	22:25	2h
10679	Air India	27/04/2019	Kolkata	Banglore	CCU → BLR	20:45	23:20	2h
10680	Jet Airways	27/04/2019	Banglore	Delhi	BLR → DEL	08:20	11:20	
10681	Vistara	01/03/2019	Banglore	New Delhi	BLR → DEL	11:30	14:10	2h
10682	Air India	9/05/2019	Delhi	Cochin	DEL → GOI → BOM → COK	10:55	19:15	8h

10683 rows × 11 columns



Preview of data

In [3]:

```
df.sample(5)
```

Out[3]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Durati
5687	Vistara	3/05/2019	Chennai	Kolkata	MAA → CCU	07:05	09:20	2h 14
4179	Jet Airways	1/06/2019	Delhi	Cochin	DEL → BOM → COK	07:05	12:35	5h 30
823	Jet Airways	9/06/2019	Delhi	Cochin	DEL → JAI → BOM → COK	19:30	04:25 10 Jun	8h 54
7435	Air India	9/06/2019	Delhi	Cochin	DEL → BOM → COK	20:00	19:15 10 Jun	23h 14
1944	Air India	21/03/2019	Delhi	Cochin	DEL → LKO → BOM → COK	18:30	19:15 22 Mar	24h 44

Basic Understanding of Data

How big is the data ?

In [4]:

```
df.shape
```

Out[4]:

```
(10683, 11)
```

-Observation:

- There are total **11 Attributes/columns** available in the dataset.

- There are total **10683 Records/Rows** available in the dataset.

Fetching Column Names

In [5]:

```
df.columns
```

Out[5]:

```
Index(['Airline', 'Date_of_Journey', 'Source', 'Destination', 'Route',
      'Dep_Time', 'Arrival_Time', 'Duration', 'Total_Stops',
      'Additional_Info', 'Price'],
      dtype='object')
```

Checking the column names

In [6]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 11 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Airline                10683 non-null  object
1   Date_of_Journey        10683 non-null  object
2   Source                 10683 non-null  object
3   Destination            10683 non-null  object
4   Route                  10682 non-null  object
5   Dep_Time               10683 non-null  object
6   Arrival_Time           10683 non-null  object
7   Duration               10683 non-null  object
8   Total_Stops            10682 non-null  object
9   Additional_Info        10683 non-null  object
10  Price                  10683 non-null  int64
dtypes: int64(1), object(10)
memory usage: 918.2+ KB
```

-Observation:

- In this data all the columns are of object type except Price , and we can see here the column names Date_of_Journey,Dep_Time,Arrival_Time need to be converted into datetime
- there is a duration column which should be in numeric

Checking the unique values

```
for i in df.columns:
```

```
#print(i,"-----",df[i].unique())
```

-Observation:

- From this output we can see there is anomaly in airline names : vistara premium economy with vistara multiple carriers with multiple carriers premium economy jet airways with jet airways business
- From this output we can see there is anomaly in destination: new Delhi with Delhi

We need to resolve this above mentioned inconsistency

Preprocessing

Cleaning

Checking the Missing Values

In [8]:

```
df.isnull().sum().to_frame().rename(columns={0:"missing_values_count"}).T
```

Out[8]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arriv
missing_values_count	0	0	0	0	1	0	

In [9]:

```
# to frame () is used to represent the data without the index
```

In [10]:

```
df["Route"].mode()
```

Out[10]:

```
0    DEL → BOM → COK
Name: Route, dtype: object
```

In [11]:

```
df["Total_Stops"].mode()
```

Out[11]:

```
0    1 stop
Name: Total_Stops, dtype: object
```

In [12]:

```
df[df["Route"].isnull()]
# means route and total stops has none values
```

Out[12]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
9039	Air India	6/05/2019	Delhi	Cochin	NaN	09:45	09:25 07 May	23h 40r

Handling Missing Values

In [13]:

```
df[["Route", "Total_Stops"]].dtypes
```

Out[13]:

```
Route      object
Total_Stops object
dtype: object
```

In [14]:

```
df["Route"].mode()
```

Out[14]:

```
0    DEL → BOM → COK
Name: Route, dtype: object
```

In [15]:

```
df[df["Route"].isnull()]
```

Out[15]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration
9039	Air India	6/05/2019	Delhi	Cochin	NaN	09:45	09:25 07 May	23h 40r

In [16]:

```
df["Route"] = df["Route"].fillna(df["Route"].mode()[0])
df["Total_Stops"] = df["Total_Stops"].fillna(df["Total_Stops"].mode()[0])
```

In [17]:

```
df.isnull().sum().sum()
```

Out[17]:

0

In [18]:

```
df.dtypes
```

Out[18]:

```
Airline      object
Date_of_Journey  object
Source       object
Destination  object
Route        object
Dep_Time     object
Arrival_Time  object
Duration     object
Total_Stops  object
Additional_Info object
Price        int64
dtype: object
```

In [114]:

```
#df["Duration"].unique()
```

In [20]:

```
s='2h 50m'
s=s.replace('h','*h').replace('m','*1').replace(' ','+')
print(s)
h=60
```

2*h+50*1

In [21]:

```
eval(s)
```

Out[21]:

170

In [22]:

```
df[df['Duration']=='5m']
```

Out[22]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duratio
6474	Air India	6/03/2019	Mumbai	Hyderabad	BOM → GOI → PNQ → HYD	16:50	16:55	5r

In [23]:

```
for i in df["Duration"].unique():
    if (len(i)==3 and 'm' in i) or (len(i)==2 and 'm' in i) :
        print(i)
```

5m

In [24]:

```
eval("3+8*2")
```

Out[24]:

19

In [25]:

```
df["Duration"]=df["Duration"].str.replace("h","*60").str.replace(" ","+").str.replace("m","")
```

In [26]:

```
df["Duration"]
```

Out[26]:

```
0      170
1      445
2     1140
3      325
4      285
...
10678   150
10679   155
10680   180
10681   160
10682   500
Name: Duration, Length: 10683, dtype: int64
```

Feature Engineering

In [27]:

```
## change the types of arrival time,departure time, date in datetime and extarct the day
```

In [28]:

```
df['Date_of_Journey']=pd.to_datetime(df['Date_of_Journey'])
df["Dep_Time"]=pd.to_datetime(df['Dep_Time'])
df["Arrival_Time"]=pd.to_datetime(df['Arrival_Time'])
```

Checking Dtypes of above columns

In [29]:

```
df.dtypes.to_frame().rename(columns={0:"dtypes"}).T
```

Out[29]:

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	I
dtypes	object	datetime64[ns]	object	object	object	datetime64[ns]	datetime64[ns]	

In [30]:

```
# Extracting day
df["day_of_journey"] = df["Date_of_Journey"].dt.day

# Extracting month
df["month_of_journey"] = df["Date_of_Journey"].dt.month

## Extracting departure Hours
df["Dep_hour"] = df["Dep_Time"].dt.hour

# Extracting departure Minutes
df["Dep_min"] = df["Dep_Time"].dt.minute

## Extracting arrival Hours
df["arrival_hour"] = df["Arrival_Time"].dt.hour

# Extracting arrival Minutes
df["arrival_min"] = df["Arrival_Time"].dt.minute
```

In [31]:

```
df.drop(columns=["Date_of_Journey","Dep_Time","Arrival_Time"],inplace=True)
df.shape
```

Out[31]:

(10683, 14)

Handling Inconsistency

In [32]:

```
df["Airline"].unique()
```

Out[32]:

```
array(['IndiGo', 'Air India', 'Jet Airways', 'SpiceJet',  
      'Multiple carriers', 'GoAir', 'Vistara', 'Air Asia',  
      'Vistara Premium economy', 'Jet Airways Business',  
      'Multiple carriers Premium economy', 'Trujet'], dtype=object)
```

In [33]:

```
df["Airline"] = df["Airline"].str.replace("Vistara Premium economy", "Vistara")  
df["Airline"] = df["Airline"].str.replace("Jet Airways Business", "Jet Airways")  
df["Airline"] = df["Airline"].str.replace("Multiple carriers Premium economy", "Multiple ca
```

In [34]:

```
df["Airline"].unique()
```

Out[34]:

```
array(['IndiGo', 'Air India', 'Jet Airways', 'SpiceJet',  
      'Multiple carriers', 'GoAir', 'Vistara', 'Air Asia', 'Trujet'],  
      dtype=object)
```

In [35]:

```
df["Destination"].unique()
```

Out[35]:

```
array(['New Delhi', 'Bangalore', 'Cochin', 'Kolkata', 'Delhi', 'Hyderaba  
d'],  
      dtype=object)
```

In [36]:

```
df["Destination"] = df["Destination"].replace("New Delhi", "Delhi")
```

In [37]:

```
df["Destination"].unique()
```

Out[37]:

```
array(['Delhi', 'Bangalore', 'Cochin', 'Kolkata', 'Hyderabad'],  
      dtype=object)
```

In [38]:

```
df["Total_Stops"].unique()
```

Out[38]:

```
array(['non-stop', '2 stops', '1 stop', '3 stops', '4 stops'],  
      dtype=object)
```

In [39]:

```
df["Total_Stops"]=df["Total_Stops"].map({"non-stop":0,"2 stops":2,"1 stop":1,"3 stops":3
```

In [40]:

```
df["Total_Stops"].unique()
```

Out[40]:

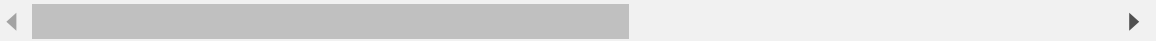
```
array([0, 2, 1, 3, 4], dtype=int64)
```

In [41]:

```
df.head()
```

Out[41]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	day_of
0	IndiGo	Banglore	Delhi	BLR → DEL	170	0	No info	3897	
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445	2	No info	7662	
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140	2	No info	13882	
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325	1	No info	6218	
4	IndiGo	Banglore	Delhi	BLR → NAG → DEL	285	1	No info	13302	



INSIGHTS

Descriptive Statistics

In [42]:

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

Out[42]:

	count	mean	std	min	25%	50%	75%	max
Duration	10683.0	643.093232	507.862001	5.0	170.0	520.0	930.0	2860
Total_Stops	10683.0	0.824207	0.675199	0.0	0.0	1.0	1.0	4
Price	10683.0	9087.064121	4611.359167	1759.0	5277.0	8372.0	12373.0	79512
day_of_journey	10683.0	12.682205	8.803701	3.0	5.0	6.0	21.0	27
month_of_journey	10683.0	5.534775	2.987489	1.0	3.0	5.0	6.0	12
Dep_hour	10683.0	12.490686	5.748650	0.0	8.0	11.0	18.0	23
Dep_min	10683.0	24.411214	18.767980	0.0	5.0	25.0	40.0	55
arrival_hour	10683.0	13.348778	6.859125	0.0	8.0	14.0	19.0	23
arrival_min	10683.0	24.690630	16.506036	0.0	10.0	25.0	35.0	55

In [43]:

```
df[df["Duration"]==5.0]
```

Out[43]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	day
6474	Air India	Mumbai	Hyderabad	BOM → GOI → PNQ → HYD	5	2	No info	17327	

In [44]:

```
x=df[(df["Airline"]=="Air India")&(df["Source"]=="Mumbai")&(df["Destination"]=="Hyderaba
```

In [45]:

```
x
```

Out[45]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	day_of_journey
597	Air India	Mumbai	Hyderabad	BOM → JDH → DEL → HYD	1775	2	No info	25139	3
1417	Air India	Mumbai	Hyderabad	BOM → AMD → ISK → HYD	795	2	No info	9977	3
				BOM					

In [46]:

```
x["Duration"].mean()
```

Out[46]:

1070.6521739130435

In [47]:

```
1070.65/60      # 84-60min(1hr) =24 min = 17+1 = 18h 24 min
```

Out[47]:

17.844166666666667

In [48]:

```
round(x["Duration"].mean(),2)
```

Out[48]:

1070.65

In [49]:

```
18*60  # 18H 24 m
```

Out[49]:

1080

In [50]:

```
df.loc[6474,"Duration"]=round(x["Duration"].mean(),2)
df.iloc[6474]
```

Out[50]:

Airline	Air India
Source	Mumbai
Destination	Hyderabad
Route	BOM → GOI → PNQ → HYD
Duration	1070.65
Total_Stops	2
Additional_Info	No info
Price	17327
day_of_journey	3
month_of_journey	6
Dep_hour	16
Dep_min	50
arrival_hour	16
arrival_min	55

Name: 6474, dtype: object

In [51]:

```
df.loc[6474,"arrival_hour"]=18
df.loc[6474,"arrival_min"]=24
df.iloc[6474]
```

Out[51]:

Airline	Air India
Source	Mumbai
Destination	Hyderabad
Route	BOM → GOI → PNQ → HYD
Duration	1070.65
Total_Stops	2
Additional_Info	No info
Price	17327
day_of_journey	3
month_of_journey	6
Dep_hour	16
Dep_min	50
arrival_hour	18
arrival_min	24

Name: 6474, dtype: object

Correlation

In [52]:

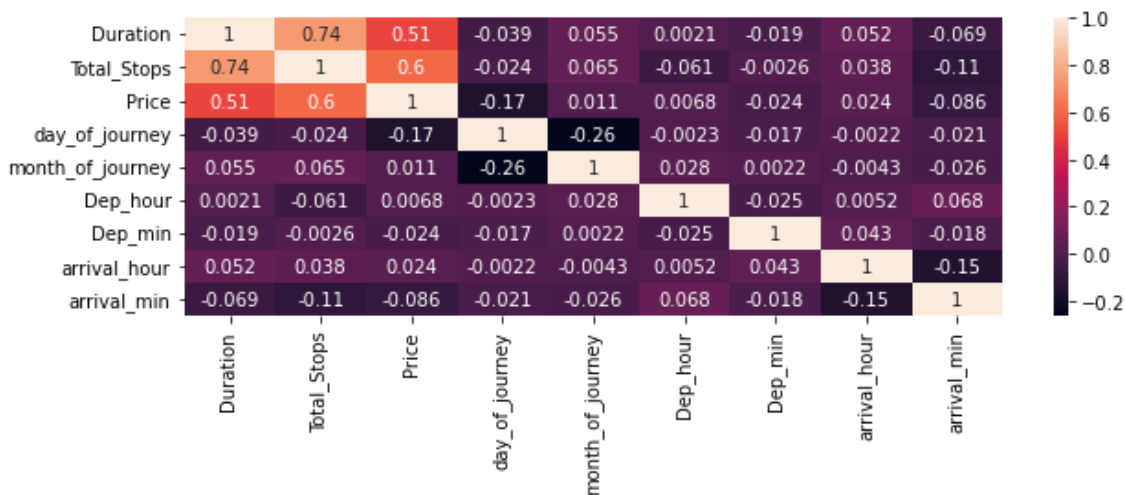
df.corr()

Out[52]:

	Duration	Total_Stops	Price	day_of_journey	month_of_journey	Dep_
Duration	1.000000	0.738397	0.506743	-0.038697	0.055185	0.002121
Total_Stops	0.738397	1.000000	0.603883	-0.024156	0.065498	-0.061490
Price	0.506743	0.603883	1.000000	-0.165412	0.010700	0.006819
day_of_journey	-0.038697	-0.024156	-0.165412	1.000000	-0.264899	-0.002251
month_of_journey	0.055185	0.065498	0.010700	-0.264899	1.000000	0.028180
Dep_hour	0.002121	-0.061490	0.006819	-0.002251	0.028180	1.000000
Dep_min	-0.018671	-0.002591	-0.024492	-0.016521	0.002152	-0.002154
arrival_hour	0.051534	0.038170	0.024312	-0.002154	-0.004334	0.000434
arrival_min	-0.069454	-0.107262	-0.086483	-0.021234	-0.025817	0.000434

In [53]:

```
plt.figure(figsize=(10,3))
sns.heatmap(df.corr(),cmap=None,annot=True);
# near to the one shows it is highly correlated i.e, duration and total stops are highly
# with increase of total_stops there is an increase in duration and vice-versa
```



-Observation:

- From this output we can see there is strong positive correlation between total stops and duration and it is also valid as total stops increases the duration also increase
- also positive correlation found between price and total stops, and Duration and Price

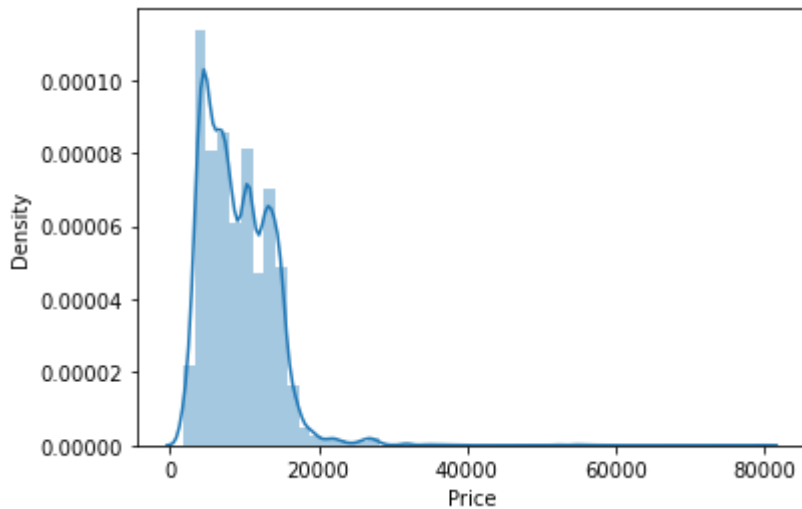
Since Price is the Main aspect so we do EDA w.r.t price column

In [54]:

```
sns.distplot(df["Price"])
```

Out[54]:

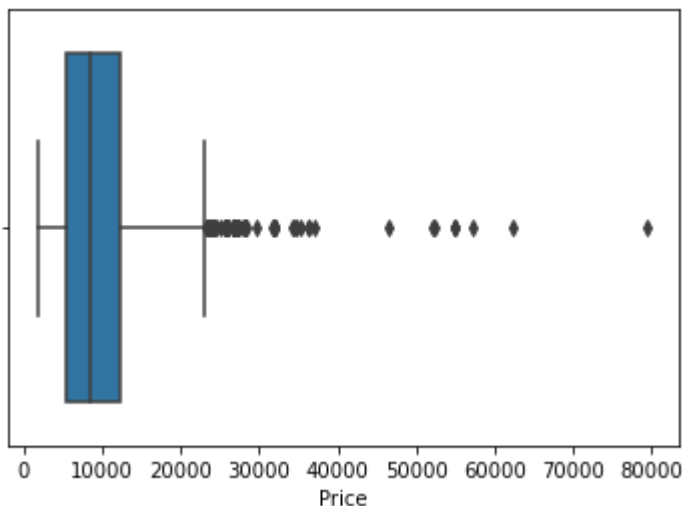
<AxesSubplot:xlabel='Price', ylabel='Density'>

**-Observation:**

- the distribution of price is right skewed so it has outliers and most of the flights price is foundm around 8k to 10k and there are some flights which are extremely expensive

In [55]:

```
sns.boxplot(x="Price", data=df);
```



FIND THE DETAIL OF EXPENSIVE FLIGHT.

In [56]:

```
df[df["Price"]==df["Price"].max()]
```

Out[56]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	d
2924	Jet Airways	Banglore	Delhi	BLR → BOM → DEL	340.0	1	Business class	79512	

FIND OUT THE MIN , MAX AND THE AVERAGE PRICE OF FLIGHTS

In [57]:

```
df["Price"].agg(["max","min","mean"])
```

Out[57]:

max 79512.000000
min 1759.000000
mean 9087.064121
Name: Price, dtype: float64

DETAILS OF THE CHEAPEST FLIGHT

In [58]:

```
df[df["Price"]==df["Price"].min()]
```

Out[58]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	d
4066	SpiceJet	Mumbai	Hyderabad	BOM → HYD	85.0	0	No info	1759	
4274	SpiceJet	Mumbai	Hyderabad	BOM → HYD	85.0	0	No info	1759	
4839	SpiceJet	Mumbai	Hyderabad	BOM → HYD	90.0	0	No info	1759	
10513	SpiceJet	Mumbai	Hyderabad	BOM → HYD	80.0	0	No info	1759	

since there are outliers in the data so mean has been highly effected by outliers so we are taking the data excluding high extreme price' flights

In [59]:

```
x_mean=df[df["Price"]<45000]["Price"].mean()  
x_mean
```

Out[59]:

9046.512647554806

TOTAL NO. OF FLIGHTS WHOSE PRICE IS LESS THAN THE AVERAGE PRICE

In [60]:

```
len(df[df["Price"]<x_mean])
```

Out[60]:

5793

In [61]:

```
len(df[df["Price"]>=x_mean])
```

Out[61]:

4890

FIND OUT THE TOTAL NO. OF FLIGHTS OF EACH COMPANY or THE MOST DEMANDING FLIGHT COMPANY

In [62]:

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

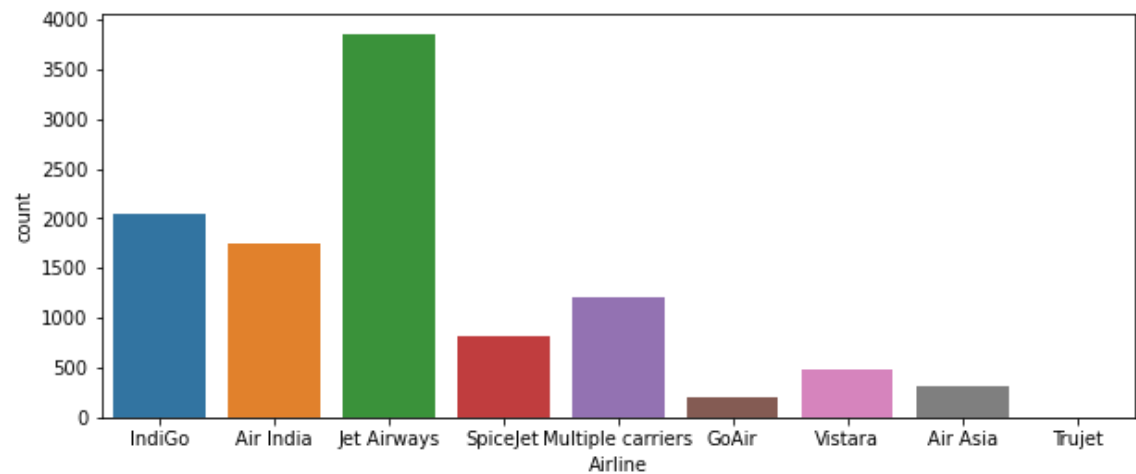
Out[62]:

Jet Airways	3855
IndiGo	2053
Air India	1752
Multiple carriers	1209
SpiceJet	818
Vistara	482
Air Asia	319
GoAir	194
Trujet	1

Name: Airline, dtype: int64

In [63]:

```
plt.figure(figsize=(10,4))
sns.countplot(x="Airline",data=df);
```



THE MOST EXPENSIVE FLIGHT W.R.T THE COMPANY

In [64]:

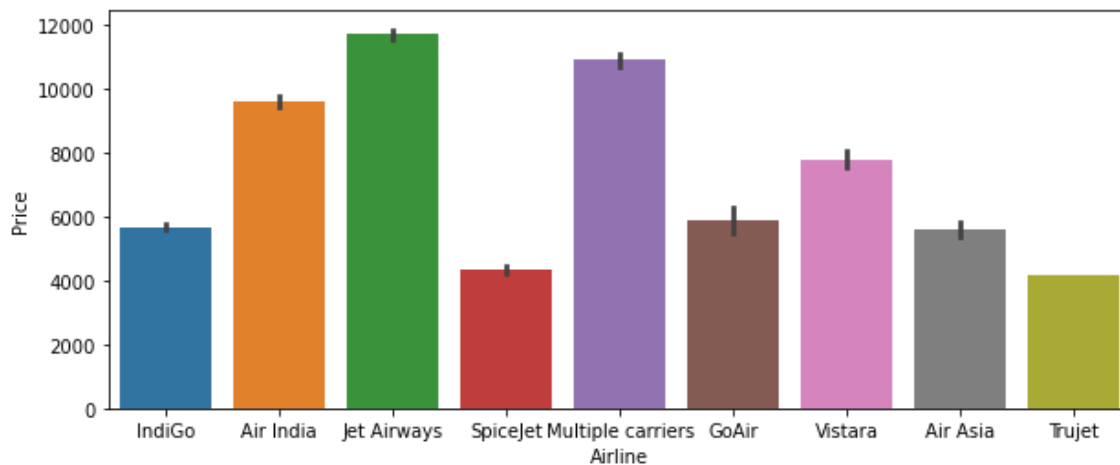
```
df.groupby('Airline')['Price'].agg(['mean']).sort_values(by='mean',ascending=False)
```

Out[64]:

mean	
Airline	
Jet Airways	11716.631128
Multiple carriers	10908.228288
Air India	9611.210616
Vistara	7803.605809
GoAir	5861.056701
IndiGo	5673.682903
Air Asia	5590.260188
SpiceJet	4338.284841
Trujet	4140.000000

In [65]:

```
plt.figure(figsize=(10,4))
sns.barplot(x="Airline",y="Price",data=df);
```

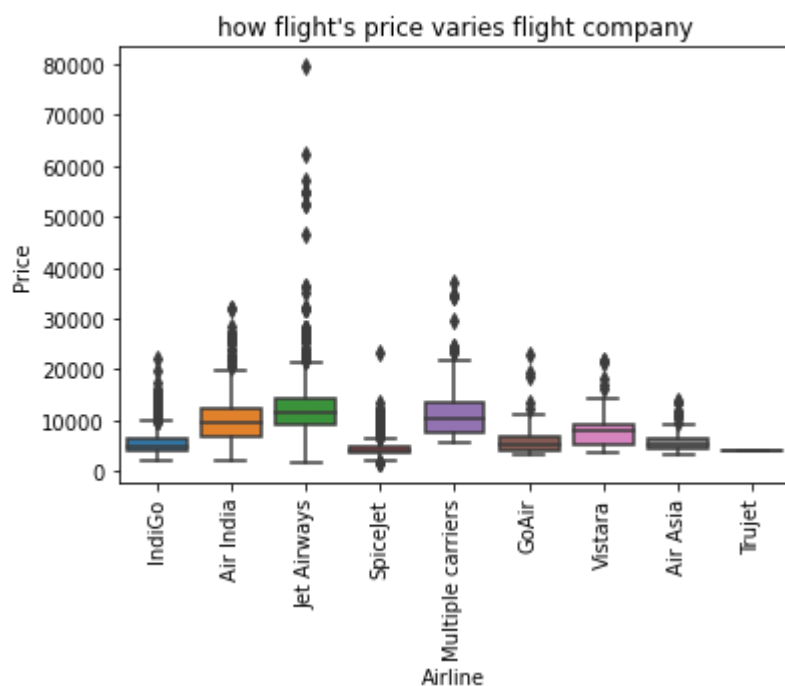


From "customer" point of view expensive flight is "jet airways" From "business" point of "jet airways" is found to be most selling company

HOW PRICE VARIES W.R.T EACH FLIGHTS' COMPANY

In [66]:

```
sns.boxplot(x="Airline",y="Price",data=df);
plt.xticks(rotation=90)
plt.title("how flight's price varies flight company");
```



The expensive flight is "jet-airways business" The cheapest flight is "Trujet company"

FIND OUT THE PRICE OF THE EXPENSIVE FLIGHTS' AND THE CHEAPEST FLIGHTS' COMPANY

In [67]:

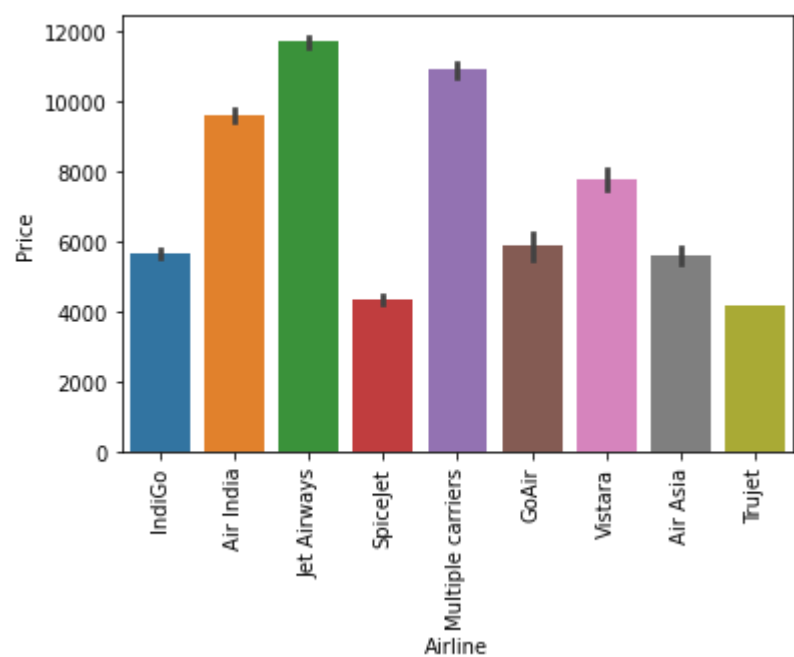
```
df.groupby('Airline')['Price'].agg(['min','max'])
```

Out[67]:

	min	max
Airline		
Air Asia	3383	13774
Air India	2050	31945
GoAir	3398	22794
IndiGo	2227	22153
Jet Airways	1840	79512
Multiple carriers	5797	36983
SpiceJet	1759	23267
Trujet	4140	4140
Vistara	3687	21730

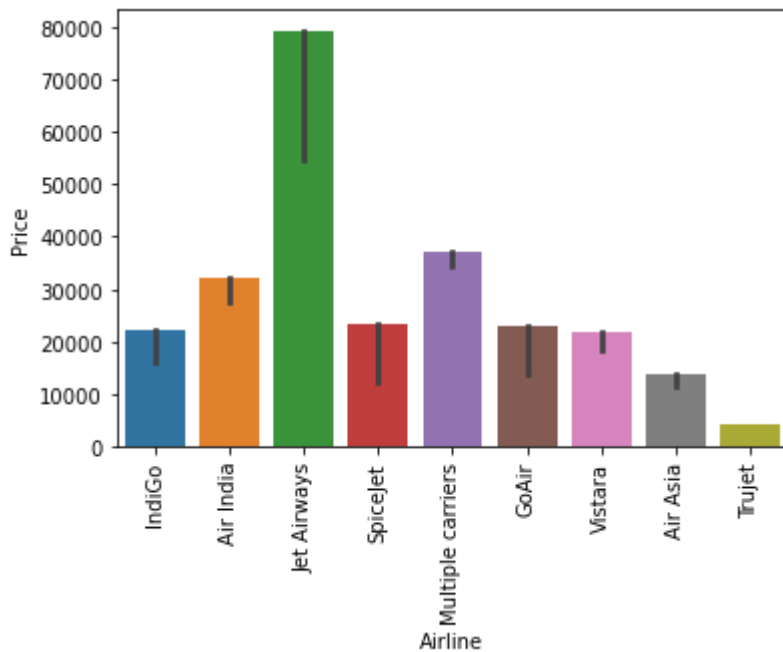
In [68]:

```
sns.barplot(x="Airline",y="Price",data=df);  
plt.xticks(rotation=90);  
# by default it takes avg
```



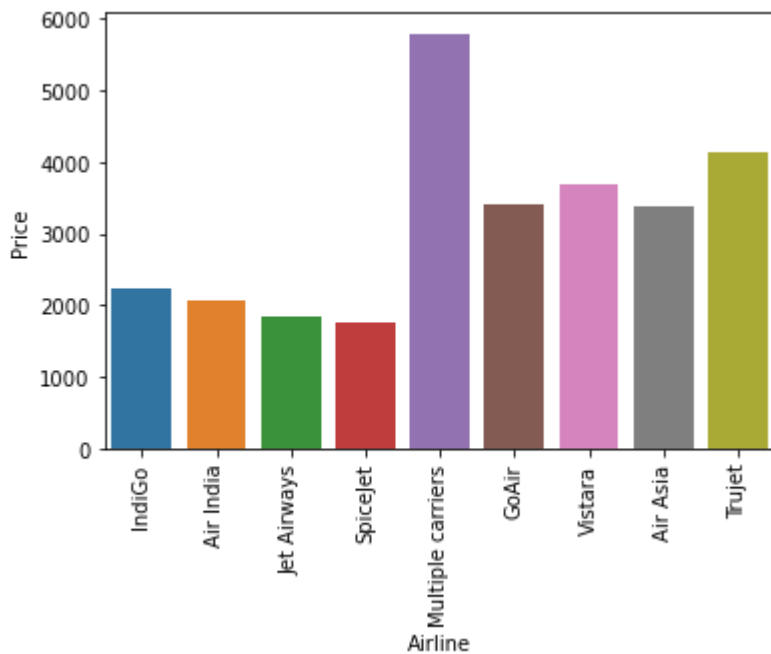
In [69]:

```
sns.barplot(x="Airline",y="Price",estimator=max,data=df);  
plt.xticks(rotation=90);
```



In [70]:

```
sns.barplot(x="Airline",y="Price",estimator=min,data=df,ci=None);  
plt.xticks(rotation=90);
```



NO.OF FLIGHTS W.R.T THEIR STOPPAGES

In [71]:

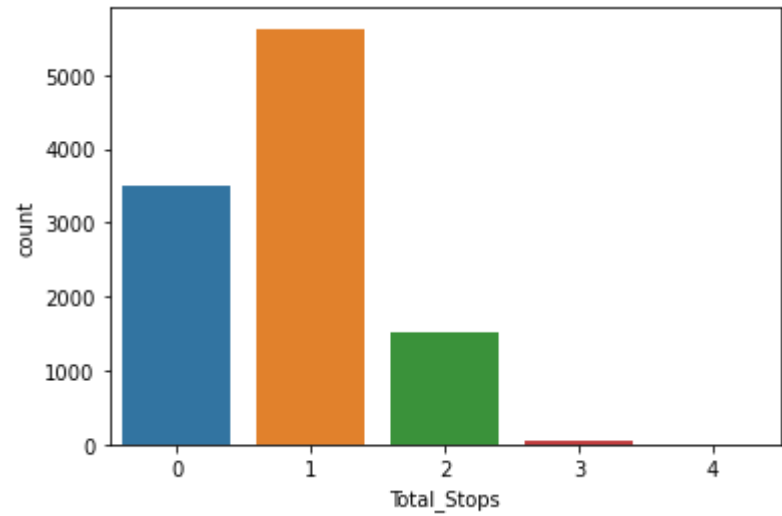
```
df['Total_Stops'].value_counts()
```

Out[71]:

```
1    5626
0    3491
2    1520
3      45
4       1
Name: Total_Stops, dtype: int64
```

In [72]:

```
sns.countplot(x='Total_Stops',data=df);
```



In [73]:

```
df[df["Total_Stops"]==4]
```

Out[73]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	da
9182	Air India	Banglore	Delhi	BLR	1770.0	4	Change airports	17686	
				→					
				CCU					
				→					
				BBi					
				→					
				HYD					
→									
				VGA					
				→					
				DEL					

We can see that no. of flights and their stoppages, In this data maximum flights have 1 stoppages and there are few flights which have 3 to 4 stoppages

FIND THE FLIGHTS AVAILABILITY W.R.T NO. OF STOPPAGES

In [74]:

```
df.groupby(["Total_Stops"])[ "Airline"].value_counts()
```

Out[74]:

```
Total_Stops  Airline
0            IndiGo          1241
             SpiceJet           670
             Jet Airways        623
             Air India          417
             Vistara           267
             Air Asia           181
             GoAir              92
1            Jet Airways       2539
             Multiple carriers  1158
             IndiGo             793
             Air India          541
             Vistara           215
             SpiceJet          148
             Air Asia          129
             GoAir             102
             Trujet             1
2            Air India          756
             Jet Airways        693
             Multiple carriers   43
             IndiGo             19
             Air Asia            9
3            Air India          37
             Multiple carriers    8
4            Air India            1
Name: Airline, dtype: int64
```

In [75]:

```
df.groupby('Airline')['Total_Stops'].agg(['min', 'max'])
```

Out[75]:

	min	max
Airline		
Air Asia	0	2
Air India	0	4
GoAir	0	1
IndiGo	0	2
Jet Airways	0	2
Multiple carriers	1	3
SpiceJet	0	1
Trujet	1	1
Vistara	0	1

"Indigo" has highest number of flights available with 0 stoppages, Jet Airways highest number of flights available with 1 stoppages, Air india highest number of flights available with 2 stoppages, Air india highest number of flights available with 3 stoppages, Air india is the only flight available with 4 stoppages.

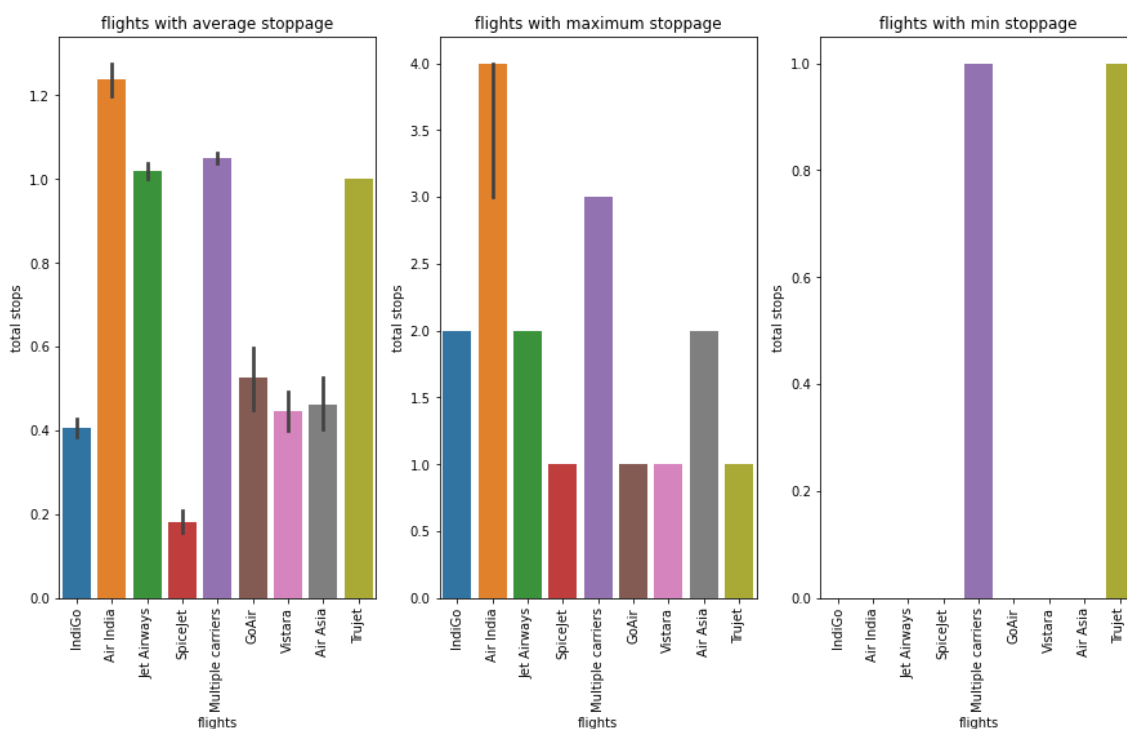
FIND THE MIN, MAX AND AVERAGE STOPPAGE OF EACH FLIGHT

In [76]:

```
plt.figure(figsize=(15,8))
plt.subplot(1,3,1)
sns.barplot(x="Airline",y="Total_Stops",data=df)
plt.title("flights with average stoppage")
plt.xlabel("flights")
plt.ylabel("total stops")
plt.xticks(rotation=90)

plt.subplot(1,3,2)
sns.barplot(x="Airline",y="Total_Stops",data=df,estimator=max)
plt.title("flights with maximum stoppage")
plt.xlabel("flights")
plt.ylabel("total stops")
plt.xticks(rotation=90)

plt.subplot(1,3,3)
sns.barplot(x="Airline",y="Total_Stops",data=df,estimator=min)
plt.title("flights with min stoppage")
plt.xlabel("flights")
plt.ylabel("total stops")
plt.xticks(rotation=90);
```



There is only one flight named "Vistara Premium economy" having "0" stoppage and "Air India" is the flight which has "maximum" stoppages

FIND THE DETAILS OF VISTARA PREMIUM ECONOMY FLIGHT

In [77]:

```
df[df['Airline']=='Vistara']
```

Out[77]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	d
28	Vistara	Banglore	Delhi	BLR → DEL	170.0	0	No info	4668	
29	Vistara	Chennai	Kolkata	MAA → CCU	135.0	0	No info	3687	
30	Vistara	Chennai	Kolkata	MAA → CCU	135.0	0	No info	3687	
57	Vistara	Chennai	Kolkata	MAA → CCU	135.0	0	No info	7414	
59	Vistara	Mumbai	Hyderabad	BOM → DEL → HYD	1505.0	1	No info	12395	
...	
10624	Vistara	Kolkata	Banglore	CCU → DEL → BLR	1580.0	1	No info	8662	
10656	Vistara	Banglore	Delhi	BLR → DEL	160.0	0	No info	5613	
10659	Vistara	Banglore	Delhi	BLR → DEL	170.0	0	No info	4668	
10660	Vistara	Banglore	Delhi	BLR → DEL	175.0	0	No info	4878	
10681	Vistara	Banglore	Delhi	BLR → DEL	160.0	0	No info	12648	

482 rows × 14 columns



HOW DOES THE NO. OF STOPPAGES AND DURATION VARIES WITH THE PRICE?

In [78]:

```
print(df.corr()["Total_Stops"]["Price"])
print(df.corr()["Duration"]["Price"])
```

0.6038830640858682

0.5067431283617874

WHICH FLIGHT IS AVAILABLE FROM SOURCE TO DESTINATION or CHECKING THE AVAILABILITY OF FLIGHT W.R.T THE SOURCE AND DESTINATION

In [79]:

```
df.groupby(["Source", "Destination"])["Airline"].value_counts()
```

Out[79]:

Source	Destination	Airline	
Bangalore	Delhi	Jet Airways	792
		IndiGo	523
		Air India	332
		Vistara	187
		SpiceJet	181
		GoAir	93
		Air Asia	89
Chennai	Kolkata	IndiGo	184
		SpiceJet	128
		Vistara	44
		Air India	25
		Jet Airways	1588
Delhi	Cochin	Multiple carriers	1209
		Air India	747
		IndiGo	705
		SpiceJet	87
		Air Asia	80
		GoAir	76
		Vistara	45
Kolkata	Bangalore	Jet Airways	1256
		Air India	512
		IndiGo	445
		SpiceJet	300
		Vistara	183
		Air Asia	150
		GoAir	25
Mumbai	Hyderabad	Jet Airways	219
		IndiGo	196
		Air India	136
		SpiceJet	122
		Vistara	23
		Trujet	1

Name: Airline, dtype: int64

FIND THE MAX , MIN AND THE AVERAGE PRICE OF DIFFERENT FLIGHT FROM
SOURCE TO DESTINATION

In [80]:

```
k=df.groupby(["Source", "Destination", "Airline"])[ "Price"].agg(["mean", "max", "min"])
k.reset_index(inplace=True)
k
```

Out[80]:

	Source	Destination	Airline	mean	max	min
0	Banglore	Delhi	Air Asia	4574.280899	10873	3383
1	Banglore	Delhi	Air India	9238.198795	31783	3758
2	Banglore	Delhi	GoAir	4948.881720	18558	3398
3	Banglore	Delhi	IndiGo	5274.112811	22153	3359
4	Banglore	Delhi	Jet Airways	11283.462121	79512	3359
5	Banglore	Delhi	SpiceJet	4550.292818	23267	3257
6	Banglore	Delhi	Vistara	6211.037433	21730	4353
7	Chennai	Kolkata	Air India	5895.640000	19630	3145
8	Chennai	Kolkata	IndiGo	4538.766304	8580	3384
9	Chennai	Kolkata	SpiceJet	3993.523438	7718	3332
10	Chennai	Kolkata	Vistara	7528.500000	11982	3687
11	Delhi	Cochin	Air Asia	7804.175000	13774	6151
12	Delhi	Cochin	Air India	10000.068273	28322	4487
13	Delhi	Cochin	GoAir	6587.157895	22794	3876
14	Delhi	Cochin	IndiGo	7203.933333	16162	4729
15	Delhi	Cochin	Jet Airways	12688.871537	52285	4256
16	Delhi	Cochin	Multiple carriers	10908.228288	36983	5797
17	Delhi	Cochin	SpiceJet	5916.356322	11726	4098
18	Delhi	Cochin	Vistara	6465.644444	12411	4851
19	Kolkata	Banglore	Air Asia	5012.320000	11323	3782
20	Kolkata	Banglore	Air India	10357.324219	31945	4145
21	Kolkata	Banglore	GoAir	7047.000000	10586	3514
22	Kolkata	Banglore	IndiGo	5075.235955	12198	3480
23	Kolkata	Banglore	Jet Airways	11717.565287	15149	5608
24	Kolkata	Banglore	SpiceJet	4642.883333	12287	3815
25	Kolkata	Banglore	Vistara	9257.683060	16932	7770
26	Mumbai	Hyderabad	Air India	6260.051471	25139	2050
27	Mumbai	Hyderabad	IndiGo	3659.816327	17501	2227
28	Mumbai	Hyderabad	Jet Airways	6227.949772	24210	1840
29	Mumbai	Hyderabad	SpiceJet	2511.106557	13552	1759
30	Mumbai	Hyderabad	Trujet	4140.000000	4140	4140
31	Mumbai	Hyderabad	Vistara	12326.521739	12395	12080

In [81]:

```
k.set_index(["Source", "Destination"])
```

Out[81]:

		Airline	mean	max	min
Source	Destination				
Bangalore	Delhi	Air Asia	4574.280899	10873	3383
	Delhi	Air India	9238.198795	31783	3758
	Delhi	GoAir	4948.881720	18558	3398
	Delhi	IndiGo	5274.112811	22153	3359
	Delhi	Jet Airways	11283.462121	79512	3359
	Delhi	SpiceJet	4550.292818	23267	3257
	Delhi	Vistara	6211.037433	21730	4353
Chennai	Kolkata	Air India	5895.640000	19630	3145
	Kolkata	IndiGo	4538.766304	8580	3384
	Kolkata	SpiceJet	3993.523438	7718	3332
	Kolkata	Vistara	7528.500000	11982	3687
Delhi	Cochin	Air Asia	7804.175000	13774	6151
	Cochin	Air India	10000.068273	28322	4487
	Cochin	GoAir	6587.157895	22794	3876
	Cochin	IndiGo	7203.933333	16162	4729
	Cochin	Jet Airways	12688.871537	52285	4256
	Cochin	Multiple carriers	10908.228288	36983	5797
	Cochin	SpiceJet	5916.356322	11726	4098
	Cochin	Vistara	6465.644444	12411	4851
Kolkata	Bangalore	Air Asia	5012.320000	11323	3782
	Bangalore	Air India	10357.324219	31945	4145
	Bangalore	GoAir	7047.000000	10586	3514
	Bangalore	IndiGo	5075.235955	12198	3480
	Bangalore	Jet Airways	11717.565287	15149	5608
	Bangalore	SpiceJet	4642.883333	12287	3815
	Bangalore	Vistara	9257.683060	16932	7770
Mumbai	Hyderabad	Air India	6260.051471	25139	2050
	Hyderabad	IndiGo	3659.816327	17501	2227
	Hyderabad	Jet Airways	6227.949772	24210	1840
	Hyderabad	SpiceJet	2511.106557	13552	1759
	Hyderabad	Trujet	4140.000000	4140	4140
	Hyderabad	Vistara	12326.521739	12395	12080

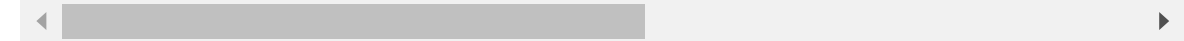
In [82]:

df

Out[82]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	Banglore	Delhi	BLR → DEL	170.0	0	No info	3897
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445.0	2	No info	7662
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140.0	2	No info	13882
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325.0	1	No info	6218
4	IndiGo	Banglore	Delhi	BLR → NAG → DEL	285.0	1	No info	13302
...
10678	Air Asia	Kolkata	Banglore	CCU → BLR	150.0	0	No info	4107
10679	Air India	Kolkata	Banglore	CCU → BLR	155.0	0	No info	4145
10680	Jet Airways	Banglore	Delhi	BLR → DEL	180.0	0	No info	7229
10681	Vistara	Banglore	Delhi	BLR → DEL	160.0	0	No info	12648
10682	Air India	Delhi	Cochin	DEL → GOI → BOM → COK	500.0	2	No info	11753

10683 rows × 14 columns



VARIATION OF FLIGHT PRICE FROM SOURCE

In [83]:

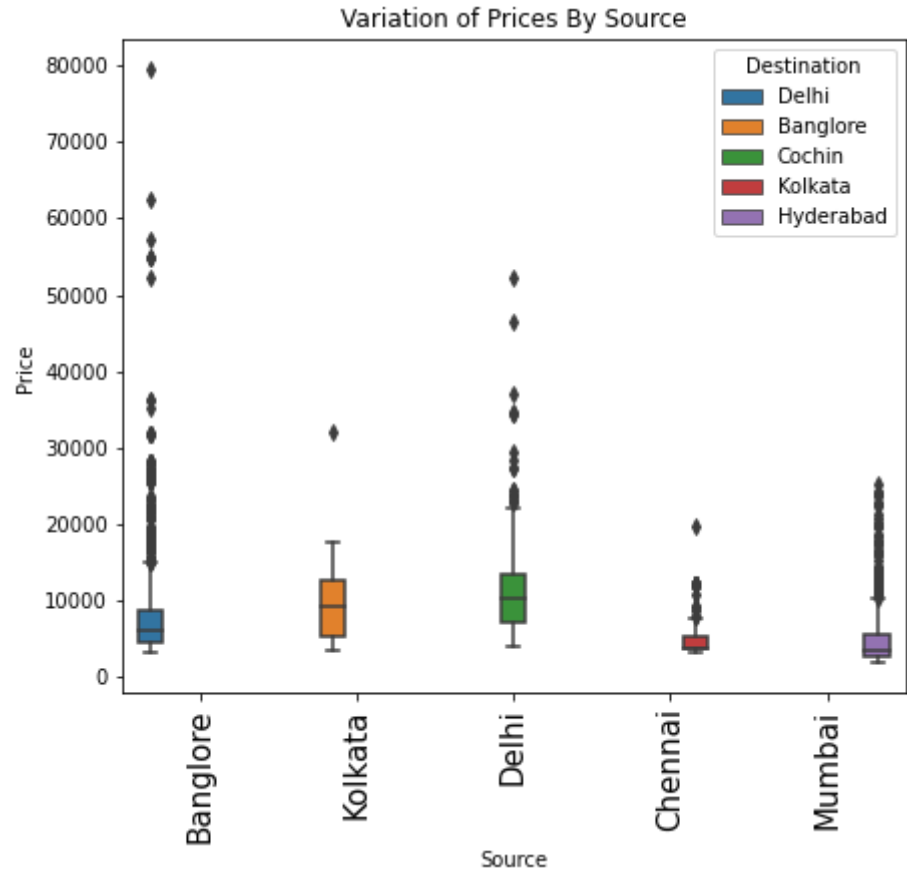
```
df.groupby(['Source', 'Destination'])['Price'].agg(['mean', 'sum'])
```

Out[83]:

		mean	sum
Source	Destination		
Banglore	Delhi	8017.464269	17614369
Chennai	Kolkata	4789.892388	1824949
Delhi	Cochin	10539.439057	47817435
Kolkata	Banglore	9158.389411	26293736
Mumbai	Hyderabad	5059.708752	3526617

In [84]:

```
plt.figure(figsize=(7,6))
sns.boxplot(x='Source',y='Price',hue='Destination',data=df)
plt.xticks(rotation=90,size=15)
plt.title('Variation of Prices By Source')
plt.show()
```



The Expensive Flight is taken off from "Bangalore" and The Cheapest flight is taken off

FIND THE DAY ON WHICH FLIGHTS TAKEN OFF WAS ON THE PEAK

Since date column have so many unique values that is why we extracted day and month from it

ON WHICH DAY AS WELL AS THE MONTH, THE FLIGHTS WERE AVAILABLE

In [85]:

```
df.head()
```

Out[85]:

	Airline	Source	Destination	Route	Duration	Total_Stops	Additional_Info	Price	day_of_journey
0	IndiGo	Banglore	Delhi	BLR → DEL	170.0	0	No info	3897	24
1	Air India	Kolkata	Banglore	CCU → IXR → BBI → BLR	445.0	2	No info	7662	5
2	Jet Airways	Delhi	Cochin	DEL → LKO → BOM → COK	1140.0	2	No info	13882	6
3	IndiGo	Kolkata	Banglore	CCU → NAG → BLR	325.0	1	No info	6218	3
4	IndiGo	Banglore	Delhi	BLR → NAG → DEL	285.0	1	No info	13302	27

In [86]:

```
df['day_of_journey'].unique()
```

Out[86]:

array([24, 5, 6, 3, 27, 18, 15, 21, 4], dtype=int64)

In [87]:

```
df['month_of_journey'].unique()
```

Out[87]:

```
array([ 3,  1,  9, 12,  6,  5,  4], dtype=int64)
```

There was no flight take off in month feb,july,aug,oct,nov

FIND THE MONTH WITH MAXIMUM FLIGHTS TAKE OFF

In [88]:

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

Out[88]:

```
6    2536
3    2211
5    2074
9    1406
1    1075
12    957
4     424
```

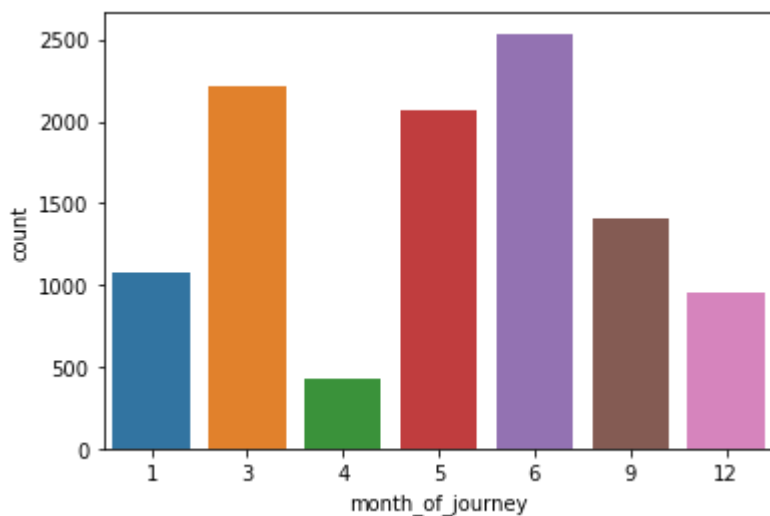
Name: month_of_journey, dtype: int64

In [89]:

```
# using countplot
sns.countplot(x="month_of_journey",data=df)
```

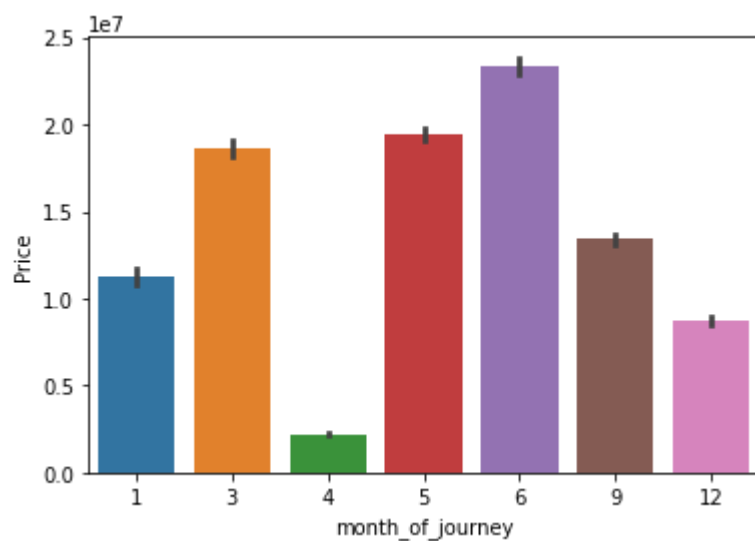
Out[89]:

<AxesSubplot:xlabel='month_of_journey', ylabel='count'>



In [90]:

```
# using barplot  
sns.barplot(x="month_of_journey",y="Price",data=df,estimator=sum);
```

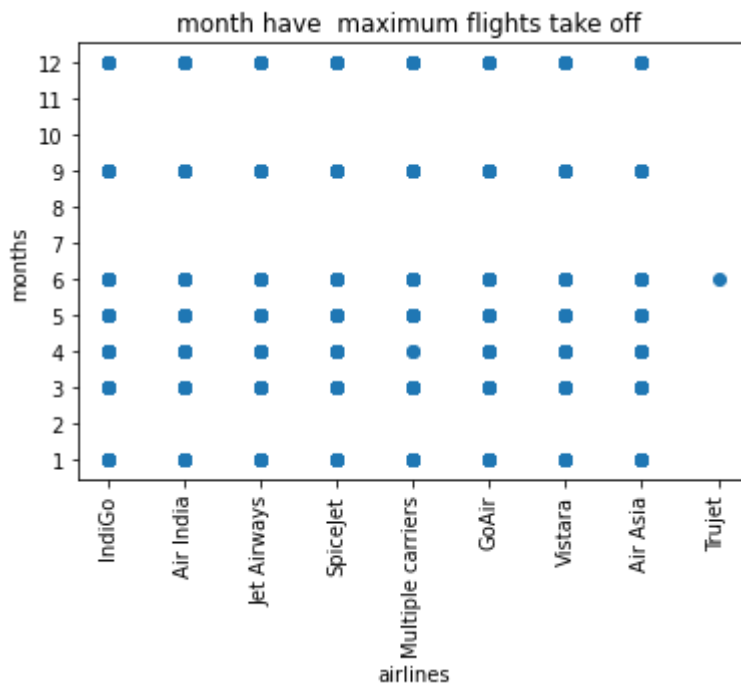


"June" was on the peak , where "maximum" flights taken off and "April" was the off season

WHICH FLIGHT ARE BEING TAKEN OFF IN WHICH MONTH?

In [91]:

```
plt.scatter(df["Airline"],df["month_of_journey"])
plt.xticks(rotation=90);
plt.ylabel("months")
plt.xlabel("airlines")
plt.yticks([1,2,3,4,5,6,7,8,9,10,11,12])
plt.title("month have maximum flights take off ");
```



HOW MANY FLIGHTS ARE TAKEN OFF IN EACH MONTH?

In [92]:

```
df.groupby(["month_of_journey"])[ "Airline"].value_counts()
```

Out[92]:

month_of_journey	Airline	
1	Jet Airways	434
	Air India	197
	IndiGo	184
	SpiceJet	76
	Multiple carriers	75
	Vistara	60
	Air Asia	28
	GoAir	21
3	Jet Airways	675
	IndiGo	513
	Air India	369
	Multiple carriers	274
	SpiceJet	193
	Vistara	85
	Air Asia	59
	GoAir	43
4	IndiGo	153
	SpiceJet	76
	Jet Airways	63
	Air India	45
	Air Asia	32
	Vistara	31
	GoAir	22
	Multiple carriers	2
5	Jet Airways	783
	Air India	352
	IndiGo	329
	Multiple carriers	292
	SpiceJet	139
	Vistara	90
	Air Asia	61
	GoAir	28
6	Jet Airways	932
	IndiGo	469
	Air India	386
	Multiple carriers	341
	SpiceJet	190
	Vistara	103
	Air Asia	69
	GoAir	45
	Trujet	1
9	Jet Airways	544
	IndiGo	253
	Air India	234
	Multiple carriers	160
	SpiceJet	90
	Vistara	61
	Air Asia	41
	GoAir	23
12	Jet Airways	424
	Air India	169
	IndiGo	152
	Multiple carriers	65
	SpiceJet	54
	Vistara	52
	Air Asia	29
	GoAir	12

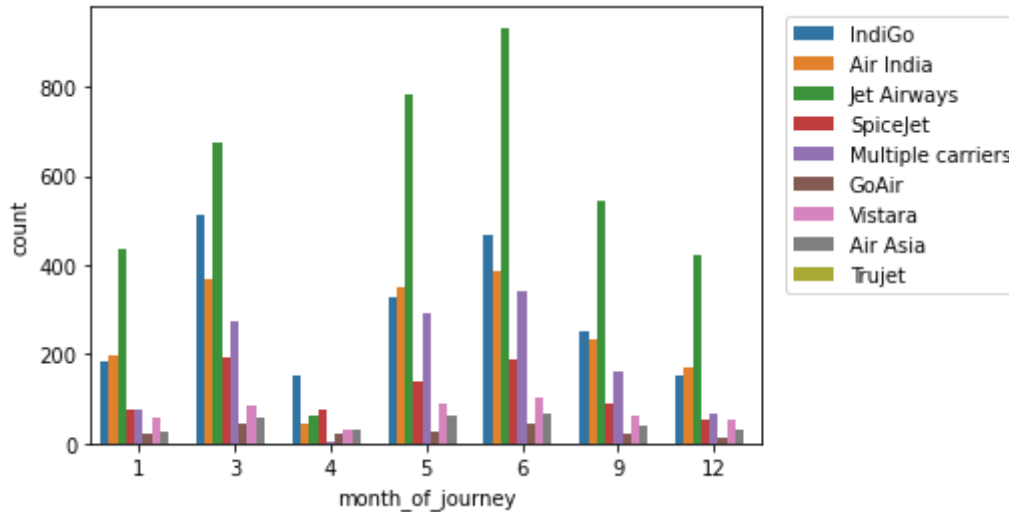
Name: Airline, dtype: int64

In [93]:

```
sns.countplot(x="month_of_journey",hue="Airline",data=df)
plt.legend(bbox_to_anchor=(1.4,1))
```

Out[93]:

```
<matplotlib.legend.Legend at 0x2006ff7c160>
```



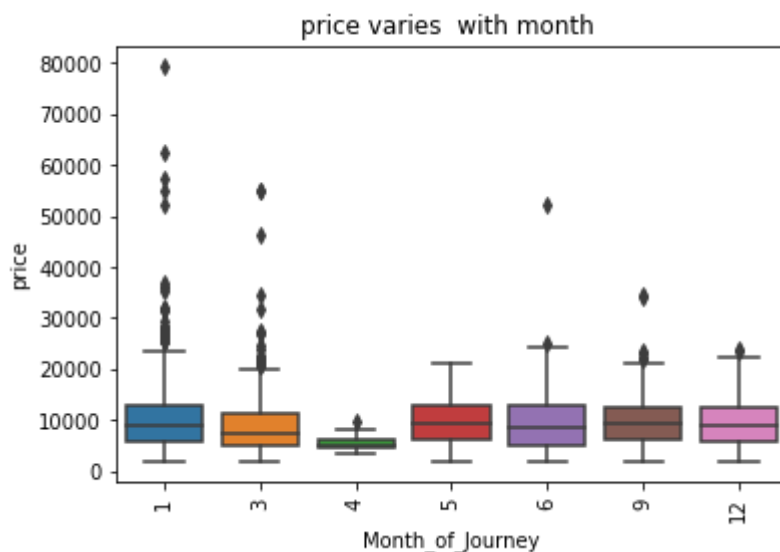
HOW MONTH OF JOURNEY VARIES WITH THE PRICE?

In [94]:

```
sns.boxplot(x="month_of_journey",y="Price",data=df)
plt.xticks(rotation=90);
plt.ylabel("price")
plt.xlabel("Month_of_Journey")
plt.title("price varies with month ")
```

Out[94]:

```
Text(0.5, 1.0, 'price varies with month ')
```



WHICH DAY OF JOURNEY WAS ON THE PEAK?

In [95]:

```
df["day_of_journey"].value_counts()
```

Out[95]:

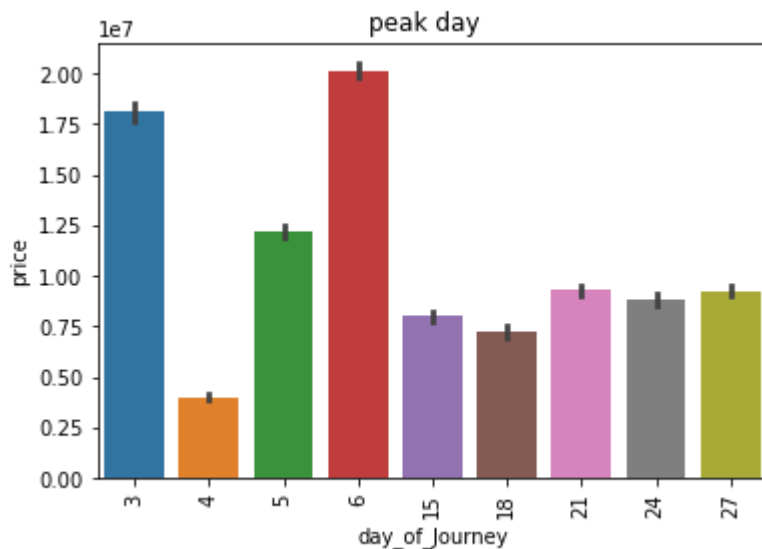
```
6      2166
5      1392
3      1361
27     1130
21     1111
24     1052
15      984
18      832
4       655
```

```
Name: day_of_journey, dtype: int64
```

6th was the day when passengers travelled the most

In [96]:

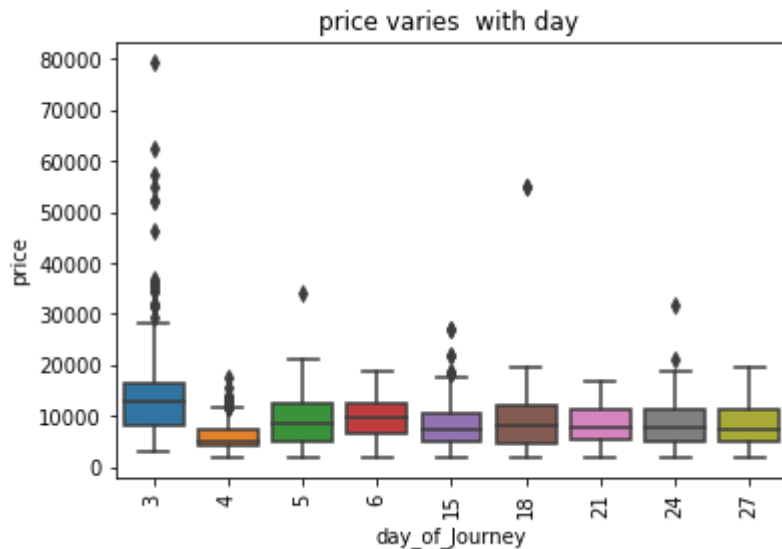
```
sns.barplot(x="day_of_journey",y="Price",estimator=sum,data=df)
plt.xticks(rotation=90);
plt.ylabel("price")
plt.xlabel("day_of_Journey")
plt.title("peak day ");
```



HOW DAY VARIES WITH THE PRICE?

In [97]:

```
sns.boxplot(x="day_of_journey",y="Price",data=df)
plt.xticks(rotation=90);
plt.ylabel("price")
plt.xlabel("day_of_Journey")
plt.title("price varies with day ");
```



WHICH DEPARTURE HOUR WAS ON THE PEAK?

In [98]:

```
df["Dep_hour"].value_counts()
```

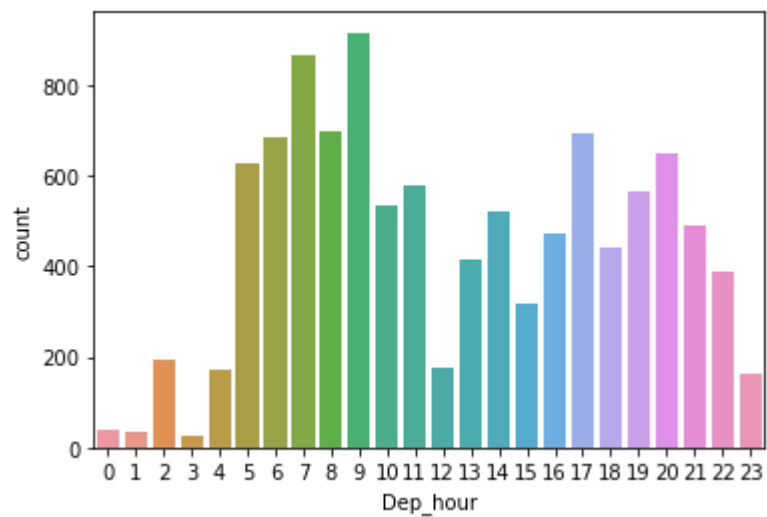
Out[98]:

```
9      916
7      867
8      697
17     695
6      687
20     651
5      629
11     580
19     567
10     536
14     523
21     492
16     472
18     444
13     417
22     387
15     319
2      194
12     178
4      170
23     161
0       40
1       37
3       24
```

Name: Dep_hour, dtype: int64

In [99]:

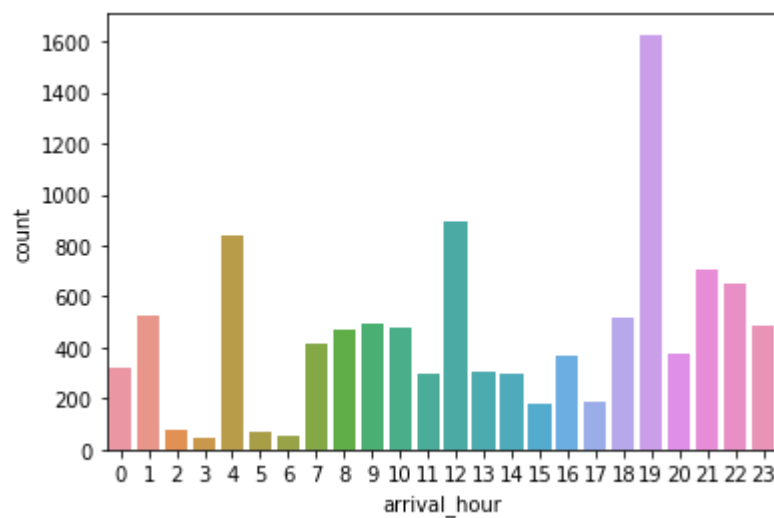
```
sns.countplot(x="Dep_hour", data=df);
```



WHICH ARRIVAL HOUR WAS ON THE PEAK?

In [100]:

```
sns.countplot(x="arrival_hour", data=df);
```



FIND THE PEAK ARRIVAL HOUR FOR DIFFERENT FLIGHTS

In [101]:

```
df.groupby(["Airline"])[ "arrival_hour"].value_counts()
```

Out[101]:

```
Airline  arrival_hour
Air Asia  22          63
          7          52
          13         36
          2          33
          1          30
          ..
Vistara   19          28
          17          23
          0           8
          16           5
          21           3
Name: arrival_hour, Length: 129, dtype: int64
```

FIND THE PEAK ARRIVAL HOUR FOE THE DIFFERENT FLIGHT W.R.T
DESTINATION ALSO

In [102]:

```
df.groupby(["Destination", "Airline"])[ "arrival_hour"].value_counts()
```

Out[102]:

```
Destination  Airline  arrival_hour
Bangalore    Air Asia  22          34
              1          30
              12         30
              23         24
              10         21
              ..
Kolkata      SpiceJet  12          41
              20         41
              Vistara  9          25
              20         18
              10         1
Name: arrival_hour, Length: 265, dtype: int64
```

In [103]:

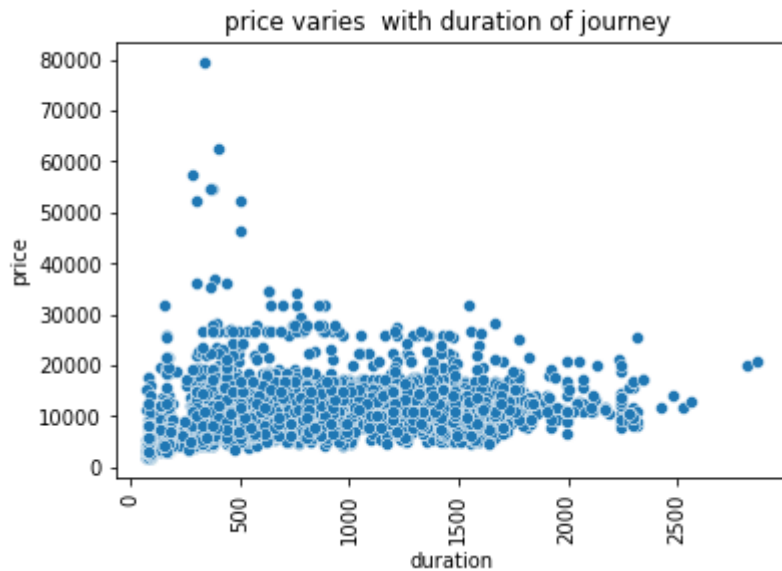
```
df.select_dtypes(["int", "float"]).columns
```

Out[103]:

```
Index(['Duration', 'Total_Stops', 'Price', 'day_of_journey',
      'month_of_journey', 'Dep_hour', 'Dep_min', 'arrival_hour',
      'arrival_min'],
      dtype='object')
```

In [104]:

```
sns.scatterplot(x="Duration",y="Price",data=df)
plt.xticks(rotation=90);
plt.ylabel("price")
plt.xlabel("duration")
plt.title("price varies with duration of journey ");
# which shows it is sort of linear
```



In [105]:

```
df.corr()["Price"]
```

Out[105]:

```
Duration          0.506743
Total_Stops       0.603883
Price             1.000000
day_of_journey    -0.165412
month_of_journey   0.010700
Dep_hour          0.006819
Dep_min          -0.024492
arrival_hour       0.024312
arrival_min       -0.086483
Name: Price, dtype: float64
```

WHICH AIRLINE HAS THE MAXIMUM PROFIT?

In [106]:

```
df.groupby('Airline')['Price'].agg(['max','sum','count']).sort_values(by='max',ascending
# jetairways is the airline which has the expensive flight and has the maximum profits at
```

Out[106]:

	max	sum	count
Airline			
Jet Airways	79512	45167613	3855
Multiple carriers	36983	13188048	1209
Air India	31945	16838841	1752
SpiceJet	23267	3548717	818
GoAir	22794	1137045	194
IndiGo	22153	11648071	2053
Vistara	21730	3761338	482
Air Asia	13774	1783293	319
Trujet	4140	4140	1

In [107]:

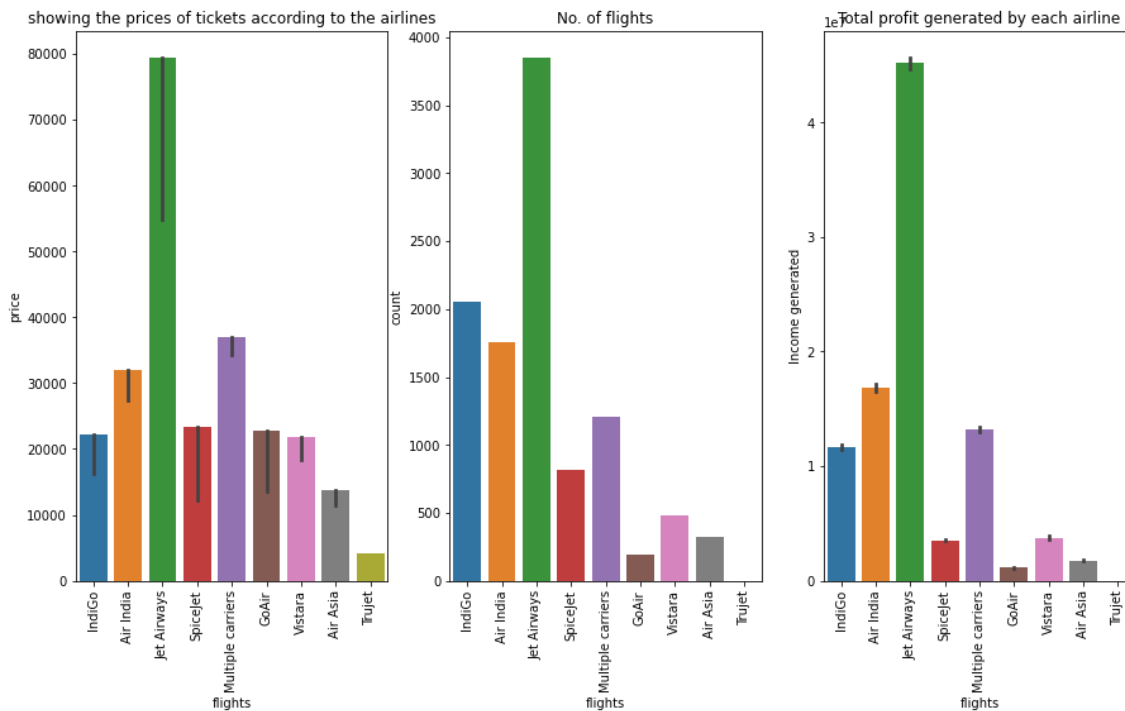
```

plt.figure(figsize=(15,8))
plt.subplot(1,3,1)
sns.barplot(x="Airline",y="Price",data=df,estimator = max)
plt.title("showing the prices of tickets according to the airlines")
plt.xlabel("flights")
plt.ylabel("price")
plt.xticks(rotation=90)

plt.subplot(1,3,2)
sns.countplot(x="Airline",data=df)
plt.title("No. of flights")
plt.xlabel("flights")
plt.ylabel("count")
plt.xticks(rotation=90)

plt.subplot(1,3,3)
sns.barplot(x="Airline",y="Price",data=df,estimator=sum)
plt.title("Total profit generated by each airline")
plt.xlabel("flights")
plt.ylabel("Income generated")
plt.xticks(rotation=90);

```



WHICH MONTH HAS THE MAXIMUM PROFIT W.R.T AIRLINES?

In [108]:

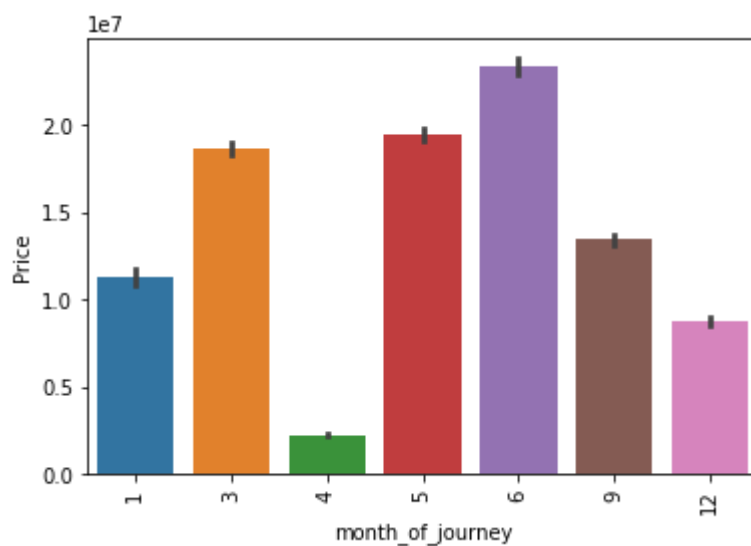
```
df.groupby('month_of_journey')['Price'].sum().sort_values(ascending=False)
```

Out[108]:

```
month_of_journey
6      23369151
5      19414875
3      18647220
9      13429373
1       11279591
12       8719011
4        2217885
Name: Price, dtype: int64
```

In [109]:

```
sns.barplot(x="month_of_journey",y="Price",estimator=sum,data=df);
plt.xticks(rotation=90);
```



In [110]:

```
df.groupby(['month_of_journey', 'Airline'])['Price'].agg(['sum', 'count']).sort_values(by=
```


Out[110]:

		sum	count
month_of_journey	Airline		
6	Jet Airways	11100376	932
5	Jet Airways	9435176	783
3	Jet Airways	7087418	675
9	Jet Airways	6523683	544
1	Jet Airways	5853653	434
12	Jet Airways	4777168	424
6	Multiple carriers	3803704	341
	Air India	3752045	386
5	Air India	3508354	352
3	IndiGo	3174457	513
	Air India	3169391	369
	Multiple carriers	3054577	274
5	Multiple carriers	2922266	292
6	IndiGo	2553771	469
9	Air India	2295491	234
1	Air India	2198523	197
9	Multiple carriers	1809841	160
5	IndiGo	1788186	329
12	Air India	1655980	169
9	IndiGo	1504417	253
1	IndiGo	1123298	184
	Multiple carriers	900258	75
3	SpiceJet	844230	193
6	Vistara	799380	103
4	IndiGo	761519	153
6	SpiceJet	757501	190
12	IndiGo	742423	152
5	Vistara	703455	90
3	Vistara	697134	85
12	Multiple carriers	685216	65
5	SpiceJet	568285	139
1	Vistara	505994	60
9	Vistara	470700	61
	SpiceJet	415805	90
12	Vistara	406085	52
4	Jet Airways	390139	63

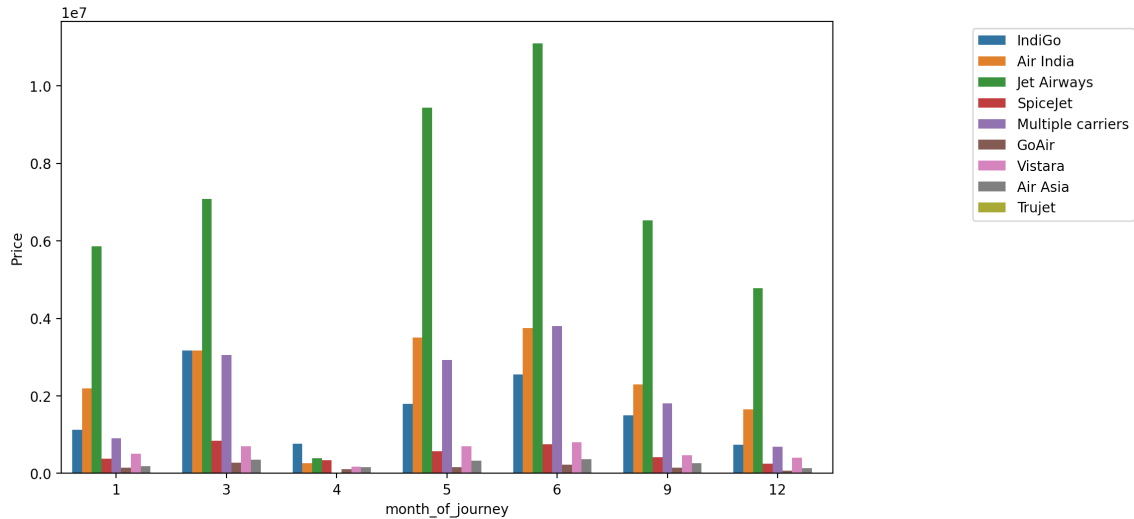
		sum	count
month_of_journey	Airline		
1	SpiceJet	372574	76
6	Air Asia	368888	69
3	Air Asia	349853	59
4	SpiceJet	343718	76
5	Air Asia	323056	61
3	GoAir	270160	43
9	Air Asia	263205	41
4	Air India	259057	45
12	SpiceJet	246604	54
6	GoAir	229346	45
1	Air Asia	179659	28
4	Vistara	178590	31
5	GoAir	166097	28
4	Air Asia	158440	32
9	GoAir	146231	23
1	GoAir	145632	21
12	Air Asia	140192	29
4	GoAir	114236	22
12	GoAir	65343	12
4	Multiple carriers	12186	2

In [111]:

```
4 Multiple carriers 12186 2
6 Trujet 4140 1
plt.figure(figsize=(10,6),dpi=200)
sns.barplot(x='month_of_journey',y = 'Price',hue='Airline',data=df,estimator=sum,ci=None)
plt.legend(bbox_to_anchor =(1.4,1))
```

Out[111]:

<matplotlib.legend.Legend at 0x20070370070>



In [112]:

```
plt.figure(figsize=(10,6),dpi=200)
sns.barplot(x='month_of_journey',y = 'Price',hue='Airline',data=df,ci=None)
plt.legend(bbox_to_anchor =(1.2,1))
# showing mean
```

Out[112]:

<matplotlib.legend.Legend at 0x200703702b0>

