Flight Data Analysis Project

This project involves analyzing flight data using various data visualization techniques. Below is a detailed explanation of the project and the steps involved:

Project Overview

The primary goal of this project is to analyze flight data to extract meaningful insights and visualize various aspects of the data. The project includes reading the data, performing exploratory data analysis (EDA), and visualizing different features of the dataset.

Steps Involved:

1. **Loading the Dataset**:

- The dataset is loaded from a CSV file named `flights_data.csv`.

2. **Initial Data Exploration**:

- Display the shape and first few rows of the dataset to understand its structure and content.

```python
flights = pd.read_csv('/content/flights_data.csv')
print(flights.shape)
flights.head()
```

3. **Visualizing the Number of Flights by Source**:

- Use a count plot to visualize the number of flights originating from different sources.

```python
sb.countplot(data=flights, x='Source')
plt.ylabel('Number of Flights', fontsize=12)
plt.xlabel('Source', fontsize=12)
```

4. **Ordered Visualization by Source**:

- Visualize the count of flights by source in a specific order based on the frequency.

```python
gen_order = flights['Source'].value_counts().index
sb.countplot(data=flights, x='Source', order=gen_order)
```

5. **Visualizing the Number of Flights by Airline**:

- Use a count plot to visualize the number of flights for each airline.

```python
sb.countplot(data=flights, x='Airline')
plt.xticks(rotation=90)
```

6. **Handling Missing Values**:

- Check for missing values in the dataset and visualize the count of missing values for each feature.

```python
na_counts = flights.isna().sum()
sb.barplot(na_counts.index.values, na_counts)
plt.xticks(rotation=90)
plt.ylabel('Number of missing values', fontsize=12)
```

7. **Visualizing Flight Destinations**:

- Use a pie chart to visualize the distribution of flight destinations.

```python
```

```
sorted_counts = flights['Destination'].value_counts()
plt.pie(sorted_counts, labels=sorted_counts.index, startangle=90, counterclock=False)
plt.axis('square')
plt.title("Flight Destination's")
```

8. **Visualizing Flight Duration**:

- Use a histogram to visualize the distribution of flight durations.

```python
plt.hist(data=flights, x='Duration(minutes)')
```

9. **Visualizing Flight Prices**:

- Use a histogram and a distplot to visualize the distribution of flight prices.

```python
plt.hist(data=flights, x='Price', bins=20)
sb.distplot(flights['Price'])
sb.distplot(flights['Price'], kde=False)
```

10. **Advanced Histogram Visualization**:

- Create a histogram with specific bin edges for a more detailed visualization of flight prices.

```python
bin_edges = np.arange(0, flights['Price'].max() + 1, 1200)
sb.distplot(flights['Price'], bins=bin_edges, kde=False, hist_kws={'alpha': 1})
```

How to Use the Code

1. **Install Required Libraries**:

- Ensure that you have the necessary libraries installed: `numpy`, `pandas`, `matplotlib`, and `seaborn`.

```bash

pip install numpy pandas matplotlib seaborn

```

2. **Run the Analysis**:

- Load the dataset and execute each step sequentially to visualize the data and gain insights.

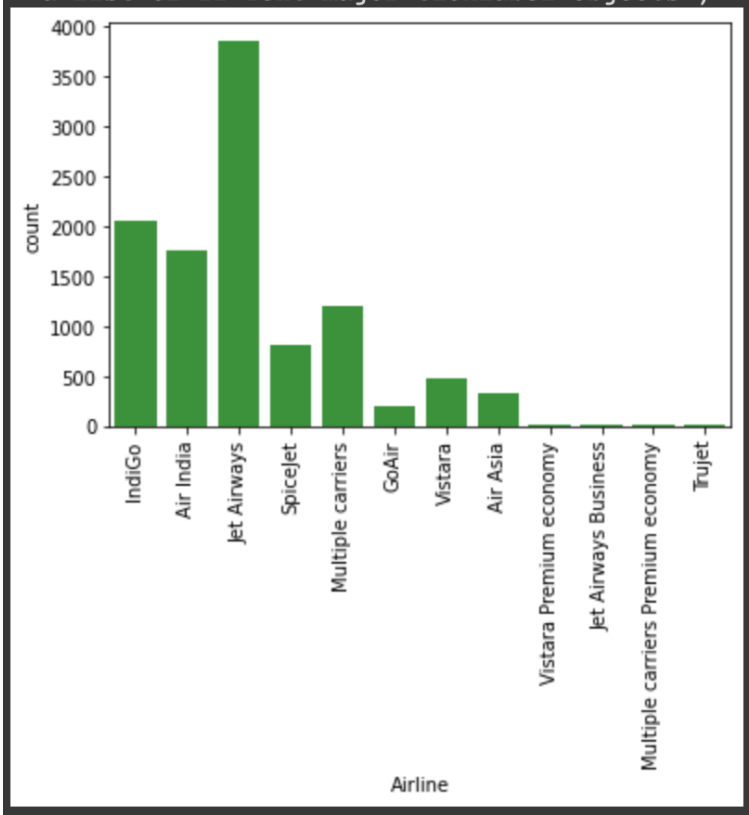
3. **Customize Visualizations**:

- Modify the visualizations as needed to better suit your analysis or presentation requirements.

Outputs:

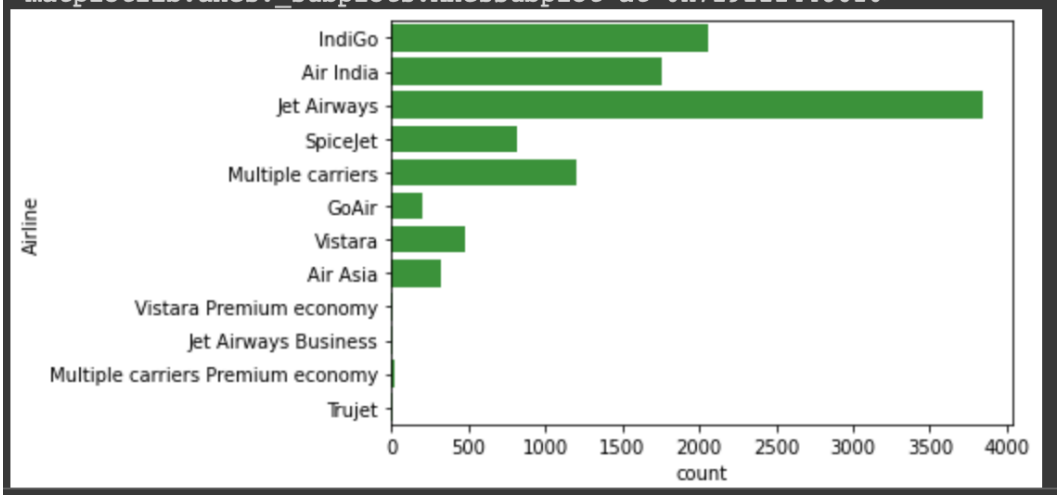
```
base_color = sb.color_palette()[2]
sb.countplot(data = flights, x = 'Airline',color=base_color)
plt.xticks(rotation=90)
```

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11]),
 <a list of 12 Text major ticklabel objects>)
```



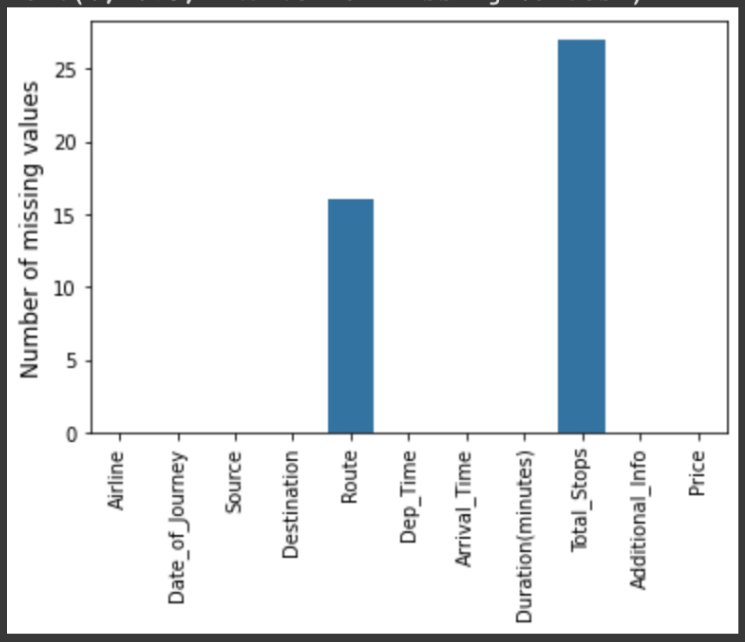

```
base_color = sb.color_palette()[2]
sb.countplot(data = flights, y = 'Airline',color=base_color)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f922f448610>
```



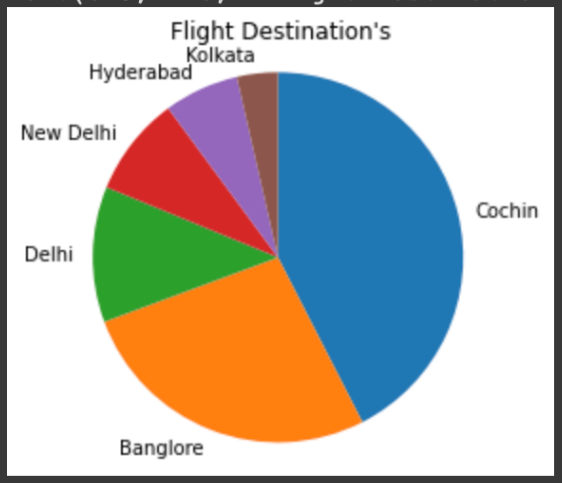
```
base_color=sb.color_palette()[0]
sb.barplot(na_counts.index.values,na_counts,color =base_color)
plt.xticks(rotation=90)
plt.ylabel('Number of missing values',fontsize=12)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
  FutureWarning
Text(0, 0.5, 'Number of missing values')
```



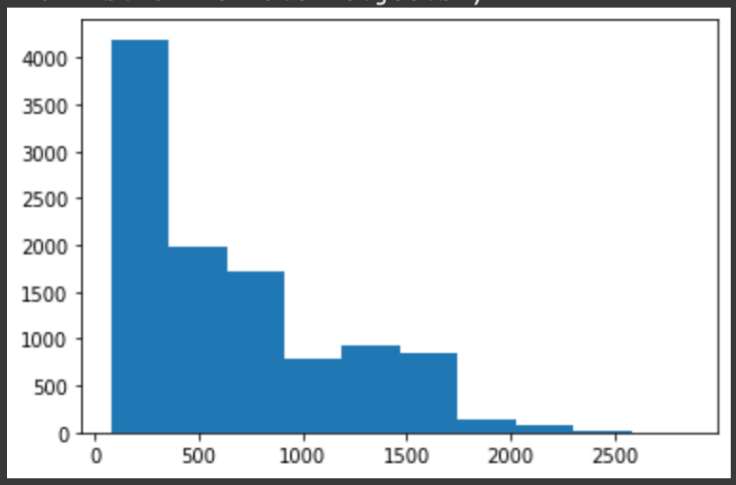

```
sorted_counts =flights['Destination'].value_counts()
plt.pie(sorted_counts,labels=sorted_counts.index,startangle=90,co
plt.axis('square')
plt.title('Flight Destination\'s')
```

```
Text(0.5, 1.0, "Flight Destination's")
```



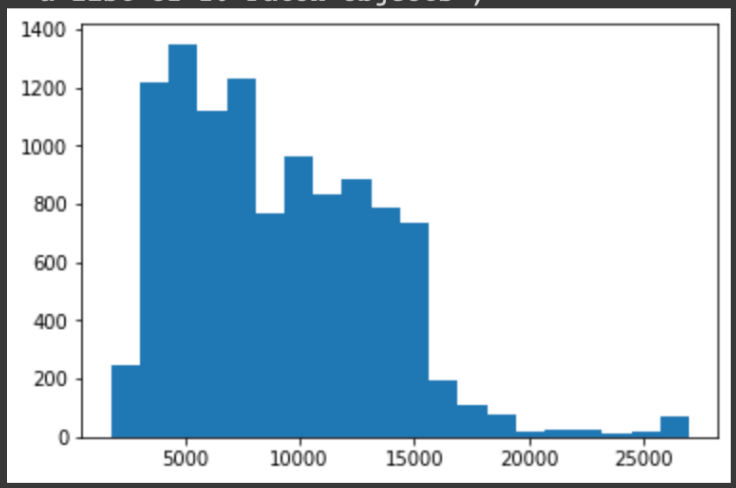

```
plt.hist(data=flights,x='Duration(minutes)')
```

```
(array([4.198e+03, 1.988e+03, 1.720e+03, 7.870e+02, 9.310e+02, 8.
        1.320e+02, 7.000e+01, 1.000e+01, 2.000e+00]),
 array([  75. ,  353.5,  632. ,  910.5, 1189. , 1467.5, 1746. , 2
        2303. , 2581.5, 2860. ]),
 <a list of 10 Patch objects>)
```

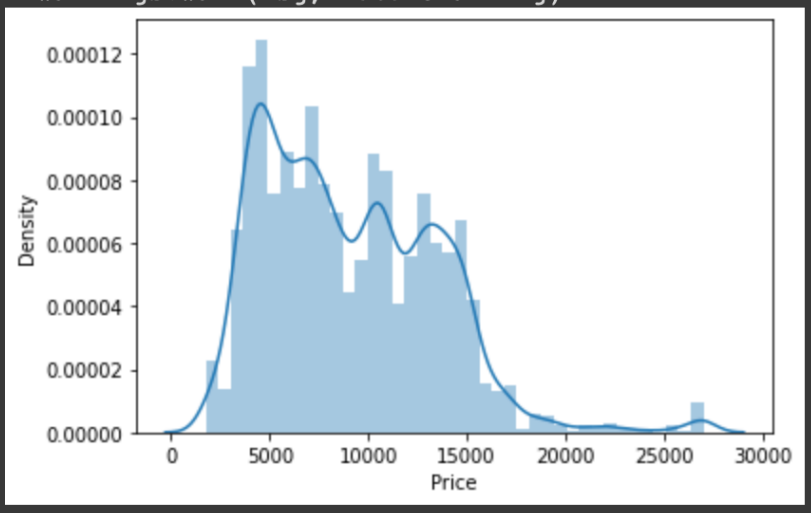



```
plt.hist(data=flights,x='Price',bins=20)
```

```
(array([ 247., 1218., 1350., 1123., 1228.,  769.,  964.,  832.,
         789.,  735.,  195.,  109.,   76.,   18.,   27.,   24.,
          15.,   68.]),
 array([ 1759.  ,  3021.05,  4283.1 ,  5545.15,  6807.2 ,  8069.2
        9331.3 , 10593.35, 11855.4 , 13117.45, 14379.5 , 15641.5
       16903.6 , 18165.65, 19427.7 , 20689.75, 21951.8 , 23213.8
       24475.9 , 25737.95, 27000.  ]),
 <a list of 20 Patch objects>)
```

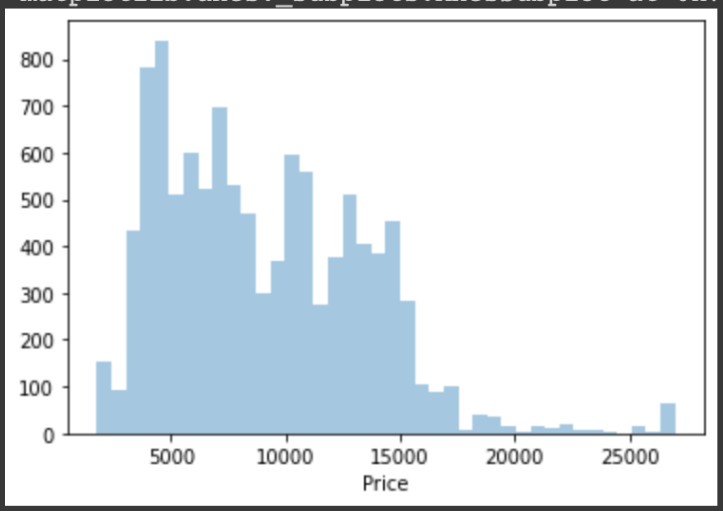


```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2
  warnings.warn(msg, FutureWarning)
```




```
sb.distplot(flights['Price'],kde= False)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f922f3e9810>
```

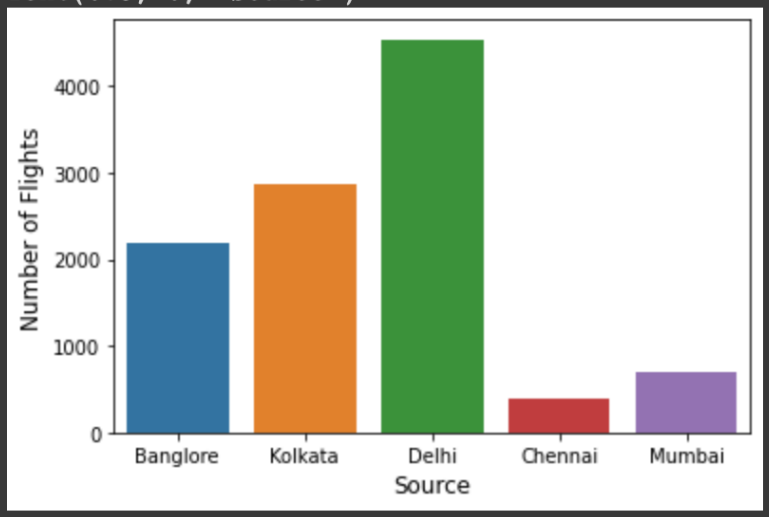



```
bin_edges = np.arange(0, flights['Price'].max()+1,1200)
sb.distplot(flights['Price'],bins =bin_edges, kde= False,hist_kws
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2
  warnings.warn(msg, FutureWarning)
```

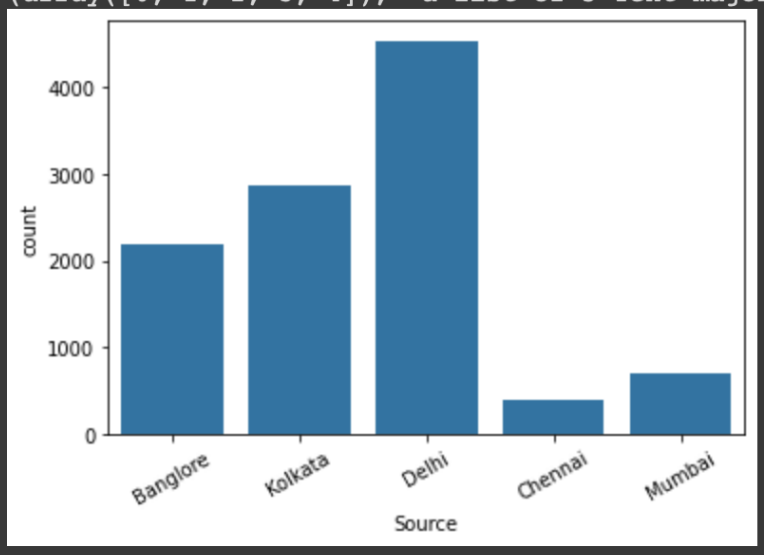
```
sb.countplot(data=flights,x = 'Source')
#plt.xticks(rotation = 30)
plt.ylabel('Number of Flights',fontsize=12)
plt.xlabel('Source',fontsize=12)
```

```
Text(0.5, 0, 'Source')
```




```
base_color = sb.color_palette()[0]
sb.countplot(data =flights,x='Source',color = base_color)
plt.xticks(rotation=30)
```

```
(array([0, 1, 2, 3, 4]), <a list of 5 Text major ticklabel objects
```




```
base_color = sb.color_palette()[1]
gen_order = flights['Source'].value_counts().index
sb.countplot(data=flights,x='Source',color = base_color,
             order = gen_order)
```