FLYZY FLIGHT CANCELLED

May 28, 2024

[1]:
```
#importing the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
plt.style.use('ggplot')
```

[2]:
```
#Loading the dataset
df= pd.read_excel('Flyzy Flight Cancellation (1).xlsx')
```

[3]:
```
#EXPLORING THE DATA
#finding the shape of the dataset
df.shape
```

[3]:
```
(3000, 14)
```

[4]:
```
#finding the head of the dataset
df.head()
```

[4]:

	Flight ID	Airline	Flight_Distance	Origin_Airport	Destination_Airport	\
0	7319483	Airline D	475	Airport 3	Airport 2	
1	4791965	Airline E	538	Airport 5	Airport 4	
2	2991718	Airline C	565	Airport 1	Airport 2	
3	4220106	Airline E	658	Airport 5	Airport 3	
4	2263008	Airline E	566	Airport 2	Airport 2	

	Scheduled_Departure_Time	Day_of_Week	Month	Airplane_Type	Weather_Score	\
0	4	6	1	Type C	0.225122	
1	12	1	6	Type B	0.060346	
2	17	3	9	Type C	0.093920	
3	1	1	8	Type B	0.656750	
4	19	7	12	Type E	0.505211	

```
   Previous_Flight_Delay_Minutes  Airline_Rating  Passenger_Load  \
0                            5.0        2.151974        0.477202
1                           68.0        1.600779        0.159718
2                           18.0        4.406848        0.256803
3                           13.0        0.998757        0.504077
4                            4.0        3.806206        0.019638

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

[5]:
```
#finding the tail of the dataset
df.tail()
```

[5]:
```
      Flight ID    Airline  Flight_Distance Origin_Airport  \
2995    1265781  Airline D              395      Airport 2
2996    5440150  Airline E              547      Airport 1
2997     779080  Airline C              461      Airport 1
2998    4044431  Airline B              464      Airport 3
2999    2806578  Airline A              369      Airport 1

     Destination_Airport  Scheduled_Departure_Time  Day_of_Week  Month  \
2995           Airport 3                        0            6      1
2996           Airport 4                       22            4      7
2997           Airport 3                        8            3      1
2998           Airport 3                        5            5      3
2999           Airport 2                        1            1     10

     Airplane_Type  Weather_Score  Previous_Flight_Delay_Minutes  \
2995        Type B       0.190018                        1.00000
2996        Type E       0.719271                       91.00000
2997        Type B       0.458724                        3.00000
2998        Type E       0.443373                       46.00000
2999        Type A       0.704563                       18.66667

      Airline_Rating  Passenger_Load  Flight_Cancelled
2995        2.451216        0.283440                 1
2996        0.027039        0.665294                 1
2997        1.131315        0.991307                 0
2998        0.968651        0.254808                 1
2999        1.879411        0.532486                 1
```

[6]:
```
df=df.drop(columns=['Flight ID'])
df
```

[6]:

	Airline	Flight_Distance	Origin_Airport	Destination_Airport	\
0	Airline D	475	Airport 3	Airport 2	
1	Airline E	538	Airport 5	Airport 4	
2	Airline C	565	Airport 1	Airport 2	
3	Airline E	658	Airport 5	Airport 3	
4	Airline E	566	Airport 2	Airport 2	
…	…	…	…	…	
2995	Airline D	395	Airport 2	Airport 3	
2996	Airline E	547	Airport 1	Airport 4	
2997	Airline C	461	Airport 1	Airport 3	
2998	Airline B	464	Airport 3	Airport 3	
2999	Airline A	369	Airport 1	Airport 2	

	Scheduled_Departure_Time	Day_of_Week	Month	Airplane_Type	\
0	4	6	1	Type C	
1	12	1	6	Type B	
2	17	3	9	Type C	
3	1	1	8	Type B	
4	19	7	12	Type E	
…	…	…	…	…	
2995	0	6	1	Type B	
2996	22	4	7	Type E	
2997	8	3	1	Type B	
2998	5	5	3	Type E	
2999	1	1	10	Type A	

	Weather_Score	Previous_Flight_Delay_Minutes	Airline_Rating	\
0	0.225122	5.00000	2.151974	
1	0.060346	68.00000	1.600779	
2	0.093920	18.00000	4.406848	
3	0.656750	13.00000	0.998757	
4	0.505211	4.00000	3.806206	
…	…	…	…	
2995	0.190018	1.00000	2.451216	
2996	0.719271	91.00000	0.027039	
2997	0.458724	3.00000	1.131315	
2998	0.443373	46.00000	0.968651	
2999	0.704563	18.66667	1.879411	

	Passenger_Load	Flight_Cancelled
0	0.477202	0
1	0.159718	1
2	0.256803	0
3	0.504077	1
4	0.019638	0
…	…	…
2995	0.283440	1

```
2996        0.665294                 1
2997        0.991307                 0
2998        0.254808                 1
2999        0.532486                 1

[3000 rows x 13 columns]
```

[7]:
```
df.columns
```

[7]:
```
Index(['Airline', 'Flight_Distance', 'Origin_Airport', 'Destination_Airport',
       'Scheduled_Departure_Time', 'Day_of_Week', 'Month', 'Airplane_Type',
       'Weather_Score', 'Previous_Flight_Delay_Minutes', 'Airline_Rating',
       'Passenger_Load', 'Flight_Cancelled'],
      dtype='object')
```

[8]:
```
len(df.columns)
```

[8]:
```
13
```

[9]:
```
df.nunique()
```

[9]:
```
Airline                          5
Flight_Distance                470
Origin_Airport                   5
Destination_Airport              4
Scheduled_Departure_Time         24
Day_of_Week                      7
Month                            12
Airplane_Type                    5
Weather_Score                 2999
Previous_Flight_Delay_Minutes    304
Airline_Rating                2999
Passenger_Load                2995
Flight_Cancelled                 2
dtype: int64
```

[10]:
```
#gathering the information of the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 13 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Airline                         3000 non-null   object
 1   Flight_Distance                 3000 non-null   int64
 2   Origin_Airport                  3000 non-null   object
```

```
 3   Destination_Airport            3000 non-null   object
 4   Scheduled_Departure_Time       3000 non-null   int64
 5   Day_of_Week                    3000 non-null   int64
 6   Month                          3000 non-null   int64
 7   Airplane_Type                  3000 non-null   object
 8   Weather_Score                  3000 non-null   float64
 9   Previous_Flight_Delay_Minutes  3000 non-null   float64
 10  Airline_Rating                 3000 non-null   float64
 11  Passenger_Load                 3000 non-null   float64
 12  Flight_Cancelled               3000 non-null   int64
dtypes: float64(4), int64(5), object(4)
memory usage: 304.8+ KB
```

[11]:
```
from sklearn.preprocessing import LabelEncoder
label_encoder=LabelEncoder()
```

[12]:
```
df['Airline']=label_encoder.fit_transform(df['Airline'])
df['Origin_Airport']=label_encoder.fit_transform(df['Origin_Airport'])
df['Destination_Airport']=label_encoder.fit_transform(df['Destination_Airport'])
df['Airplane_Type']=label_encoder.fit_transform(df['Airplane_Type'])
df.dtypes
```

[12]:
```
Airline                          int64
Flight_Distance                  int64
Origin_Airport                   int64
Destination_Airport              int64
Scheduled_Departure_Time         int64
Day_of_Week                      int64
Month                            int64
Airplane_Type                    int64
Weather_Score                  float64
Previous_Flight_Delay_Minutes  float64
Airline_Rating                 float64
Passenger_Load                 float64
Flight_Cancelled                 int64
dtype: object
```

[ ]:
```
# CONVERING MY DATA FROM CATEGORIC TO NUMERIC
df['Code'] = pd.Factorize(df.Flight_Cancelled)[0]
```

[14]:
```
df.Flight_Cancelled.value_counts()
```

[14]:
```
1    2072
0     928
Name: Flight_Cancelled, dtype: int64
```

[15]:
```
#cleaning the data
df.isnull().sum()
```

[15]:
```
Airline                          0
Flight_Distance                  0
Origin_Airport                   0
Destination_Airport              0
Scheduled_Departure_Time         0
Day_of_Week                      0
Month                            0
Airplane_Type                    0
Weather_Score                    0
Previous_Flight_Delay_Minutes    0
Airline_Rating                   0
Passenger_Load                   0
Flight_Cancelled                 0
dtype: int64
```

[16]:
```
df.describe()
```

[16]:

	Airline	Flight_Distance	Origin_Airport	Destination_Airport
count	3000.000000	3000.000000	3000.000000	3000.000000
mean	1.567333	498.909333	1.631667	0.911667
std	1.513350	98.892266	1.499805	1.147012
min	0.000000	138.000000	0.000000	0.000000
25%	0.000000	431.000000	0.000000	0.000000
50%	1.000000	497.000000	1.000000	0.000000
75%	3.000000	566.000000	3.000000	2.000000
max	4.000000	864.000000	4.000000	3.000000

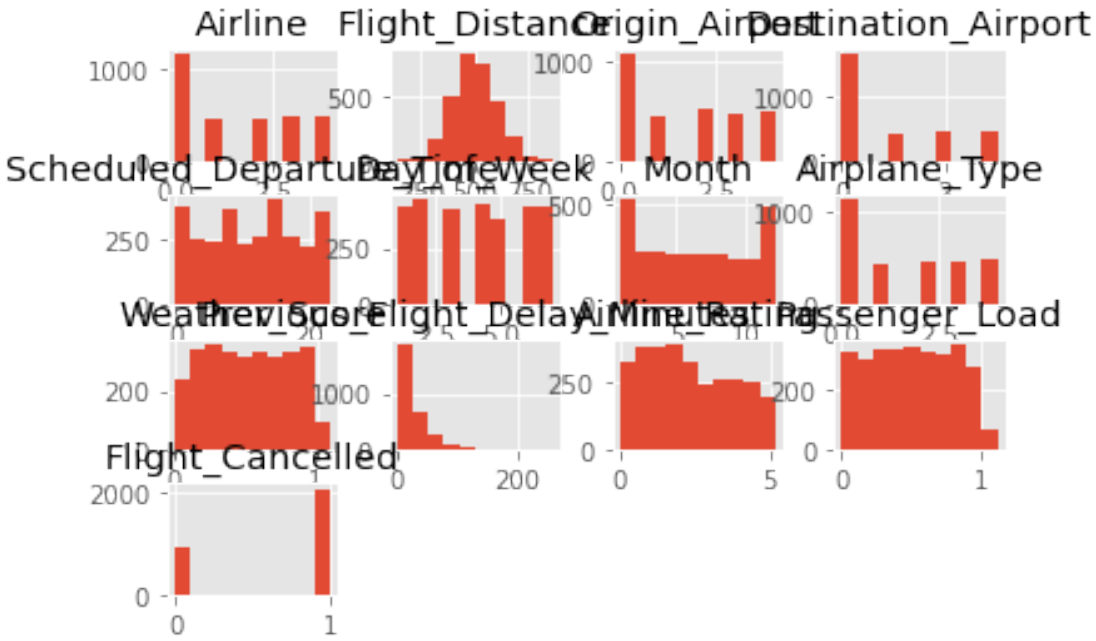
	Scheduled_Departure_Time	Day_of_Week	Month	Airplane_Type
count	3000.000000	3000.000000	3000.000000	3000.000000
mean	11.435000	3.963000	6.381000	1.582000
std	6.899298	2.016346	3.473979	1.515049
min	0.000000	1.000000	1.000000	0.000000
25%	6.000000	2.000000	3.000000	0.000000
50%	12.000000	4.000000	6.000000	1.000000
75%	17.000000	6.000000	9.000000	3.000000
max	23.000000	7.000000	12.000000	4.000000

	Weather_Score	Previous_Flight_Delay_Minutes	Airline_Rating
count	3000.000000	3000.000000	3000.000000
mean	0.524023	26.793383	2.317439
std	0.290694	27.874733	1.430386
min	0.000965	0.000000	0.000103
25%	0.278011	7.000000	1.092902
50%	0.522180	18.000000	2.126614

```
75%          0.776323                    38.000000        3.525746
max          1.099246                   259.000000        5.189038

       Passenger_Load  Flight_Cancelled
count     3000.000000       3000.000000
mean         0.515885          0.690667
std          0.295634          0.462296
min          0.001039          0.000000
25%          0.265793          0.000000
50%          0.517175          1.000000
75%          0.770370          1.000000
max          1.123559          1.000000
```

[17]:
```
#distribution of data
df.hist()
```

[17]:
```
array([[<AxesSubplot: title={'center': 'Airline'}>,
        <AxesSubplot: title={'center': 'Flight_Distance'}>,
        <AxesSubplot: title={'center': 'Origin_Airport'}>,
        <AxesSubplot: title={'center': 'Destination_Airport'}>],
       [<AxesSubplot: title={'center': 'Scheduled_Departure_Time'}>,
        <AxesSubplot: title={'center': 'Day_of_Week'}>,
        <AxesSubplot: title={'center': 'Month'}>,
        <AxesSubplot: title={'center': 'Airplane_Type'}>],
       [<AxesSubplot: title={'center': 'Weather_Score'}>,
        <AxesSubplot: title={'center': 'Previous_Flight_Delay_Minutes'}>,
        <AxesSubplot: title={'center': 'Airline_Rating'}>,
        <AxesSubplot: title={'center': 'Passenger_Load'}>],
       [<AxesSubplot: title={'center': 'Flight_Cancelled'}>,
        <AxesSubplot: >, <AxesSubplot: >, <AxesSubplot: >]], dtype=object)
```

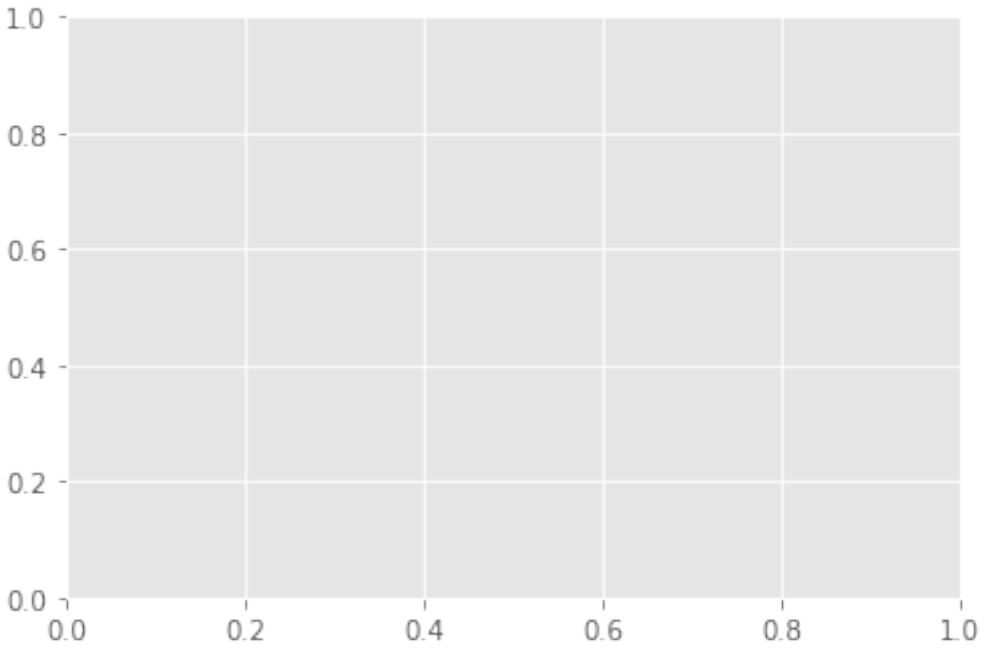


[21]:
```
#plotting the scatter plots of the dataset to get the relationship between␣
 ↪features
df.plot.scatter(x='Scheduled_Departure_Time', y='Flight ID')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.10/site-packages/pandas/core/indexes/base.py in␣
 ↪get_loc(self, key, method, tolerance)
   3801             try:
-> 3802                 return self._engine.get_loc(casted_key)
   3803             except KeyError as err:

/usr/local/lib/python3.10/site-packages/pandas/_libs/index.pyx in pandas._libs.
 ↪index.IndexEngine.get_loc()

/usr/local/lib/python3.10/site-packages/pandas/_libs/index.pyx in pandas._libs.
 ↪index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.
 ↪PyObjectHashTable.get_item()

KeyError: 'Flight ID'
```

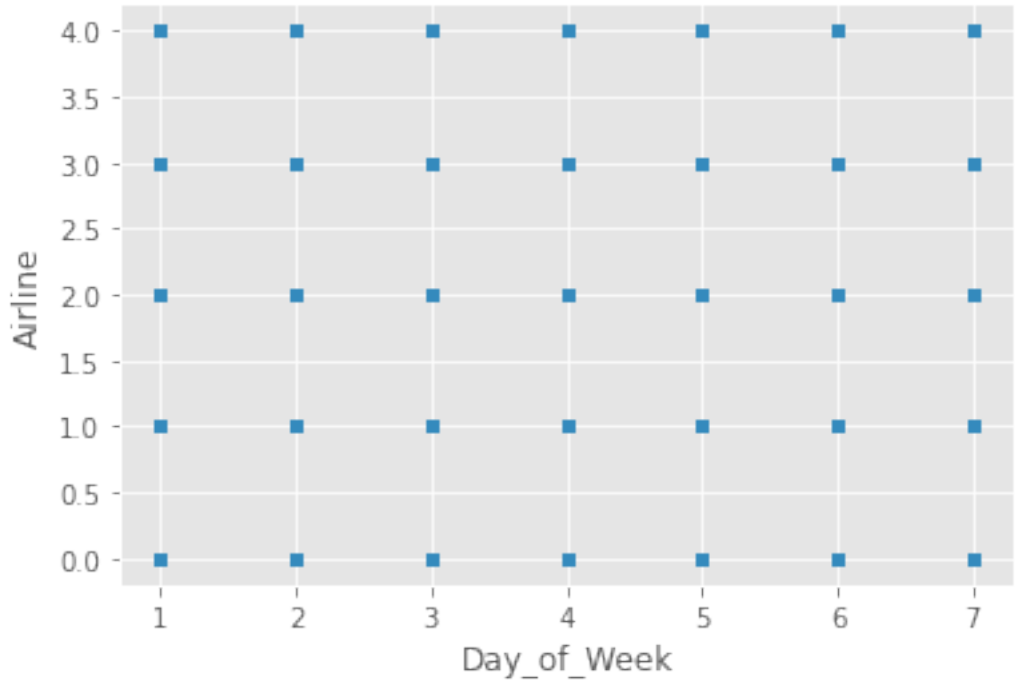
```
The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
/tmp/ipykernel_121/3824218344.py in <cell line: 2>()
      1 #plotting the scatter plots of the dataset to get the relationship␣
 ↪between features
----> 2 df.plot.scatter(x='Scheduled_Departure_Time', y='Flight ID')

/usr/local/lib/python3.10/site-packages/pandas/plotting/_core.py in␣
 ↪scatter(self, x, y, s, c, **kwargs)
   1695             …                         colormap='viridis')
   1696         """
-> 1697         return self(kind="scatter", x=x, y=y, s=s, c=c, **kwargs)
   1698
   1699     def hexbin(

/usr/local/lib/python3.10/site-packages/pandas/plotting/_core.py in␣
 ↪__call__(self, *args, **kwargs)
    943         if kind in self._dataframe_kinds:
    944             if isinstance(data, ABCDataFrame):
--> 945                 return plot_backend.plot(data, x=x, y=y, kind=kind,␣
 ↪**kwargs)
    946             else:
    947                 raise ValueError(f"plot kind {kind} can only be used fo␣
 ↪data frames")

/usr/local/lib/python3.10/site-packages/pandas/plotting/_matplotlib/__init__.py
 ↪in plot(data, kind, **kwargs)
     69             kwargs["ax"] = getattr(ax, "left_ax", ax)
     70     plot_obj = PLOT_CLASSES[kind](data, **kwargs)
---> 71     plot_obj.generate()
     72     plot_obj.draw()
     73     return plot_obj.result

/usr/local/lib/python3.10/site-packages/pandas/plotting/_matplotlib/core.py in␣
 ↪generate(self)
    450         self._compute_plot_data()
    451         self._setup_subplots()
--> 452         self._make_plot()
    453         self._add_table()
    454         self._make_legend()

/usr/local/lib/python3.10/site-packages/pandas/plotting/_matplotlib/core.py in␣
 ↪_make_plot(self)
   1259         scatter = ax.scatter(
   1260             data[x].values,
-> 1261             data[y].values,
   1262             c=c_values,
```

```
   1263             label=label,

/usr/local/lib/python3.10/site-packages/pandas/core/frame.py in␣
 ↪__getitem__(self, key)
   3805             if self.columns.nlevels > 1:
   3806                 return self._getitem_multilevel(key)
-> 3807             indexer = self.columns.get_loc(key)
   3808             if is_integer(indexer):
   3809                 indexer = [indexer]

/usr/local/lib/python3.10/site-packages/pandas/core/indexes/base.py in␣
 ↪get_loc(self, key, method, tolerance)
   3802                 return self._engine.get_loc(casted_key)
   3803             except KeyError as err:
-> 3804                 raise KeyError(key) from err
   3805             except TypeError:
   3806                 # If we have a listlike key, _check_indexing_error will␣
 ↪raise

KeyError: 'Flight ID'
```



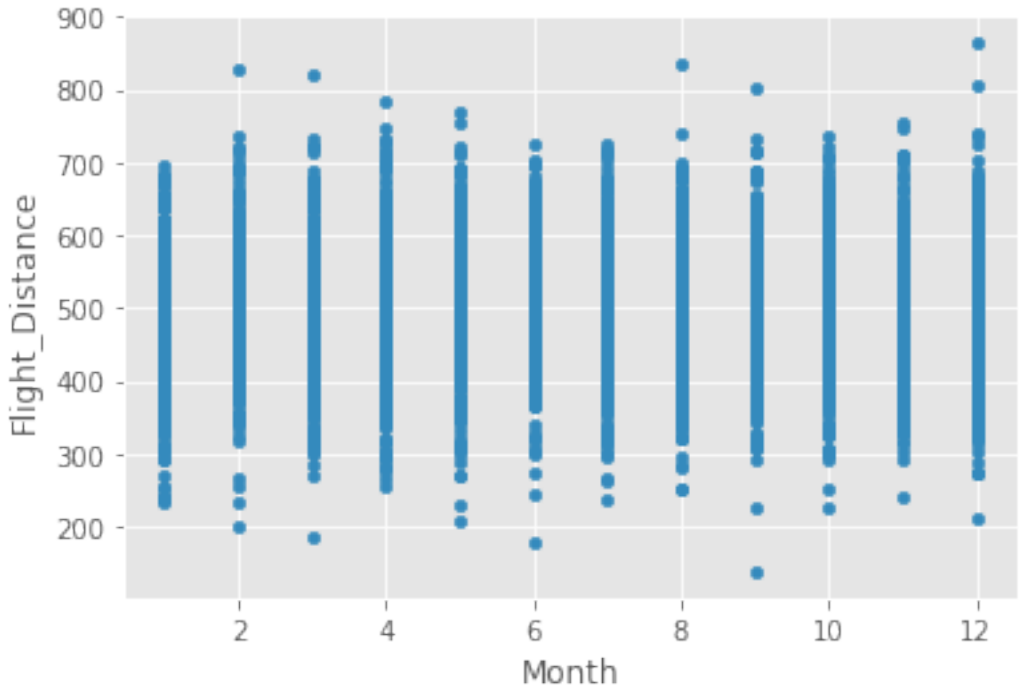
[20]:
```
df.plot.scatter(x='Day_of_Week', y='Airline')
```

[20]: <AxesSubplot: xlabel='Day_of_Week', ylabel='Airline'>



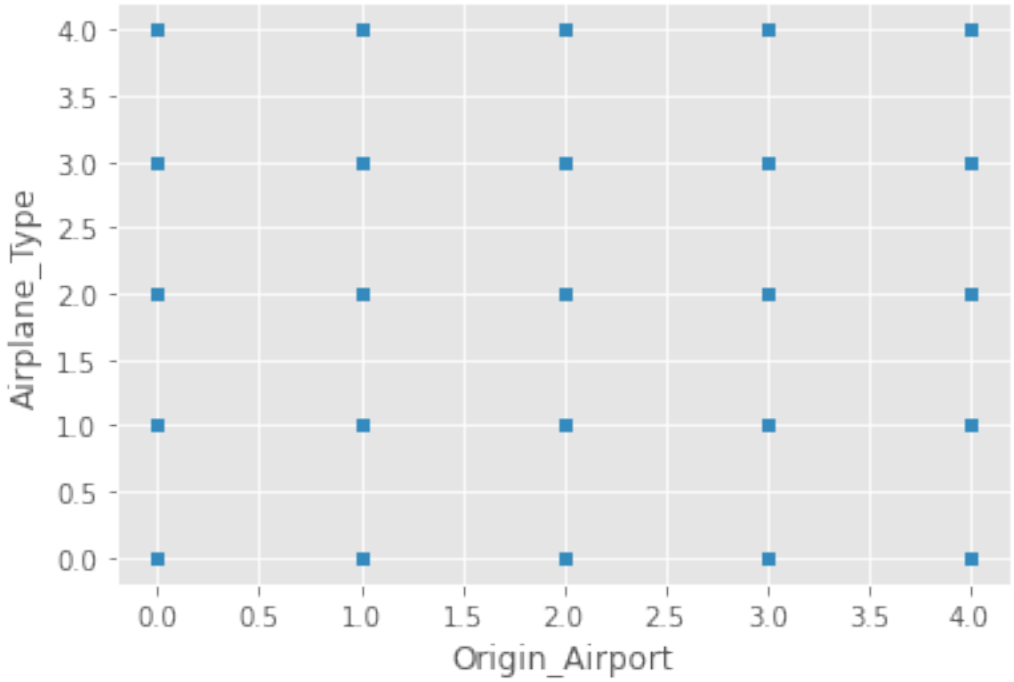
[22]:
```
df.plot.scatter(x='Month', y='Flight_Distance')
```

[22]: <AxesSubplot: xlabel='Month', ylabel='Flight_Distance'>



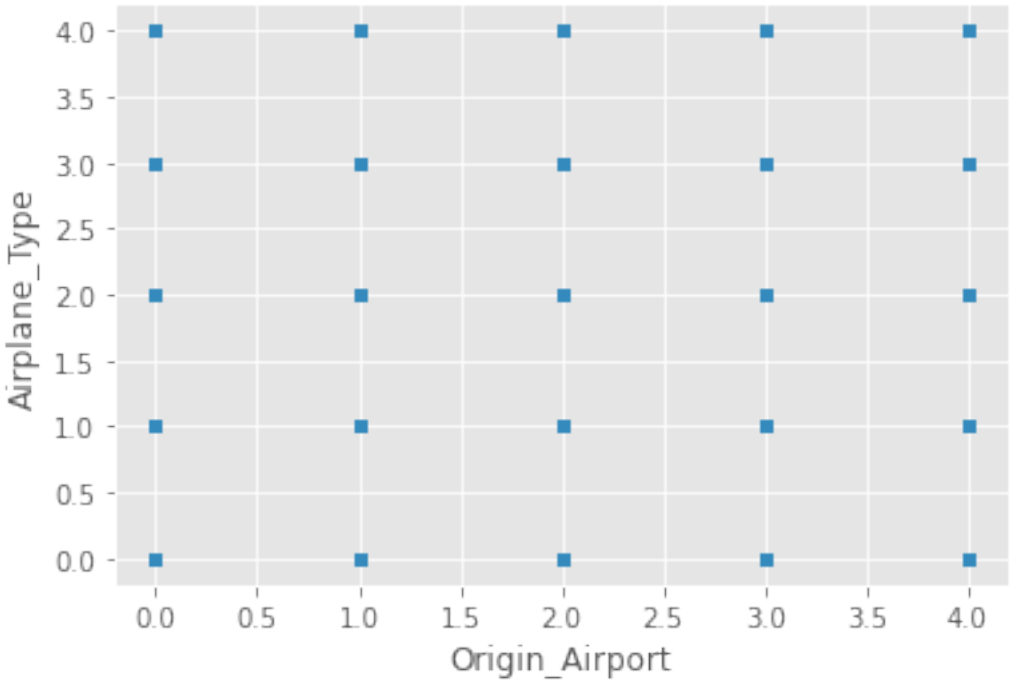
[23]:
```
df.plot.scatter(x='Origin_Airport', y='Airplane_Type')
```

[23]: <AxesSubplot: xlabel='Origin_Airport', ylabel='Airplane_Type'>



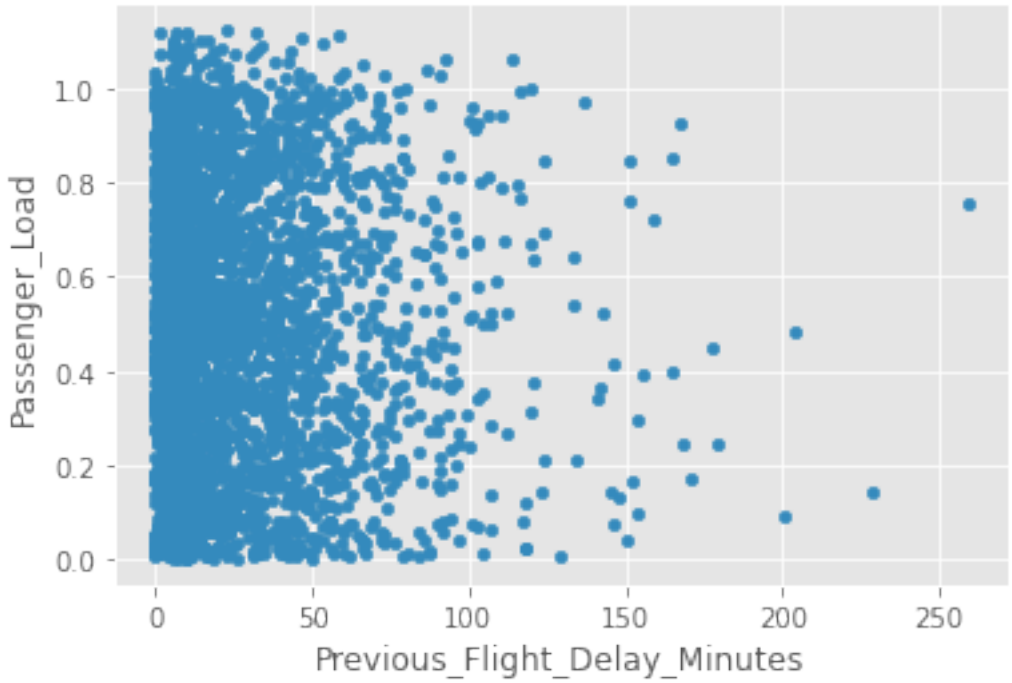
[24]:
```
df.plot.scatter(x='Origin_Airport', y='Airplane_Type')
```

[24]: <AxesSubplot: xlabel='Origin_Airport', ylabel='Airplane_Type'>



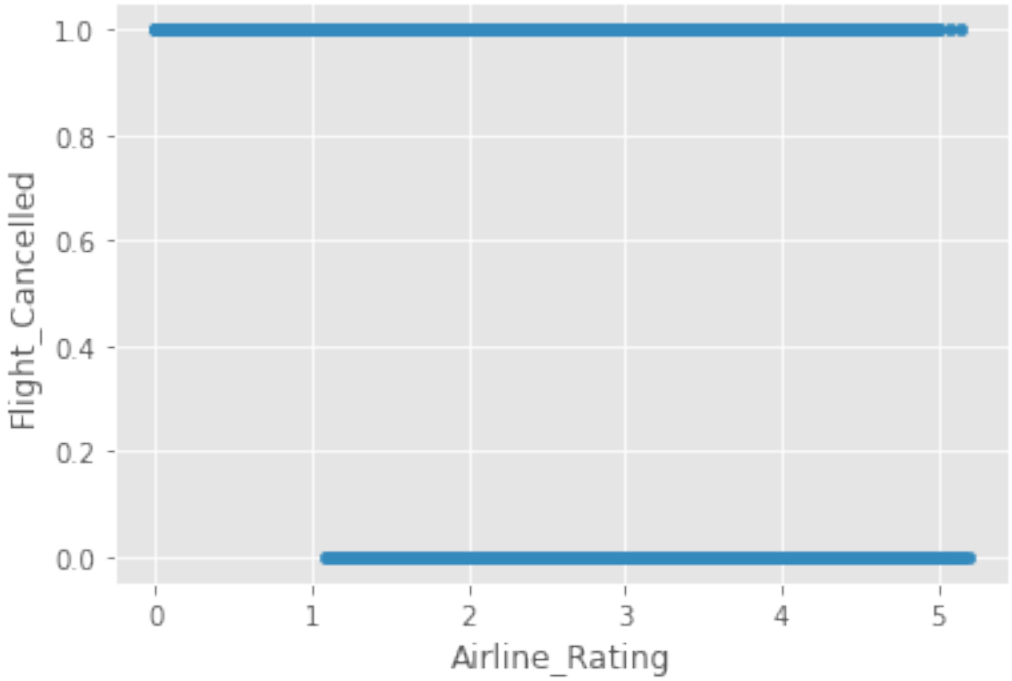
[26]:
```
df.plot.scatter(x='Previous_Flight_Delay_Minutes', y='Passenger_Load')
```

[26]: <AxesSubplot: xlabel='Previous_Flight_Delay_Minutes', ylabel='Passenger_Load'>



[27]:
```
df.plot.scatter(x='Airline_Rating', y='Flight_Cancelled')
```

[27]:
```
<AxesSubplot: xlabel='Airline_Rating', ylabel='Flight_Cancelled'>
```



[35]:
```
df= pd.read_excel('Flyzy Flight Cancellation (1).xlsx')
df.head()
```

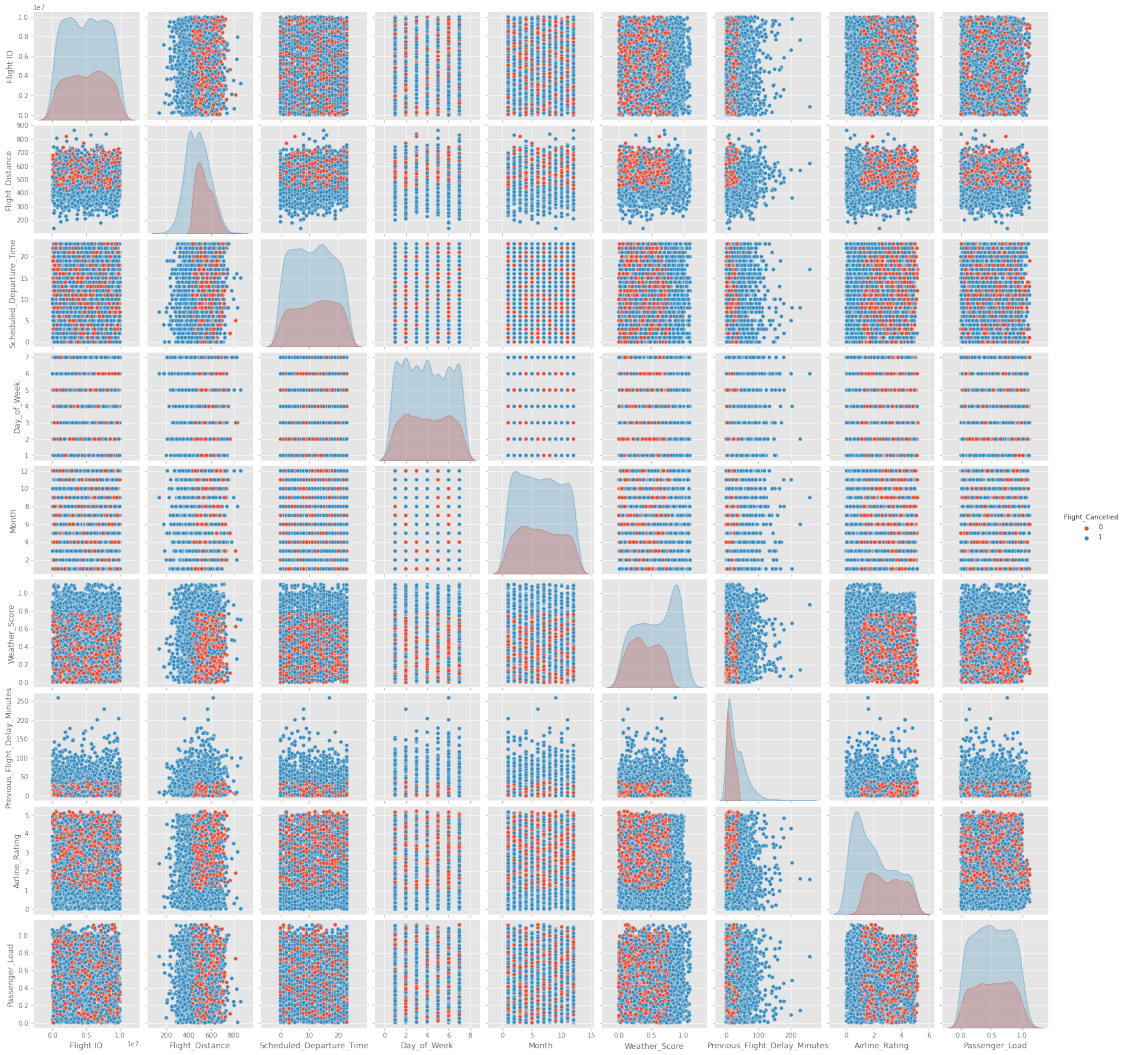
[35]:

	Flight ID	Airline	Flight_Distance	Origin_Airport	Destination_Airport	Scheduled_Departure_Time	Day_of_Week	Month	Airplane_Type	Weather_Score	Previous_Flight_Delay_Minutes	Airline_Rating	Passenger_Load
0	7319483	Airline D	475	Airport 3	Airport 2	4	6	1	Type C	0.225122	5.0	2.151974	0.477202
1	4791965	Airline E	538	Airport 5	Airport 4	12	1	6	Type B	0.060346			
2	2991718	Airline C	565	Airport 1	Airport 2	17	3	9	Type C	0.093920			
3	4220106	Airline E	658	Airport 5	Airport 3	1	1	8	Type B	0.656750			
4	2263008	Airline E	566	Airport 2	Airport 2	19	7	12	Type E	0.505211			

```
1                          68.0         1.600779         0.159718
2                          18.0         4.406848         0.256803
3                          13.0         0.998757         0.504077
4                           4.0         3.806206         0.019638

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

[36]:
```
sns.pairplot(df, hue='Flight_Cancelled')
```

[36]:
```
<seaborn.axisgrid.PairGrid at 0x7fa0724a2e30>
```



[37]:
```
#Relationship analysis
corelation=df.corr()
```

```
/tmp/ipykernel_121/1794495283.py:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  corelation=df.corr()
```

[38]:
```
sns.heatmap(corelation, xticklabels=corelation.columns, yticklabels=corelation.
↪columns
            ,annot=True)
```

[38]: <AxesSubplot: >