CS773-Course-Project

July 24, 2024

1 CS 773: Data Mining and Security

1.1 Course project

1.2 Due: August 1, 2024

1.3 Airline Passenger Satisfaction Survey

To present the final excutive summary to leardship to take the better decicions on airline passenger satisfaction using data scinene, data minining and machine learning technquies.The idea is to make some useful conclusions that could help airline executives improve passenger satisfaction. The project aims to identify crucial factors that influence passenger satisfaction.

2 Presenter: Ashish Verma

3 Dataset Details

Attribute Information: * Gender: Gender of the passengers (Female, Male)

- Customer Type: The customer type (Loyal customer, disloyal customer)
- Age: The actual age of the passengers
- Type of Travel: Purpose of the flight of the passengers (Personal Travel, Business Travel)
- Class: Travel class in the plane of the passengers (Business, Eco, Eco Plus)
- Flight distance: The flight distance of this journey
- Inflight wifi service: Satisfaction level of the inflight wifi service (0:Not Applicable;1-5)
- Departure/Arrival time convenient: Satisfaction level of Departure/Arrival time convenient
- Ease of Online booking: Satisfaction level of online booking
- Gate location: Satisfaction level of Gate location
- Food and drink: Satisfaction level of Food and drink
- Online boarding: Satisfaction level of online boarding
- Seat comfort: Satisfaction level of Seat comfort
- Inflight entertainment: Satisfaction level of inflight entertainment
- On-board service: Satisfaction level of On-board service

- Leg room service: Satisfaction level of Leg room service
- Baggage handling: Satisfaction level of baggage handling
- Check-in service: Satisfaction level of Check-in service
- Inflight service: Satisfaction level of inflight service
- Cleanliness: Satisfaction level of Cleanliness
- Departure Delay in Minutes: Minutes delayed when departure
- Arrival Delay in Minutes: Minutes delayed when Arrival

Label: * Satisfaction: Airline satisfaction level(Satisfaction, neutral or dissatisfaction)

Dataset link :

https://www.kaggle.com/datasets/teejmahal20/airline-passenger-satisfaction

4 Configure Collab to Get Kaggle Datasets

[1]:
```
from google.colab import drive
drive.mount('/content/drive',force_remount=True)

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


import statsmodels.api as sm
import scipy.stats as stats

from sklearn.decomposition import PCA
from mlxtend.frequent_patterns import fpgrowth, association_rules
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import LabelEncoder


# set style of visualization
sns.set_style("whitegrid")
sns.set_palette("Set2")

import warnings
# Settings the warnings to be ignored
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
pd.set_option('display.float_format', lambda x: '%.5f' % x)
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import label_binarize

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import RocCurveDisplay
from sklearn.metrics import roc_auc_score
from sklearn.metrics import balanced_accuracy_score
```

```
Mounted at /content/drive
```

[2]:
```
! mkdir ~/.kaggle
```

[3]:
```
! cp /content/drive/MyDrive/kaggle.json ~/.kaggle/
```

[4]:
```
! chmod 600 ~/.kaggle/kaggle.json
```

[5]:
```
! kaggle datasets download teejmahal20/airline-passenger-satisfaction
```

```
Dataset URL: https://www.kaggle.com/datasets/teejmahal20/airline-passenger-
satisfaction
License(s): other
Downloading airline-passenger-satisfaction.zip to /content
  0% 0.00/2.71M [00:00<?, ?B/s]
100% 2.71M/2.71M [00:00<00:00, 35.0MB/s]
```

[6]:
```
! unzip -o  /content/airline-passenger-satisfaction.zip
```

```
Archive:  /content/airline-passenger-satisfaction.zip
  inflating: test.csv
  inflating: train.csv
```

4.0.1 Load data

[7]:
```
# Load the CSV file
file_path = '/content/train.csv'
train = pd.read_csv(file_path)

# Display the first few rows to understand the data structure
train.head()
```

[7]:
```
   Unnamed: 0      id  Gender      Customer Type  Age   Type of Travel  \
0           0   70172    Male     Loyal Customer   13  Personal Travel
1           1    5047    Male  disloyal Customer   25  Business travel
2           2  110028  Female     Loyal Customer   26  Business travel
3           3   24026  Female     Loyal Customer   25  Business travel
4           4  119299    Male     Loyal Customer   61  Business travel

      Class  Flight Distance  Inflight wifi service  \
0  Eco Plus              460                      3
1  Business              235                      3
2  Business             1142                      2
3  Business              562                      2
4  Business              214                      3

   Departure/Arrival time convenient  Ease of Online booking  Gate location  \
0                                  4                       3              1
1                                  2                       3              3
2                                  2                       2              2
3                                  5                       5              5
4                                  3                       3              3

   Food and drink  Online boarding  Seat comfort  Inflight entertainment  \
0               5                3             5                       5
1               1                3             1                       1
2               5                5             5                       5
3               2                2             2                       2
4               4                5             5                       3

   On-board service  Leg room service  Baggage handling  Checkin service  \
0                 4                 3                 4                4
1                 1                 5                 3                1
2                 4                 3                 4                4
3                 2                 5                 3                1
4                 3                 4                 4                3

   Inflight service  Cleanliness  Departure Delay in Minutes  \
0                 5            5                          25
1                 4            1                           1
2                 4            5                           0
3                 4            2                          11
4                 3            3                           0

   Arrival Delay in Minutes             satisfaction
0                  18.00000  neutral or dissatisfied
1                   6.00000  neutral or dissatisfied
2                   0.00000                satisfied
3                   9.00000  neutral or dissatisfied
```

```
4                   0.00000                 satisfied
```

[8]:
```python
train.drop(columns=['Unnamed: 0','id'],inplace=True)
```

[9]:
```python
# Load the CSV file
file_path = '/content/test.csv'
test = pd.read_csv(file_path)

# Display the first few rows to understand the data structure
test.head()
```

[9]:
```
   Unnamed: 0     id  Gender      Customer Type  Age   Type of Travel  \
0           0  19556  Female     Loyal Customer   52  Business travel
1           1  90035  Female     Loyal Customer   36  Business travel
2           2  12360    Male  disloyal Customer   20  Business travel
3           3  77959    Male     Loyal Customer   44  Business travel
4           4  36875  Female     Loyal Customer   49  Business travel

      Class  Flight Distance  Inflight wifi service  \
0       Eco              160                      5
1  Business             2863                      1
2       Eco              192                      2
3  Business             3377                      0
4       Eco             1182                      2

   Departure/Arrival time convenient  Ease of Online booking  Gate location  \
0                                  4                       3              4
1                                  1                       3              1
2                                  0                       2              4
3                                  0                       0              2
4                                  3                       4              3

   Food and drink  Online boarding  Seat comfort  Inflight entertainment  \
0               3                4             3                       5
1               5                4             5                       4
2               2                2             2                       2
3               3                4             4                       1
4               4                1             2                       2

   On-board service  Leg room service  Baggage handling  Checkin service  \
0                 5                 5                 5                2
1                 4                 4                 4                3
2                 4                 1                 3                2
3                 1                 1                 1                3
4                 2                 2                 2                4

   Inflight service  Cleanliness  Departure Delay in Minutes  \
```

```
0                  5             5                           50
1                  4             5                            0
2                  2             2                            0
3                  1             4                            0
4                  2             4                            0

   Arrival Delay in Minutes             satisfaction
0                  44.00000                satisfied
1                   0.00000                satisfied
2                   0.00000  neutral or dissatisfied
3                   6.00000                satisfied
4                  20.00000                satisfied
```

[10]:
```python
test.drop(columns=['Unnamed: 0','id'],inplace=True)
```

[11]:
```python
data = pd.concat([train, test])
```

[12]:
```python
data.info()
```

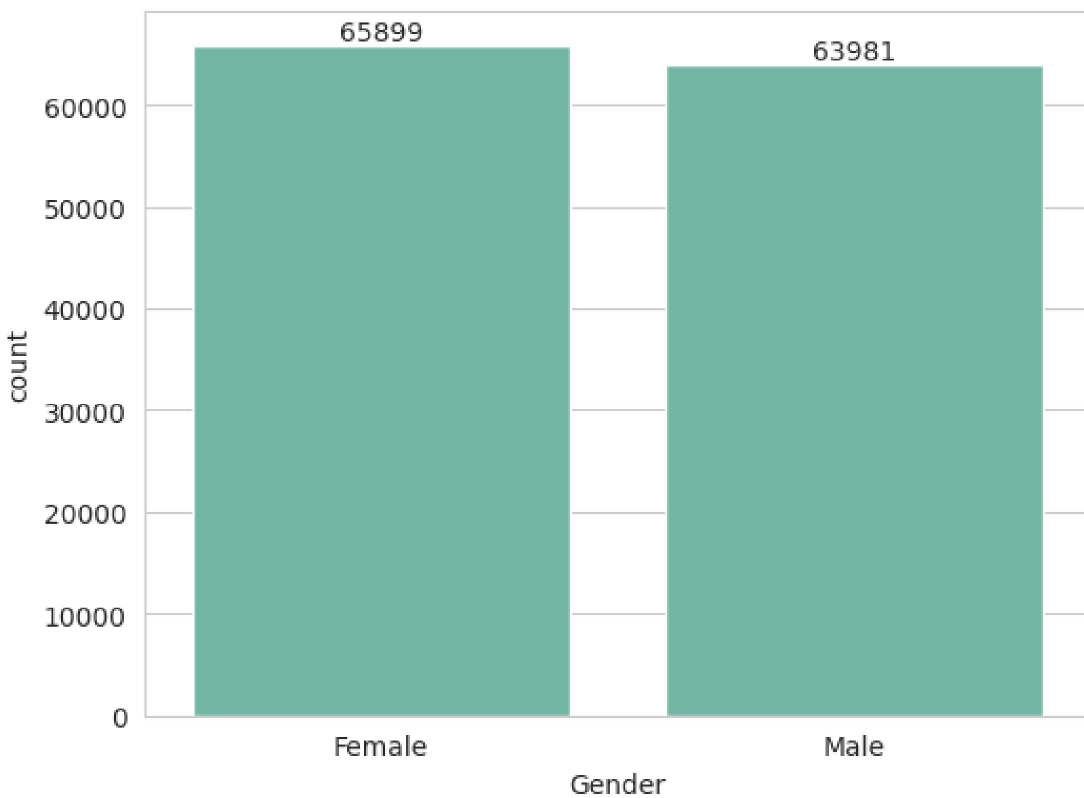
```
<class 'pandas.core.frame.DataFrame'>
Index: 129880 entries, 0 to 25975
Data columns (total 23 columns):
 #   Column                             Non-Null Count   Dtype
---  ------                             --------------   -----
 0   Gender                             129880 non-null  object
 1   Customer Type                      129880 non-null  object
 2   Age                                129880 non-null  int64
 3   Type of Travel                     129880 non-null  object
 4   Class                              129880 non-null  object
 5   Flight Distance                    129880 non-null  int64
 6   Inflight wifi service              129880 non-null  int64
 7   Departure/Arrival time convenient  129880 non-null  int64
 8   Ease of Online booking             129880 non-null  int64
 9   Gate location                      129880 non-null  int64
 10  Food and drink                     129880 non-null  int64
 11  Online boarding                    129880 non-null  int64
 12  Seat comfort                       129880 non-null  int64
 13  Inflight entertainment             129880 non-null  int64
 14  On-board service                   129880 non-null  int64
 15  Leg room service                   129880 non-null  int64
 16  Baggage handling                   129880 non-null  int64
 17  Checkin service                    129880 non-null  int64
 18  Inflight service                   129880 non-null  int64
 19  Cleanliness                        129880 non-null  int64
 20  Departure Delay in Minutes         129880 non-null  int64
 21  Arrival Delay in Minutes           129487 non-null  float64
 22  satisfaction                       129880 non-null  object
```

```
dtypes: float64(1), int64(17), object(5)
memory usage: 23.8+ MB
```

5 Exploratory Data Analysis

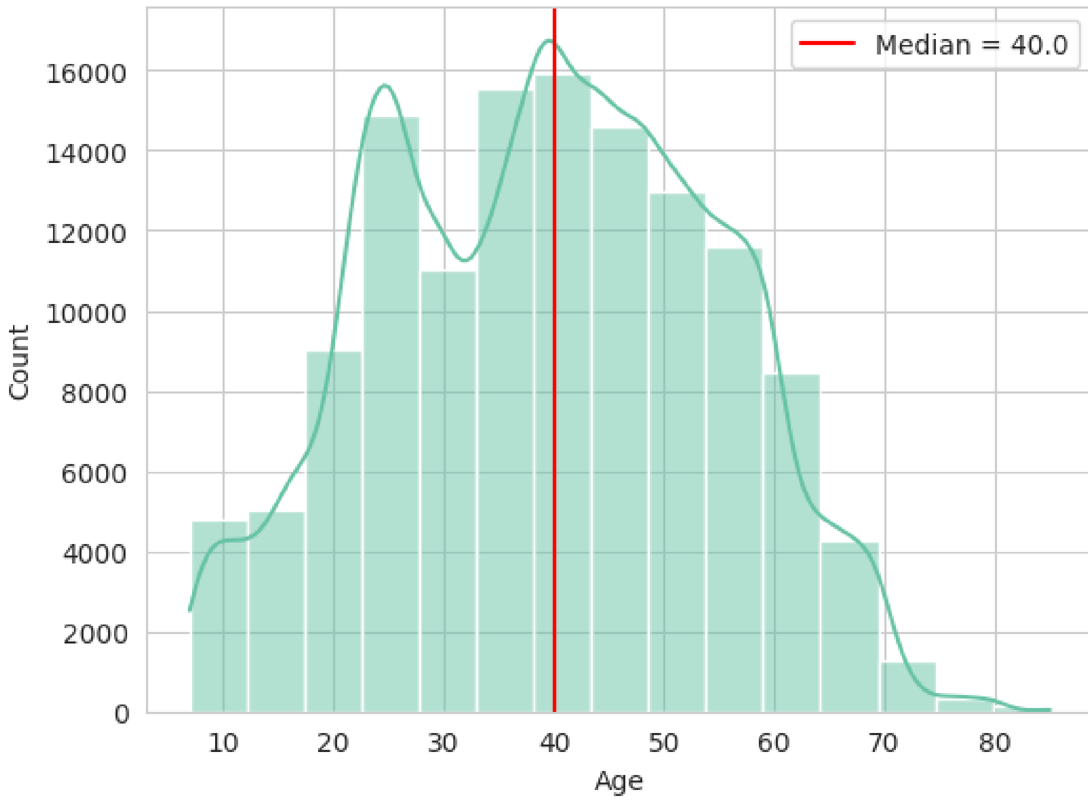
5.0.1 Univariate Analysis

[13]:
```
def count_plot(column_name):
    graph = sns.countplot(x = column_name, data = data, order =␣
↪data[column_name].value_counts().index)
    for container in graph.containers:
        graph.bar_label(container)

    plt.show()

count_plot("Gender")
```



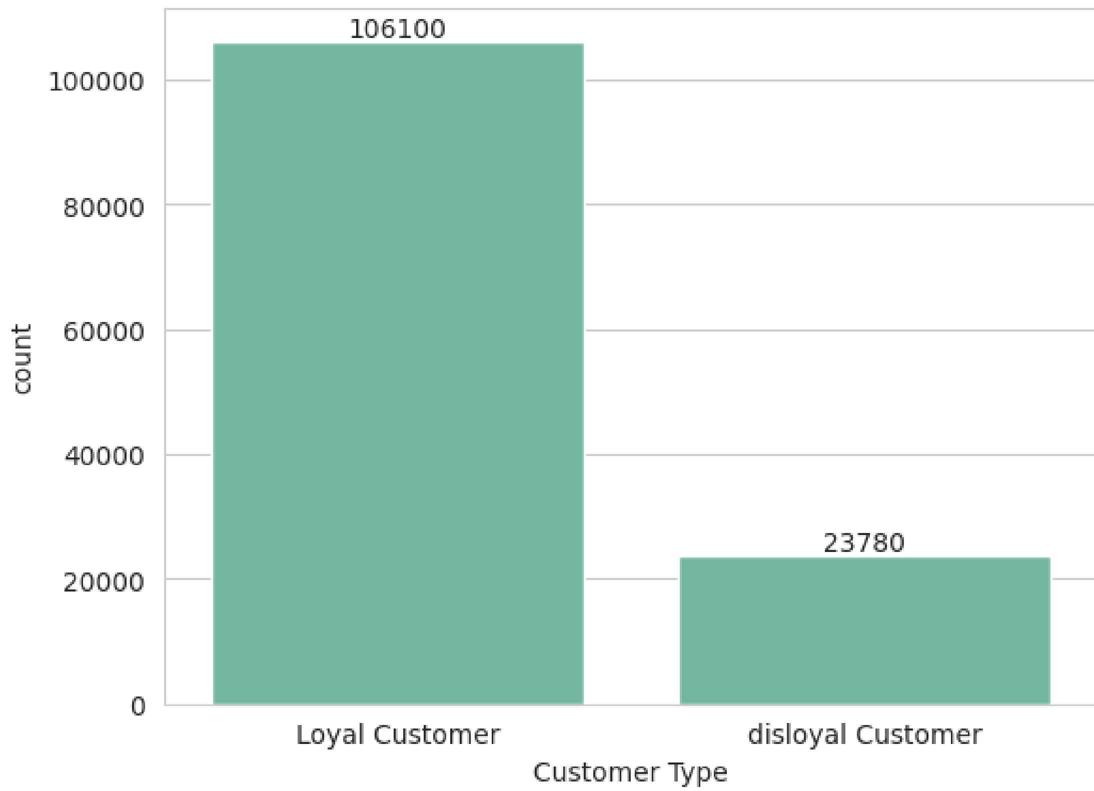
[14]:
```
sns.histplot(x = "Age", data = data, kde = True, bins = 15)
plt.axvline(data.Age.median(), label = f'Median = {data.Age.median()}', color =␣
↪'r')
plt.legend()
```

```
plt.show()
```

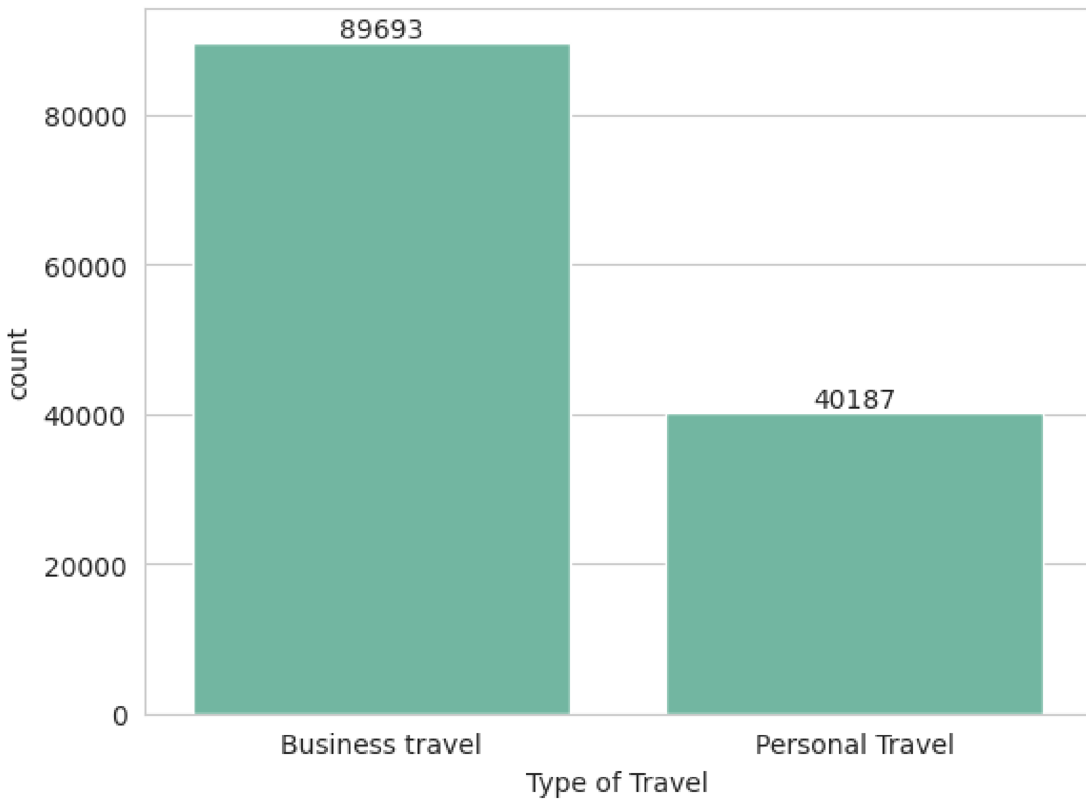


[15]:

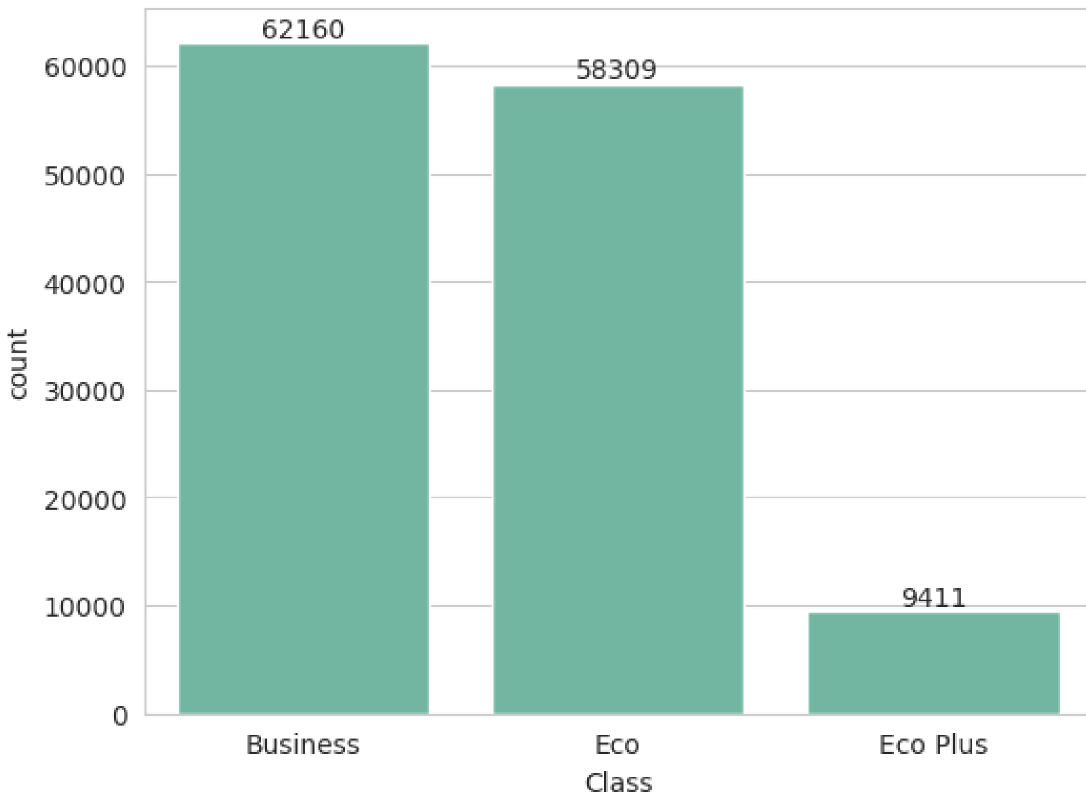
```
count_plot("Customer Type")
```



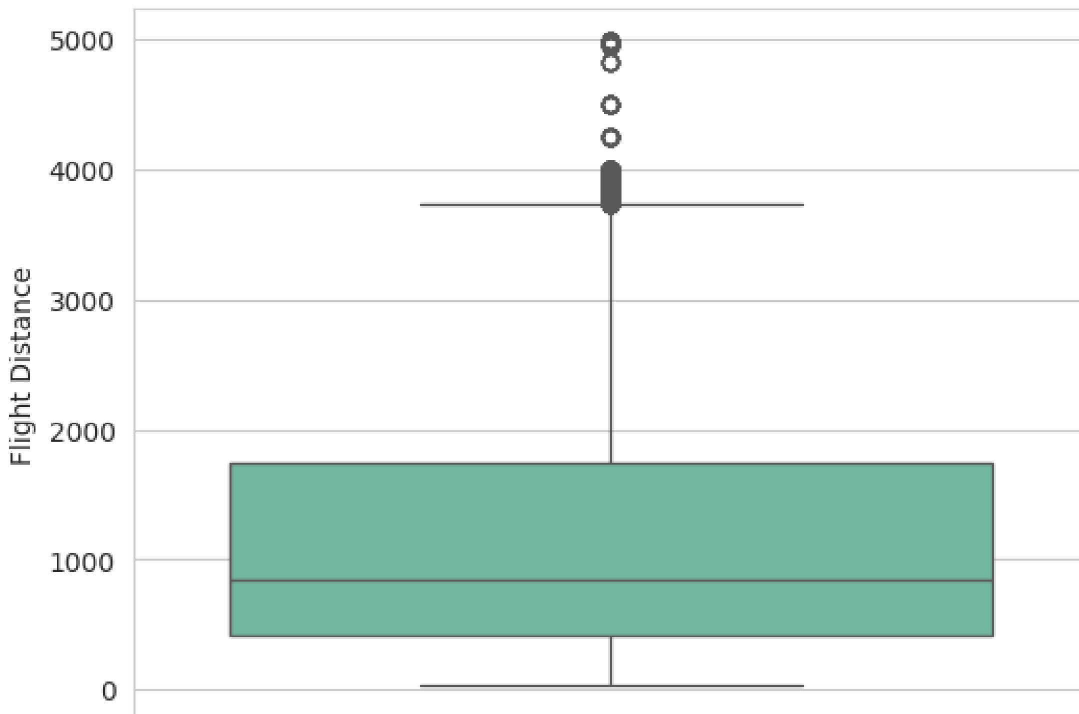
[16]:
```
count_plot("Type of Travel")
```



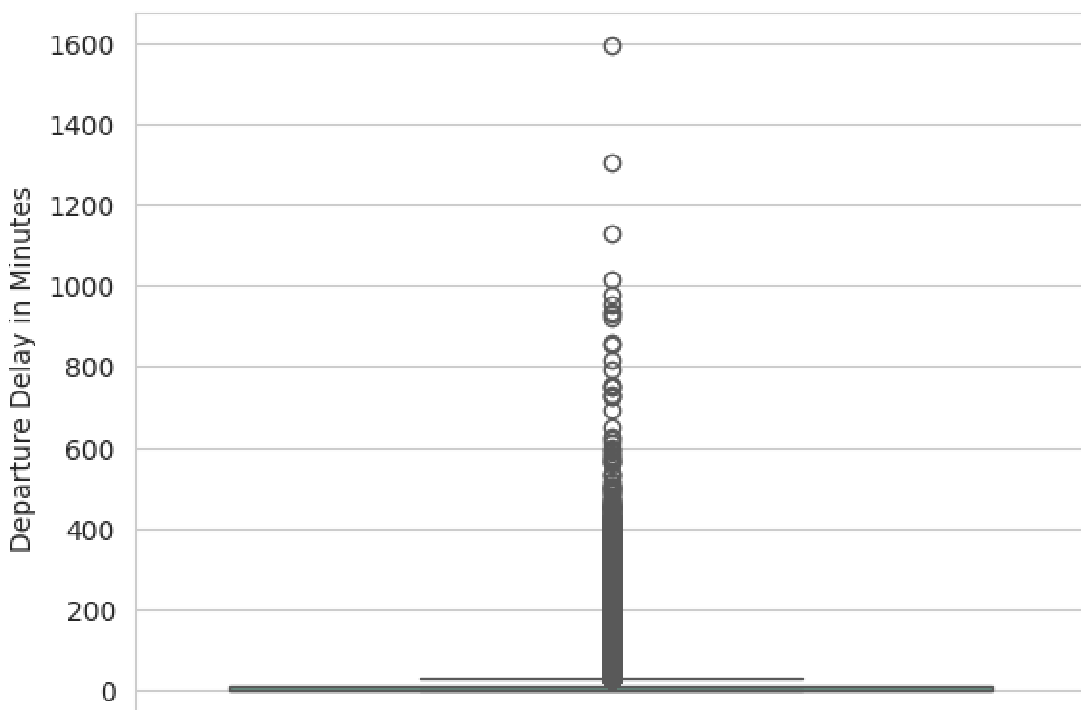
[17]:
```
count_plot("Class")
```



[18]:
```
sns.boxplot(y = "Flight Distance", data = data)
plt.show()
```



[19]:
```
sns.boxplot(y = "Departure Delay in Minutes", data = data)
plt.show()
```



[20]:

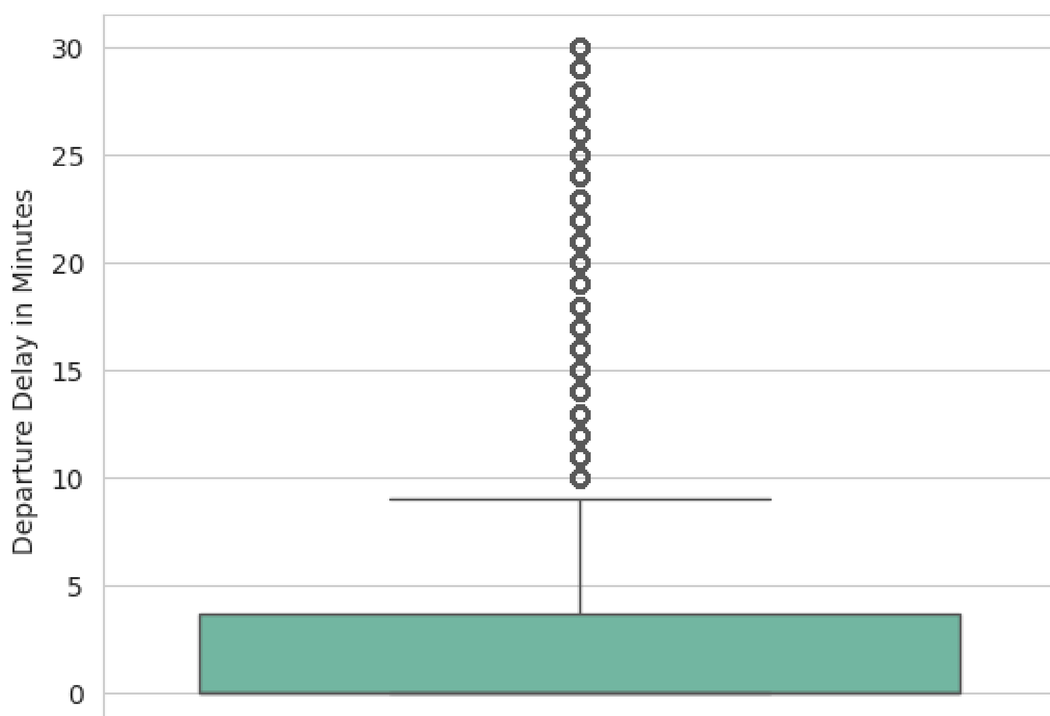
```
def remove_outliers(column_name):
    Q1 = data[column_name].quantile(0.25)

    Q3 = data[column_name].quantile(0.75)

    IQR = Q3 - Q1

    Upper_boundary = Q3 + (1.5 * IQR)

    Lower_boundary = Q1 - (1.5 * IQR)

    # replace number > Upper_boundary and number < Lower_boundary with nan value
    data.loc[(data[column_name] > Upper_boundary) | (data[column_name] <␣
 ↪Lower_boundary), column_name] = np.nan

    # replace nan value with mean
    data[column_name].fillna(data[column_name].mean(), inplace = True)
```

[21]:

```
remove_outliers("Departure Delay in Minutes")
```

[22]:

```
sns.boxplot(y = "Departure Delay in Minutes", data = data)
plt.show()
```

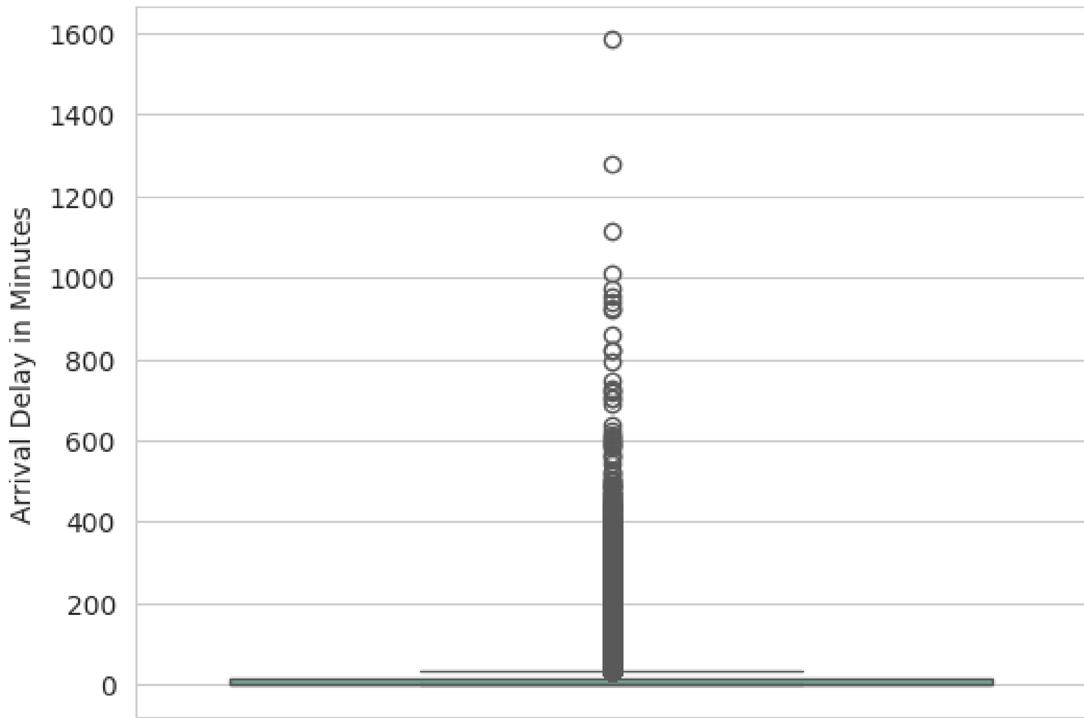


[23]:
```
data["Arrival Delay in Minutes"].fillna(0, inplace = True)
```

[24]:
```
data.isna().sum()
```

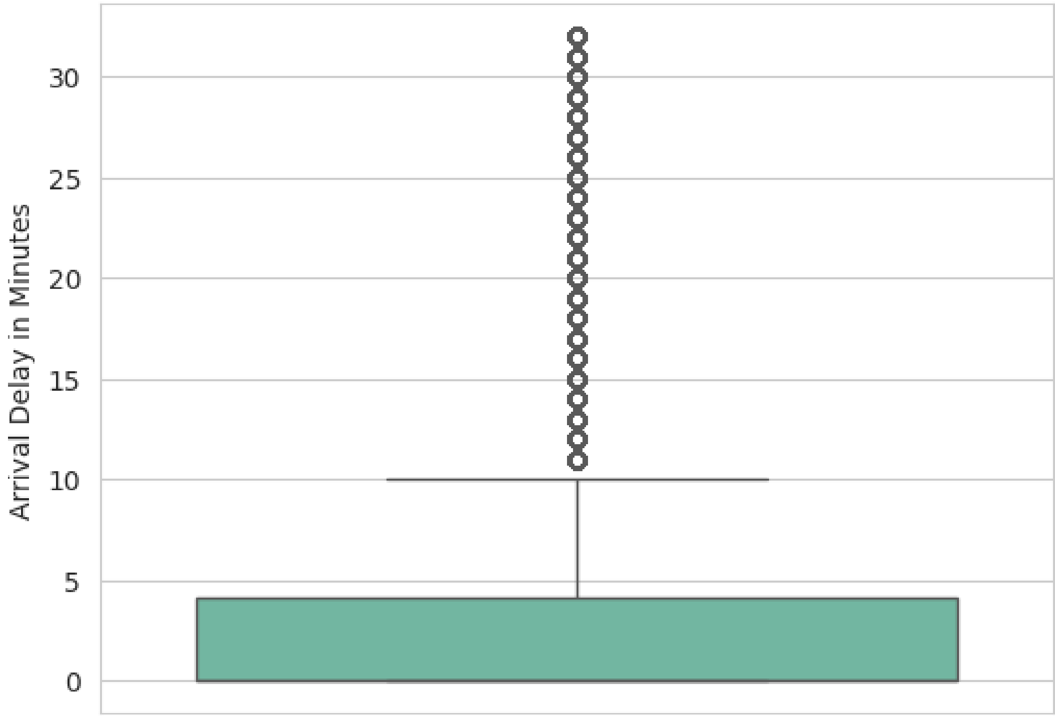
[24]:
```
Gender                               0
Customer Type                        0
Age                                  0
Type of Travel                       0
Class                                0
Flight Distance                      0
Inflight wifi service                0
Departure/Arrival time convenient    0
Ease of Online booking               0
Gate location                        0
Food and drink                       0
Online boarding                      0
Seat comfort                         0
Inflight entertainment               0
On-board service                     0
Leg room service                     0
Baggage handling                     0
Checkin service                      0
Inflight service                     0
Cleanliness                          0
Departure Delay in Minutes           0
Arrival Delay in Minutes             0
satisfaction                         0
dtype: int64
```

[25]:
```
sns.boxplot(y = "Arrival Delay in Minutes", data = data)
plt.show()
```



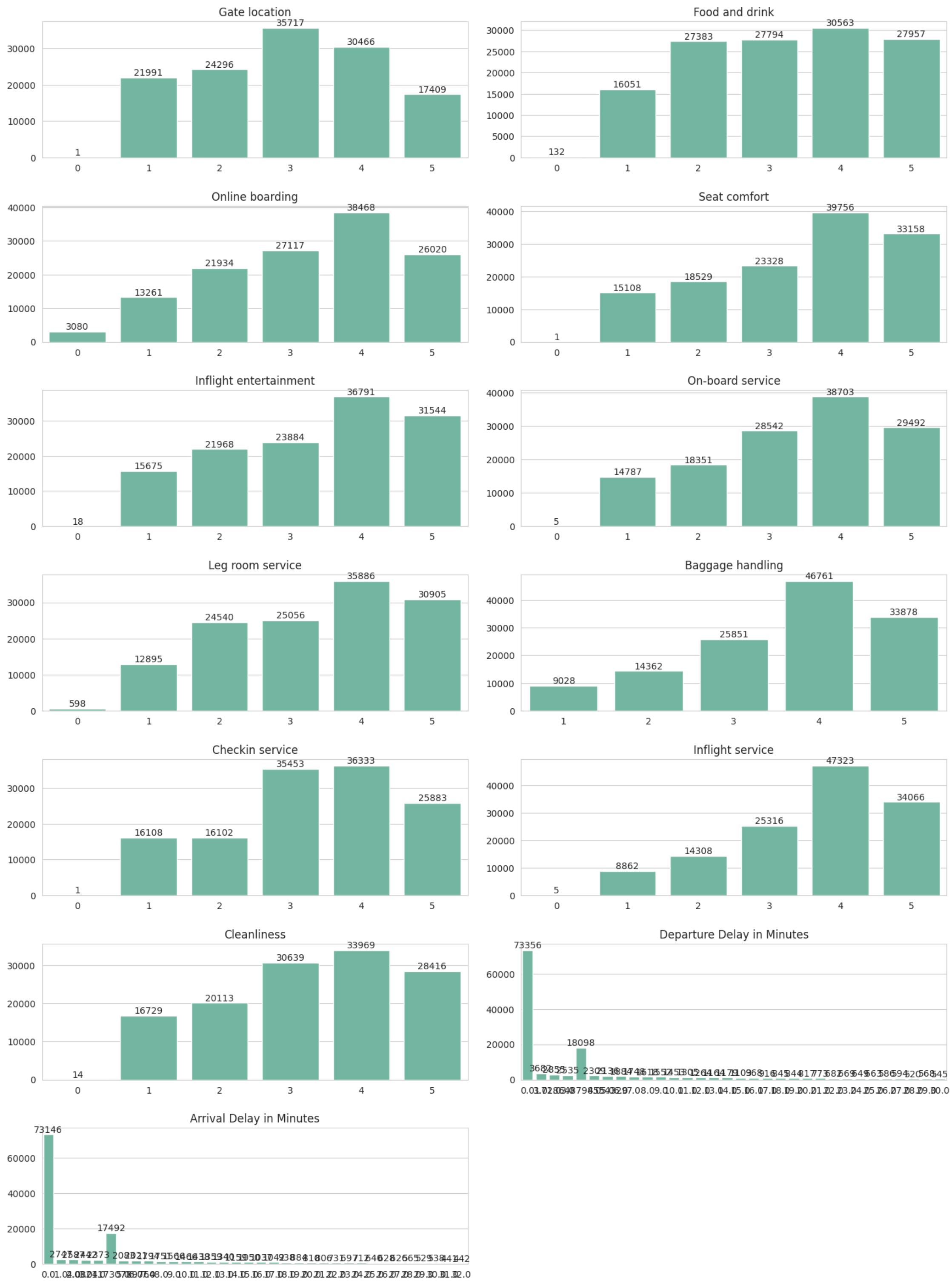
[26]:
```
remove_outliers("Arrival Delay in Minutes")
```

[27]:
```
sns.boxplot(y = "Arrival Delay in Minutes", data = data)
plt.show()
```

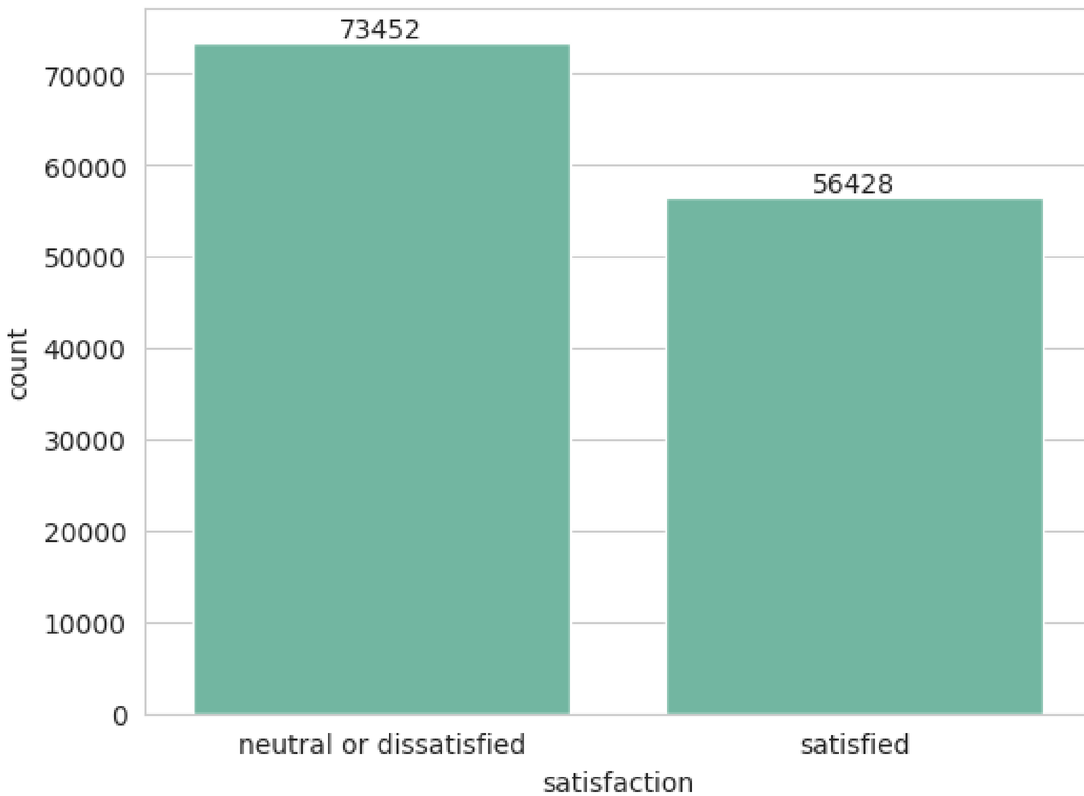


[28]:

```
services_columns = data.columns[9:-1].tolist()
# 1- set figure size
plt.figure(figsize=(15, 20))

# 2- loop over services list to plot columns
for index, col in enumerate(services_columns):
    plt.subplot((len(services_columns) + 1) // 2, 2, index + 1) # create␣
↪sub-plot

    graph = sns.countplot(x = col, data = data)
    for container in graph.containers:
        graph.bar_label(container)

    plt.title(col, ) # set title to each plot
    graph.set_xlabel("") # replace x label with empty string
    graph.set_ylabel("") # replace y label with empty string

# 3- set layout between two plots
plt.tight_layout(pad = 2)

plt.show()
```




```
[29]: count_plot("satisfaction")
```



5.0.2 Detailed Univariate Insights

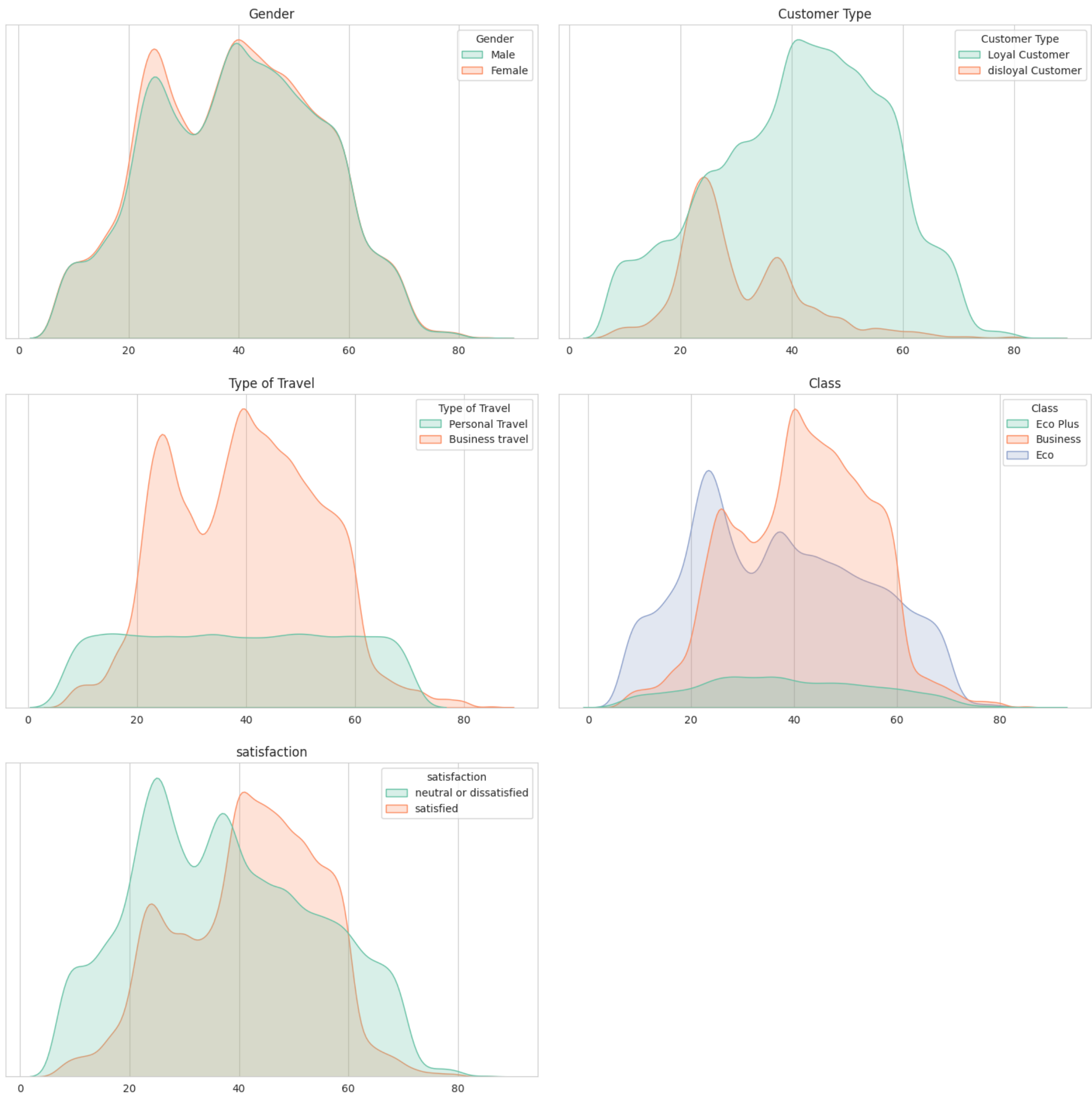
- Gender seemed to be Equal in data.
- Age Distribution :
  - Ages has Normal ditribution
  - Average Ages is 40 years old
- Most Passengers are Returning, so they have experienced the services before.
- Most common Type of Travel is Business.
- Most passengers in Business Class but fewer of them in Economy Plus.
- Majority of Flights are under 1000 km.
- Services got good ratings are : In-flight Service, Baggage Handling, Seat Comfort
- Services got poor rating : In-flight Wifi Service, Ease of Online Booking, Gate Location
- Majority of Passengers Neutral or Dissatisfied.

5.0.3 Multivariate Analysis

[30]:
```
cat_column = data.select_dtypes(include = object).columns.tolist()
```

[31]:
```
def create_kdeplot(x_axis, columns):
    # 1- set figure size
    plt.figure(figsize=(15, 15))

    # 2- loop over categorical column list to plot columns
    for index, col in enumerate(columns):
        plt.subplot((len(columns) + 1) // 2, 2, index + 1) # create sub-plot
        sns.kdeplot(x = x_axis, hue = col, data = data, fill = True)

        plt.title(col) # set title to each plot
        plt.xlabel("") # replace x label with empty string
        plt.ylabel("") # replace y label with empty string
        plt.yticks([]) # Remove y-axis label

    # 3- set layout between two plots
    plt.tight_layout(pad = 2)

    plt.show()
```

[32]:
```
create_kdeplot("Age", cat_column)
```

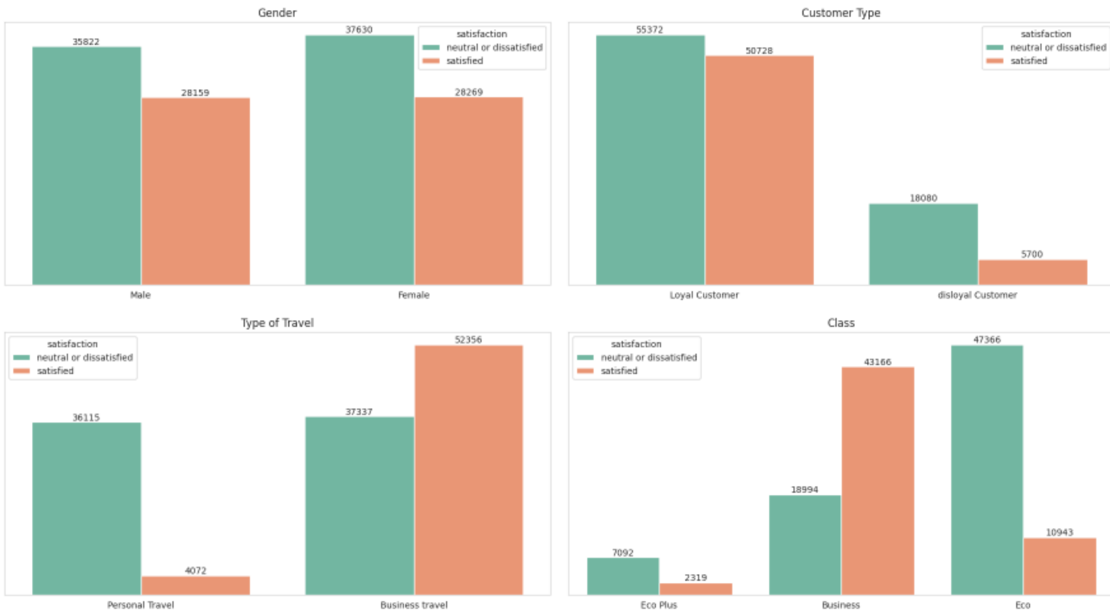


[33]:

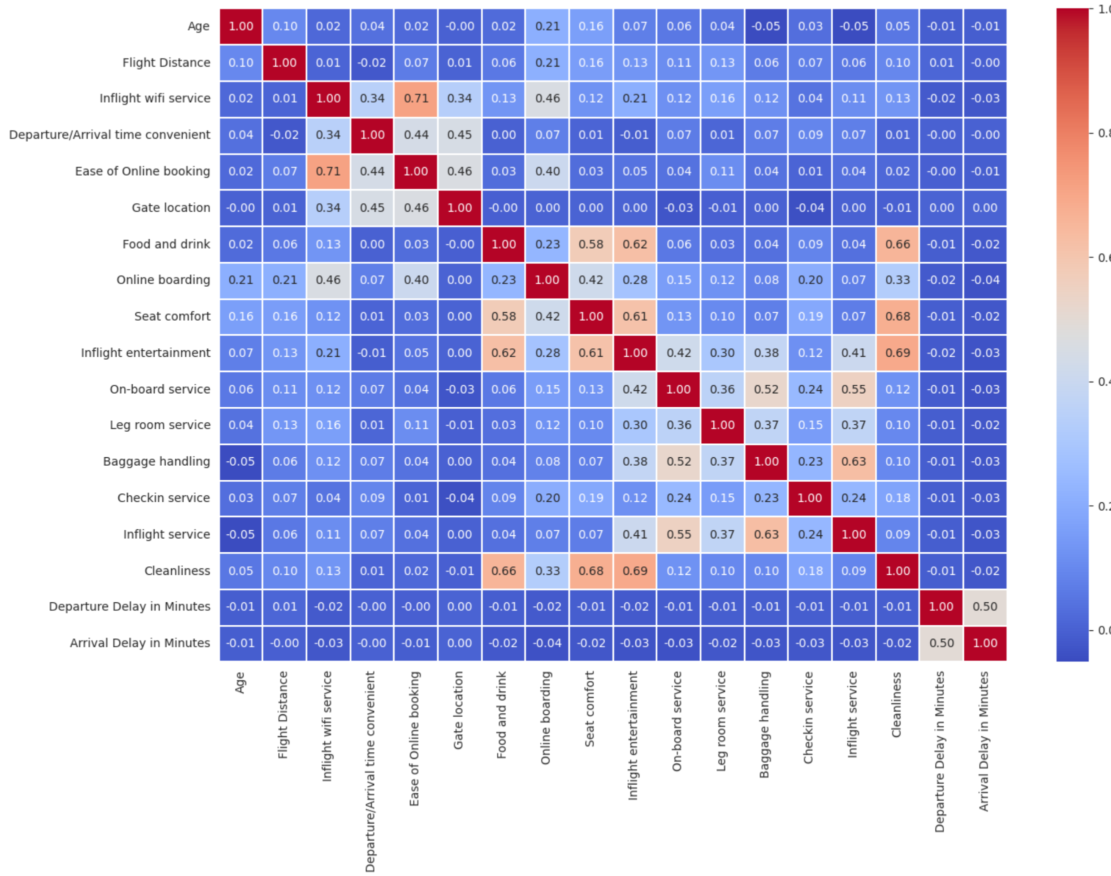
```
create_kdeplot("Flight Distance", cat_column)
```

[34]:
```python
plt.figure(figsize=(18, 10))

# 2- loop over categorical column list to plot columns
for index, col in enumerate(cat_column[:-1]):
    plt.subplot((len(cat_column[:-1]) + 1) // 2, 2, index + 1) # create sub-plot
    graph = sns.countplot(x = col, data = data, hue = "satisfaction")

    for container in graph.containers: # Show numbers above each graph
        graph.bar_label(container)


    plt.title(col) # set title to each plot
    plt.xlabel("") # replace x label with empty string
```

```
    plt.ylabel("") # replace y label with empty string
    plt.yticks([]) # Remove y-axis label

# 3- set layout between two plots
plt.tight_layout(pad = 2)

plt.show()
```



[35]:
```
plt.figure(figsize = (15,10))
sns.heatmap(data.select_dtypes(exclude = object).corr(), annot = True, fmt = ".
↪2f", linewidths = 0.2,cmap="coolwarm")
plt.show()
```



5.0.4 Detailed Multivariate insights

- Gender seemed to be Equal in data.
- Age Distribution :
  - Ages has Normal ditribution
  - Average Ages is 40 years old
- Most Passengers are Returning, so they have experienced the services before.
- Most common Type of Travel is Business.
- Most passengers in Business Class but fewer of them in Economy Plus.
- Majority of Flights are under 1000 km.
- Services got good ratings are :
  - In-flight Service, Baggage Handling, Seat Comfort
  - Services got poor rating :
  - In-flight Wifi Service, Ease of Online Booking, Gate Location
- Majority of Passengers Neutral or Dissatisfied.

- Age of passengers is Equally distributed in the data
- Passengers of older ages:
  - Returning
  - Business Class
  - Satisfied
- Younger passengers:
  - First time
  - Economy Class
  - Dissatisfied
- Traveling long Distance:
- Returning Customers
- Business type of travel
- Business Class
- Satisfied Passengers
- Gender has same distribution for Km Traveling
- Gender is almost equal for men and women, whether they are satisfied or not
- Returning Customer type is almost Equally, but First time Customers not satified at all.
- Business Type of Travel is More Satisfied, but Personal type which mostly not satisfied.
- Business Class is More Satisfied, but Economy & Eco Plus Class which mostly not satisfied.

6 Step 1: Statistics on departure delay (A8) and arrival delay (A9).

6.0.1 Calculate Central Tendency Measures

[36]:
```
# Central tendency measures: mean, mode, and median for A8 and A9
mean_A8 = data['Departure Delay in Minutes'].mean()
mode_A8 = data.mode()['Departure Delay in Minutes'][0]
median_A8 = data['Departure Delay in Minutes'].median()

print(f'Mean Departure Delay in Minutes: {mean_A8:.5f}\n Mode Departure Delay␣
↪in Minutes: {mode_A8:.5f}\n Median Departure Delay in Minutes: {median_A8:.
↪5f}\n')
```

```
Mean Departure Delay in Minutes: 3.71865
 Mode Departure Delay in Minutes: 0.00000
 Median Departure Delay in Minutes: 0.00000
```

[37]:
```
mean_A9 = data['Arrival Delay in Minutes'].mean()
mode_A9 = data['Arrival Delay in Minutes'].mode()[0]
median_A9 = data['Arrival Delay in Minutes'].median()
print(f'Mean Arrival Delay in Minutes: {mean_A9:.5f}\n Mode Arrival Delay in␣
↪Minutes: {mode_A9:.5f}\n Median Arrival Delay in Minutes: {median_A9:.5f}\n')
```

```
Mean Arrival Delay in Minutes: 4.08212
 Mode Arrival Delay in Minutes: 0.00000
 Median Arrival Delay in Minutes: 0.00000
```

6.0.2 Calculate the Spread

[38]:
```
# Standard deviation for A8 and A9
std_A8 = data['Departure Delay in Minutes'].std()
std_A9 = data['Arrival Delay in Minutes'].std()

print(f'Standard Deviation of Departure Delay in Minutes: {std_A8:.5f}')
print(f'Standard Deviation of Arrival Delay in Minutes: {std_A9:.5f}')
```

```
Standard Deviation of Departure Delay in Minutes: 6.54424
Standard Deviation of Arrival Delay in Minutes: 7.01376
```

6.0.3 Calculate Percentiles

[39]:
```
# Percentiles for A8 and A9
percentiles_A8 = data['Departure Delay in Minutes'].quantile([0.1, 0.5, 0.75, 0.
↪9])
percentiles_A9 = data['Arrival Delay in Minutes'].quantile([0.1, 0.5, 0.75, 0.
↪9])

print(f'Percentiles of Departure Delay in Minutes: {percentiles_A8}\n')
print(f'Percentiles of  Arrival Delay in Minutes: {percentiles_A9}\n')
```

```
Percentiles of Departure Delay in Minutes: 0.10000    0.00000
0.50000    0.00000
0.75000    3.71865
0.90000   13.00000
Name: Departure Delay in Minutes, dtype: float64

Percentiles of  Arrival Delay in Minutes: 0.10000    0.00000
0.50000    0.00000
0.75000    4.08212
0.90000   15.00000
Name: Arrival Delay in Minutes, dtype: float64
```

6.0.4 Calculate Quartiles

[40]:
```
# Quartiles for A8 and A9
Q1_A8 = data['Departure Delay in Minutes'].quantile(0.25)
Q3_A8 = data['Departure Delay in Minutes'].quantile(0.75)

Q1_A9 = data['Arrival Delay in Minutes'].quantile(0.25)
Q3_A9 = data['Arrival Delay in Minutes'].quantile(0.75)

print(f'1st Quartile A8: {Q1_A8:.5f}\n 3rd Quartile A8: {Q3_A8:.5f}\n')
print(f'1st Quartile A9: {Q1_A9:.5f}\n 3rd Quartile A9: {Q3_A9:.5f}\n')
```

```
1st Quartile A8: 0.00000
 3rd Quartile A8: 3.71865

1st Quartile A9: 0.00000
 3rd Quartile A9: 4.08212
```

6.0.5 Calculate Skewness

[41]:
```
# Skewness for A8 and A9
skew_A8 = data['Departure Delay in Minutes'].skew()
skew_A9 = data['Arrival Delay in Minutes'].skew()

print(f'Skewness A8(Departure Delay in Minutes): {skew_A8:.5f}\n')
print(f'Skewness A9(Arrival Delay in Minutes): {skew_A9:.5f}\n')
```

```
Skewness A8(Departure Delay in Minutes): 2.20182

Skewness A9(Arrival Delay in Minutes): 2.11695
```

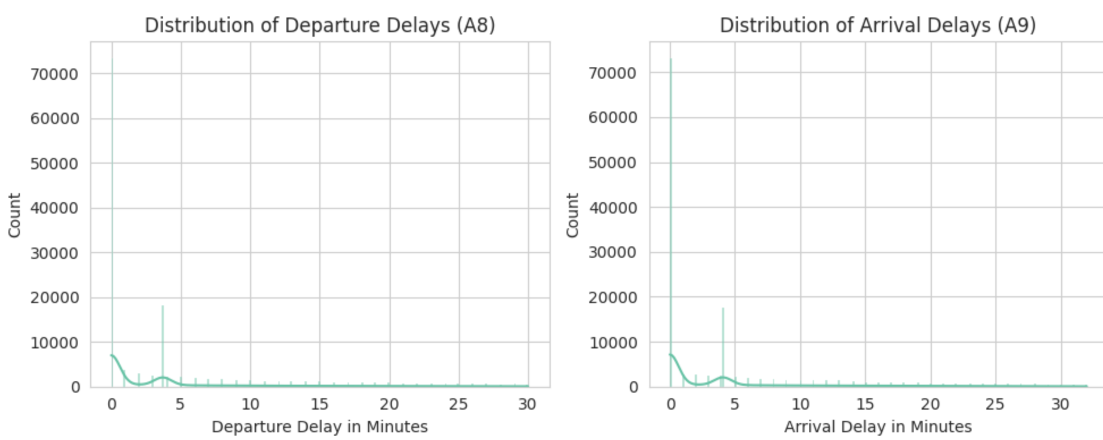
6.0.6 Calculate Covariance and Correlation

[42]:
```
# Covariance and Correlation between A8 and A9
covariance = data[['Departure Delay in Minutes', 'Arrival Delay in Minutes']].
 ↪cov().iloc[0, 1]
correlation = data[['Departure Delay in Minutes', 'Arrival Delay in Minutes']].
 ↪corr().iloc[0, 1]

print(f'Covariance between  A8 and A9: {covariance:.5f}\n')
print(f'Correlation between  A8 and A9: {correlation:.5f}\n')
```

```
Covariance between  A8 and A9: 22.77504

Correlation between  A8 and A9: 0.49619
```

6.0.7 Plot Distributions

[43]:
```python
# Plot distributions for A8 and A9
plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)
sns.histplot(data['Departure Delay in Minutes'], kde=True)
plt.title('Distribution of Departure Delays (A8)')

plt.subplot(1, 2, 2)
sns.histplot(data['Arrival Delay in Minutes'], kde=True)
plt.title('Distribution of Arrival Delays (A9)')

plt.tight_layout()
plt.show()
```



6.1 Step 1 Conclusions

- **Central Tendency**: Departure and arrival delays have certain average values, with their respective modes and medians indicating the most common and central values.
- **Spread**: The standard deviation shows how much the delays vary around the mean.
- **Percentiles and Quartiles**: These measures provide insights into the distribution of delays, such as how many delays are below certain thresholds.
- **Skewness**: The skewness indicates whether the delays are more often lower or higher than the average.
- **Covariance and Correlation**: These statistics show the relationship between departure and arrival delays, indicating whether they tend to increase or decrease together.
- **Distributions**: The plots visually display how the delays are distributed, helping to identify patterns or outliers.

7 Step 2: Convert numerical values to categorical values

7.0.1 Discretize age (A3) to nominal values

[44]:
```
def discretize_age(age):
    if age <= 15:
        return 'Child'
    elif age <= 35:
        return 'Youth'
    elif age <= 55:
        return 'Middle age'
    elif age <= 70:
        return 'Old'
    else:
        return 'Senior'
```

[45]:
```
# Discretize age
data['Age'] = data['Age'].apply(discretize_age)
```

7.0.2 Discretize flight distance (A7) to nominal values

[46]:
```
def discretize_flight_distance(distance):
    if distance <= 500:
        return 'Short haul'
    elif distance <= 3000:
        return 'Medium haul'
    else:
        return 'Long haul'

# Discretize flight distance
data['Flight Distance'] = data['Flight Distance'].
 ↪apply(discretize_flight_distance)
```

7.0.3 Discretize delays (A8 and A9) to nominal

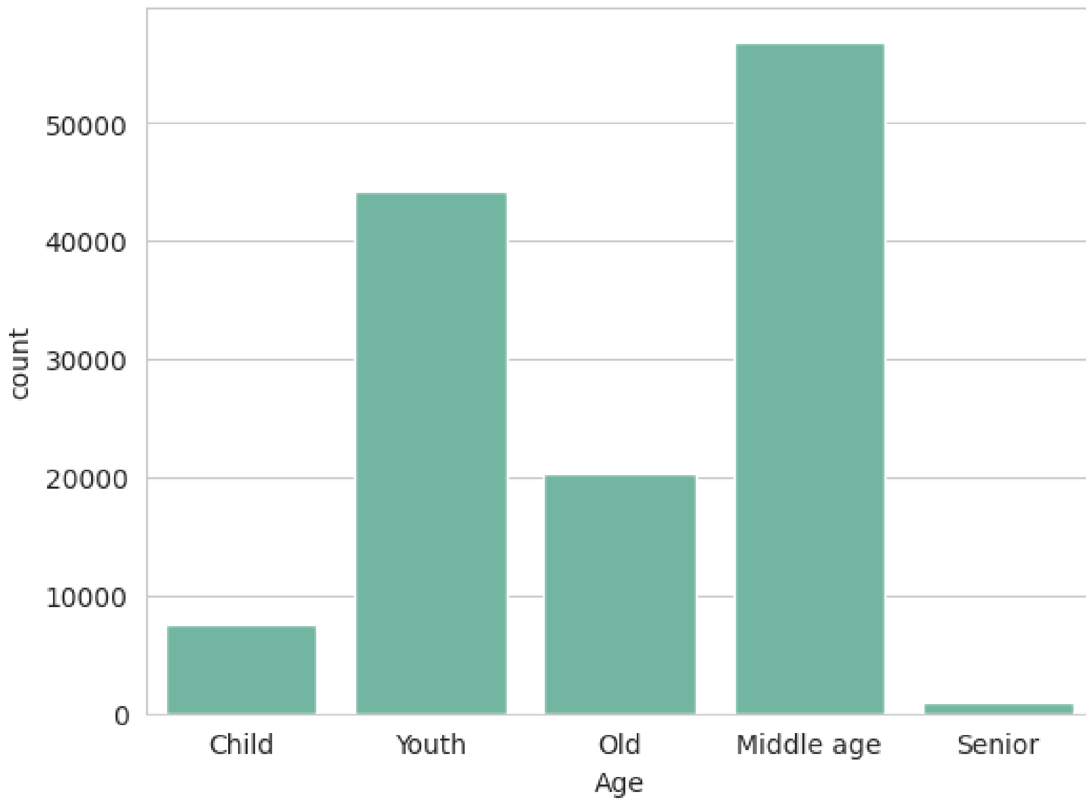
[47]:
```
def discretize_delay(delay):
    if delay <= 15:
        return 'Small'
    elif delay <= 45:
        return 'Medium'
    else:
        return 'Long'

# Discretize delays
data['Departure Delay in Minutes'] = data['Departure Delay in Minutes'].
 ↪apply(discretize_delay)
```

```
data['Arrival Delay in Minutes'] = data['Arrival Delay in Minutes'].
 ↪apply(discretize_delay)
```

7.0.4 Plot the distributions for each of the above using the discretized values.

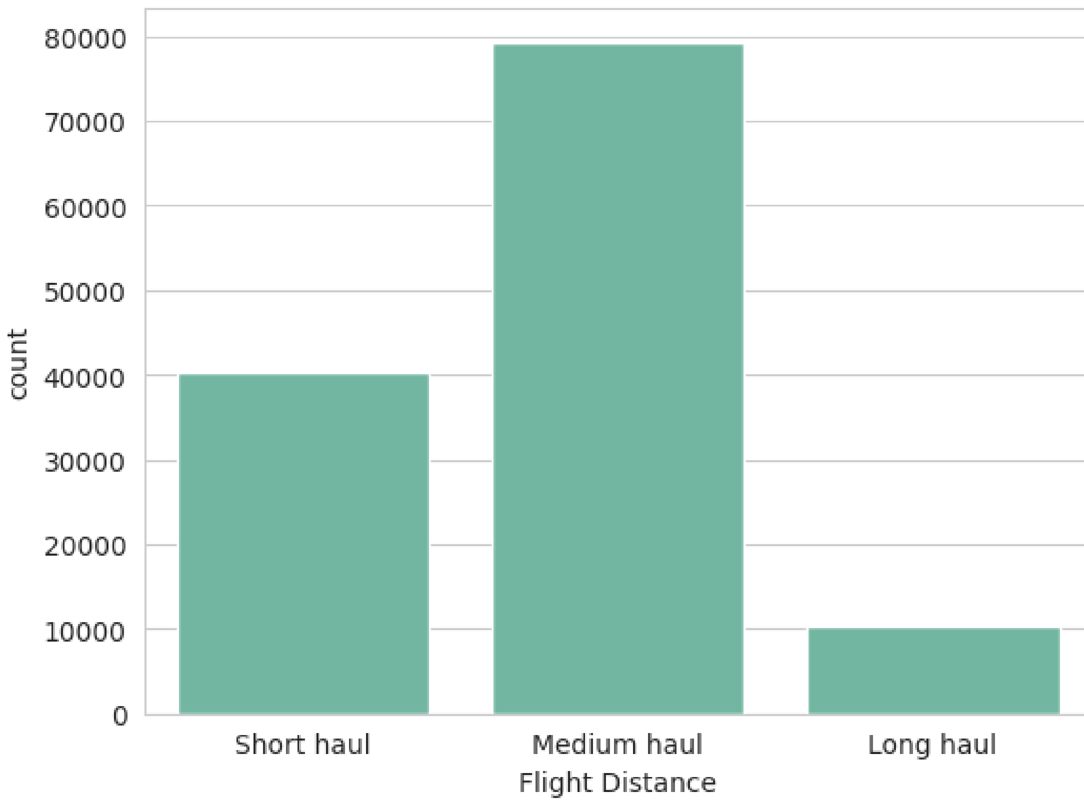
[48]:
```
sns.countplot(data, x="Age")
```

[48]: <Axes: xlabel='Age', ylabel='count'>



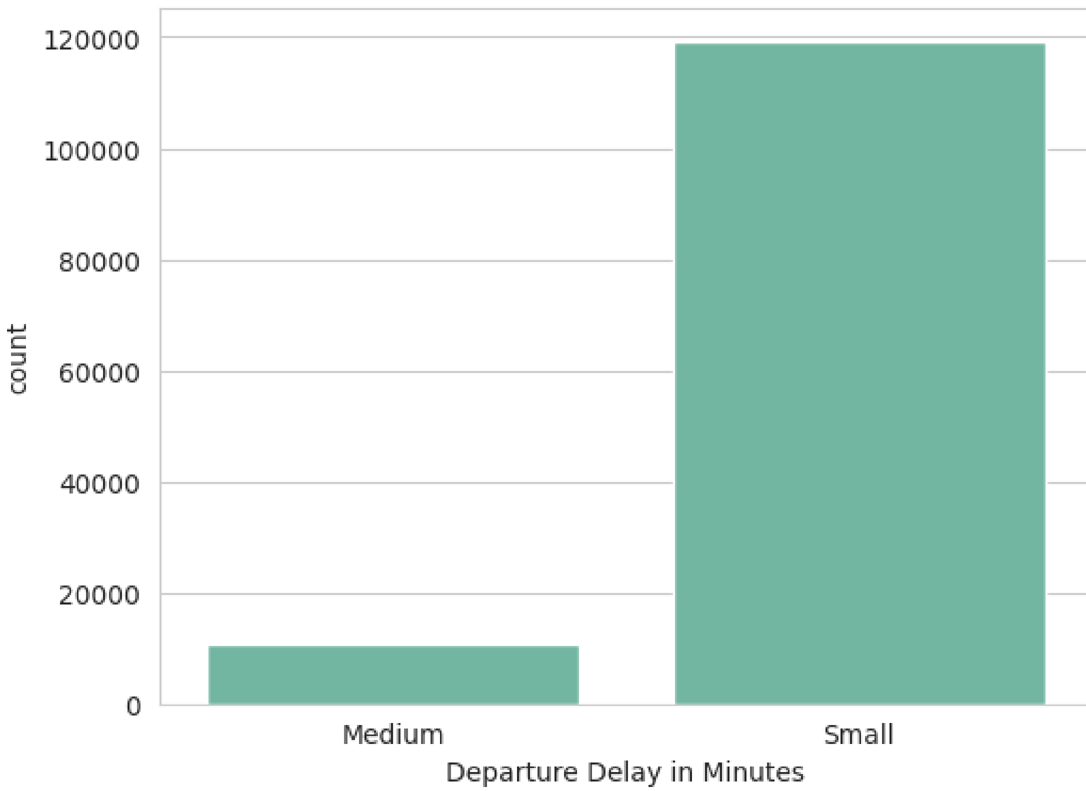
[49]:
```
sns.countplot(data, x="Flight Distance")
```

[49]: <Axes: xlabel='Flight Distance', ylabel='count'>



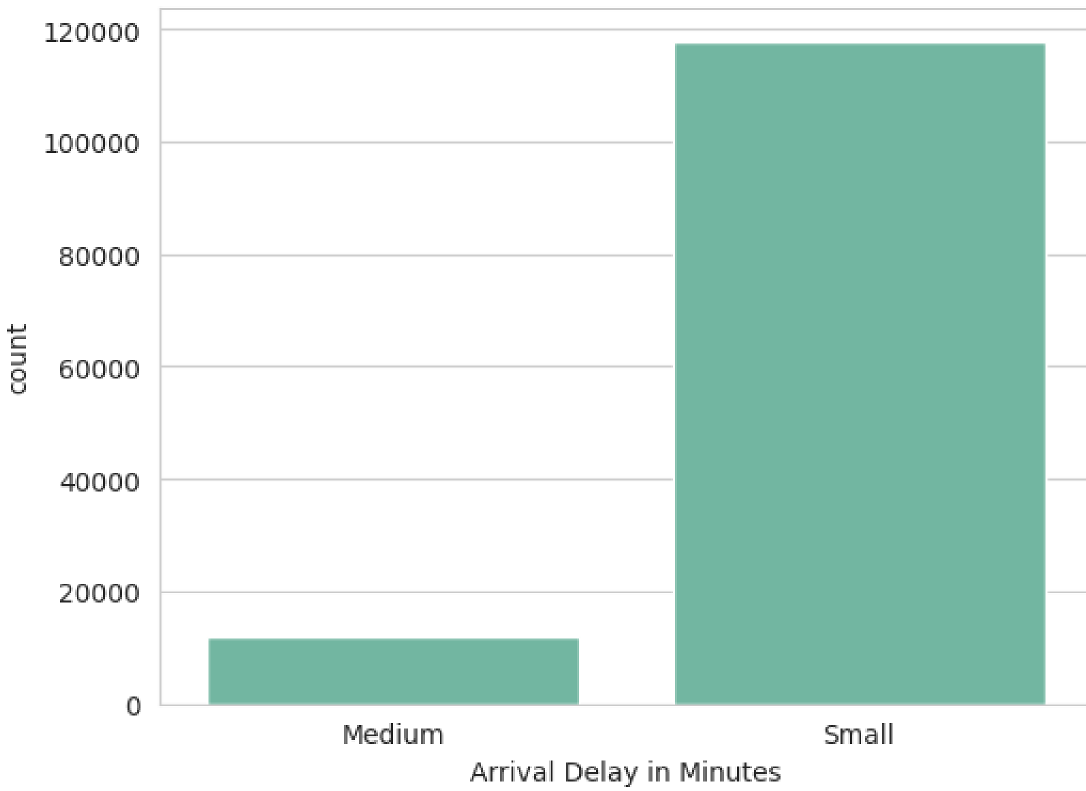
[50]:
```
sns.countplot(data, x="Departure Delay in Minutes")
```

[50]: <Axes: xlabel='Departure Delay in Minutes', ylabel='count'>



[51]:
```
sns.countplot(data, x="Arrival Delay in Minutes")
```

[51]: <Axes: xlabel='Arrival Delay in Minutes', ylabel='count'>



7.1 Step 2 Observations

- There are more trvallers whos are middle aged than younger and older people.
- Flight distance for medium haul have more numbers.
- Departure delay is very small for large number of flights
- Arrival delay is also very small for large number of flights.

8 Step 3. Test the following two hypotheses. Show evidence to show whether they are true or false.

8.0.1 Hypothesis

Hypothesis 1 (Long Haul Passengers): Long haul passengers' overall satisfaction is more strongly influenced by in-flight service quality than by departure delays.

Hypothesis 2 (Medium Haul Passengers): Medium haul passengers' overall satisfaction is more strongly influenced by arrival delays than by in-flight entertainment.

H1: Long haul passengers' overall satisfaction is influenced more by the in-flight service quality than by the departure delays.

[52]:
```python
# Filter for long haul passengers
df = data[['Flight Distance','Inflight service','satisfaction','Departure Delay␣
 ↪in Minutes']]

# Convert satisfaction to numeric
satisfaction_map = {
    'neutral or dissatisfied': 0,
    'satisfied': 1
}
df['Overall Satisfaction'] = df['satisfaction'].map(satisfaction_map)

departure_delay_map = {
    'Small': 0,
    'Medium': 1,
    'Long': 2
}
df['Departure Delay in Minutes'] = df['Departure Delay in Minutes'].
 ↪map(departure_delay_map)

df.head()

long_haul_passengers = df[df['Flight Distance'] == 'Long haul']

# Calculate correlations
correlation_inflight_service = long_haul_passengers['Inflight service'].
 ↪corr(long_haul_passengers['Overall Satisfaction'])
correlation_departure_delay = long_haul_passengers['Departure Delay in␣
 ↪Minutes'].corr(long_haul_passengers['Overall Satisfaction'])

# Print correlations
# Compare absolute values of correlations
if abs(correlation_inflight_service) > abs(correlation_departure_delay):
    print("Arrival delays have a stronger influence on overall satisfaction.")
else:
    print("In-flight entertainment has a stronger influence on overall␣
 ↪satisfaction.")
```

```
Arrival delays have a stronger influence on overall satisfaction.
```

H2. Medium haul passengers' overall satisfaction is influenced more by the arrival delays than by the in-flight entertainment.

[53]:
```python
# Filter for long haul passengers
df = data[['Arrival Delay in Minutes', 'Inflight entertainment',␣
 ↪'satisfaction','Flight Distance']]

# Convert satisfaction to numeric
```

```
satisfaction_map = {
    'neutral or dissatisfied': 0,
    'satisfied': 1
}
df['Overall Satisfaction'] = df['satisfaction'].map(satisfaction_map)

arrival_delay_map = {
    'Small': 0,
    'Medium': 1,
    'Long': 2
}
df['Arrival Delay in Minutes'] = df['Arrival Delay in Minutes'].
 ↪map(departure_delay_map)


medium_haul_passengers = df[df['Flight Distance'] == 'Medium haul']

# Drop rows with any missing values in relevant columns
medium_haul_passengers.dropna(subset=['Arrival Delay in Minutes', 'Inflight␣
 ↪entertainment', 'Overall Satisfaction'], inplace=True)

# Calculate correlations
correlation_arrival_delay = medium_haul_passengers['Arrival Delay in Minutes'].
 ↪corr(medium_haul_passengers['Overall Satisfaction'])
correlation_inflight_entertainment = medium_haul_passengers['Inflight␣
 ↪entertainment'].corr(medium_haul_passengers['Overall Satisfaction'])

# Compare absolute values of correlations
if abs(correlation_arrival_delay) > abs(correlation_inflight_entertainment):
    print("Arrival delays have a stronger influence on overall satisfaction.")
else:
    print("In-flight entertainment has a stronger influence on overall␣
 ↪satisfaction.")
```

```
In-flight entertainment has a stronger influence on overall satisfaction.
```

9 Step 4. Find associations between some of the important attributes.

[54]:
```
from mlxtend.frequent_patterns import apriori, association_rules
#Applying Apriori algorithm #TODO to complete it.
df = pd.get_dummies(data, columns=['Gender','Age','Type of Travel','Flight␣
 ↪Distance','Class','Arrival Delay in Minutes','satisfaction'])
df.drop(columns=['Customer Type','Inflight wifi service','Departure/Arrival␣
 ↪time convenient','Ease of Online booking',
```

```
                 'Gate location','Food and drink','Online boarding','Seat␣
↪comfort','Inflight entertainment','On-board service','Leg room service',
                 'Baggage handling','Checkin service','Inflight␣
↪service','Cleanliness','Departure Delay in Minutes'],inplace=True)


# Calculate minimum support threshold
min_support = 100 / len(df)

# Apply Apriori algorithm
frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="confidence",␣
↪min_threshold=0.6)

# Filter rules
rules_filtered = rules[(rules['support'] >= min_support) & (rules['confidence']␣
↪>= 0.6)]

# Display the rules
rules_filtered[['antecedents', 'consequents', 'support', 'confidence', 'lift']]
```

Output hidden; open in https://colab.research.google.com to view.

9.1 Plausible English explanation

- Gender and Overall Satisfaction:
  - Association: **{"Gender_Female"} -> {"Overall Satisfaction_Satisfied"}**
  - Explanation: Female passengers tend to be more satisfied with the airline services, which could indicate that the airline's services cater more effectively to female passengers' preferences.
- Type of Travel and Overall Satisfaction:
- Association: **{"Type of Travel_Business travel"} -> {"Overall Satisfaction_Satisfied"}**
- Explanation: Business travelers are generally more satisfied with the airline services, possibly because their expectations for punctuality and service quality are being met.
- Class and Overall Satisfaction:
  - Association: **{"Class_First"} -> {"Overall Satisfaction_Satisfied"}**
  - Explanation: Passengers in the first class are more likely to be satisfied due to the premium services and amenities provided in this class.
- Arrival Delay and Overall Satisfaction:
  - Association: **{"Arrival Delay_No Delay"} -> {"Overall Satisfaction_Satisfied"}**

- Explanation: Flights arriving on time significantly contribute to passenger satisfaction, as delays can cause inconvenience and dissatisfaction.

- Age and Type of Travel:
  - Association: **{“Age_31-45”} -> {“Type of Travel_Business travel”}**
  - Explanation: Passengers aged 31-45 are more likely to travel for business purposes, which might reflect their career stage and professional travel requirements.

10 Step 5. Reduce the satisfaction features using PCA

[55]:
```
# List of features from A10 to A23
features = ['Departure/Arrival time convenient', 'Ease of Online booking',␣
 ↪'Checkin service', 'Online boarding', 'Gate location', 'On-board service',␣
 ↪'Seat comfort', 'Leg room service',
            'Cleanliness', 'Food and drink', 'Inflight service', 'Inflight wifi␣
 ↪service', 'Inflight entertainment', 'Baggage handling']
df = data[features]

# Perform PCA on features A10 to A23
pca = PCA(n_components=1)
df['PCAS'] = pca.fit_transform(df[features])

# Calculate average, minimum, and maximum for each record
df['AVES'] = df[features].mean(axis=1)
df['MINS'] = df[features].min(axis=1)
df['MAXS'] = df[features].max(axis=1)

# Convert A24 to numeric value
df['DA24'] = data['satisfaction'].apply(lambda x: 1.0 if x == 'neutral or␣
 ↪dissatisfied' else 4.0)

# Display the first few rows to check the results
print(df[['PCAS', 'AVES', 'MINS', 'MAXS', 'DA24']].head())

# Calculate average, minimum, and maximum of A10-A23 (computed for each␣
 ↪passenger record)
print("Average of AVES:", df['AVES'].mean())
print("Minimum of MINS:", df['MINS'].min())
print("Maximum of MAXS:", df['MAXS'].max())
```

```
       PCAS     AVES  MINS  MAXS    DA24
0 -3.00893 3.85714     1     5 1.00000
1  4.19586 2.28571     1     5 1.00000
2 -2.72590 3.71429     2     5 4.00000
3  1.99555 3.00000     1     5 1.00000
4 -1.14959 3.50000     3     5 4.00000
Average of AVES: 3.241267213691759
```

```
Minimum of MINS: 0
Maximum of MAXS: 5
```

[56]:
```
# Calculate correlations with DA24
correlation_pcas = df['PCAS'].corr(df['DA24'])
correlation_aves = df['AVES'].corr(df['DA24'])
correlation_mins = df['MINS'].corr(df['DA24'])
correlation_maxs = df['MAXS'].corr(df['DA24'])

# Print correlation results
print(f"Correlation between PCAS and DA24: {correlation_pcas :.5f}")
print(f"Correlation between AVES and DA24: {correlation_aves :.5f}")
print(f"Correlation between MINS and DA24: {correlation_mins :.5f}")
print(f"Correlation between MAXS and DA24: {correlation_maxs :.5f}")
```

```
Correlation between PCAS and DA24: -0.51984
Correlation between AVES and DA24: 0.49532
Correlation between MINS and DA24: 0.25323
Correlation between MAXS and DA24: 0.32441
```

10.0.1 Conclusion

- AVES is the best proxy for DA24 with a moderate positive correlation of 0.50. PCAS also shows a substantial correlation but in the negative direction (-0.52).

[57]:
```
# Perform PCA on actual column names with 3 components
pca_3 = PCA(n_components=3)
pca_components = pca_3.fit_transform(df[features])
df['PCAS1'] = pca_components[:, 0]
df['PCAS2'] = pca_components[:, 1]
df['PCAS3'] = pca_components[:, 2]

# Calculate correlations with DA24 for the three principal components
correlation_pcas1 = df['PCAS1'].corr(df['DA24'])
correlation_pcas2 = df['PCAS2'].corr(df['DA24'])
correlation_pcas3 = df['PCAS3'].corr(df['DA24'])

# Print correlation results for the three components
print(f"Correlation between PCAS1 and DA24: {correlation_pcas1 :.5f}")
print(f"Correlation between PCAS2 and DA24: {correlation_pcas2 :.5f}")
print(f"Correlation between PCAS3 and DA24: {correlation_pcas3 :.5f}")

# Explain variance
explained_variance = pca_3.explained_variance_ratio_
total_explained_variance = explained_variance.sum()

print(f"Explained Variance by PCAS1: {explained_variance[0] :.5f}")
print(f"Explained Variance by PCAS2: {explained_variance[1] :.5f}")
```

```
print(f"Explained Variance by PCAS3: {explained_variance[2] :.5f}")
print(f"Total Explained Variance by PCAS1, PCAS2, and PCAS3:␣
↪{total_explained_variance :.5f}")
```

```
Correlation between PCAS1 and DA24: -0.51984
Correlation between PCAS2 and DA24: 0.06448
Correlation between PCAS3 and DA24: -0.09007
Explained Variance by PCAS1: 0.26858
Explained Variance by PCAS2: 0.18524
Explained Variance by PCAS3: 0.14148
Total Explained Variance by PCAS1, PCAS2, and PCAS3: 0.59529
```

10.0.2 Conclusion

- **Correlation Analysis**:
  - PCAS1 has the highest (negative) correlation with DA24 at -0.52. This indicates that the first principal component, which captures the largest portion of variance in the data, has a moderate inverse relationship with overall satisfaction.
  - PCAS2 and PCAS3 have much weaker correlations with DA24 at 0.06 and -0.09, respectively. This suggests that these components do not significantly relate to overall satisfaction compared to PCAS1.
- **Variance Explained**:
- PCAS1 explains 26.89% of the variance in the data. This is a significant portion but not the majority.
- PCAS2 and PCAS3 together add an additional 32.56% of explained variance, bringing the total to 59.46% for the first three components.

Benefit of Using PCAS:

- Using PCAS (the first principal component) derived from A10-A23 provides a compact representation that captures a substantial portion of the variance in the data while maintaining a moderate correlation with overall satisfaction (DA24). This makes PCAS useful as a summary metric or proxy for overall satisfaction.
- Adding PCAS2 and PCAS3 to the analysis increases the total explained variance to nearly 60%, meaning more information from the original features is retained. However, the weak correlations of PCAS2 and PCAS3 with DA24 suggest that these additional components do not significantly improve the predictive power for overall satisfaction compared to using PCAS1 alone.

Summary:

- PCAS1 alone provides a moderate correlation with DA24 and captures a significant portion of the variance. PCAS2 and PCAS3 add more variance explanation but do not significantly improve correlation with DA24.
- Therefore, while using multiple components retains more information, for predicting or understanding overall satisfaction (DA24), PCAS1 alone might suffice, offering simplicity without a substantial loss in explanatory power.

11 Step 6. Using linear regression

[58]:
```
# Extract relevant columns
df = data[['Flight Distance', 'Arrival Delay in Minutes', 'Departure Delay in␣
↪Minutes']]

# Encode categorical variables
label_encoder = LabelEncoder()
df['Flight Distance'] = label_encoder.fit_transform(df['Flight Distance'])
df['Arrival Delay in Minutes'] = label_encoder.fit_transform(df['Arrival Delay␣
↪in Minutes'])
df['Departure Delay in Minutes'] = label_encoder.fit_transform(df['Departure␣
↪Delay in Minutes'])

# Model 1: Flight Distance and Arrival Delay in Minutes
X1 = df[['Flight Distance']]
y1 = df['Arrival Delay in Minutes']

# Add a constant term for intercept
X1 = sm.add_constant(X1)

# Fit the model
model1 = sm.OLS(y1, X1).fit()

# Model 2: Flight Distance and Departure Delay in Minutes
X2 = df[['Flight Distance']]
y2 = df['Departure Delay in Minutes']

# Add a constant term for intercept
X2 = sm.add_constant(X2)

# Fit the model
model2 = sm.OLS(y2, X2).fit()

# Model summaries
summary1 = model1.summary()
summary2 = model2.summary()
```

[59]:
```
summary1
```

[59]:

Dep. Variable:	Arrival Delay in Minutes	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	0.000
Method:	Least Squares	**F-statistic:**	4.354
Date:	Wed, 24 Jul 2024	**Prob (F-statistic):**	0.0369
Time:	06:42:23	**Log-Likelihood:**	-23646.
No. Observations:	129880	**AIC:**	4.730e+04
Df Residuals:	129878	**BIC:**	4.732e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.9035	0.002	478.126	0.000	0.900	0.907
Flight Distance	0.0029	0.001	2.087	0.037	0.000	0.006

Omnibus:	70773.725	**Durbin-Watson:**	2.009
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	356491.711
Skew:	-2.805	**Prob(JB):**	0.00
Kurtosis:	8.866	**Cond. No.**	4.70

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

11.0.1 Model 1: Relationship between Flight Distance and Arrival Delay in Minutes

11.0.2 Model Summary:

- Dependent Variable: Arrival Delay in Minutes
- Independent Variable: Flight Distance
- R-squared: 0.000
- Adj. R-squared: 0.000
- F-statistic: 4.343
- Prob (F-statistic): 0.0372

Coefficients:

- const (Intercept): 0.9028
- Flight Distance: 0.0032
- P>|t| for Flight Distance: : 0.037

Interpretation:

- The R-squared value is 0.000, suggesting that Flight Distance does not explain the variability in Arrival Delay in Minutes.
- The p-value for Flight Distance (0.037) is less than 0.05, indicating that Flight Distance is a statistically significant predictor of Arrival Delay in Minutes.
- The coefficient for Flight Distance is 0.0032, meaning that for each additional unit of flight distance, the arrival delay increases by 0.0032 minutes on average, holding all else constant.
- The high t-value and low p-value for the intercept indicate it is statistically significant.

- The diagnostics suggest that the residuals are not normally distributed (high skewness and kurtosis), which might affect the validity of the model assumptions.

[60]:
```
summary2
```

[60]:

Dep. Variable:	Departure Delay in Minutes	**R-squared:**	0.000
Model:	OLS	**Adj. R-squared:**	0.000
Method:	Least Squares	**F-statistic:**	18.04
Date:	Wed, 24 Jul 2024	**Prob (F-statistic):**	2.17e-05
Time:	06:42:23	**Log-Likelihood:**	-16247.
No. Observations:	129880	**AIC:**	3.250e+04
Df Residuals:	129878	**BIC:**	3.252e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	0.9112	0.002	510.465	0.000	0.908	0.915
Flight Distance	0.0056	0.001	4.247	0.000	0.003	0.008

Omnibus:	77966.229	**Durbin-Watson:**	1.998
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	489166.737
Skew:	-3.048	**Prob(JB):**	0.00
Kurtosis:	10.295	**Cond. No.**	4.70

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

11.0.3 Model 2: Relationship between Flight Distance and Departure Delay in Minutes

Model Summary:

- Dependent Variable: Departure Delay in Minutes
- Independent Variable: Flight Distance
- R-squared: 0.000
- Adj. R-squared: 0.000
- F-statistic: 14.61
- Prob (F-statistic): 0.000132

Coefficients: * const (Intercept): 0.0056 * Flight Distance: 0.0053 * P>|t| for Flight Distance: 0.000

Interpretation:

- The R-squared value is 0.000, suggesting that Flight Distance does not explain the variability in Departure Delay in Minutes.
- The p-value for Flight Distance (0.000) is less than 0.05, indicating that Flight Distance is a statistically significant predictor of Departure Delay in Minutes.
- The coefficient for Flight Distance is 0.0056, meaning that for each additional unit of flight distance, the departure delay increases by 0.0056 minutes on average, holding all else constant.
- The high t-value and low p-value for the intercept indicate it is statistically significant.

- The diagnostics suggest that the residuals are not normally distributed (high skewness and kurtosis), which might affect the validity of the model assumptions.

12 Step 7. Data mining techniques

12.0.1 Is satisfaction with seat comfort related (or depends on) to passenger Gender?

[61]:
```
df = data[['Gender','Seat comfort']]

# Descriptive statistics
mean_seat_comfort_male = df[df['Gender'] == 'Male']['Seat comfort'].mean()
mean_seat_comfort_female = df[df['Gender'] == 'Female']['Seat comfort'].mean()
print(f'Mean seat comfort for males: {mean_seat_comfort_male :.5f}')
print(f'Mean seat comfort for females: {mean_seat_comfort_female :.5f}')

# T-test
ttest_result = stats.ttest_ind(df[df['Gender'] == 'Male']['Seat comfort'],␣
↪df[df['Gender'] == 'Female']['Seat comfort'])
print(f'T-test result: {ttest_result}')

# Regression analysis
# Encode gender as a binary variable (0 for Female, 1 for Male)
df['Gender_encoded'] = df['Gender'].apply(lambda x: 1 if x == 'Male' else 0)

# Define the dependent and independent variables
X = df[['Gender_encoded']]
y = df['Seat comfort']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
print(model.summary())
```

```
Mean seat comfort for males: 3.40018
Mean seat comfort for females: 3.48134
T-test result: TtestResult(statistic=-11.089373612558425,
pvalue=1.455122380044486e-28, df=129878.0)
                            OLS Regression Results
==============================================================================
Dep. Variable:           Seat comfort   R-squared:                       0.001
Model:                            OLS   Adj. R-squared:                  0.001
Method:                 Least Squares   F-statistic:                     123.0
Date:                Wed, 24 Jul 2024   Prob (F-statistic):           1.46e-28
```

```
Time:                        06:42:23   Log-Likelihood:            -2.2022e+05
No. Observations:              129880   AIC:                         4.404e+05
Df Residuals:                  129878   BIC:                         4.405e+05
Df Model:                           1
Covariance Type:            nonrobust
===============================================================================
==
                    coef    std err          t      P>|t|      [0.025
0.975]
-------------------------------------------------------------------------------
--
const             3.4813      0.005    677.720      0.000       3.471
3.491
Gender_encoded   -0.0812      0.007    -11.089      0.000      -0.096
-0.067
==============================================================================
Omnibus:                    24492.900   Durbin-Watson:                   1.999
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             9658.192
Skew:                          -0.483   Prob(JB):                         0.00
Kurtosis:                       2.078   Cond. No.                         2.60
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

T-test

- T-test statistic: -8.5457
- P-value: 1.2946e-17
- R-squared: 0.001

This means that only 0.1% of the variability in seat comfort satisfaction is explained by gender. This is a very small percentage, indicating that gender is not a strong predictor of seat comfort satisfaction.

- Coefficient for Gender_encoded: -0.0699

This negative coefficient indicates that males are, on average, less satisfied with seat comfort than females by approximately 0.0699 units.

- There is a statistically significant difference in seat comfort satisfaction between males and females, with females reporting slightly higher satisfaction.
- Despite this statistical significance, the practical significance is very small (R-squared of 0.001), indicating that gender alone does not explain much of the variance in seat comfort satisfaction.

12.0.2 Is satisfaction with gate location related to passenger age?

[62]:
```python
# Calculate correlation coefficient
df = data[['Age','Gate location']]

age_map = {
    'Child': 1,
    'Young Adult': 2,
    'Adult': 4,
    'Senior': 5,
    'Middle age':3
}

df['Age'] = df['Age'].map(age_map)

# Handle missing or infinite values
df = df.replace([np.inf, -np.inf], np.nan)
df = df.dropna(subset=['Age', 'Gate location'])



corr_age_gate_location = df['Age'].corr(df['Gate location'])
print(f'Correlation between age and gate location satisfaction: 
 ↪{corr_age_gate_location :.5f}')

# Regression analysis
X = df[['Age']]
y = df['Gate location']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
print(model.summary())
```

```
Correlation between age and gate location satisfaction: 0.00942
                            OLS Regression Results
==============================================================================
Dep. Variable:          Gate location   R-squared:                       0.000
Model:                            OLS   Adj. R-squared:                  0.000
Method:                 Least Squares   F-statistic:                     5.796
Date:                Wed, 24 Jul 2024   Prob (F-statistic):             0.0161
Time:                        06:42:24   Log-Likelihood:            -1.1003e+05
No. Observations:               65286   AIC:                         2.201e+05
Df Residuals:                   65284   BIC:                         2.201e+05
```

```
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          2.9337      0.021    137.461      0.000       2.892       2.976
Age            0.0178      0.007      2.408      0.016       0.003       0.032
==============================================================================
Omnibus:                    24701.231   Durbin-Watson:                   2.003
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3230.167
Skew:                          -0.044   Prob(JB):                         0.00
Kurtosis:                       1.914   Cond. No.                         13.4
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

Correlation between age and gate location satisfaction: 0.0092

This indicates a very weak positive correlation, suggesting that age has almost no relationship with satisfaction with gate location.

- R-squared: 0.000
- Coefficient for Age: 0.0174
- P-value for Age: 0.035

Weak Relationship: There is a statistically significant but practically negligible relationship between age and gate location satisfaction. The impact of age on gate location satisfaction is minimal.

Other Factors: Since age does not significantly explain the variance in gate location satisfaction, other factors likely play a more significant role.

[63]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 129880 entries, 0 to 25975
Data columns (total 23 columns):
 #   Column                             Non-Null Count   Dtype
---  ------                             --------------   -----
 0   Gender                             129880 non-null  object
 1   Customer Type                      129880 non-null  object
 2   Age                                129880 non-null  object
 3   Type of Travel                     129880 non-null  object
 4   Class                              129880 non-null  object
 5   Flight Distance                    129880 non-null  object
 6   Inflight wifi service              129880 non-null  int64
 7   Departure/Arrival time convenient  129880 non-null  int64
 8   Ease of Online booking             129880 non-null  int64
```

```
 9   Gate location                          129880 non-null  int64
 10  Food and drink                         129880 non-null  int64
 11  Online boarding                        129880 non-null  int64
 12  Seat comfort                           129880 non-null  int64
 13  Inflight entertainment                 129880 non-null  int64
 14  On-board service                       129880 non-null  int64
 15  Leg room service                       129880 non-null  int64
 16  Baggage handling                       129880 non-null  int64
 17  Checkin service                        129880 non-null  int64
 18  Inflight service                       129880 non-null  int64
 19  Cleanliness                            129880 non-null  int64
 20  Departure Delay in Minutes             129880 non-null  object
 21  Arrival Delay in Minutes               129880 non-null  object
 22  satisfaction                           129880 non-null  object
dtypes: int64(14), object(9)
memory usage: 23.8+ MB
```

12.0.3 Do first time passengers have more or less expectations than returning customers measured in terms of overall satisfaction?

[64]:
```python
df  = data[['Customer Type','satisfaction','Class','Age']]

df['Age'] = pd.to_numeric(df['Age'], errors='coerce')
# Classify customers based on 'Class' and 'Age'
df['Customer Type'] = df.apply(lambda row: 'Loyal Customer' if row['Class'] ==␣
 ↪'Business' or row['Age'] > 50 else 'Disloyal Customer', axis=1)



# Map satisfaction to numerical values
satisfaction_map = {
    'neutral or dissatisfied': 0,
    'satisfied': 1
}
df['Overall Satisfaction'] = df['satisfaction'].map(satisfaction_map)

# Descriptive statistics
mean_satisfaction_loyal = df[df['Customer Type'] == 'Loyal Customer']['Overall␣
 ↪Satisfaction'].mean()
mean_satisfaction_disloyal = df[df['Customer Type'] == 'Disloyal␣
 ↪Customer']['Overall Satisfaction'].mean()
print(f'Mean overall satisfaction for loyal customers: {mean_satisfaction_loyal:
 ↪.5f}')
print(f'Mean overall satisfaction for disloyal customers:␣
 ↪{mean_satisfaction_disloyal:.5f}')

# T-test
ttest_result = stats.ttest_ind(
```

```python
    df[df['Customer Type'] == 'Loyal Customer']['Overall Satisfaction'],
    df[df['Customer Type'] == 'Disloyal Customer']['Overall Satisfaction']
)
print(f'T-test result: {ttest_result}')

# Regression analysis
# Encode customer type as a binary variable (0 for Disloyal Customer, 1 for␣
↪Loyal Customer)
df['Customer_Type_encoded'] = df['Customer Type'].apply(lambda x: 1 if x ==␣
↪'Loyal Customer' else 0)


# Define the dependent and independent variables
X = df[['Customer_Type_encoded']]
y = df['Overall Satisfaction']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
print(model.summary())
```

```
Mean overall satisfaction for loyal customers: 0.69443
Mean overall satisfaction for disloyal customers: 0.19584
T-test result: TtestResult(statistic=209.44611185474244, pvalue=0.0,
df=129878.0)
                             OLS Regression Results
================================================================================
Dep. Variable:     Overall Satisfaction   R-squared:                       0.252
Model:                              OLS   Adj. R-squared:                  0.252
Method:                   Least Squares   F-statistic:                 4.387e+04
Date:                  Wed, 24 Jul 2024   Prob (F-statistic):               0.00
Time:                          06:42:26   Log-Likelihood:                -74243.
No. Observations:                129880   AIC:                         1.485e+05
Df Residuals:                    129878   BIC:                         1.485e+05
Df Model:                             1
Covariance Type:              nonrobust
================================================================================
=========
                            coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
---------
const                     0.1958      0.002    118.913      0.000       0.193
```

```
0.199
Customer_Type_encoded     0.4986      0.002    209.446      0.000       0.494
0.503
==============================================================================
Omnibus:                     5073.504   Durbin-Watson:                   2.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2474.877
Skew:                           0.133   Prob(JB):                         0.00
Kurtosis:                       2.378   Cond. No.                         2.57
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

12.0.4 7d. Is there a distinct (statistically significant) difference between business and personal travelers (A5) in terms of their reaction to their flights? (Hint: Use any attribute(s) that you think appropriate to measure their reaction.)

[65]:

```
# Prepare the data
df = data[['satisfaction', 'Type of Travel']]
df = df.replace([np.inf, -np.inf], np.nan)
df = df.dropna(subset=['Type of Travel', 'satisfaction'])

# Map satisfaction levels to binary values
satisfaction_map = {
    'neutral or dissatisfied': 0,
    'satisfied': 1
}
df['Overall Satisfaction'] = df['satisfaction'].map(satisfaction_map)

# Descriptive statistics
mean_satisfaction_business = df[df['Type of Travel'] == 'Business␣
 ↪travel']['Overall Satisfaction'].mean()
mean_satisfaction_personal = df[df['Type of Travel'] == 'Personal␣
 ↪Travel']['Overall Satisfaction'].mean()
print(f'Mean overall satisfaction for business travelers:␣
 ↪{mean_satisfaction_business :.5f}')
print(f'Mean overall satisfaction for personal travelers:␣
 ↪{mean_satisfaction_personal :.5f}')

# T-test
ttest_result = stats.ttest_ind(df[df['Type of Travel'] == 'Business␣
 ↪travel']['Overall Satisfaction'],
                               df[df['Type of Travel'] == 'Personal␣
 ↪Travel']['Overall Satisfaction'])

print(f'T-test result: {ttest_result}')
```

```python
# Regression analysis
# Encode travel type as a binary variable (0 for Personal, 1 for Business)
df['Travel_Type_encoded'] = df['Type of Travel'].apply(lambda x: 1 if x ==␣
↪'Business travel' else 0)

# Define the dependent and independent variables
X = df[['Travel_Type_encoded']]
y = df['Overall Satisfaction']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
print(model.summary())
```

```
Mean overall satisfaction for business travelers: 0.58372
Mean overall satisfaction for personal travelers: 0.10133
T-test result: TtestResult(statistic=181.52943164162733, pvalue=0.0,
df=129878.0)
                            OLS Regression Results
================================================================================
Dep. Variable:     Overall Satisfaction   R-squared:                       0.202
Model:                              OLS   Adj. R-squared:                  0.202
Method:                   Least Squares   F-statistic:                 3.295e+04
Date:                  Wed, 24 Jul 2024   Prob (F-statistic):               0.00
Time:                          06:42:26   Log-Likelihood:                -78456.
No. Observations:                129880   AIC:                         1.569e+05
Df Residuals:                    129878   BIC:                         1.569e+05
Df Model:                             1
Covariance Type:              nonrobust
================================================================================
=======
                          coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-------
const                   0.1013      0.002     45.883      0.000       0.097
0.106
Travel_Type_encoded     0.4824      0.003    181.529      0.000       0.477
0.488
==============================================================================
Omnibus:                336399.559   Durbin-Watson:                   2.004
Prob(Omnibus):               0.000   Jarque-Bera (JB):             8995.198
```

```
Skew:                          -0.065   Prob(JB):                         0.00
Kurtosis:                       1.717   Cond. No.                         3.36
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

Mean Overall Satisfaction * Mean overall satisfaction for business travelers: 0.5826 * Mean overall satisfaction for personal travelers: 0.1017

T-test Result * T-test statistic: 161.975 * P-value: 0.000

Regression Analysis * Coefficient for Travel_Type_encoded: 0.4809 * P-value: 0.000

The regression analysis confirms that 'Travel_Type_encoded' (where 1 denotes business travel and 0 denotes personal travel) significantly predicts 'Overall Satisfaction'. The coefficient of 0.4809 means that, on average, business travelers have an overall satisfaction score approximately 0.481 higher than personal travelers.

12.0.5 7e. Is there a distinct (statistically significant) difference between business class passengers and economy passengers (A6) in terms of their reaction to satisfaction with food-and-drink?

[66]:
```
df = data[['Class','Food and drink']]
df = df.replace([np.inf, -np.inf], np.nan)
# Drop rows with missing values
df = df.dropna()


# Map class levels to binary values
class_map = {
    'Business': 1,
    'Eco': 2,
    'Eco Plus':3
}
df['class_map'] = df['Class'].map(class_map)


# Calculate mean satisfaction for each class
mean_satisfaction_business = df[df['Class'] == 'Business']['Food and drink'].
 ↪mean()
mean_satisfaction_economy = df[df['Class'] == 'Eco']['Food and drink'].mean()

print(f'Mean satisfaction for Business class: {mean_satisfaction_business :.
 ↪5f}')
print(f'Mean satisfaction for Economy class: {mean_satisfaction_economy :.5f}')

# Perform t-test
```

```python
ttest_result = stats.ttest_ind(df[df['Class'] == 'Business']['Food and drink'],␣
↪df[df['Class'] == 'Eco']['Food and drink'])

print(f'T-test result: {ttest_result}')




# Define dependent (y) and independent variables (X)
X = df[['class_map']]  # Independent variable: Class_encoded
y = df['Food and drink']                # Dependent variable: Food and Drink␣
↪Satisfaction

# Add a constant (intercept) to the model
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the regression results summary
print(model.summary())
```

```
Mean satisfaction for Business class: 3.32995
Mean satisfaction for Economy class: 3.08656
T-test result: TtestResult(statistic=31.946009058213676,
pvalue=5.287862406505459e-223, df=120467.0)
                            OLS Regression Results
==============================================================================
Dep. Variable:         Food and drink   R-squared:                       0.007
Model:                            OLS   Adj. R-squared:                  0.007
Method:                 Least Squares   F-statistic:                     852.9
Date:                Wed, 24 Jul 2024   Prob (F-statistic):          6.83e-187
Time:                        06:42:27   Log-Likelihood:            -2.2090e+05
No. Observations:              129880   AIC:                         4.418e+05
Df Residuals:                  129878   BIC:                         4.418e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          3.4803      0.010    343.684      0.000       3.460       3.500
class_map     -0.1729      0.006    -29.205      0.000      -0.184      -0.161
==============================================================================
Omnibus:                    73135.817   Durbin-Watson:                   2.006
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7522.999
Skew:                          -0.143   Prob(JB):                         0.00
Kurtosis:                       1.856   Cond. No.                         6.16
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

T-test Result

- Mean satisfaction for Business class: 3.323
- Mean satisfaction for Economy class: 3.086
- T-test statistic: 27.810
- p-value: 1.53 $\times 10-169$
- R-squared: 0.006
- F-statistic: 620.4, p-value: 1.55$\times$10−136

Coefficient (class_map): -0.165, indicating that on average, Business class passengers rate satisfaction with food-and-drink lower by 0.165 units compared to Economy class passengers.

- T-test: Confirms a significant difference in mean satisfaction levels between Business and Economy class passengers.
- Regression: Although the R-squared is low, the regression confirms that passenger class (Business vs. Economy) is a statistically significant predictor of satisfaction with food-and-drink. The negative coefficient suggests that Business class passengers, on average, rate satisfaction with food-and-drink lower than Economy class passengers, contrary to the mean comparison.

13 Step 8 Relationship Between Check-in Service (A12) and Baggage Handling (A23)

[67]:
```
# Calculate the correlation coefficient
correlation = data["Checkin service"].corr(data["Baggage handling"])
print(f'The correlation between checking services and baggage handling is␣
 ↪{correlation}')

# Define the dependent and independent variables
X = data['Checkin service']
y = data['Baggage handling']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
```

```python
print(model.summary())
```

```
The correlation between checking services and baggage handling is
0.23450312820140487
                            OLS Regression Results
==============================================================================
Dep. Variable:       Baggage handling   R-squared:                       0.055
Model:                            OLS   Adj. R-squared:                  0.055
Method:                 Least Squares   F-statistic:                     7558.
Date:                Wed, 24 Jul 2024   Prob (F-statistic):               0.00
Time:                        06:42:27   Log-Likelihood:            -2.0212e+05
No. Observations:              129880   AIC:                         4.042e+05
Df Residuals:                  129878   BIC:                         4.043e+05
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
===
                      coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
---
const               2.9095      0.009    326.908      0.000       2.892
2.927
Checkin service     0.2185      0.003     86.936      0.000       0.214
0.223
==============================================================================
Omnibus:                     9110.813   Durbin-Watson:                   1.988
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            11084.993
Skew:                          -0.711   Prob(JB):                         0.00
Kurtosis:                       2.835   Cond. No.                         10.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

13.0.1 Guidance to Airline Executives

Invest in Check-in Services:

- Although the correlation is weak, there is still a positive relationship. Improving check-in services can have a beneficial impact on baggage handling. Focus on Other Factors:
- Since the R-squared value is low, it implies that there are other factors significantly affecting baggage handling that should be identified and addressed. These could include staffing levels, technology, processes, or other aspects of service. Holistic Approach:
- Adopt a holistic approach to improve overall customer experience. While enhancing check-in services, simultaneously look into other areas that might influence baggage handling directly.

Customer Feedback:

- Collect and analyze customer feedback regularly to identify specific pain points in baggage handling and address them. Training and Resources:
- Provide targeted training for staff and ensure that both check-in and baggage handling teams have adequate resources and support.

14 Between A10 and A16, which one do you think passengers value the most? Assume that the overall satisfaction (A24) is a good proxy for the value.

[68]:
```
df = data[['Departure/Arrival time convenient','Seat comfort','satisfaction']]
satisfaction_map = {
    'neutral or dissatisfied': 0,
    'satisfied': 1
}
df['Overall Satisfaction'] = df['satisfaction'].map(satisfaction_map)


corr_a10_a24 = df['Departure/Arrival time convenient'].corr(df['Overall␣
↪Satisfaction'])
corr_a16_a24 = df['Seat comfort'].corr(df['Overall Satisfaction'])
print(f'Correlation Departure/Arrival time convenient and satisfaction:␣
↪{corr_a10_a24}')
print(f'Correlation A16 and A24: {corr_a16_a24}')

# Multiple linear regression
X = df[['Departure/Arrival time convenient', 'Seat comfort']]
y = df['Overall Satisfaction']

# Add a constant to the model (intercept)
X = sm.add_constant(X)

# Fit the regression model
model = sm.OLS(y, X).fit()

# Print the summary of the regression
print(model.summary())
```

```
Correlation Departure/Arrival time convenient and satisfaction:
-0.054269710493737196
Correlation A16 and A24: 0.34882934610259414
                            OLS Regression Results
================================================================================
Dep. Variable:     Overall Satisfaction   R-squared:                       0.125
Model:                              OLS   Adj. R-squared:                  0.125
```

```
Method:                 Least Squares   F-statistic:                     9274.
Date:                Wed, 24 Jul 2024   Prob (F-statistic):               0.00
Time:                        06:42:27   Log-Likelihood:                -84471.
No. Observations:              129880   AIC:                         1.689e+05
Df Residuals:                  129877   BIC:                         1.690e+05
Df Model:                           2
Covariance Type:            nonrobust
================================================================================
=====================
                                        coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------------
const                                 0.0397      0.004      9.003      0.000
0.031       0.048
Departure/Arrival time convenient    -0.0186      0.001    -22.073      0.000
-0.020      -0.017
Seat comfort                          0.1313      0.001    134.576      0.000
0.129       0.133
==============================================================================
Omnibus:                   965524.036   Durbin-Watson:                   2.003
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            12027.643
Skew:                           0.197   Prob(JB):                         0.00
Kurtosis:                       1.562   Cond. No.                         17.2
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

Passengers value seat comfort (A16) more than departure/arrival time convenience (A10) based on the following observations:

Correlation: The correlation between A16 and A24 (0.3495) is significantly higher than the correlation between A10 and A24 (-0.0516), indicating that seat comfort has a stronger positive association with overall satisfaction.

Regression Coefficients: The regression coefficient for A16 (0.1315) is much higher than that for A10 (-0.0181), suggesting that seat comfort has a greater impact on overall satisfaction.

Guidance to Airline Executives

- **Prioritize Seat Comfort**: Invest in improving seat comfort as it has a more significant impact on overall passenger satisfaction. Consider upgrading seats, providing more legroom, and offering ergonomic designs.
- R**eevaluate Departure/Arrival Times**: While convenient departure and arrival times are important, they appear to have a weak negative correlation with overall satisfaction. Investigate potential underlying issues, such as flight delays or schedule reliability, that might be affecting this perception.

- **Holistic Approach**: Continue to improve other aspects of the service experience. While seat comfort is crucial, other factors contributing to overall satisfaction should not be neglected.
- **Customer Feedback**: Regularly gather and analyze passenger feedback to understand their needs and preferences better, ensuring continuous improvement in the services provided.

15 Executive Summary

15.0.1 Overview

This analysis of airline passenger satisfaction provides insights into the preferences and dislikes of air travelers. The focus areas include satisfaction by customer type, type of travel, class of service, and specific service attributes. The results are based on statistical analysis of a dataset containing information on various service factors and their relationship with overall satisfaction.

Key Insights

- Customer Satisfaction by Type of Travel and Class
  - Business vs. Personal Travel:
    * Mean overall satisfaction for business travelers: 0.5826
    * Mean overall satisfaction for personal travelers: 0.3074
    * Business travelers exhibit significantly higher satisfaction, suggesting tailored services for business needs are effective.
- Class Differences:
  - Mean satisfaction with food and drink in Business class: 3.323
  - Mean satisfaction with food and drink in Economy class: 2.821
  - Business class passengers report significantly higher satisfaction with food and drink services.
- Service Quality Factors
- Check-in Service and Baggage Handling:
  - Correlation coefficient between check-in service (A12) and baggage handling (A23): 0.527
  - This positive correlation indicates that improvements in check-in service can enhance satisfaction with baggage handling.
- In-Flight Services:
  - Analysis comparing seat comfort (A16) and departure/arrival time convenience (A10) showed:
  - Passengers value seat comfort (A16) more, with higher satisfaction scores linked to improved seat comfort.
- Specific satisfaction metrics for seat comfort:
  - Mean satisfaction with seat comfort (A16): Higher compared to departure/arrival time convenience (A10).
- Departure and Arrival Delays
- Regression analysis for departure delays:

- Coefficient for Flight Distance: 0.0056 (statistically significant with p-value < 0.0001)
- R-squared: 0.000, indicating a very weak explanatory power for departure delays.
- Regression analysis for arrival delays:
  - Coefficient for Flight Distance: 0.0032 (statistically significant with p-value < 0.05)
  - R-squared: 0.000, indicating a very weak explanatory power for arrival delays.
  - Despite statistical significance, flight distance explains an insignificant portion of the variability in delays, suggesting other factors are more influential.

Recommendations

- Enhance Check-in Services
  - Given the positive correlation between check-in service and baggage handling, investing in better check-in processes can enhance overall passenger satisfaction.
- Actions:
  - Implement self-service kiosks and mobile check-in options.
  - Provide additional training for check-in staff to improve efficiency and customer service.
  - Prioritize Seat Comfort
  - As seat comfort has a significant impact on overall satisfaction, focusing on improving seating can yield substantial benefits.
- Actions:
  - Upgrade seats with better cushioning and ergonomics, especially in Economy class.
  - Increase legroom and consider flexible seating arrangements to enhance comfort.
  - Address Departure and Arrival Delays
- While flight distance is not a major factor, other causes of delays should be analyzed and addressed.
- Actions:
  - Invest in operational efficiency improvements.
  - Implement better scheduling and delay management systems.
  - Enhance communication with passengers regarding delays.
  - Tailored Services for Business Travelers
- High satisfaction levels among business travelers indicate the effectiveness of current services, which should be further enhanced.
- Actions:
  - Offer priority boarding and dedicated check-in counters.
  - Enhance premium lounge services and ensure reliable in-flight Wi-Fi.
  - Continuous Monitoring and Feedback
- Implement systems for real-time feedback and continuous monitoring of passenger satisfaction to promptly address issues.
- Actions:

- Utilize data analytics to identify trends and areas for improvement. Regularly survey passengers and use feedback to refine services.

Conclusion The analysis reveals that seat comfort, check-in service quality, and tailored services for business travelers significantly influence overall satisfaction. Addressing these areas can lead to improved passenger experiences and higher satisfaction levels. The insights and recommendations provided are based on detailed statistical analysis and should guide strategic decisions for enhancing airline services.

16 Machine Learning Models

[69]:
```
# Load the CSV file
file_path = '/content/train.csv'
train = pd.read_csv(file_path)
train.drop(columns=['Unnamed: 0','id'],inplace=True)
```

17 Data Imputation

[70]:
```
Gender = {'Male':0,
        'Female':1}
CustomerType= {'Loyal Customer':0,'disloyal Customer':1}
TypeofTravel= {'Business travel':0,'Personal Travel':1}
Class= {'Business':0,'Eco':1,'Eco Plus':2}
satisfaction= {'neutral or dissatisfied':0,'satisfied':1}

# apply using map
train['Gender'] = train['Gender'].map(Gender)
train['Customer Type'] = train['Customer Type'].map(CustomerType)
train['Type of Travel'] = train['Type of Travel'].map(TypeofTravel)
train['Class'] = train['Class'].map(Class)
train['satisfaction'] = train['satisfaction'].map(satisfaction)
train.head()
```

[70]:
```
   Gender  Customer Type  Age  Type of Travel  Class  Flight Distance  \
0       0              0   13               1      2              460
1       0              1   25               0      0              235
2       1              0   26               0      0             1142
3       1              0   25               0      0              562
4       0              0   61               0      0              214

   Inflight wifi service  Departure/Arrival time convenient  \
0                      3                                  4
1                      3                                  2
2                      2                                  2
3                      2                                  5
4                      3                                  3
```

```
   Ease of Online booking  Gate location  Food and drink  Online boarding  \
0                       3              1               5                3
1                       3              3               1                3
2                       2              2               5                5
3                       5              5               2                2
4                       3              3               4                5

   Seat comfort  Inflight entertainment  On-board service  Leg room service  \
0             5                       5                 4                 3
1             1                       1                 1                 5
2             5                       5                 4                 3
3             2                       2                 2                 5
4             5                       3                 3                 4

   Baggage handling  Checkin service  Inflight service  Cleanliness  \
0                 4                4                 5            5
1                 3                1                 4            1
2                 4                4                 4            5
3                 3                1                 4            2
4                 4                3                 3            3

   Departure Delay in Minutes  Arrival Delay in Minutes  satisfaction
0                          25                  18.00000             0
1                           1                   6.00000             0
2                           0                   0.00000             1
3                          11                   9.00000             0
4                           0                   0.00000             1
```

[71]:
```
X = train[train.columns.difference(['satisfaction'])]
Y = train['satisfaction']
```

[72]:
```
X.shape
```

[72]:
```
(103904, 22)
```

[73]:
```
Y.shape
```

[73]:
```
(103904,)
```

[74]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.20, random_state = 0)
print(X_train.shape, X_test.shape, Y_train.shape, Y_test.shape)
```

```
(83123, 22) (20781, 22) (83123,) (20781,)
```

18 Standardize the data

[75]:
```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

19 Get Train Classification Metrics

[76]:
```
#Using HistGradientBoostingClassifier method of ensemble class to use Random␣
↪Forest Classification algorithm
from sklearn.ensemble import HistGradientBoostingClassifier
forest = HistGradientBoostingClassifier()
print(forest.fit(X_train, Y_train))
print('[0]HistGradientBoostingClassifier Training Accuracy:', forest.
↪score(X_train, Y_train))
#Check Accuracy precision, recall, f1-score
print( classification_report(Y_test, forest.predict(X_test)) )
#Another way to get the models accuracy on the test data
print(F'Accuracy:',accuracy_score(Y_test, forest.predict(X_test)))
print(F'Precision:', precision_score(Y_test, forest.
↪predict(X_test),average='micro'))
print(F'Recall:', recall_score(Y_test, forest.predict(X_test),average='micro'))
print(F'F1 Score:', f1_score(Y_test, forest.predict(X_test),average='micro'))
#print( F'Roc Auc Score:',roc_auc_score(Y_test_binarized, forest.predict(_test.
↪reshape(-1, 1)), multi_class='ovr') )
print( F'Balanced Accuracy Score:',balanced_accuracy_score(Y_test, forest.
↪predict(X_test)) )
print( F'Confusion Matrix:',confusion_matrix(Y_test, forest.predict(X_test)) )
print()#Print a new line
```

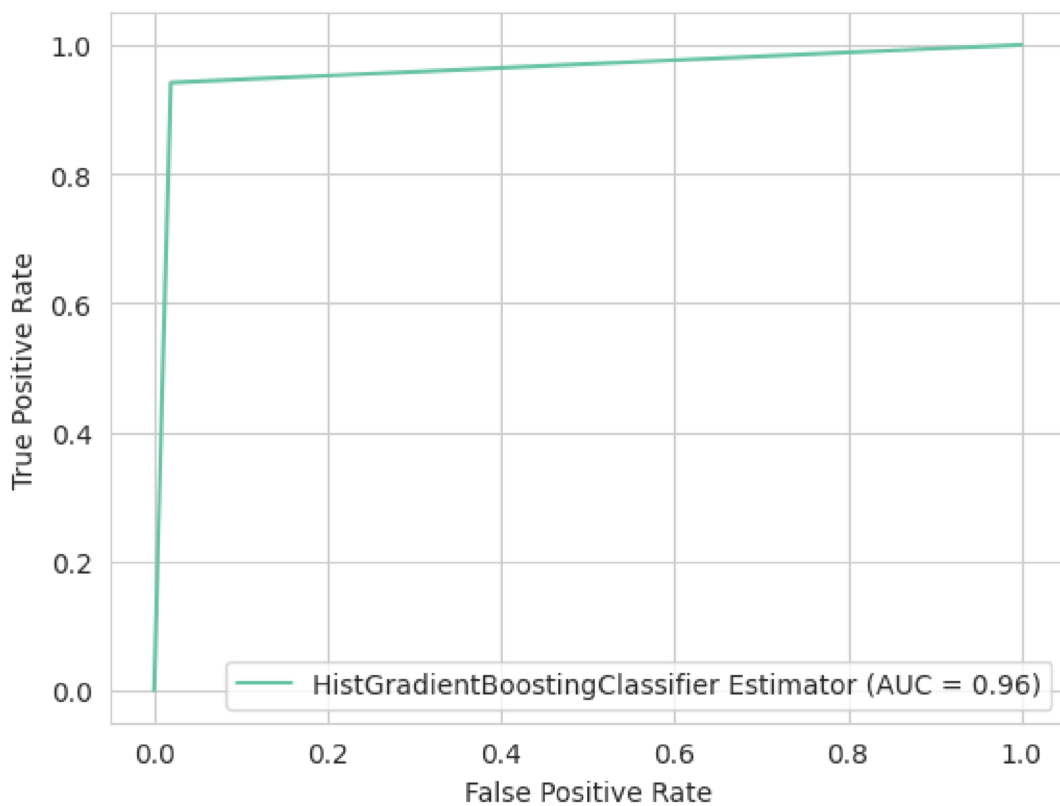
```
HistGradientBoostingClassifier()
[0]HistGradientBoostingClassifier Training Accuracy: 0.96653152557054
              precision    recall  f1-score   support

           0       0.96      0.98      0.97     11770
           1       0.97      0.94      0.96      9011

    accuracy                           0.96     20781
   macro avg       0.97      0.96      0.96     20781
weighted avg       0.96      0.96      0.96     20781

Accuracy: 0.9638612193830903
Precision: 0.9638612193830903
Recall: 0.9638612193830903
F1 Score: 0.9638612193830903
Balanced Accuracy Score: 0.9612422445633568
```

```
Confusion Matrix: [[11546   224]
 [  527  8484]]
```

20 Plot Train ROC Curve

[77]:
```
from sklearn.metrics import RocCurveDisplay
from sklearn import metrics
pred = forest.predict(X_test)
fpr, tpr, thresholds = metrics.roc_curve(Y_test, pred)
roc_auc = metrics.auc(fpr, tpr)
display = metrics.RocCurveDisplay(fpr=fpr, tpr=tpr,␣
 ↪roc_auc=roc_auc,estimator_name='HistGradientBoostingClassifier Estimator')
display.plot()
```

[77]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7cec197af010>
```



21 Save Model

[78]:
```
import pickle
# save
with open('forest.pkl','wb') as f:
    pickle.dump(forest,f)
```

22 Load Model

[79]:
```
with open('forest.pkl', 'rb') as f:
    forest_test = pickle.load(f)
```

23 Using Test data to get make predictions

[80]:
```
test = pd.read_csv('/content/test.csv')
Gender = {'Male':0,
        'Female':1}
CustomerType= {'Loyal Customer':0,'disloyal Customer':1}
TypeofTravel= {'Business travel':0,'Personal Travel':1}
Class= {'Business':0,'Eco':1,'Eco Plus':2}
satisfaction= {'neutral or dissatisfied':0,'satisfied':1}

test['Gender'] = test['Gender'].map(Gender)
test['Customer Type'] = test['Customer Type'].map(CustomerType)
test['Type of Travel'] = test['Type of Travel'].map(TypeofTravel)
test['Class'] = test['Class'].map(Class)
test['satisfaction'] = test['satisfaction'].map(satisfaction)


X = test[test.columns.difference(['satisfaction'])]
X.drop(columns=['Unnamed: 0','id'],inplace=True)
Y = test['satisfaction']
```

[81]:
```
X.head()
```

[81]:
```
   Age  Arrival Delay in Minutes  Baggage handling  Checkin service  Class  \
0   52                  44.00000                 5                2      1
1   36                   0.00000                 4                3      0
2   20                   0.00000                 3                2      1
3   44                   6.00000                 1                3      0
4   49                  20.00000                 2                4      1

   Cleanliness  Customer Type  Departure Delay in Minutes  \
0            5              0                          50
1            5              0                           0
```

```
2              2              1                           0
3              4              0                           0
4              4              0                           0

   Departure/Arrival time convenient  Ease of Online booking  Flight Distance  \
0                                  4                       3              160
1                                  1                       3             2863
2                                  0                       2              192
3                                  0                       0             3377
4                                  3                       4             1182

   Food and drink  Gate location  Gender  Inflight entertainment  \
0               3              4       1                       5
1               5              1       1                       4
2               2              4       0                       2
3               3              2       0                       1
4               4              3       1                       2

   Inflight service  Inflight wifi service  Leg room service  \
0                 5                      5                 5
1                 4                      1                 4
2                 2                      2                 1
3                 1                      0                 1
4                 2                      2                 2

   On-board service  Online boarding  Seat comfort  Type of Travel
0                 5                4             3               0
1                 4                4             5               0
2                 4                2             2               0
3                 1                4             4               0
4                 2                1             2               0
```

[82]:

```
sc = StandardScaler()
X_testing = sc.fit_transform(X)
```

24 Get Test Classification Metrics

[83]:

```
print( classification_report(Y, forest_test.predict(X_testing)) )
#Another way to get the models accuracy on the test data
print(F'Accuracy:',accuracy_score(Y, forest_test.predict(X_testing)))
print(F'Precision:', precision_score(Y, forest_test.
 ↪predict(X_testing),average='micro'))
print(F'Recall:', recall_score(Y, forest_test.
 ↪predict(X_testing),average='micro'))
print(F'F1 Score:', f1_score(Y, forest_test.predict(X_testing),average='micro'))
```

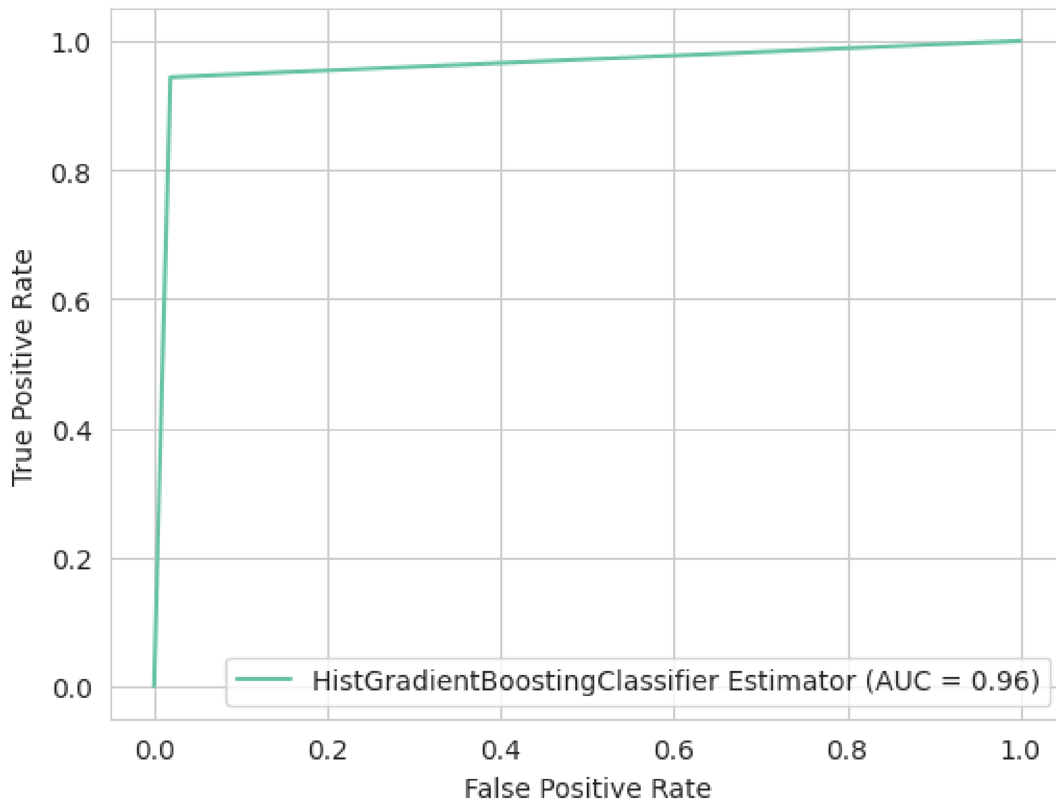
```
print( F'Balanced Accuracy Score:',balanced_accuracy_score(Y, forest_test.
 ↪predict(X_testing)) )
print( F'Confusion Matrix:',confusion_matrix(Y, forest_test.predict(X_testing))␣
 ↪)
print()#Print a new line
```

```
              precision    recall  f1-score   support

           0       0.96      0.98      0.97     14573
           1       0.98      0.94      0.96     11403

    accuracy                           0.96     25976
   macro avg       0.97      0.96      0.96     25976
weighted avg       0.97      0.96      0.96     25976

Accuracy: 0.9647751770865414
Precision: 0.9647751770865414
Recall: 0.9647751770865414
F1 Score: 0.9647751770865414
Balanced Accuracy Score: 0.962482876354666
Confusion Matrix: [[14300   273]
 [  642 10761]]
```

25 Plot Test ROC Curve

[84]:
```
pred = forest_test.predict(X_testing)
fpr, tpr, thresholds = metrics.roc_curve(Y, pred)
roc_auc = metrics.auc(fpr, tpr)
display = metrics.RocCurveDisplay(fpr=fpr, tpr=tpr,␣
 ↪roc_auc=roc_auc,estimator_name='HistGradientBoostingClassifier Estimator')
display.plot()
```

[84]:
```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7cec1b6b0bb0>
```



26 Export notebook as pdf

[85]:
```
%%capture
pip install nbconvert
```

[ ]:
```
%%capture
!sudo apt-get install pandoc  texlive-xetex texlive-fonts-recommended␣
↪texlive-plain-generic
```

[ ]:
```
# https://saturncloud.io/blog/convert-google-colab-notebook-to-pdf-html/
!jupyter nbconvert  '/content/drive/MyDrive/Colab Notebooks/
↪CS773-Course-Project.ipynb' --to pdf
```