flight-price-predicition

April 3, 2024

1 Flight Price Prediction

In this notebook, we will consider the problem of modelling flight price predicition based on the data from Kaggle website.

1.1 Import Necessary Libraries

First, we need to import the libraries that will be used throughout this notebook.

[ ]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import sklearn as skl
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error,mean_absolute_error, r2_score
from sklearn.preprocessing import PolynomialFeatures
from sklearn.feature_selection import mutual_info_regression
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_predict
from sklearn.model_selection import cross_validate
```

1.2 Load Datasets

Now, let's load the datasets that we will be using for our analysis.

[ ]:

```
# Load datasets
business_df = pd.read_csv('../datasets/business.csv')
economy_df = pd.read_csv('../datasets/economy.csv')
clean_dataset = pd.read_csv('../datasets/Clean_Dataset.csv')
business_df.head()
economy_df.head()
clean_dataset.head()
```

```
[ ]:    Unnamed: 0   airline   flight source_city departure_time stops  \
     0           0  SpiceJet  SG-8709       Delhi        Evening  zero
     1           1  SpiceJet  SG-8157       Delhi  Early_Morning  zero
     2           2   AirAsia   I5-764       Delhi  Early_Morning  zero
     3           3   Vistara   UK-995       Delhi        Morning  zero
     4           4   Vistara   UK-963       Delhi        Morning  zero

         arrival_time destination_city    class  duration  days_left  price
     0          Night           Mumbai  Economy      2.17          1   5953
     1        Morning           Mumbai  Economy      2.33          1   5953
     2  Early_Morning           Mumbai  Economy      2.17          1   5956
     3      Afternoon           Mumbai  Economy      2.25          1   5955
     4        Morning           Mumbai  Economy      2.33          1   5955
```

```
[ ]: clean_dataset.shape
     clean_dataset.describe(include='all')
```

```
[ ]:            Unnamed: 0  airline  flight source_city departure_time   stops  \
     count   300153.000000   300153  300153      300153         300153  300153
     unique            NaN        6    1561           6              6       3
     top               NaN  Vistara  UK-706       Delhi        Morning     one
     freq              NaN   127859    3235       61343          71146  250863
     mean    150076.000000      NaN     NaN         NaN            NaN     NaN
     std      86646.852011      NaN     NaN         NaN            NaN     NaN
     min          0.000000      NaN     NaN         NaN            NaN     NaN
     25%      75038.000000      NaN     NaN         NaN            NaN     NaN
     50%     150076.000000      NaN     NaN         NaN            NaN     NaN
     75%     225114.000000      NaN     NaN         NaN            NaN     NaN
     max     300152.000000      NaN     NaN         NaN            NaN     NaN

            arrival_time destination_city    class       duration      days_left  \
     count        300153           300153   300153  300153.000000  300153.000000
     unique            6                6        2            NaN            NaN
     top           Night           Mumbai  Economy            NaN            NaN
     freq          91538            59097   206666            NaN            NaN
     mean            NaN              NaN      NaN      12.221021      26.004751
     std             NaN              NaN      NaN       7.191997      13.561004
     min             NaN              NaN      NaN       0.830000       1.000000
     25%             NaN              NaN      NaN       6.830000      15.000000
     50%             NaN              NaN      NaN      11.250000      26.000000
     75%             NaN              NaN      NaN      16.170000      38.000000
     max             NaN              NaN      NaN      49.830000      49.000000

                     price
     count   300153.000000
     unique            NaN
     top               NaN
```

```
freq              NaN
mean     20889.660523
std      22697.767366
min       1105.000000
25%       4783.000000
50%       7425.000000
75%      42521.000000
max     123071.000000
```

```
[ ]: clean_dataset.dropna(inplace=True)
     clean_dataset.shape
```

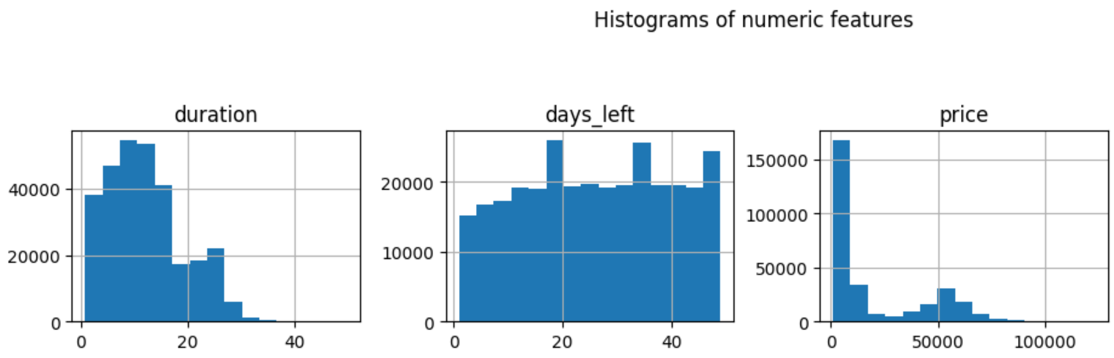
```
[ ]: (300153, 12)
```

```
[ ]: clean_dataset.isnull().sum()
```

```
[ ]: Unnamed: 0          0
     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
```

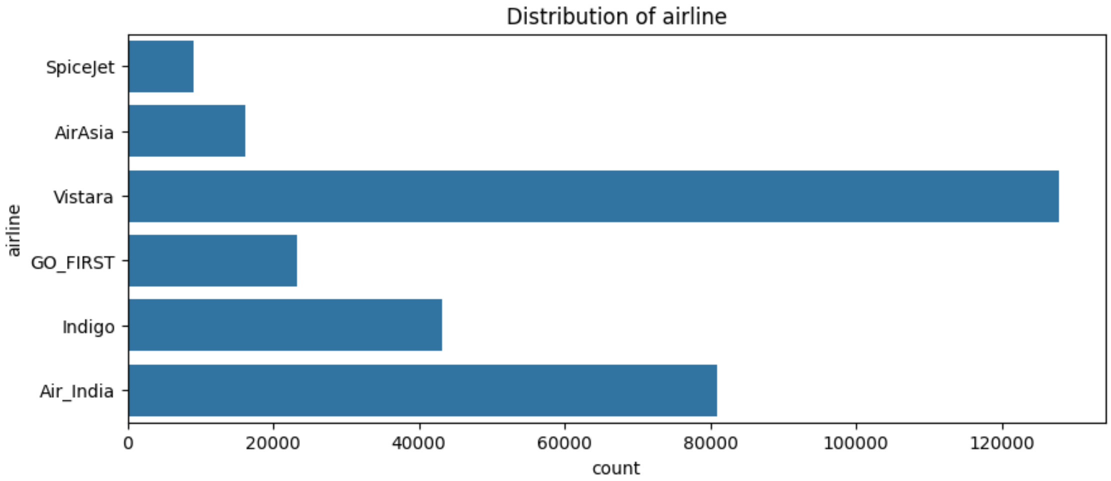
2 Let's visualize the first few rows of the dataset

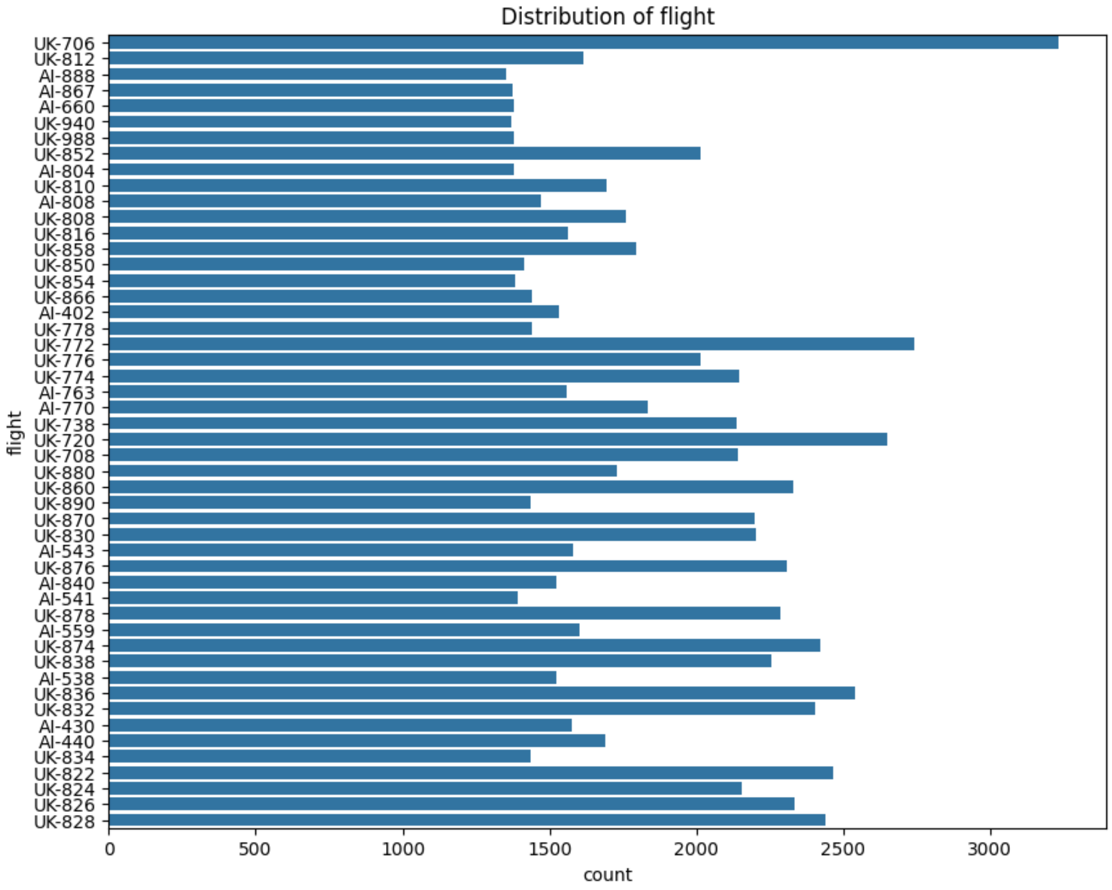
```
[ ]: # Plotting histograms for all numeric features to understand distributions
     # exclude the unnamed column
     clean_dataset.drop('Unnamed: 0', axis=1, inplace=True)
     clean_dataset.hist(bins=15, figsize=(15, 10), layout=(4, 4))
     plt.suptitle('Histograms of numeric features')
     plt.show()
```

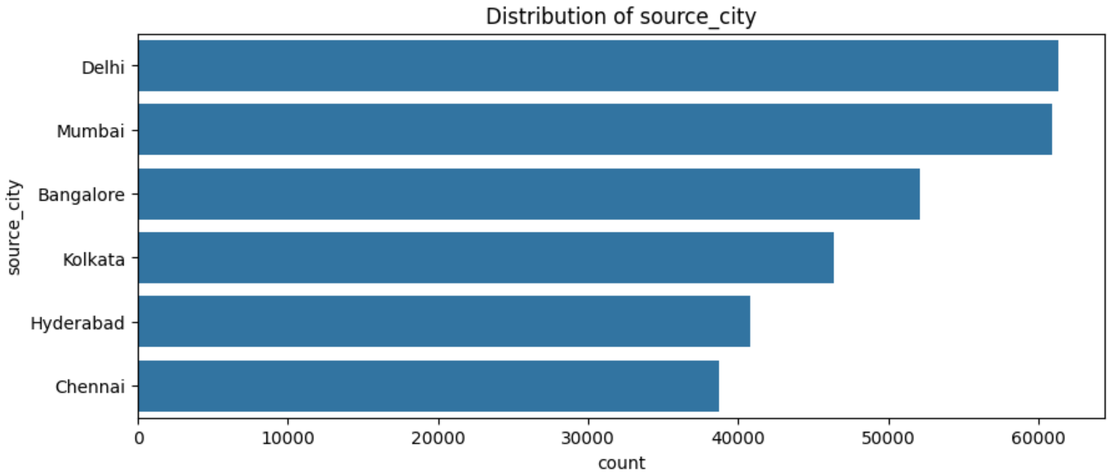


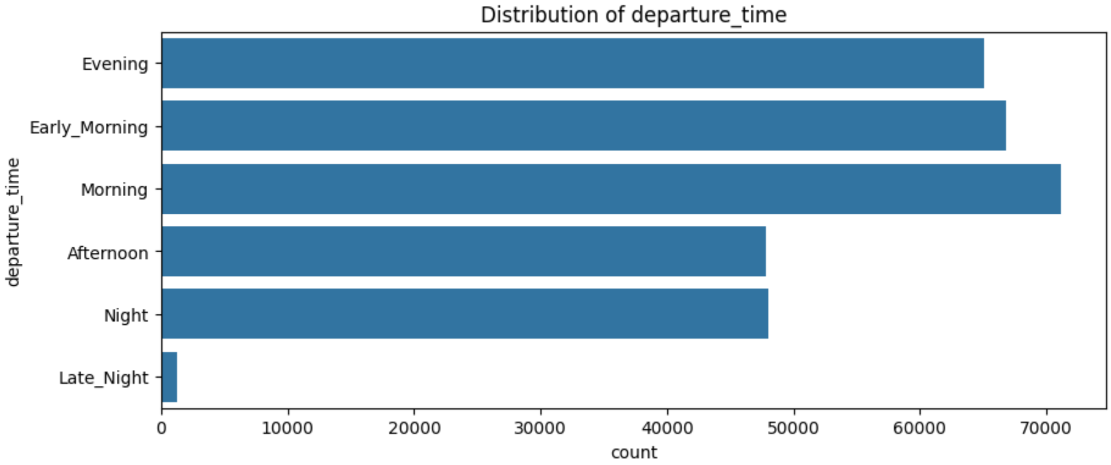
[ ]:

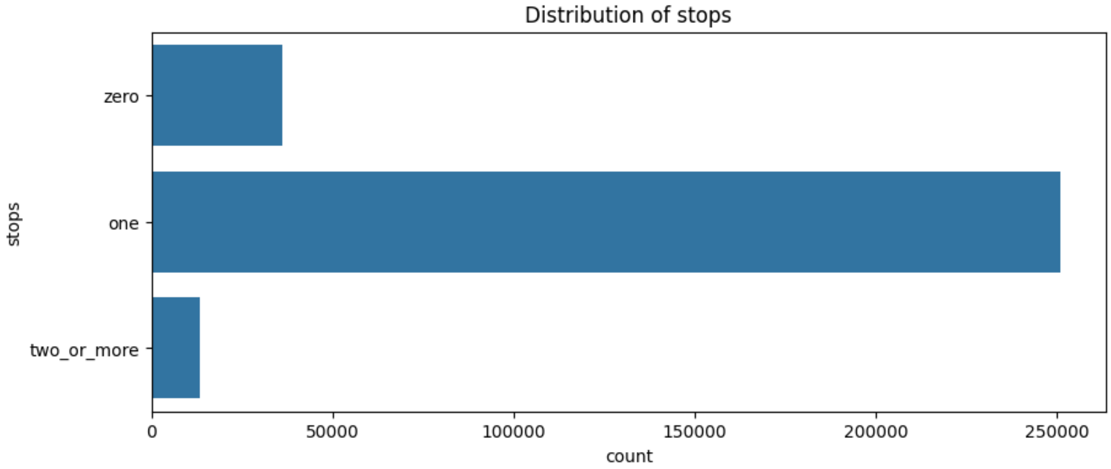
```
# For categorical data, we can use count plots to understand the distribution␣
 ↪of categories
for column in clean_dataset.select_dtypes(include=['object']).columns:
    # Plotting count plots for all categorical features
    # If the number of categories is too high,e.g., flight, we can filter the␣
 ↪top 50 categories to make the plot more readable
    if column != 'flight':
        plt.figure(figsize=(10, 4))
        sns.countplot(y=column, data=clean_dataset)
        plt.title(f'Distribution of {column}')
        plt.show()
    else:
        top_categories = clean_dataset[column].value_counts().index[:50]  # Get␣
 ↪top 50 categories
        filtered_data = clean_dataset[clean_dataset[column].
 ↪isin(top_categories)]
        plt.figure(figsize=(10, 8))
        sns.countplot(y=column, data=filtered_data)
        plt.yticks(fontsize=10)
        plt.title(f'Distribution of {column}')
        plt.show()
```

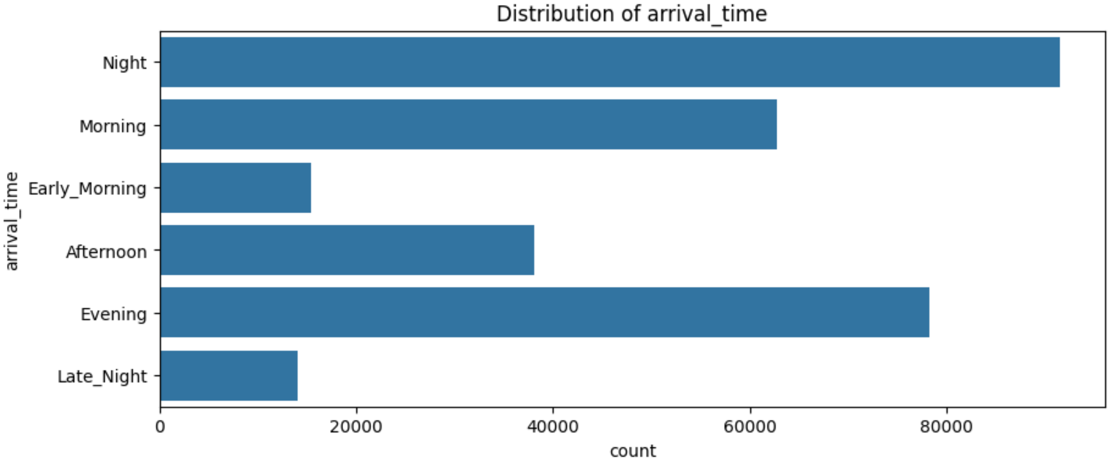

Distribution of airline
airline
SpiceJet
AirAsia
Vistara
GO_FIRST
Indigo
Air_India
0
20000
40000
60000
80000
100000
120000
count

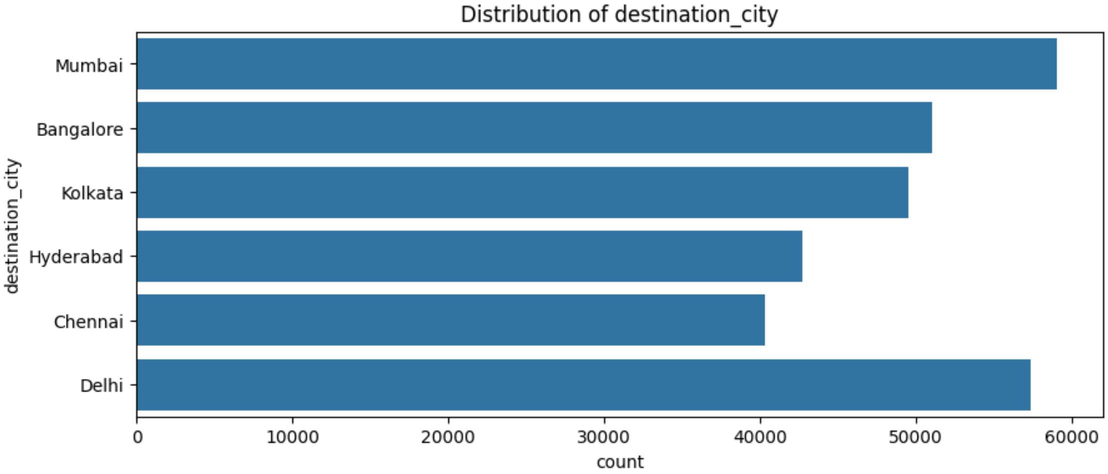

Distribution of flight
flight
UK-706
UK-812
AI-888
AI-867
AI-660
UK-940
UK-988
UK-852
AI-804
UK-810
AI-808
UK-808
UK-816
UK-858
UK-850
UK-854
UK-866
AI-402
UK-778
UK-772
UK-776
UK-774
AI-763
AI-770
UK-738
UK-720
UK-708
UK-880
UK-860
UK-890
UK-870
UK-830
AI-543
UK-876
AI-840
AI-541
UK-878
AI-559
UK-874
UK-838
AI-538
UK-836
UK-832
AI-430
AI-440
UK-834
UK-822
UK-824
UK-826
UK-828
0
500
1000
1500
2000
2500
3000
count

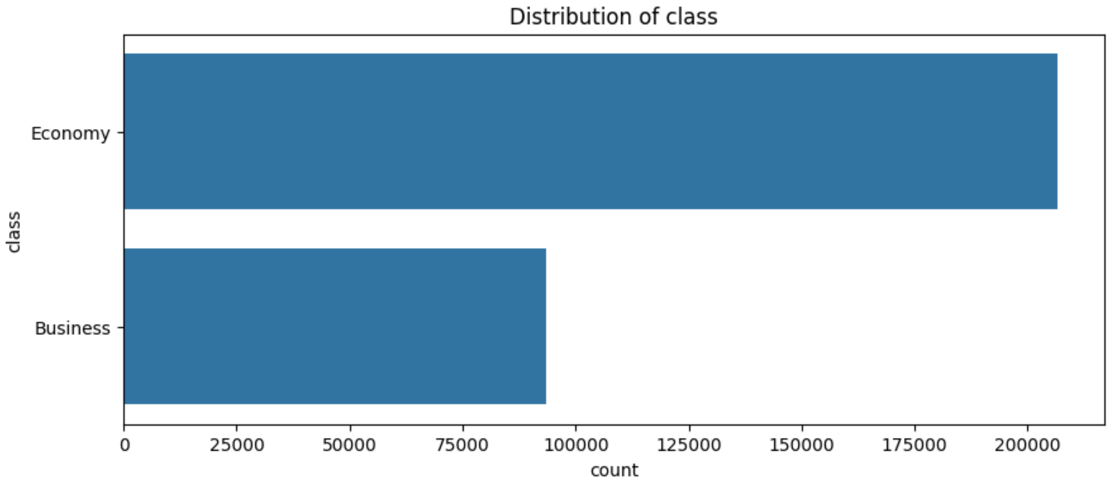

Distribution of source_city
source_city
Delhi
Mumbai
Bangalore
Kolkata
Hyderabad
Chennai
0
10000
20000
30000
40000
50000
60000
count


Distribution of departure_time
departure_time
Evening
Early_Morning
Morning
Afternoon
Night
Late_Night
0
10000
20000
30000
40000
50000
60000
70000
count


Distribution of stops
stops
zero
one
two_or_more
0
50000
100000
150000
200000
250000
count

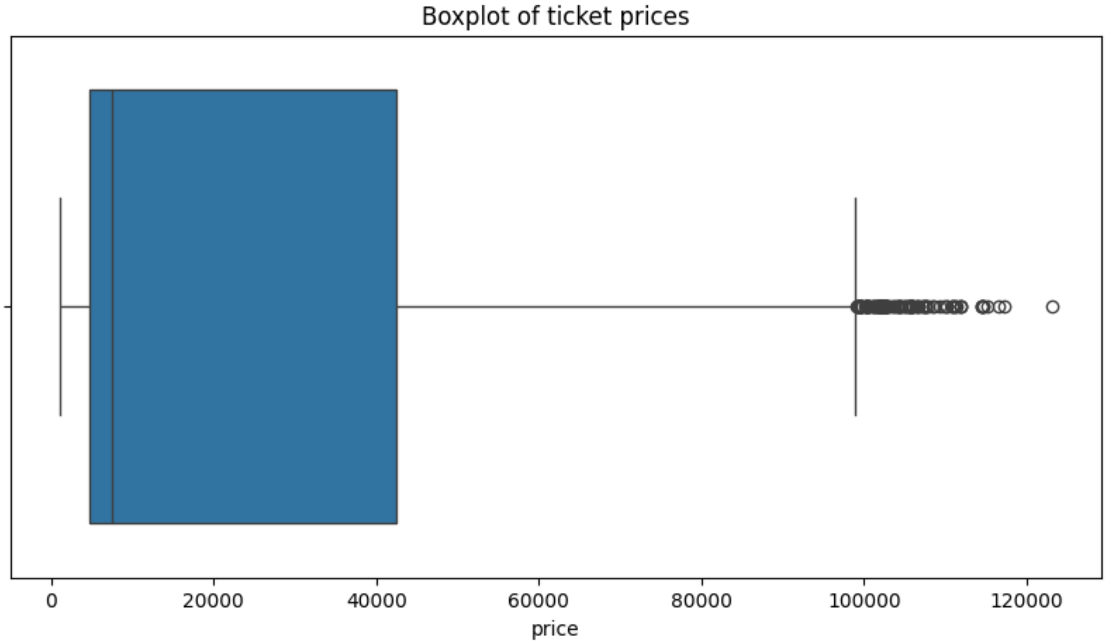

Distribution of arrival_time
arrival_time
Night
Morning
Early_Morning
Afternoon
Evening
Late_Night
0
20000
40000
60000
80000
count


Distribution of destination_city
destination_city
Mumbai
Bangalore
Kolkata
Hyderabad
Chennai
Delhi
0
10000
20000
30000
40000
50000
60000
count



[ ]:

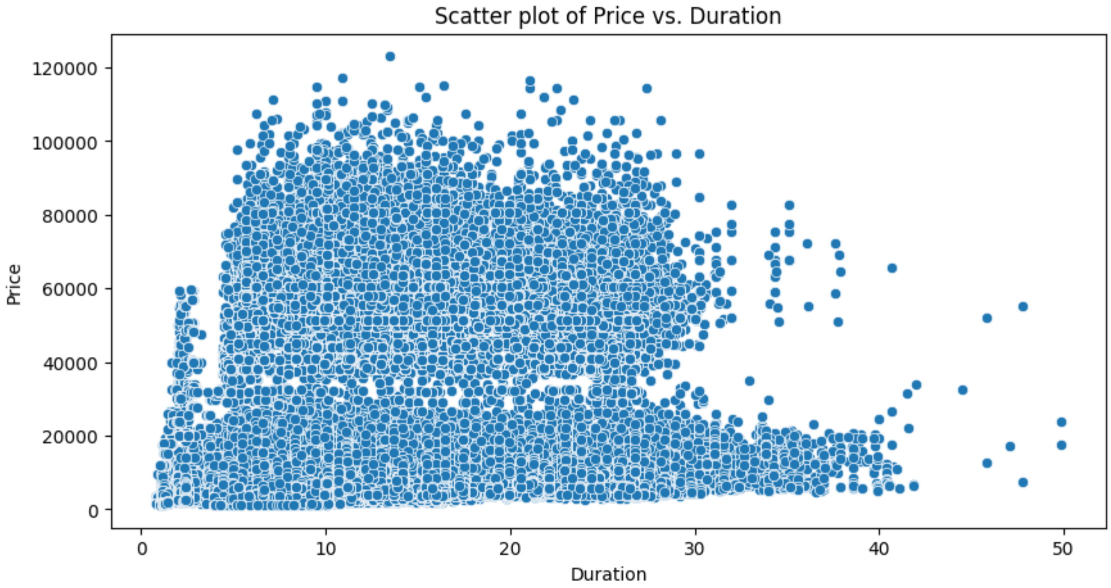
```
# Boxplot for the price column to see its distribution and spot any outliers
plt.figure(figsize=(10, 5))
sns.boxplot(x=clean_dataset['price'])
plt.title('Boxplot of ticket prices')
plt.show()
```



[ ]:

```
# A scatter plot to visualize the relationship between two variables, for␣
↪example, price and duration
```

```
plt.figure(figsize=(10, 5))
sns.scatterplot(x=clean_dataset['duration'], y=clean_dataset['price'])
plt.title('Scatter plot of Price vs. Duration')
plt.xlabel('Duration')
plt.ylabel('Price')
plt.show()
```

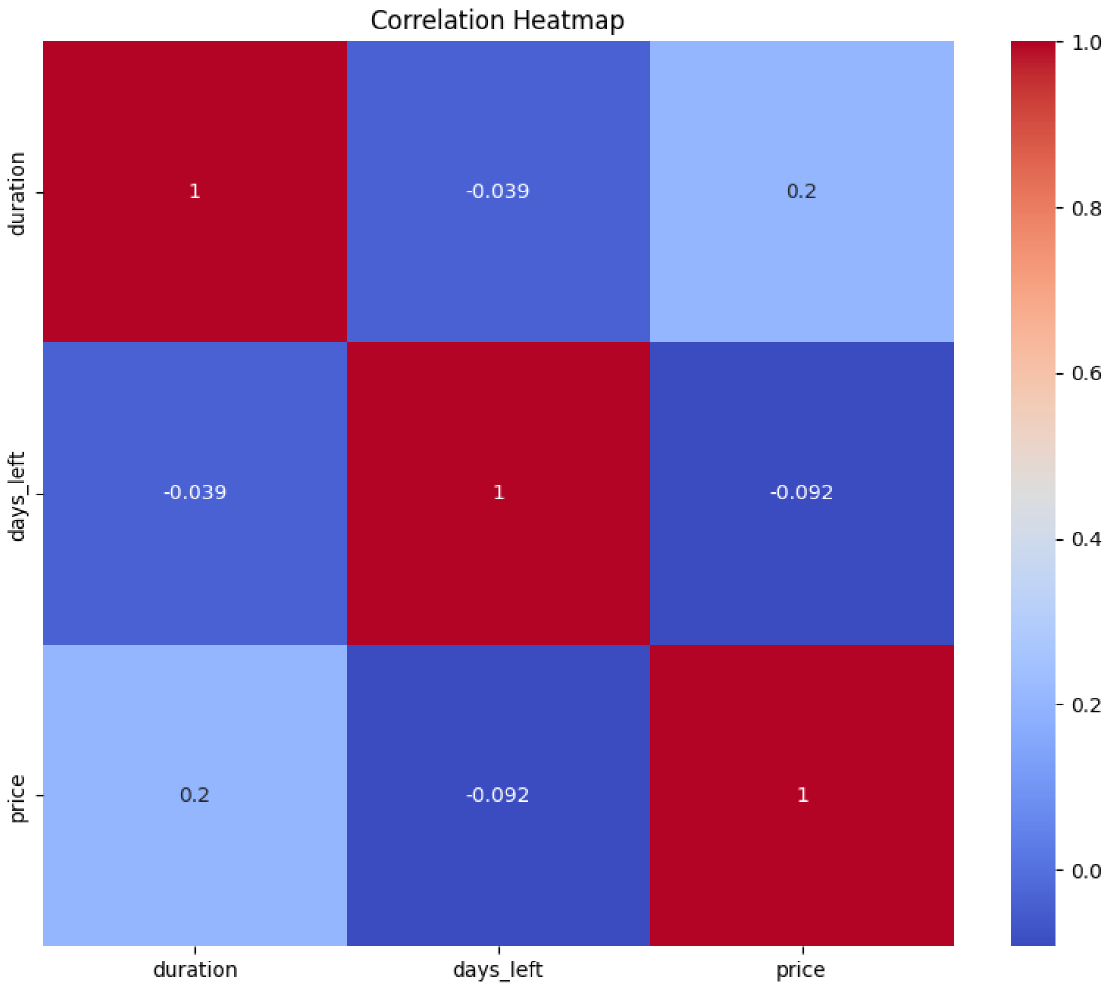


[ ]:

```
# Correlation heatmap to understand the relationships between variables
# Select only the numeric columns for correlation
numeric_dataset = clean_dataset.select_dtypes(include=[np.number])
correlation_matrix = numeric_dataset.corr()
```

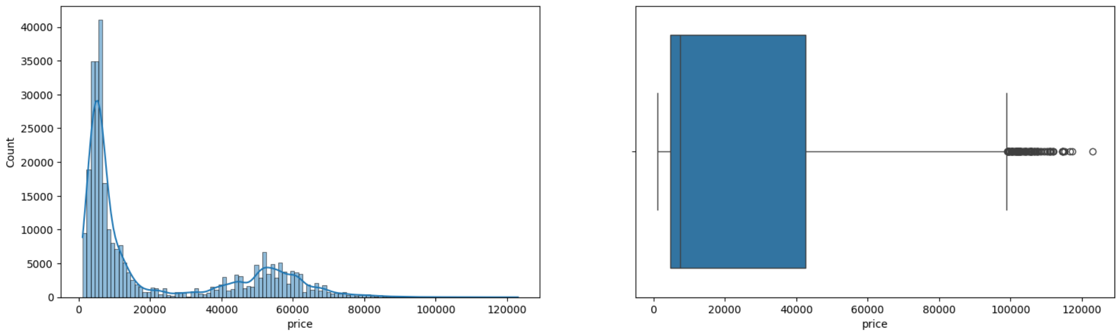
[ ]:

```
# Visualize the correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



```
[ ]: plt.figure(figsize = (18,5))
     plt.subplot(1,2,1)
     sns.histplot(x = 'price', data = clean_dataset, kde = True)
     plt.subplot(1,2,2)
     sns.boxplot(x = 'price', data = clean_dataset)
```

```
[ ]: <Axes: xlabel='price'>
```

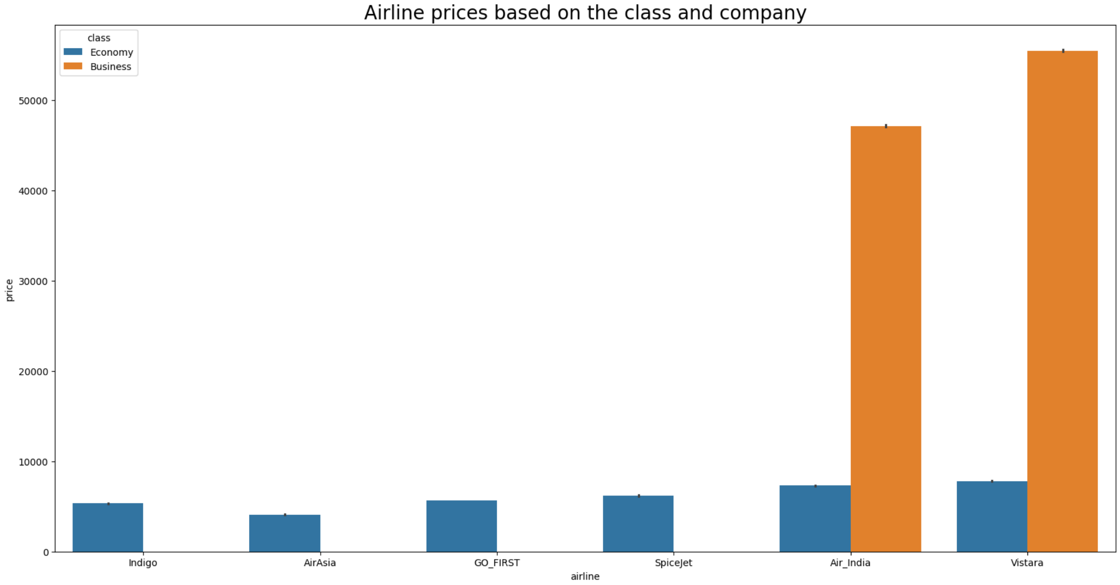


[ ]:

```
plt.figure(figsize=(20, 10))
sns.barplot(x='airline',y='price',hue="class",data=clean_dataset.
  ↪sort_values("price")).set_title('Airline prices based on the class and␣
  ↪company',fontsize=20)
```

[ ]:

```
Text(0.5, 1.0, 'Airline prices based on the class and company')
```



3 Make data transformation

[ ]:

```
transformed_dataset = clean_dataset.copy()
transformed_dataset['Economy'] = clean_dataset['class'] == 'Economy'
transformed_dataset.drop('class', axis=1, inplace=True)
```

[ ]:

```
#transformed_dataset['source_city'].unique()
```

```
[ ]: city_size = { # this is for year 2011 - https://en.wikipedia.org/wiki/
      ↪List_of_cities_in_India_by_population
         'Delhi': 110,
         'Mumbai': 124,
         'Bangalore': 84,
         'Kolkata': 44,
         'Hyderabad': 69,
         'Chennai' : 46
     }
     transformed_dataset['source_size'] = transformed_dataset['source_city'].
      ↪replace(city_size)
     transformed_dataset.drop('source_city', axis=1, inplace=True)
     transformed_dataset['destination_size'] =␣
      ↪transformed_dataset['destination_city'].replace(city_size)
     transformed_dataset.drop('destination_city', axis=1, inplace=True)
```

```
[ ]: transformed_dataset = pd.
      ↪get_dummies(transformed_dataset,columns=['departure_time','arrival_time'])
```

```
[ ]: stops = {
         'zero': 0,
         'one': 1,
         'two_or_more': 2,
     }
     transformed_dataset['stops_num'] = transformed_dataset['stops'].replace(stops)
     transformed_dataset.drop('stops', axis=1, inplace=True)
```

```
[ ]: transformed_dataset = pd.get_dummies(transformed_dataset,columns=['airline'])
```

```
[ ]: transformed_dataset['flight_num'] = pd.
      ↪factorize(transformed_dataset['flight'])[0]
     transformed_dataset.drop('flight', axis=1, inplace=True)
```

```
[ ]: transformed_dataset.head()
```

```
[ ]:    duration  days_left  price  Economy  source_size  destination_size  \
     0      2.17          1   5953     True          110               124
     1      2.33          1   5953     True          110               124
     2      2.17          1   5956     True          110               124
     3      2.25          1   5955     True          110               124
     4      2.33          1   5955     True          110               124

        departure_time_Afternoon  departure_time_Early_Morning  \
     0                     False                         False
     1                     False                          True
     2                     False                          True
     3                     False                         False
```

```
4                     False                         False

   departure_time_Evening  departure_time_Late_Night  …  \
0                    True                      False  …
1                   False                      False  …
2                   False                      False  …
3                   False                      False  …
4                   False                      False  …

   arrival_time_Morning  arrival_time_Night  stops_num  airline_AirAsia  \
0                 False                True          0            False
1                  True               False          0            False
2                 False               False          0             True
3                 False               False          0            False
4                  True               False          0            False

   airline_Air_India  airline_GO_FIRST  airline_Indigo  airline_SpiceJet  \
0              False             False           False              True
1              False             False           False              True
2              False             False           False             False
3              False             False           False             False
4              False             False           False             False

  airline_Vistara  flight_num
0           False           0
1           False           1
2           False           2
3            True           3
4            True           4

[5 rows x 26 columns]
```

[ ]:

```
transformed_dataset.describe()
# output the transformed dataset to a new CSV file
transformed_dataset.to_csv('../datasets/Transformed_Dataset.csv', index=False)
```