
FLYING WITH DATA:
INSIGHTS FROM AIRLINES
FLIGHT RECORDS

Table of Content:

1)Objective:

The objective of this project is to efficiently extract actionable insights from airline flight data to improve route performance, optimize pricing, and enhance customer experience through data-driven analysis

2)Data Description:

->**Index** – Row ID, not useful for analysis

->**airline**– Airline name (e.g., IndiGo, SpiceJet).

->**flight** – Flight number/code.

->**source_city** – Departure city.

->**departure_time** – Time of departure(Morning, Evening)

->**stops** – Number of stops (direct/connecting)

->**arrival_time** – Time of arrival.

->**destination_city** – Arrival city.

->**class** – Ticket type (Economy/Business).

->**days_left** – Days before departure when booked..

->**duration** – Flight duration (hours).

->**price** – Ticket price (₹)..

```python
#importing the libraries which used for the analysis of this project
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

```python
data=pd.read_csv('airlines_flights_data.csv')
```

```
data
```

	index	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
0	0	SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
1	1	SpiceJet	SG-8157	Delhi	Early_Morning	zero	Morning	Mumbai	Economy	2.33	1	5953
2	2	AirAsia	I5-764	Delhi	Early_Morning	zero	Early_Morning	Mumbai	Economy	2.17	1	5956
3	3	Vistara	UK-995	Delhi	Morning	zero	Afternoon	Mumbai	Economy	2.25	1	5955
4	4	Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955
...	...	...	...	...	...	...	...	...	...	...	...	...
300148	300148	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	69265
300149	300149	Vistara	UK-826	Chennai	Afternoon	one	Night	Hyderabad	Business	10.42	49	77105
300150	300150	Vistara	UK-832	Chennai	Early_Morning	one	Night	Hyderabad	Business	13.83	49	79099
300151	300151	Vistara	UK-828	Chennai	Early_Morning	one	Evening	Hyderabad	Business	10.00	49	81585
300152	300152	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	81585

300153 rows × 12 columns

Basic Queries to Analyze

```
data.shape   #---getting how many rows and columns
```

```
(300153, 11)
```

```
data.info()   #---fetching the information of the dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300153 entries, 0 to 300152
Data columns (total 11 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   airline           300153 non-null  object 
 1   flight            300153 non-null  object 
 2   source_city       300153 non-null  object 
 3   departure_time    300153 non-null  object 
 4   stops             300153 non-null  object 
 5   arrival_time      300153 non-null  object 
 6   destination_city  300153 non-null  object 
 7   class             300153 non-null  object 
 8   duration          300153 non-null  float64
 9   days_left         300153 non-null  int64  
 10  price             300153 non-null  int64  
dtypes: float64(1), int64(2), object(8)
memory usage: 25.2+ MB
```

```
data.columns       #---look for the columns
```

```
Index(['airline', 'flight', 'source_city', 'departure_time', 'stops',
       'arrival_time', 'destination_city', 'class', 'duration', 'days_left',
       'price'],
      dtype='object')
```

```
data.describe()    #---describing about the dataset
```

	duration	days_left	price
count	300153.000000	300153.000000	300153.000000
mean	12.221021	26.004751	20889.660523
std	7.191997	13.561004	22697.767366
min	0.830000	1.000000	1105.000000
25%	6.830000	15.000000	4783.000000
50%	11.250000	26.000000	7425.000000
75%	16.170000	38.000000	42521.000000
max	49.830000	49.000000	123071.000000

Basic Queries to Analyze

```
data.isnull().sum()    #----Check for the null values
```

```
airline           0
flight            0
source_city       0
departure_time    0
stops             0
arrival_time      0
destination_city  0
class             0
duration          0
days_left         0
price             0
dtype: int64
```

```
data.head()
```

	index	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
0	0	SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
1	1	SpiceJet	SG-8157	Delhi	Early_Morning	zero	Morning	Mumbai	Economy	2.33	1	5953
2	2	AirAsia	I5-764	Delhi	Early_Morning	zero	Early_Morning	Mumbai	Economy	2.17	1	5956
3	3	Vistara	UK-995	Delhi	Morning	zero	Afternoon	Mumbai	Economy	2.25	1	5955
4	4	Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955

```
# checking how many airlines are in the dataset

data['airline'].nunique()
```

```
6
```

```
# showing the names of the airlines in the dataset

data['airline'].unique()
```

```
array(['SpiceJet', 'AirAsia', 'Vistara', 'GO_FIRST', 'Indigo',
       'Air_India'], dtype=object)
```

```
# no.of.counts in each airline_name

data['airline'].value_counts()
```

```
airline
Vistara      127859
Air_India     80892
Indigo        43120
GO_FIRST      23173
AirAsia       16098
SpiceJet       9011
Name: count, dtype: int64
```

```
data[data['duration'] == 49.830000]  #---flights having highest travelling duration
```

	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
193889	Air_India	AI-672	Chennai	Evening	two_or_more	Evening	Bangalore	Economy	49.83	2	23891
194359	Air_India	AI-672	Chennai	Evening	one	Evening	Bangalore	Economy	49.83	9	17538

```
data[data['price'] == 123071.000000]   #---flights having highest ticket price
```

	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
261377	Vistara	UK-772	Kolkata	Morning	one	Night	Delhi	Business	13.5	3	123071

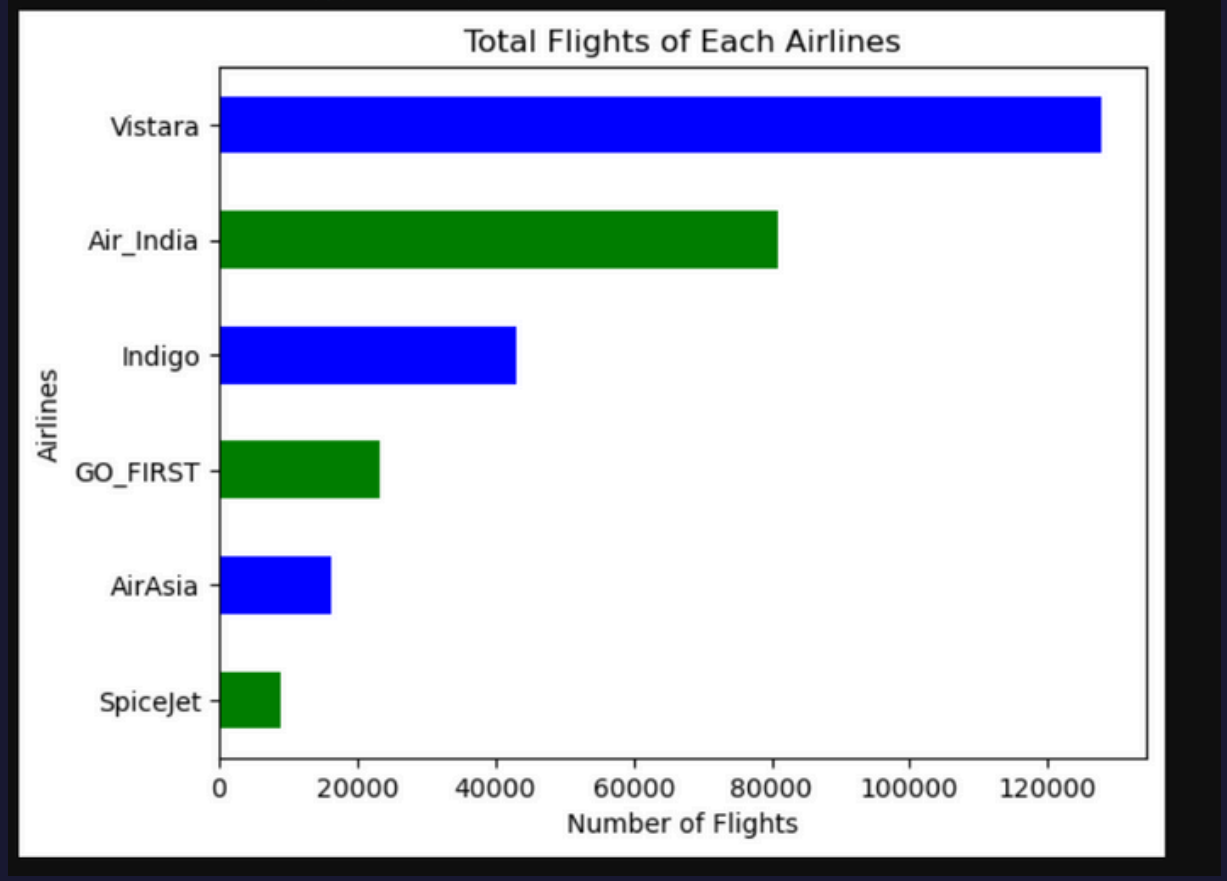
```
data[data['price'] == 1105.000000]   #---flights having lowest ticket price
```

	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
203807	AirAsia	I5-517	Chennai	Morning	zero	Morning	Hyderabad	Economy	1.17	16	1105
203808	GO_FIRST	G8-505	Chennai	Evening	zero	Evening	Hyderabad	Economy	1.25	16	1105
203908	AirAsia	I5-517	Chennai	Morning	zero	Morning	Hyderabad	Economy	1.17	17	1105
203909	GO_FIRST	G8-505	Chennai	Evening	zero	Evening	Hyderabad	Economy	1.25	17	1105
204003	AirAsia	I5-517	Chennai	Morning	zero	Morning	Hyderabad	Economy	1.17	18	1105
...	...	...	...	...	...	...	...	...	...	...	...
206601	Indigo	6E-7261	Chennai	Morning	one	Evening	Hyderabad	Economy	7.92	49	1105
206602	Indigo	6E-611	Chennai	Evening	one	Late_Night	Hyderabad	Economy	8.25	49	1105
206603	Indigo	6E-581	Chennai	Morning	one	Evening	Hyderabad	Economy	9.17	49	1105
206604	Indigo	6E-7127	Chennai	Afternoon	one	Night	Hyderabad	Economy	9.50	49	1105
206605	Indigo	6E-7261	Chennai	Morning	one	Night	Hyderabad	Economy	10.08	49	1105

3)Basic Queries with Visualization Charts

```
# 1)no.of.counts in each airline_name via bar graph

data['airline'].value_counts(ascending=True).plot.barh(color=['green','blue'])
plt.title("Total Flights of Each Airlines")
plt.xlabel('Number of Flights')
plt.ylabel('Airlines')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



->**Vistara Airlines** have more no.of.flights

->**SpiceJet Airlines** have less no.of.flights

Finding the max & min count of flights -based on time

```
# showing the no.of.flights according to departure time

data['departure_time'].value_counts()
```

```
departure_time
Morning          71146
Early_Morning    66790
Evening          65102
Night            48015
Afternoon        47794
Late_Night        1306
Name: count, dtype: int64
```

```
# showing the no.of.flights according to arrival time

data['arrival_time'].value_counts()
```

```
arrival_time
Night            91538
Evening          78323
Morning          62735
Afternoon        38139
Early_Morning    15417
Late_Night       14001
Name: count, dtype: int64
```

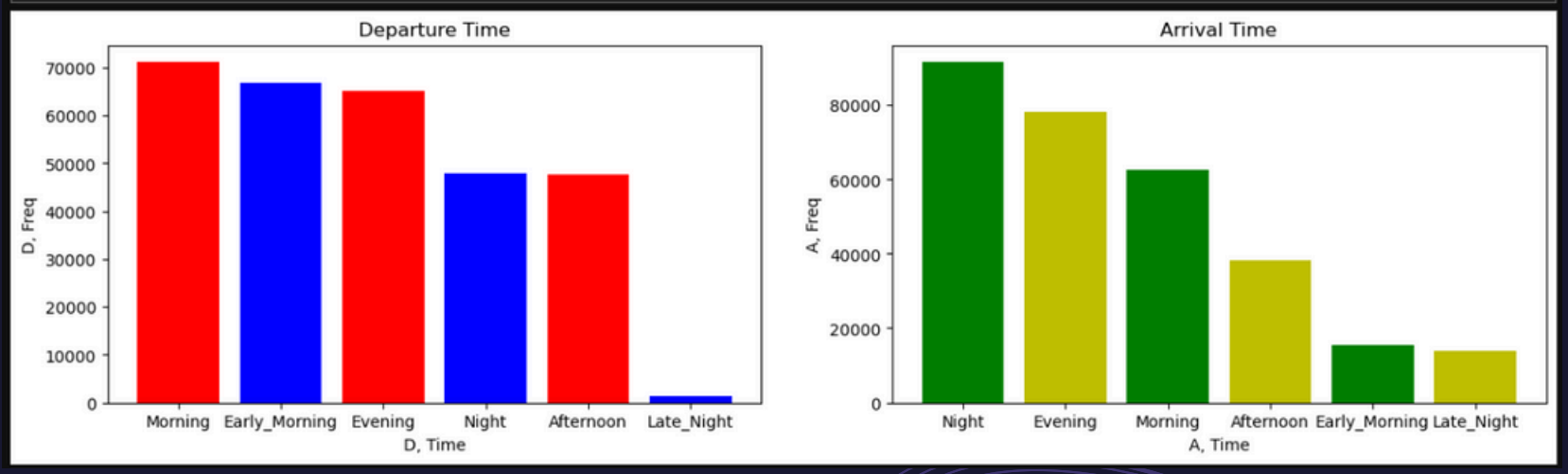
In morning, more no.of.flights are departing

In late night, more no.of.flights are departing

In night, more no.of.flights are arriving

In late night, less no.of.flights are arriving

```
# showing the departime time & arrival time for the flights with their counts

plt.figure(figsize=(16,4))

plt.subplot(1,2,1)

plt.bar(data['departure_time'].value_counts().index , data['departure_time'].value_counts().values,color=['r','b'])

plt.title('Departure Time')
plt.xlabel('D, Time')
plt.ylabel('D, Freq')
plt.subplot(1,2,2)
plt.bar(data['arrival_time'].value_counts().index, data['arrival_time'].value_counts().values,color=['g','y'])
plt.title('Arrival Time')
plt.xlabel('A, Time')
plt.ylabel('A, Freq')
plt.show()
```



Finding which city has max & min flights as departing and arriving

```
# showing the no.of.flights of each source city

data['source_city'].value_counts()
```

```
source_city
Delhi        61343
Mumbai       60896
Bangalore    52061
Kolkata      46347
Hyderabad    40806
Chennai      38700
Name: count, dtype: int64
```

```
# showing the no.of.flights of each destination city

data['destination_city'].value_counts()
```

```
destination_city
Mumbai       59097
Delhi        57360
Bangalore    51068
Kolkata      49534
Hyderabad    42726
Chennai      40368
Name: count, dtype: int64
```

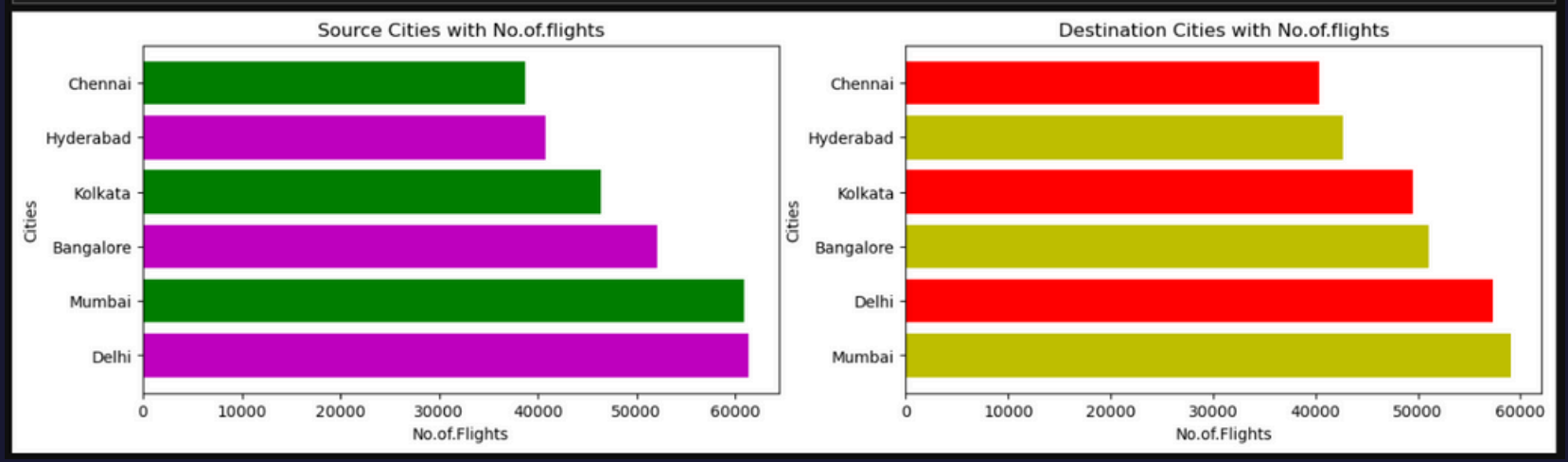
Delhi have more no.of.flights as departing

Chennai have less no.of.flights as departing

Mumbai have more no.of.flights as arriving

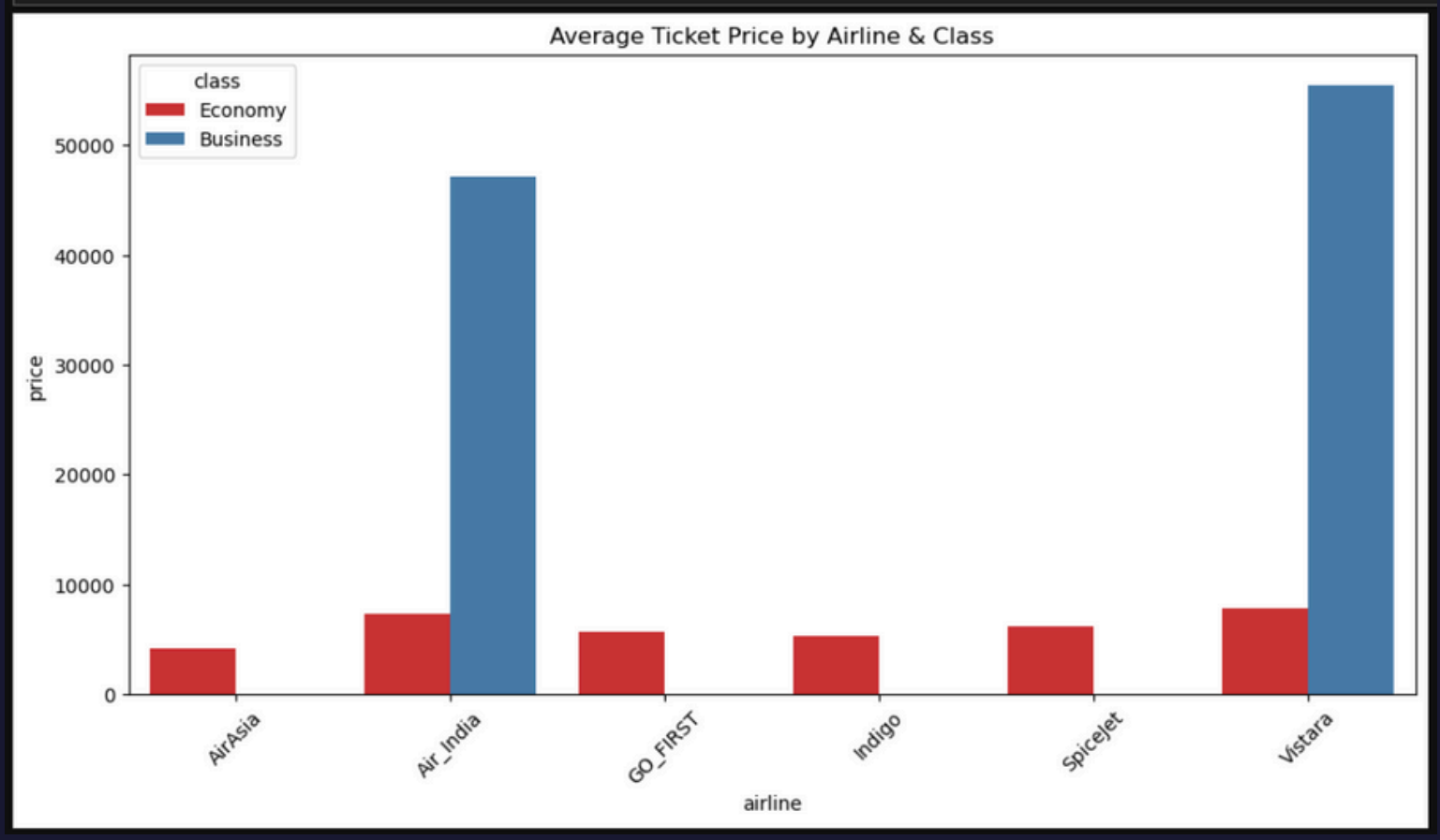
Chennai have less no.of.flights as arriving

```
# showing the source and destination city for the flights with their counts

plt.figure(figsize=(16,4))

plt.subplot(1,2,1)

plt.barh(data['source_city'].value_counts().index,data['source_city'].value_counts().values,color=['m','g'])
plt.title('Source Cities with No.of.flights')
plt.ylabel('Cities')
plt.xlabel('No.of.Flights')

plt.subplot(1,2,2)

plt.barh(data['destination_city'].value_counts().index,data['destination_city'].value_counts().values,color=['y','r'])
plt.title('Destination Cities with No.of.flights')
plt.ylabel('Cities')
plt.xlabel('No.of.Flights')
plt.show()
```



Average Ticket Price by Airline & Class

```
avg_airline_class = data.groupby(["airline","class"])["price"].mean().reset_index()

plt.figure(figsize=(12,6))
sns.barplot(data=avg_airline_class, x="airline", y="price", hue="class", palette="Set1")
plt.title("Average Ticket Price by Airline & Class")
plt.xticks(rotation=45)
plt.show()
```



->Vistara airline costs more in both Economy & Business class

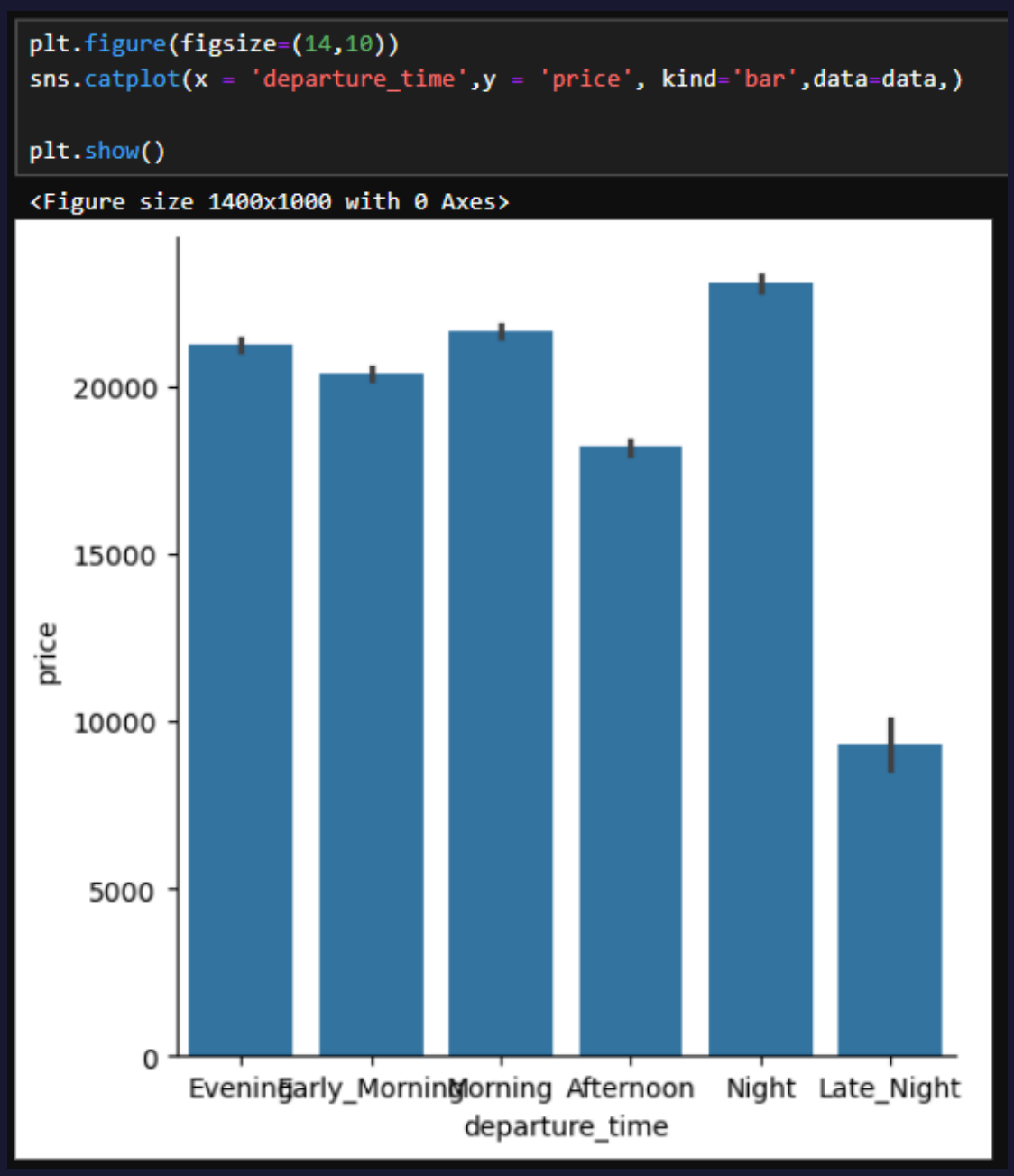
->AirAsia airline costs less in Economy class

->Also conclude that only **Air_India** & **Vistara** airlines providing both economy & business class

Checking the mean ticket price based on the departure time

```
plt.figure(figsize=(14,10))
sns.catplot(x = 'departure_time',y = 'price', kind='bar',data=data,)

plt.show()
```

```
<Figure size 1400x1000 with 0 Axes>
```



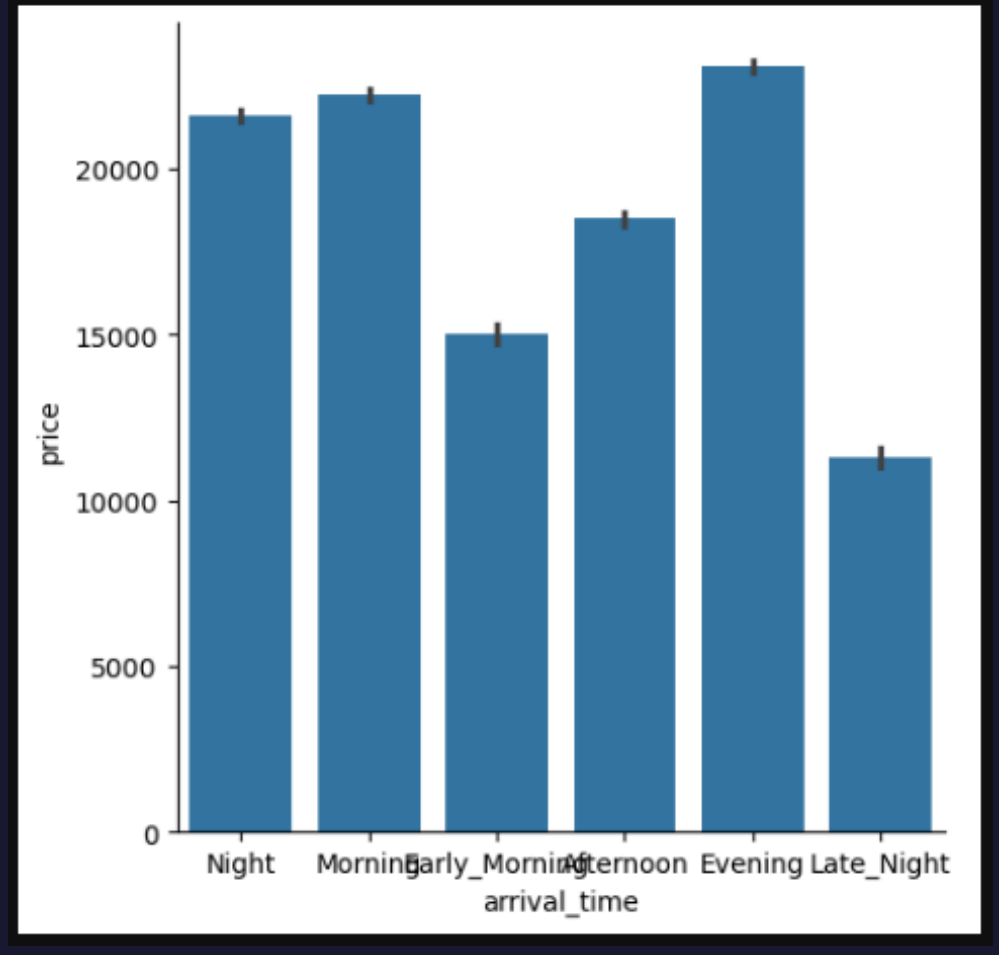
->In Night , ticket price costs more

->In Late Night , ticket price costs less

Checking the mean ticket price based on the arrival time

```
plt.figure(figsize=(12,4))
sns.catplot(x = 'arrival_time',y = 'price', kind='bar',data=data)

plt.show()
```

```
<Figure size 1200x400 with 0 Axes>
```

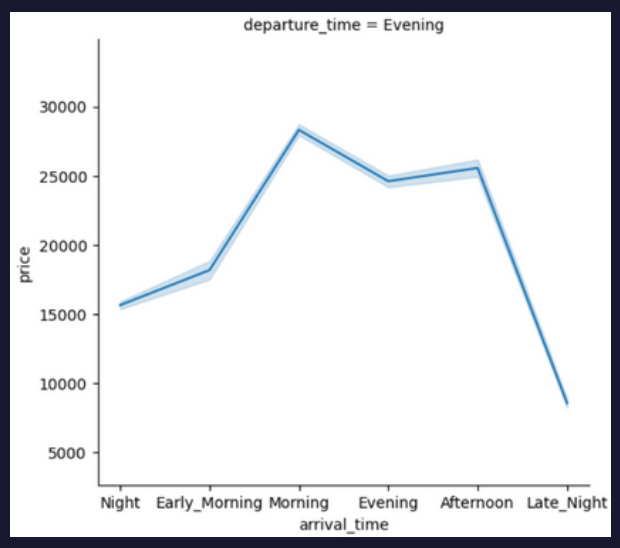


->In Evening , ticket price costs more

->In Late Night , ticket price costs less

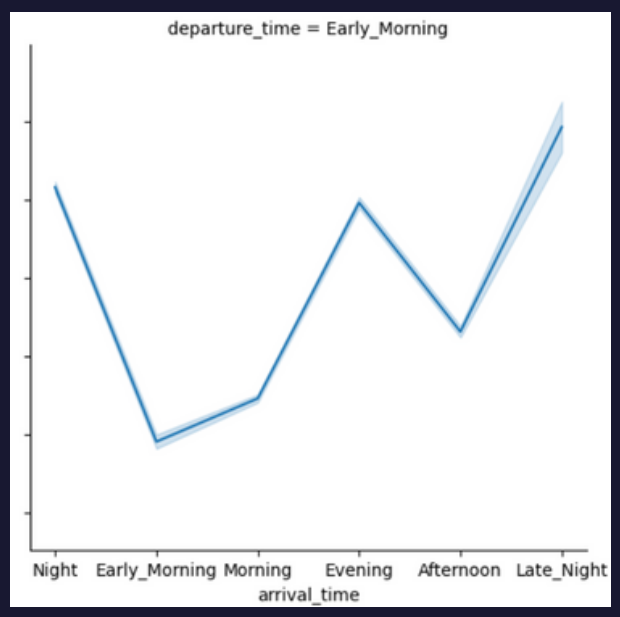
Comparing the ticket price based on the arrival & departure time

```
sns.relplot( x='arrival_time', y='price',data=data,col='departure_time',kind='line')
plt.show()
```



->Flight departing from evening and arriving morning costs more

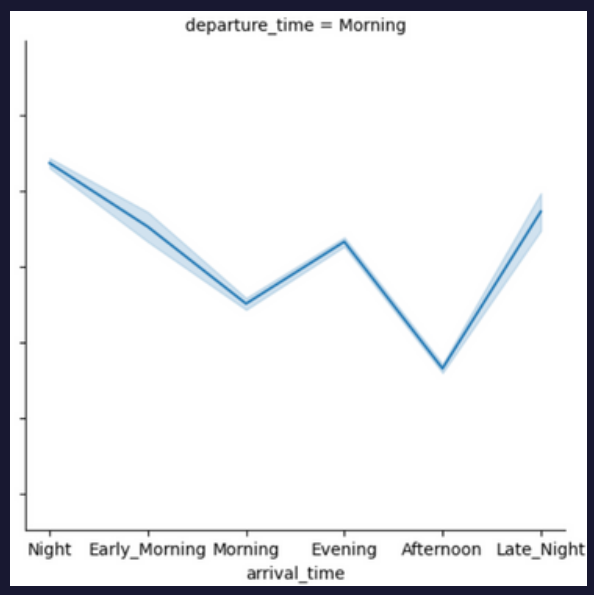
->Flight departing from evening and arriving late_night costs less



->Flight departing from early_morning and arriving late_night costs more

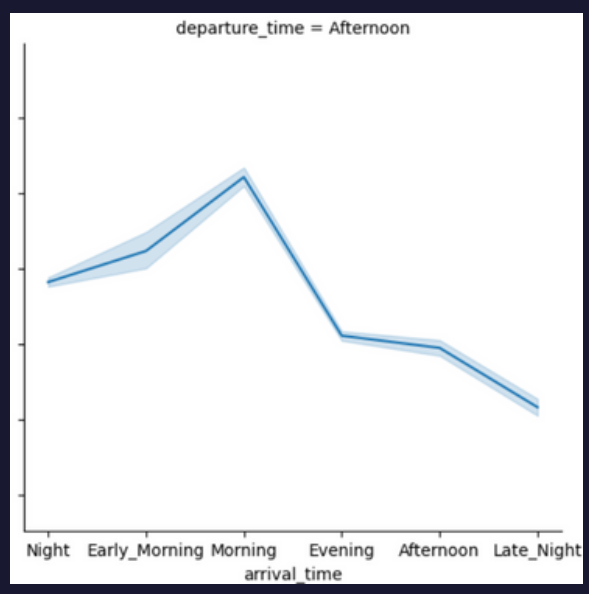
->Flight departing from early_morning and arriving early_morning costs less

Comparing the ticket price based on the arrival & departure time



->Flight departing from morning and arriving night costs more

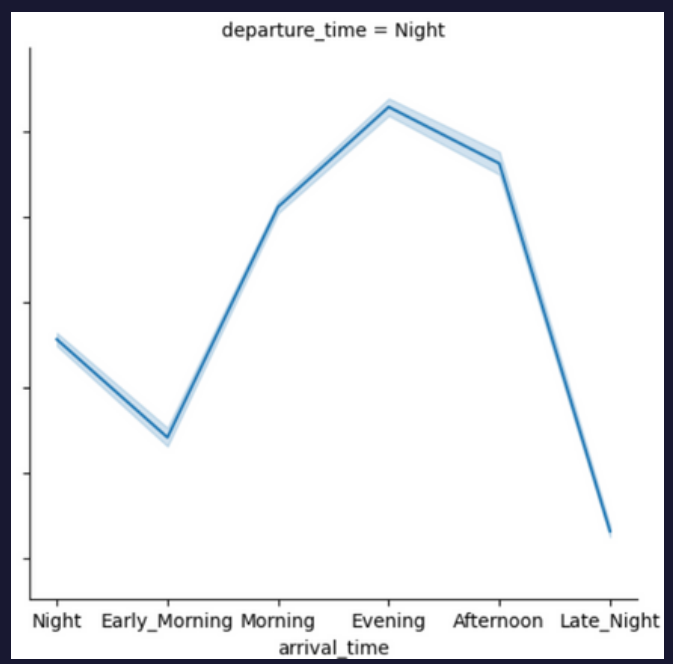
->Flight departing from morning and arriving afternoon costs less



->Flight departing from afternoon and arriving morning costs more

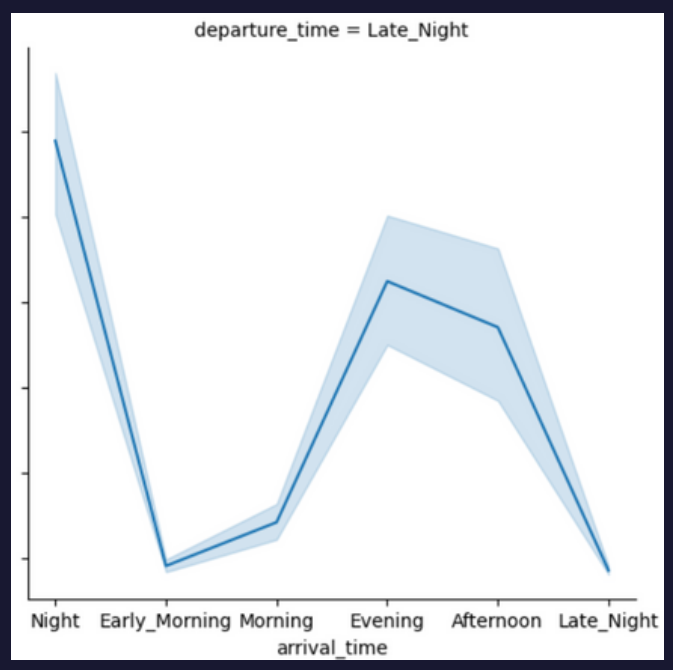
->Flight departing from afternoon and arriving late_night costs less

Comparing the ticket price based on the arrival & departure time



->Flight departing from night and arriving evening costs more

->Flight departing from night and arriving late_night costs less

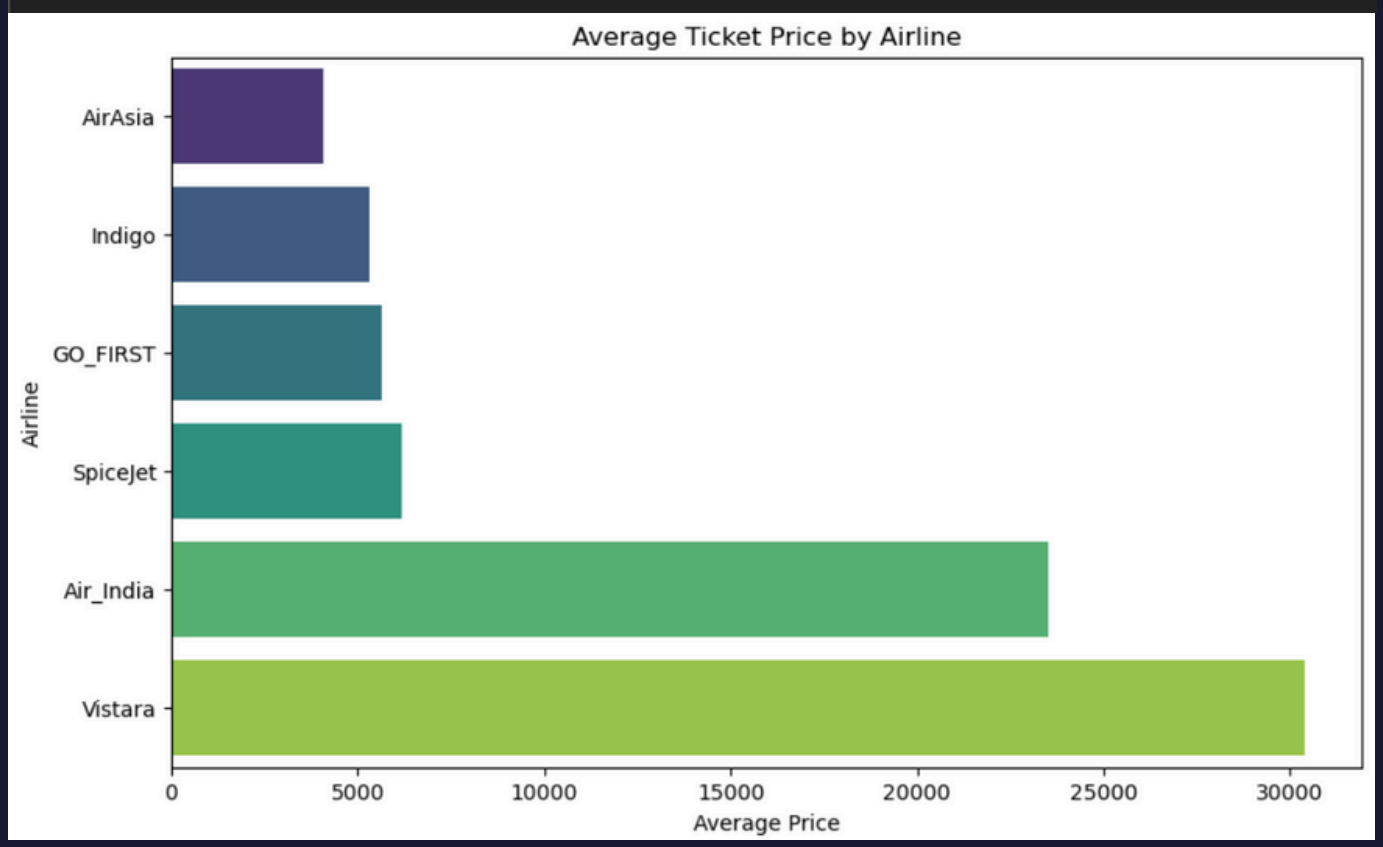


->Flight departing from late_night and arriving morning costs more

->Flight departing from late_night and arriving late_night costs less

Average Ticket Price of Each Airlines

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

airline_price = data.groupby("airline")["price"].mean().sort_values()
plt.figure(figsize=(10,6))
sns.barplot(x=airline_price.values, y=airline_price.index, palette="viridis")
plt.title("Average Ticket Price by Airline")
plt.xlabel("Average Price")
plt.ylabel("Airline")
plt.show()
```

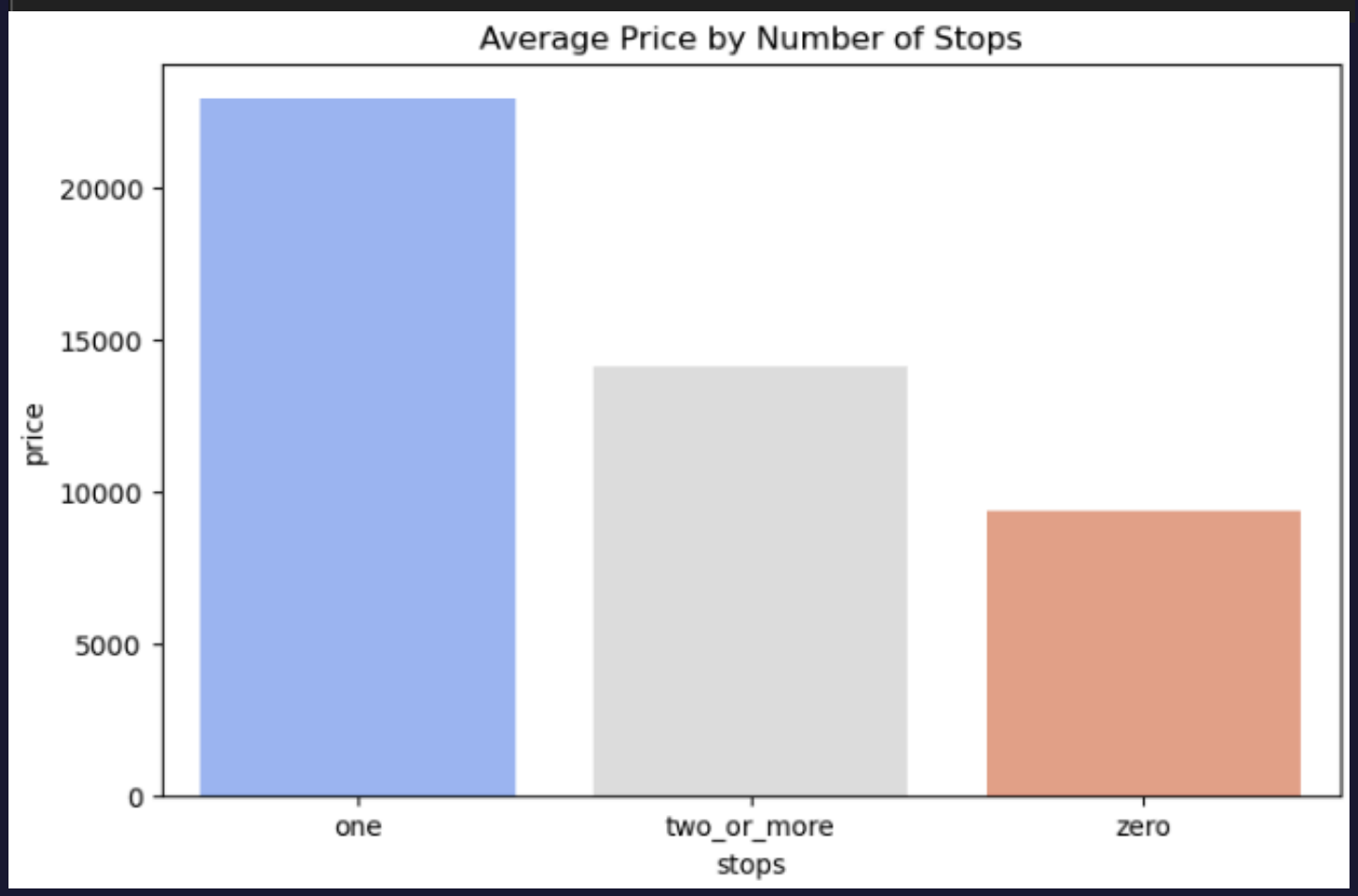


->Vistara Airline costs more Avg.Ticket price

->AirAsia Airline costs less Avg.Ticket price

Average Ticket Price by No.of.Stops

```
stops_price = data.groupby("stops")["price"].mean().reset_index()
plt.figure(figsize=(8,5))
sns.barplot(data=stops_price, x="stops", y="price", palette="coolwarm")
plt.title("Average Price by Number of Stops")
plt.show()
```

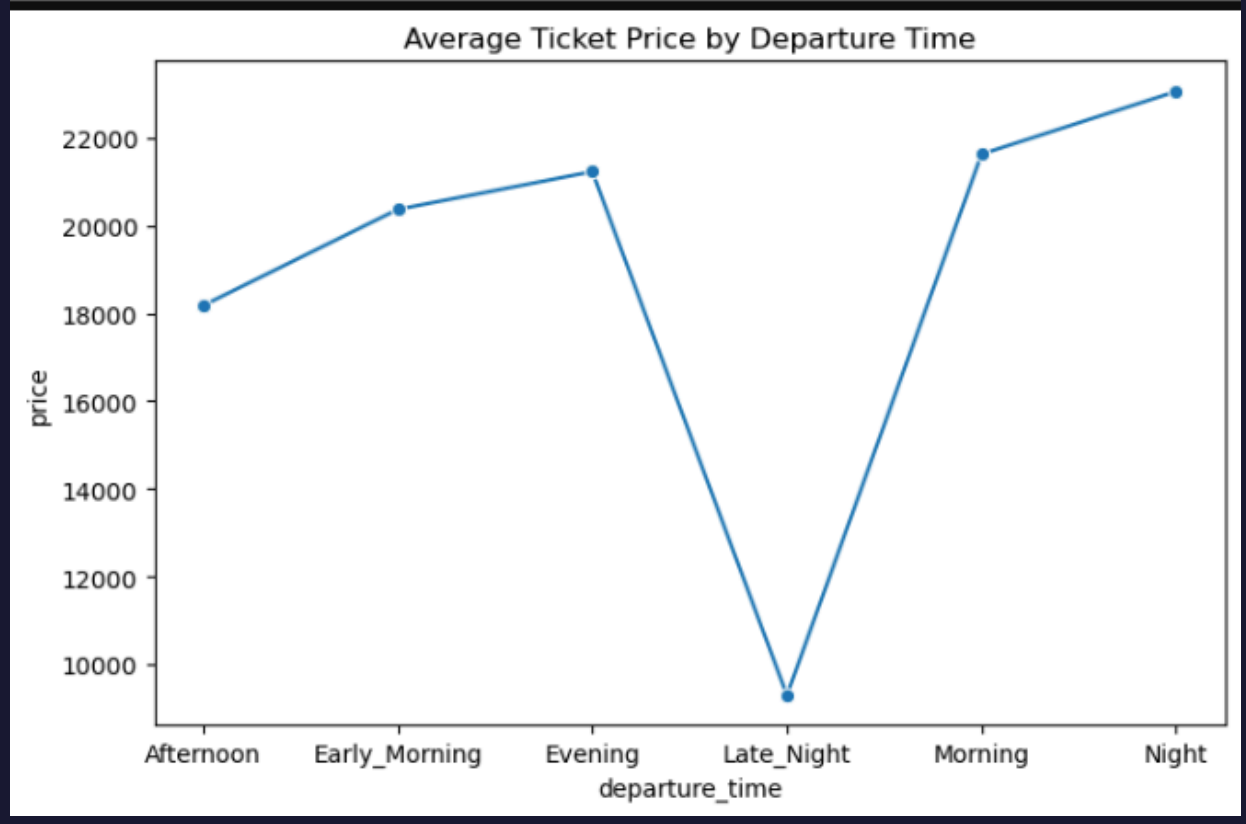


->Flights having one stop costs more ticket price

->Flights having zero stop costs less Ticket price

Average Ticket Price by Departure_time

```
time_price = data.groupby("departure_time")["price"].mean().reset_index()
plt.figure(figsize=(8,5))
sns.lineplot(data=time_price, x="departure_time", y="price", marker="o")
plt.title("Average Ticket Price by Departure Time")
plt.show()
```

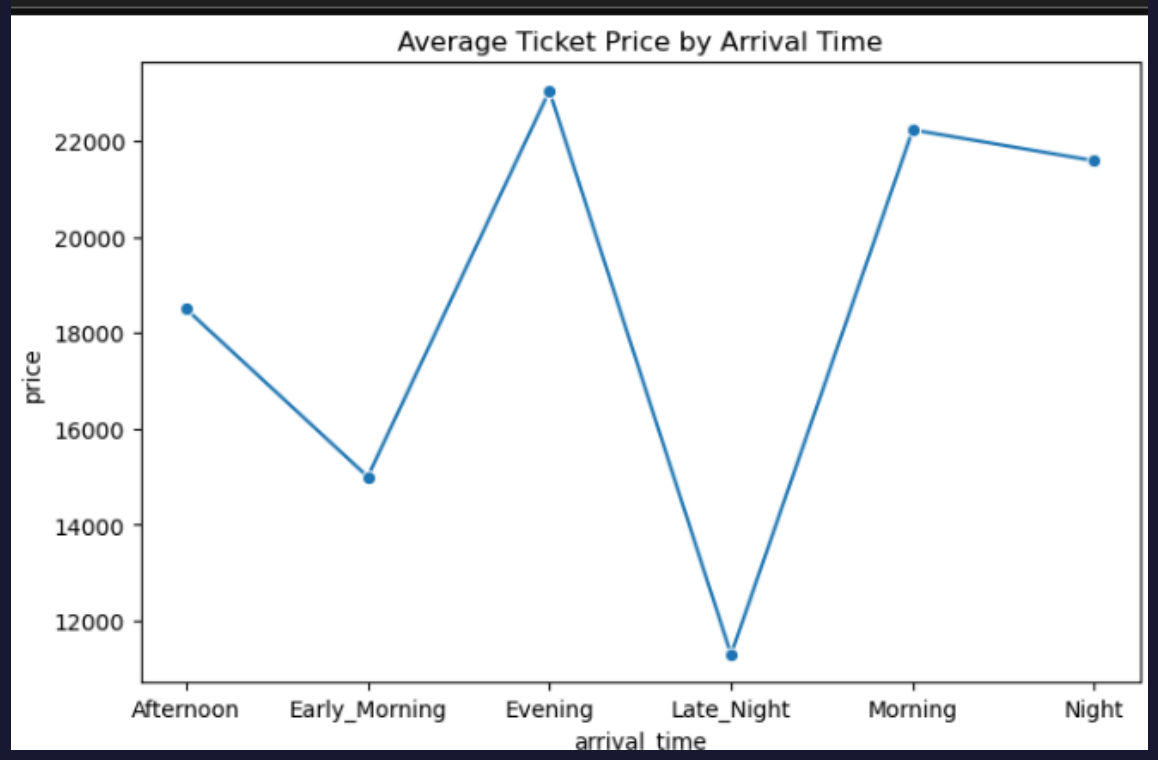


->Flights departing from night costs more ticket price

->Flights departing from late_night costs less ticket price

Average Ticket Price by Arrival_time

```
time_price = data.groupby("arrival_time")["price"].mean().reset_index()
plt.figure(figsize=(8,5))
sns.lineplot(data=time_price, x="arrival_time", y="price", marker="o")
plt.title("Average Ticket Price by Arrival Time")
plt.show()
```

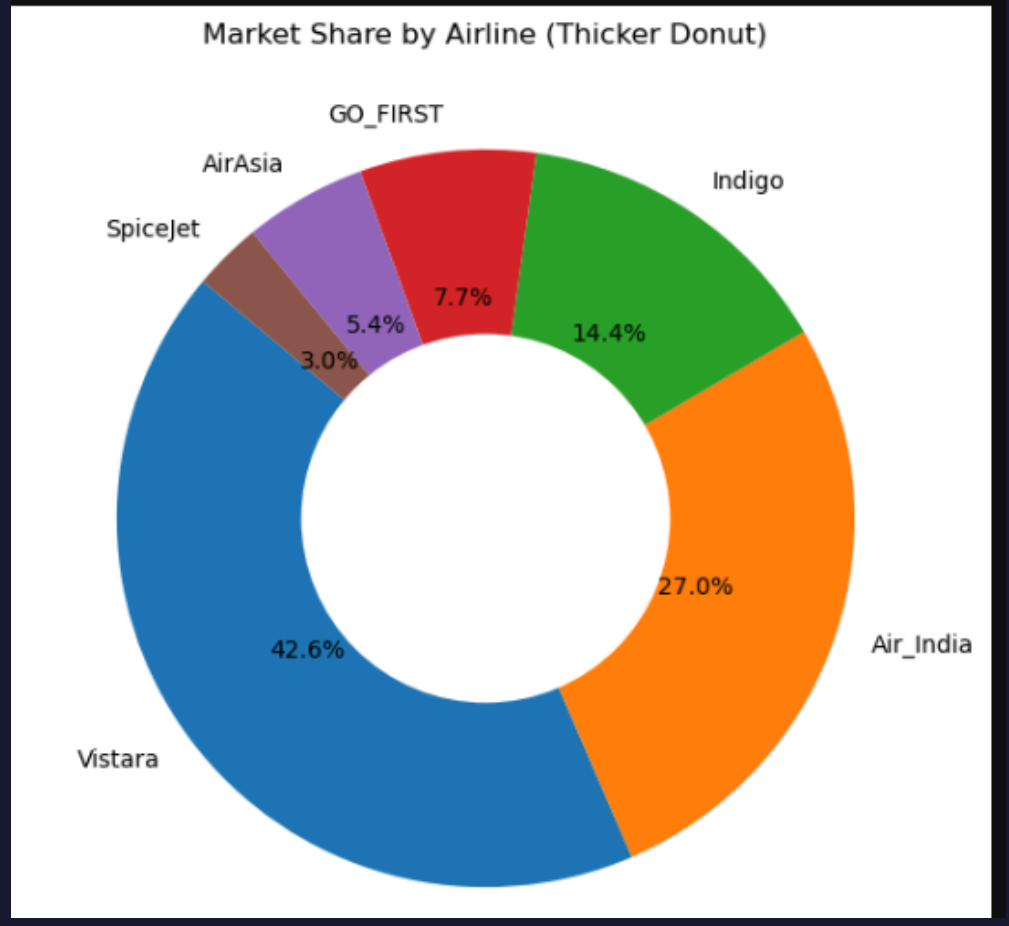


->Flights that arriving evening costs more ticket price

->Flights that arriving late_night costs less ticket price

Market Share by Each Airline

```
airline_share = data["airline"].value_counts()

plt.figure(figsize=(7,7))
# Pie chart
wedges, texts, autotexts = plt.pie(
    airline_share, labels=airline_share.index, autopct="%1.1f%%", startangle=140
)

# Make inner circle smaller (thicker donut)
centre_circle = plt.Circle((0,0), 0.50, fc="white")  # 0.70 → 0.50
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

plt.title("Market Share by Airline (Thicker Donut)")
plt.show()
```

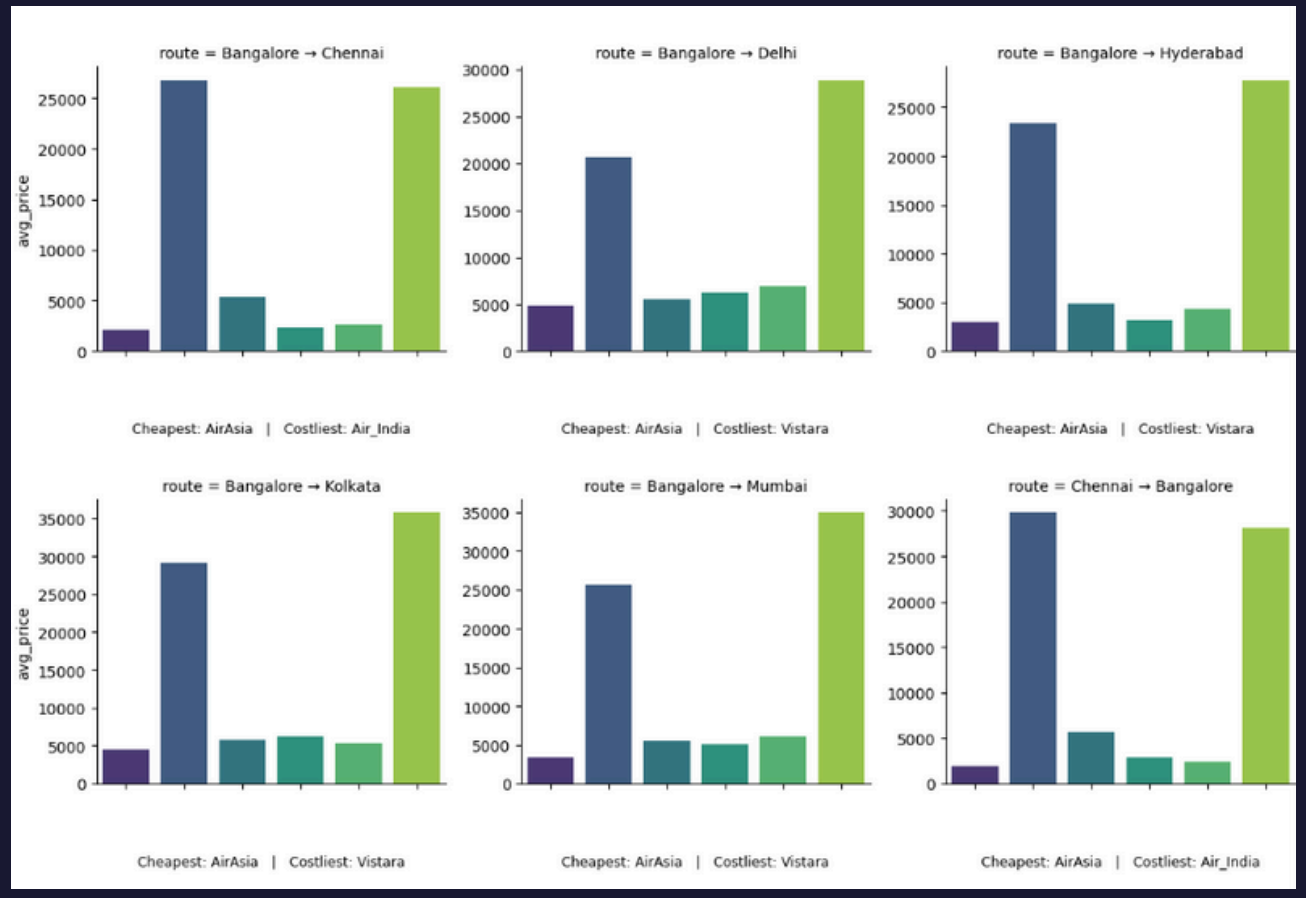


->Vistara Airline hold more market share (42.6%) among all airline

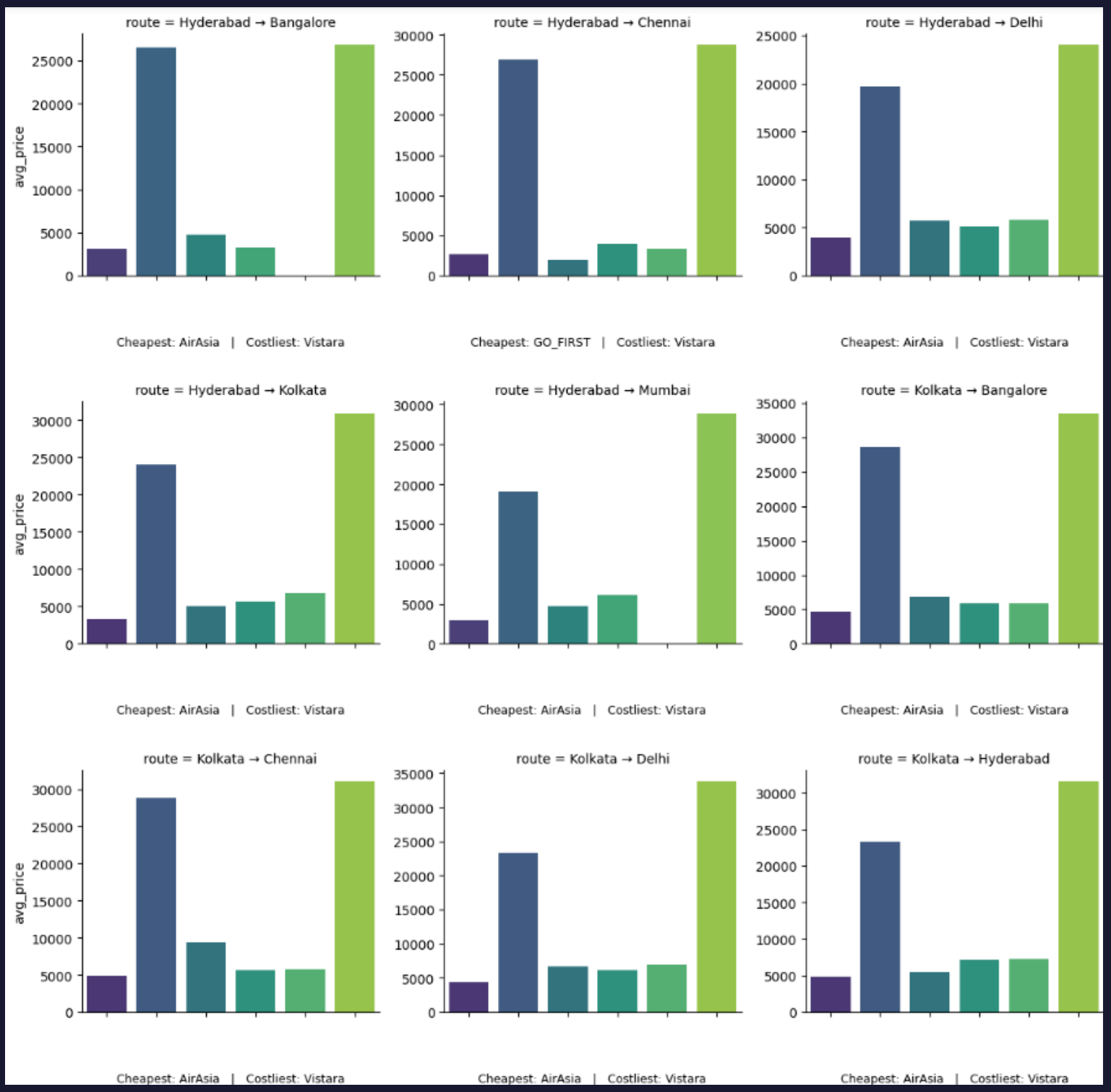
->SpiceJet Airline hold less market share (3.0%) among all airline

Costliest and Cheapest Ticket Price for different Cities

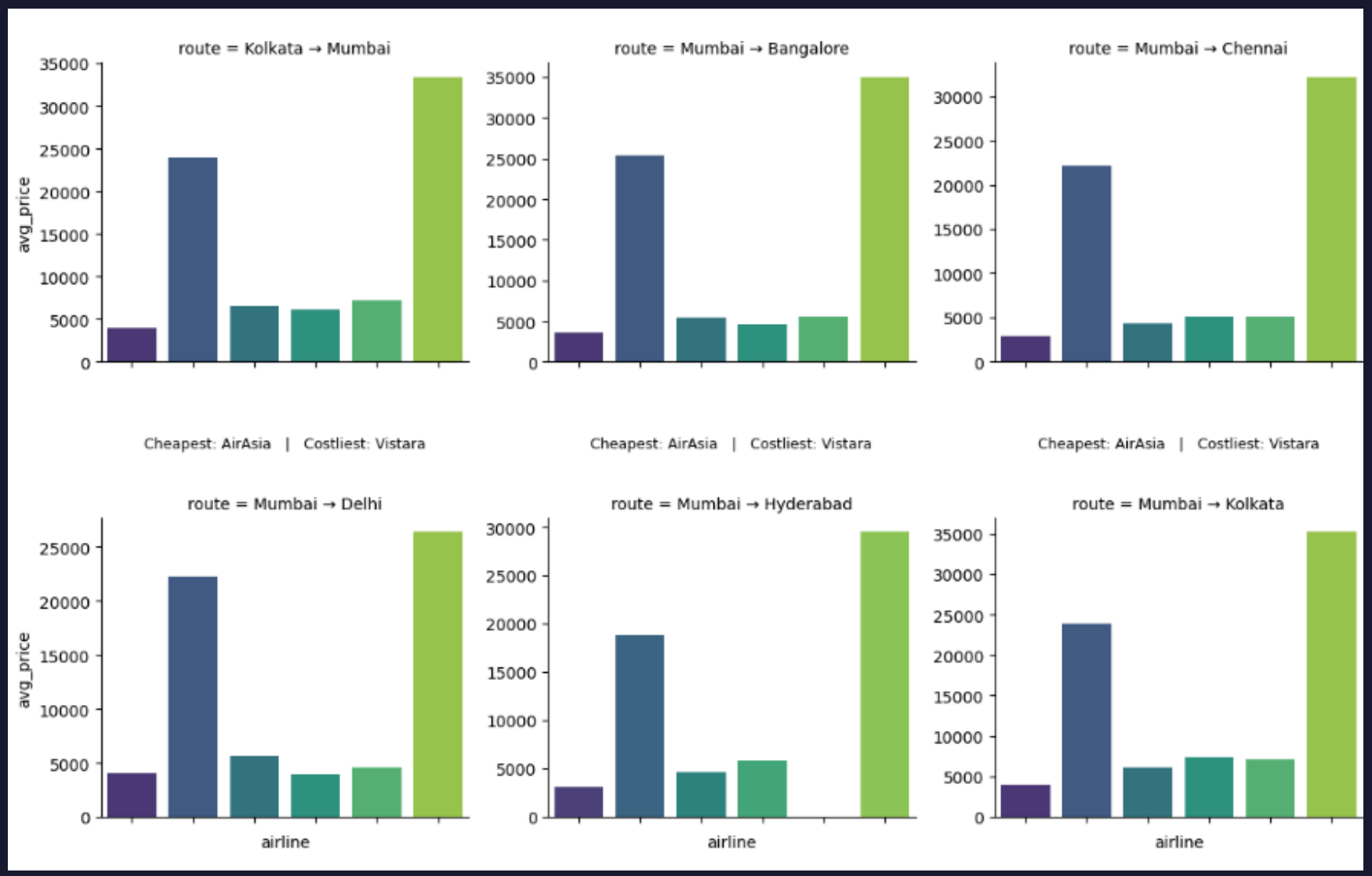
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Create route column
route_prices["route"] = route_prices["source_city"] + " → " + route_prices["destination_city"]

# Step 2: Plot with FacetGrid
g = sns.FacetGrid(route_prices, col="route", col_wrap=3, height=4, sharey=False)
g.map_dataframe(sns.barplot, x="airline", y="avg_price", palette="viridis")

# Step 3: Add labels BELOW each chart (outside bar area)
for ax, route in zip(g.axes.flatten(), route_prices["route"].unique()):
    data = route_prices[route_prices["route"] == route]
    cheapest = data["cheapest_airline"].iloc[0]
    costliest = data["costliest_airline"].iloc[0]

    # Position below x-axis
    ax.text(
        0.5, -0.25,  # x=center, y=below chart
        f"Cheapest: {cheapest}   |   Costliest: {costliest}",
        ha="center", va="top", fontsize=9, color="black",
        transform=ax.transAxes
    )

# Final touches
for ax in g.axes.flatten():
    ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha="right")

g.fig.suptitle("Average Ticket Price Comparison by Airline for Each Route", y=1.08)
plt.tight_layout()
plt.show()
```



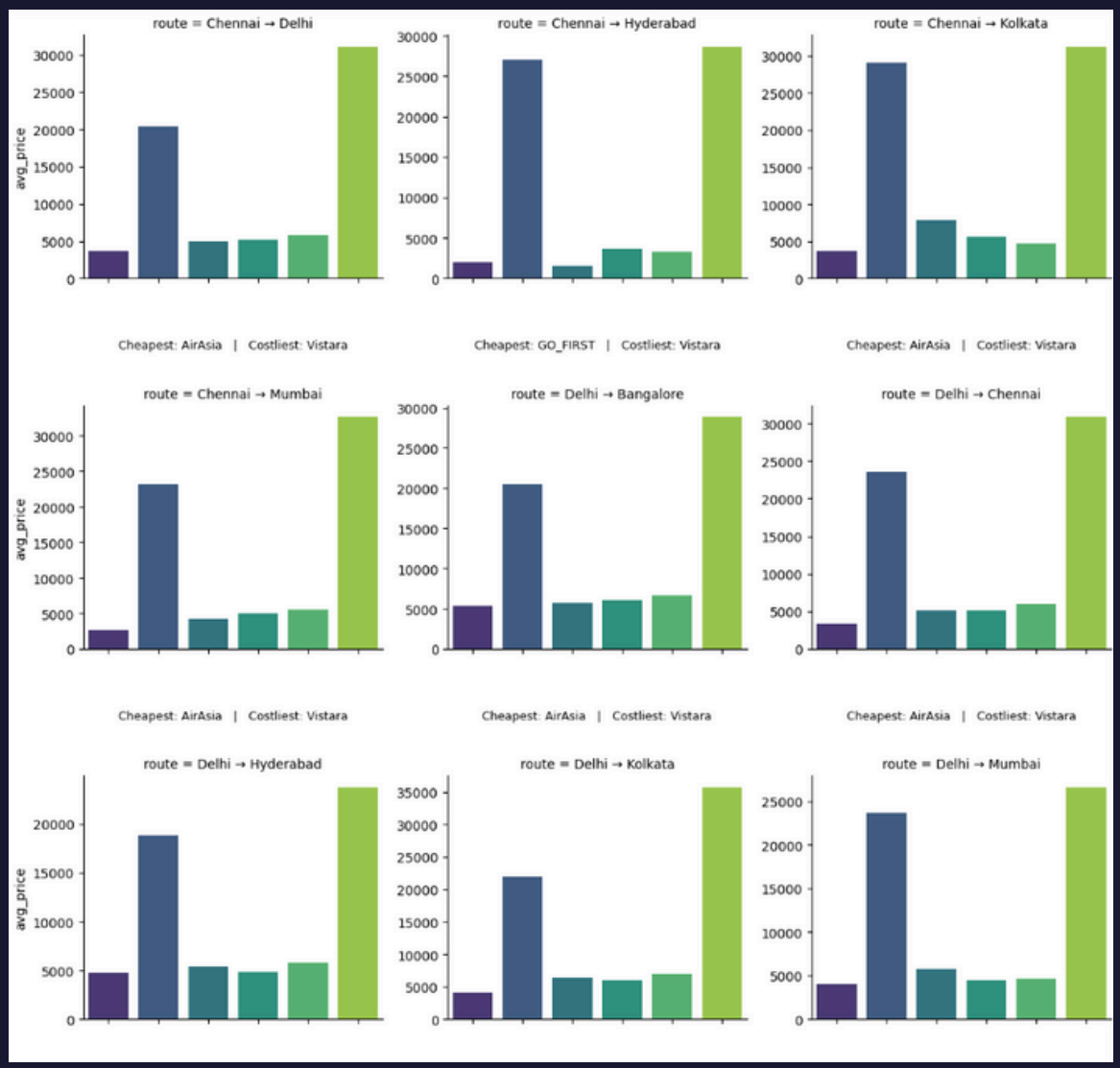
Costliest and Cheapest Ticket Price for different Cities



Costliest and Cheapest Ticket Price for different Cities



Costliest and Cheapest Ticket Price for different Cities



4)Conclusions:

> Route-wise comparison revealed clear pricing differences across airlines for the same destinations.

> Visualizations like bar charts made it easy to spot cheapest and costliest carriers per route.

> Insights help travelers choose budget-friendly options and airlines assess competitive pricing.

> This analysis supports smarter travel decisions and strategic planning for airline pricing models.

Thank you