Importing Liabraries

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

import warnings warnings.filterwarnings("ignore")

Loading Dataset

In [27]:
```
df=pd.read_csv("Air_Traffic_Passenger_Statistics.csv")
```

The Data at a Glance

In [28]:
```
df.head()
```

Out[28]:

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Deplaned	Low Fare	Terminal 1	B	27271	Deplaned
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Enplaned	Low Fare	Terminal 1	B	29131	Enplaned
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2
3	200507	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 1	B	35156	Deplaned
4	200507	Air Canada	AC	Air Canada	AC	International	Canada	Enplaned	Other	Terminal 1	B	34090	Enplaned

In [29]:
```
df.shape
```

Out[29]:
```
(15007, 16)
```

In [30]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15007 entries, 0 to 15006
Data columns (total 16 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Activity Period              15007 non-null  int64
 1   Operating Airline            15007 non-null  object
 2   Operating Airline IATA Code  14953 non-null  object
 3   Published Airline            15007 non-null  object
 4   Published Airline IATA Code  14953 non-null  object
 5   GEO Summary                  15007 non-null  object
 6   GEO Region                   15007 non-null  object
 7   Activity Type Code           15007 non-null  object
 8   Price Category Code          15007 non-null  object
 9   Terminal                     15007 non-null  object
 10  Boarding Area                15007 non-null  object
 11  Passenger Count              15007 non-null  int64
 12  Adjusted Activity Type Code  15007 non-null  object
 13  Adjusted Passenger Count     15007 non-null  int64
 14  Year                         15007 non-null  int64
 15  Month                        15007 non-null  object
dtypes: int64(4), object(12)
memory usage: 1.8+ MB
```

In [31]:
```
df.describe()
```

Out[31]:

	Activity Period	Passenger Count	Adjusted Passenger Count	Year
count	15007.000000	15007.000000	15007.000000	15007.000000
mean	201045.073366	29240.521090	29331.917105	2010.385220
std	313.336196	58319.509284	58284.182219	3.137589
min	200507.000000	1.000000	1.000000	2005.000000
25%	200803.000000	5373.500000	5495.500000	2008.000000
50%	201011.000000	9210.000000	9354.000000	2010.000000
75%	201308.000000	21158.500000	21182.000000	2013.000000
max	201603.000000	659837.000000	659837.000000	2016.000000

In [32]:
```
df.columns
```

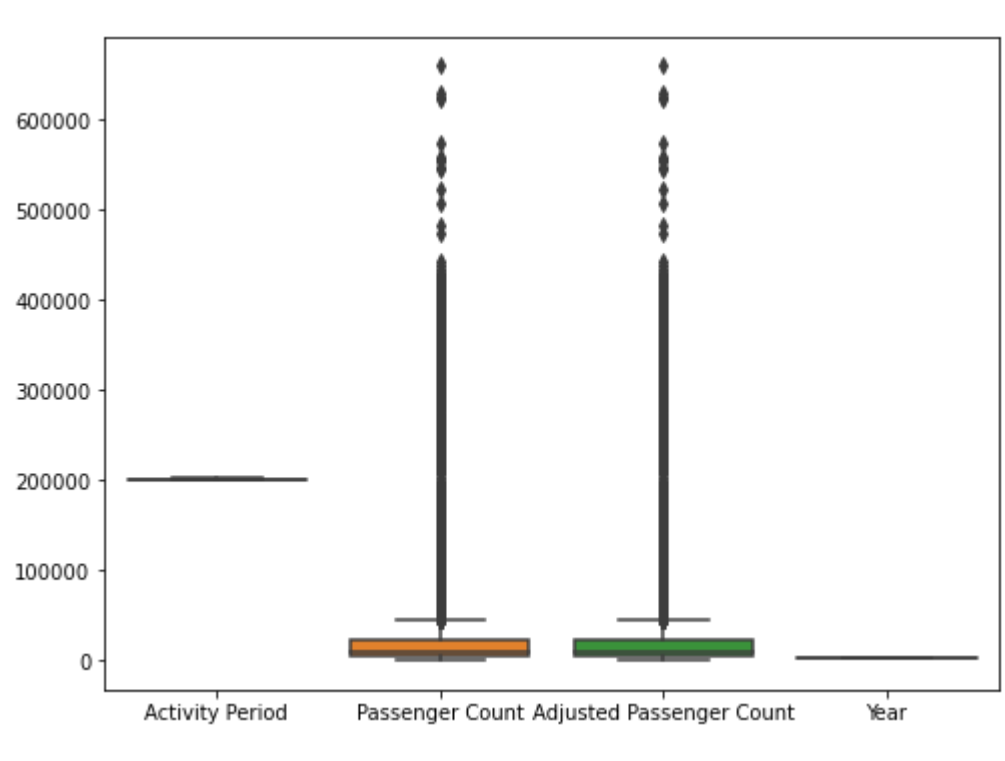
Out[32]:
```
Index(['Activity Period', 'Operating Airline', 'Operating Airline IATA Code',
       'Published Airline', 'Published Airline IATA Code', 'GEO Summary',
       'GEO Region', 'Activity Type Code', 'Price Category Code', 'Terminal',
       'Boarding Area', 'Passenger Count', 'Adjusted Activity Type Code',
       'Adjusted Passenger Count', 'Year', 'Month'],
      dtype='object')
```

In [33]:
```
df.isna().sum()
```

Out[33]:
```
Activity Period                 0
Operating Airline               0
Operating Airline IATA Code    54
Published Airline               0
Published Airline IATA Code    54
GEO Summary                     0
GEO Region                      0
Activity Type Code              0
Price Category Code             0
Terminal                        0
Boarding Area                   0
Passenger Count                 0
Adjusted Activity Type Code     0
Adjusted Passenger Count        0
Year                            0
Month                           0
dtype: int64
```

In [34]:
```
plt.figure(figsize=(8,6))
sns.boxplot(data=df)
```

Out[34]:
```
<AxesSubplot:>
```



In [35]:
```
df.drop(['Activity Period','Activity Type Code','Operating Airline IATA Code'],axis=1,inplace=True)
```

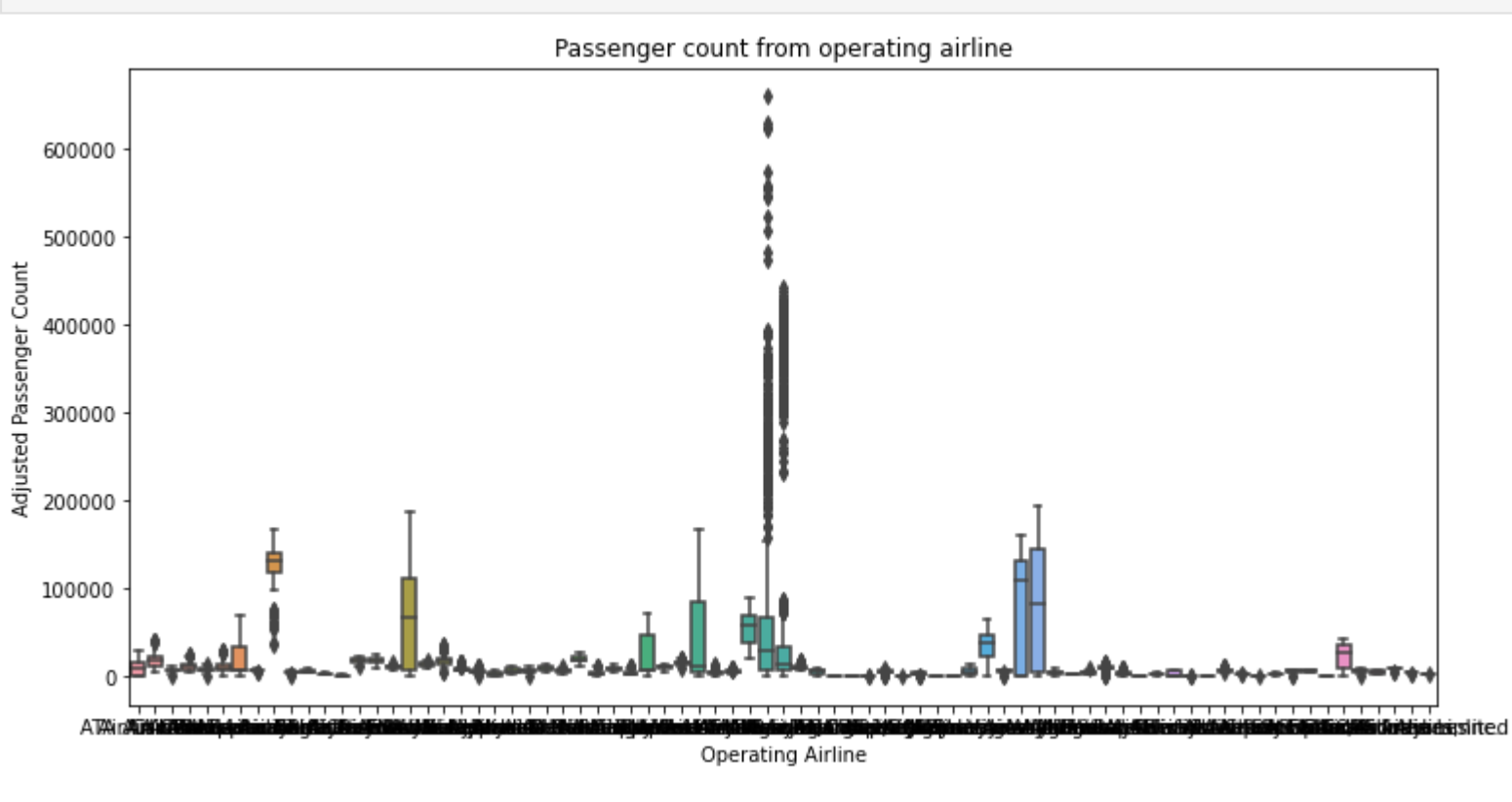
In [36]:
```
numcol=['Passenger Count','Adjusted Passenger Count', 'Year', 'Month']

catcol=[ 'Operating Airline','Published Airline', 'Published Airline IATA Code',
       'GEO Summary','GEO Region','Price Category Code', 'Terminal',
       'Boarding Area',  'Adjusted Activity Type Code']
```

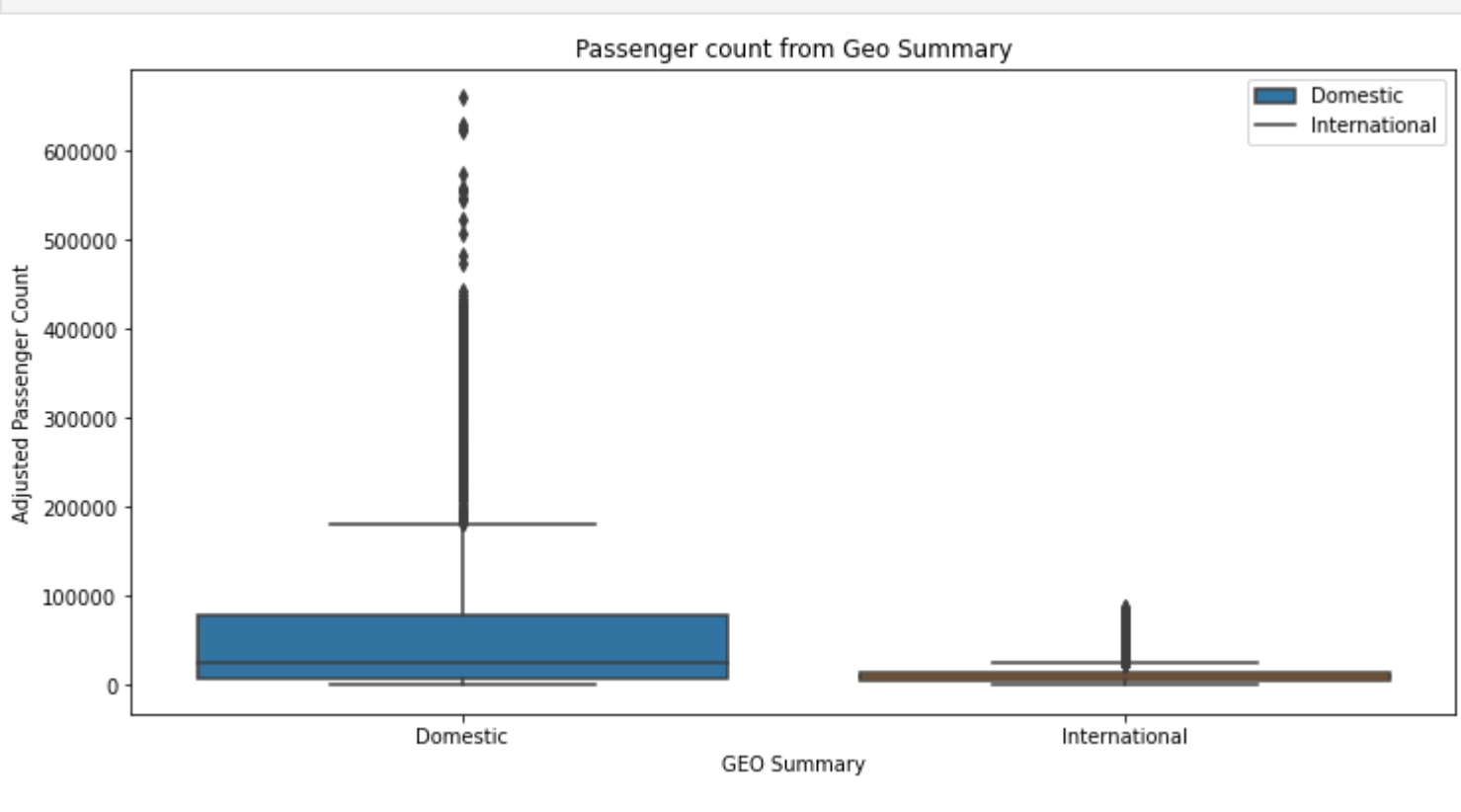
In [37]:
```
for i in catcol:
    print(df[i].value_counts())
```

```
United Airlines - Pre 07/01/2013    2154
SkyWest Airlines                     963
United Airlines                      892
Alaska Airlines                      751
Delta Air Lines                      386
                                    ...
Evergreen International Airlines       2
Atlas Air, Inc                         2
Xtra Airways                           2
Pacific Aviation                       2
Boeing Company                         1
Name: Operating Airline, Length: 77, dtype: int64
United Airlines - Pre 07/01/2013    2645
United Airlines                     1107
Alaska Airlines                      969
Delta Air Lines                      803
American Airlines                    416
                                    ...
Evergreen International Airlines       2
Atlas Air, Inc                         2
Xtra Airways                           2
Pacific Aviation                       2
Boeing Company                         1
Name: Published Airline, Length: 68, dtype: int64
UA     3752
AS      969
DL      803
AA      416
US      407
       ...
BBB      6
WO       3
5Y       2
XP       2
EZ       2
Name: Published Airline IATA Code, Length: 64, dtype: int64
International    9210
Domestic         5797
Name: GEO Summary, dtype: int64
US                     5797
Asia                   3273
Europe                 2089
Canada                 1418
Mexico                 1115
Australia / Oceania     737
Central America         274
Middle East             214
South America            90
Name: GEO Region, dtype: int64
Other       13087
Low Fare     1920
Name: Price Category Code, dtype: int64
International    9197
Terminal 1       3241
Terminal 3       2218
Terminal 2        324
Other              27
Name: Terminal, dtype: int64
A        5225
G        3992
B        1993
F        1377
C        1228
E         841
D         324
Other      27
Name: Boarding Area, dtype: int64
Deplaned              7071
Enplaned              7016
Thru / Transit * 2     920
Name: Adjusted Activity Type Code, dtype: int64
```

In [38]:
```
fig, ax = plt.subplots(figsize=(12, 6))
sns.boxplot(data=df,x='Operating Airline',y='Adjusted Passenger Count')
plt.title('Passenger count from operating airline')
plt.show()
```



In [39]:
```
fig, ax = plt.subplots(figsize=(12, 6))
sns.boxplot(data=df,x='GEO Summary',y='Adjusted Passenger Count')
plt.title('Passenger count from Geo Summary')
plt.legend(["Domestic","International"])
plt.show()
```



In [45]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15007 entries, 0 to 15006
Data columns (total 13 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Operating Airline            15007 non-null  int32
 1   Published Airline            15007 non-null  int32
 2   Published Airline IATA Code  15007 non-null  int32
 3   GEO Summary                  15007 non-null  int32
 4   GEO Region                   15007 non-null  int32
 5   Price Category Code          15007 non-null  int32
 6   Terminal                     15007 non-null  int32
 7   Boarding Area                15007 non-null  int32
 8   Passenger Count              15007 non-null  int64
 9   Adjusted Activity Type Code  15007 non-null  int32
 10  Adjusted Passenger Count     15007 non-null  int64
 11  Year                         15007 non-null  int64
 12  Month                        15007 non-null  int32
dtypes: int32(10), int64(3)
memory usage: 938.1 KB
```

In [46]:
```
x=df.iloc[:,[3,8,10,11]]
x
```

Out[46]:

	GEO Summary	Passenger Count	Adjusted Passenger Count	Year
0	0	8528	8530	0
1	0	8615	8617	0
2	0	2548	5527	0
3	1	8880	8883	0
4	1	8821	8824	0
...	...	...	...	...
15002	0	11373	11376	11
15003	1	1831	1762	11
15004	1	2111	2039	11
15005	1	6121	6091	11
15006	1	5601	5560	11

15007 rows × 4 columns

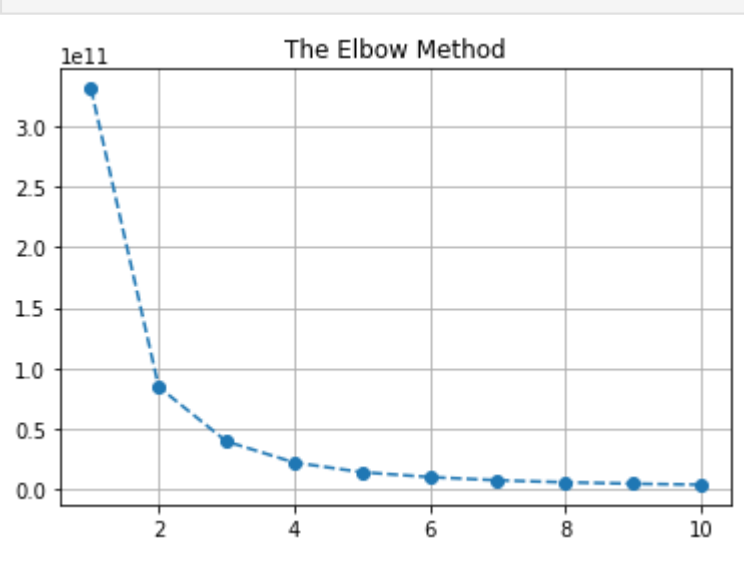
In [47]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for i in df:
    df[i]=le.fit_transform(df[i])
```

KMeans Clustering

In [48]:
```
from sklearn.cluster import KMeans
wcss = []

for i in range(1,11):
    Kmeans=KMeans(n_clusters=i,random_state=1)
    Kmeans.fit(x)
    wcss.append(Kmeans.inertia_)
```

In [49]:
```
plt.plot(range(1,11),wcss,'o--')
plt.grid()
plt.title("The Elbow Method")
plt.show()
```



In [50]:
```
kmeans=KMeans(n_clusters=5,random_state=1)
ylabel=kmeans.fit_predict(x)
```

In [51]:
```
df["ykmeans"]=ylabel
df
```

Out[51]:

	Operating Airline	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month	ykn
0	0	0	54	0	8	0	2	1	8528	0	8530	0	5	
1	0	0	54	0	8	0	2	1	8615	1	8617	0	5	
2	0	0	54	0	8	0	2	1	2548	2	5527	0	5	
3	4	4	6	1	2	1	2	1	8880	0	8883	0	5	
4	4	4	6	1	2	1	2	1	8821	1	8824	0	5	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
15002	71	62	58	0	8	0	3	3	11373	1	11376	11	7	
15003	71	62	58	1	5	0	0	0	1831	0	1762	11	7	
15004	71	62	58	1	5	0	3	3	2111	1	2039	11	7	
15005	72	63	57	1	4	1	0	0	6121	0	6091	11	7	
15006	72	63	57	1	4	1	0	0	5601	1	5560	11	7	

15007 rows × 14 columns

In [52]:
```
kmeans.cluster_centers_
```

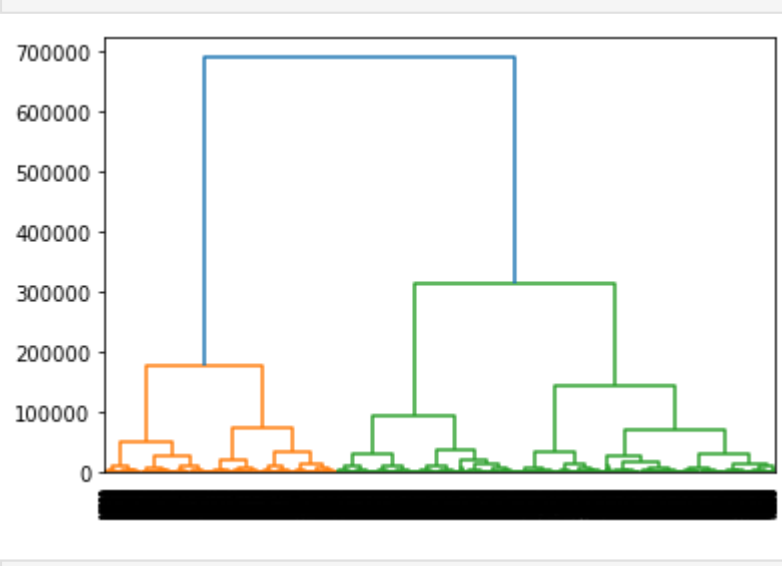
Out[52]:
```
array([[5.83176868e-01, 7.91030943e+03, 7.91116786e+03, 5.46526474e+00],
       [8.01530891e-01, 3.26162739e+03, 3.27907983e+03, 5.46145435e+00],
       [1.10156250e-01, 1.04277871e+04, 1.04307871e+04, 5.65156250e+00],
       [6.12598425e-01, 1.03643150e+03, 1.04815654e+03, 4.91716535e+00],
       [8.46492714e-01, 5.49928295e+03, 5.50408031e+03, 5.49101999e+00]])
```

In [53]:
```
df["ykmeans"].value_counts()
```

Out[53]:
```
1    3649
3    3172
4    2956
0    2672
2    2558
Name: ykmeans, dtype: int64
```

Hierarchical Clustering

In [54]:
```
from scipy.cluster import hierarchy as hi
lk=hi.linkage(x,method="ward")
ddg=hi.dendrogram(lk)
plt.show()
```



In [61]:
```
from sklearn.cluster import AgglomerativeClustering
hc=AgglomerativeClustering(n_clusters=5)
ylabel=hc.fit_predict(x)
```

In [62]:
```
df["hcy"]=ylabel
```

In [63]:
```
df[df.hcy==0].describe()
```

Out[63]:

	Operating Airline	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	A Pa
count	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019.000000	4019
mean	40.611844	35.025877	34.579995	0.833541	3.107987	0.946504	0.656133	2.536949	5150.292112	0.508087	5140
std	23.134362	20.613954	17.852020	0.372539	2.876205	0.225048	1.331434	2.740078	882.819996	0.523820	897
min	0.000000	0.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2813.000000	0.000000	3693
25%	24.000000	19.000000	16.000000	1.000000	0.000000	1.000000	0.000000	0.000000	4388.000000	0.000000	4356
50%	40.000000	31.000000	34.000000	1.000000	2.000000	1.000000	0.000000	1.000000	5116.000000	0.000000	5104
75%	60.000000	60.000000	55.000000	1.000000	5.000000	1.000000	0.000000	6.000000	5899.000000	1.000000	5903
max	73.000000	64.000000	63.000000	1.000000	8.000000	1.000000	4.000000	6.000000	6757.000000	2.000000	7744

In [64]:
```
df.groupby("hcy")[["Passenger Count","Adjusted Passenger Count"]].mean()
```

Out[64]:

hcy	Passenger Count	Adjusted Passenger Count
0	5150.292112	5140.071162
1	10299.830496	10302.830496
2	2587.450986	2621.927464
3	7809.054754	7809.973260
4	609.091181	621.279522

In [65]:
```
y=df.iloc[:,-1]
y
```

Out[65]:
```
0        3
1        3
2        2
3        3
4        3
        ..
15002    1
15003    2
15004    2
15005    0
15006    0
Name: hcy, Length: 15007, dtype: int64
```

Importing models & metrics from Sklearn for model building

In [66]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.3,random_state=1)
```

In [67]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
```

In [68]:
```
def indmodel(model):
    model.fit(xtrain,ytrain)
    ypred=model.predict(xtest)

    train=model.score(xtrain,ytrain)
    test=model.score(xtest,ytest)

    print(f' Training Accuracy : {train} \n Testing Accuracy : {test} \n')
    print(classification_report(ytest,ypred))

    return model
```

In [69]:
```
cladt=indmodel(DecisionTreeClassifier(random_state=2))
```

```
 Training Accuracy : 1.0
 Testing Accuracy : 0.9993337774816788

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1208
           1       1.00      1.00      1.00       836
           2       1.00      1.00      1.00      1188
           3       1.00      1.00      1.00       718
           4       1.00      1.00      1.00       553

    accuracy                           1.00      4503
   macro avg       1.00      1.00      1.00      4503
weighted avg       1.00      1.00      1.00      4503
```

In [70]:
```
knn=indmodel(KNeighborsClassifier(n_neighbors=5))
```

```
 Training Accuracy : 0.9998095963442498
 Testing Accuracy : 0.9997779258272262

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1208
           1       1.00      1.00      1.00       836
           2       1.00      1.00      1.00      1188
           3       1.00      1.00      1.00       718
           4       1.00      1.00      1.00       553

    accuracy                           1.00      4503
   macro avg       1.00      1.00      1.00      4503
weighted avg       1.00      1.00      1.00      4503
```

In [ ]: