Final_proj 4

April 24, 2023

[2]:
```
import pandas as pd
import seaborn as sns
import numpy as np
```

[3]:
```
airlines = pd.read_excel('Airlines.xlsx')
airports = pd.read_excel('airports.xlsx')
runways = pd.read_excel('runways.xlsx')
```

[5]:
```
airlines.head(10)
```

[5]:

	id	Airline	Flight	AirportFrom	AirportTo	DayOfWeek	Time	Length	Delay
0	1	CO	269	SFO	IAH	3	15	205	1
1	2	US	1558	PHX	CLT	3	15	222	1
2	3	AA	2400	LAX	DFW	3	20	165	1
3	4	AA	2466	SFO	DFW	3	20	195	1
4	5	AS	108	ANC	SEA	3	30	202	0
5	6	CO	1094	LAX	IAH	3	30	181	1
6	7	DL	1768	LAX	MSP	3	30	220	0
7	8	DL	2722	PHX	DTW	3	30	228	0
8	9	DL	2606	SFO	MSP	3	35	216	1
9	10	AA	2538	LAS	ORD	3	40	200	1

[7]:
```
runways.head(5)
```

[7]:

	id	airport_ref	airport_ident	length_ft	width_ft	surface	lighted	\
0	269408	6523	00A	80.0	80.0	ASPH-G	1	
1	255155	6524	00AK	2500.0	70.0	GRVL	0	
2	254165	6525	00AL	2300.0	200.0	TURF	0	
3	270932	6526	00AR	40.0	40.0	GRASS	0	
4	322128	322127	00AS	1450.0	60.0	Turf	0	

	closed	le_ident	le_latitude_deg	le_longitude_deg	le_elevation_ft	\
0	0	H1	NaN	NaN	NaN	
1	0	N	NaN	NaN	NaN	
2	0	1	NaN	NaN	NaN	
3	0	H1	NaN	NaN	NaN	
4	0	1	NaN	NaN	NaN	

```
   le_heading_degT  le_displaced_threshold_ft he_ident  he_latitude_deg  \
0              NaN                        NaN      NaN              NaN
1              NaN                        NaN        S              NaN
2              NaN                        NaN       19              NaN
3              NaN                        NaN       H1              NaN
4              NaN                        NaN       19              NaN

   he_longitude_deg  he_elevation_ft  he_heading_degT  \
0               NaN              NaN              NaN
1               NaN              NaN              NaN
2               NaN              NaN              NaN
3               NaN              NaN              NaN
4               NaN              NaN              NaN

   he_displaced_threshold_ft
0                        NaN
1                        NaN
2                        NaN
3                        NaN
4                        NaN
```

[9]:
```
airports.head()
```

[9]:
```
       id ident           type                                name  \
0    6523   00A       heliport                   Total Rf Heliport
1  323361  00AA  small_airport                  Aero B Ranch Airport
2    6524  00AK  small_airport                          Lowell Field
3    6525  00AL  small_airport                          Epps Airpark
4    6526  00AR         closed  Newport Hospital & Clinic Heliport

   latitude_deg  longitude_deg  elevation_ft continent iso_country iso_region  \
0     40.070801     -74.933601          11.0       NaN          US      US-PA
1     38.704022    -101.473911        3435.0       NaN          US      US-KS
2     59.947733    -151.692524         450.0       NaN          US      US-AK
3     34.864799     -86.770302         820.0       NaN          US      US-AL
4     35.608700     -91.254898         237.0       NaN          US      US-AR

   municipality scheduled_service gps_code iata_code local_code home_link  \
0      Bensalem                no      00A       NaN        00A       NaN
1         Leoti                no     00AA       NaN       00AA       NaN
2  Anchor Point                no     00AK       NaN       00AK       NaN
3       Harvest                no     00AL       NaN       00AL       NaN
4       Newport                no      NaN       NaN        NaN       NaN

  wikipedia_link keywords
0            NaN      NaN
```

```
1            NaN      NaN
2            NaN      NaN
3            NaN      NaN
4            NaN     00AR
```

[10]:
```
airlines.shape
```

[10]:
```
(518556, 9)
```

[7]:
```
airlines.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 518556 entries, 0 to 518555
Data columns (total 9 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   id           518556 non-null  int64
 1   Airline      518556 non-null  object
 2   Flight       518556 non-null  int64
 3   AirportFrom  518556 non-null  object
 4   AirportTo    518556 non-null  object
 5   DayOfWeek    518556 non-null  int64
 6   Time         518556 non-null  int64
 7   Length       518556 non-null  int64
 8   Delay        518556 non-null  int64
dtypes: int64(6), object(3)
memory usage: 35.6+ MB
```

[11]:
```
airports.shape
```

[11]:
```
(73805, 18)
```

[12]:
```
airports.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73805 entries, 0 to 73804
Data columns (total 18 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id                 73805 non-null  int64
 1   ident              73805 non-null  object
 2   type               73805 non-null  object
 3   name               73805 non-null  object
 4   latitude_deg       73805 non-null  float64
 5   longitude_deg      73805 non-null  float64
 6   elevation_ft       59683 non-null  float64
 7   continent          38086 non-null  object
 8   iso_country        73546 non-null  object
```

```
 9   iso_region          73805 non-null  object
 10  municipality        68739 non-null  object
 11  scheduled_service   73805 non-null  object
 12  gps_code            42996 non-null  object
 13  iata_code           9160 non-null   object
 14  local_code          32975 non-null  object
 15  home_link           3492 non-null   object
 16  wikipedia_link      10705 non-null  object
 17  keywords            13951 non-null  object
dtypes: float64(3), int64(1), object(14)
memory usage: 10.1+ MB
```

[10]:
```
runways.head()
```

[10]:

	id	airport_ref	airport_ident	length_ft	width_ft	surface	lighted	\
0	269408	6523	00A	80.0	80.0	ASPH-G	1	
1	255155	6524	00AK	2500.0	70.0	GRVL	0	
2	254165	6525	00AL	2300.0	200.0	TURF	0	
3	270932	6526	00AR	40.0	40.0	GRASS	0	
4	322128	322127	00AS	1450.0	60.0	Turf	0	

	closed	le_ident	le_latitude_deg	le_longitude_deg	le_elevation_ft	\
0	0	H1	NaN	NaN	NaN	
1	0	N	NaN	NaN	NaN	
2	0	1	NaN	NaN	NaN	
3	0	H1	NaN	NaN	NaN	
4	0	1	NaN	NaN	NaN	

	le_heading_degT	le_displaced_threshold_ft	he_ident	he_latitude_deg	\
0	NaN	NaN	NaN	NaN	
1	NaN	NaN	S	NaN	
2	NaN	NaN	19	NaN	
3	NaN	NaN	H1	NaN	
4	NaN	NaN	19	NaN	

	he_longitude_deg	he_elevation_ft	he_heading_degT	\
0	NaN	NaN	NaN	
1	NaN	NaN	NaN	
2	NaN	NaN	NaN	
3	NaN	NaN	NaN	
4	NaN	NaN	NaN	

	he_displaced_threshold_ft
0	NaN
1	NaN
2	NaN
3	NaN

```
4                          NaN
```

[11]:
```
runways.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43977 entries, 0 to 43976
Data columns (total 20 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   id                        43977 non-null  int64
 1   airport_ref               43977 non-null  int64
 2   airport_ident             43977 non-null  object
 3   length_ft                 43753 non-null  float64
 4   width_ft                  41088 non-null  float64
 5   surface                   43520 non-null  object
 6   lighted                   43977 non-null  int64
 7   closed                    43977 non-null  int64
 8   le_ident                  43793 non-null  object
 9   le_latitude_deg           15016 non-null  float64
 10  le_longitude_deg          15000 non-null  float64
 11  le_elevation_ft           12781 non-null  float64
 12  le_heading_degT           14624 non-null  float64
 13  le_displaced_threshold_ft  2883 non-null   float64
 14  he_ident                  37332 non-null  object
 15  he_latitude_deg           14971 non-null  float64
 16  he_longitude_deg          14973 non-null  float64
 17  he_elevation_ft           12620 non-null  float64
 18  he_heading_degT           16428 non-null  float64
 19  he_displaced_threshold_ft  3176 non-null   float64
dtypes: float64(12), int64(4), object(4)
memory usage: 6.7+ MB
```

[13]:
```
air_runw = pd.merge(airports, runways,left_on ='ident',right_on =␣
↪'airport_ident', how = 'left')
air_runw.head()
```

[13]:

	id_x	ident	type	name	\
0	6523	00A	heliport	Total Rf Heliport	
1	323361	00AA	small_airport	Aero B Ranch Airport	
2	6524	00AK	small_airport	Lowell Field	
3	6525	00AL	small_airport	Epps Airpark	
4	6526	00AR	closed	Newport Hospital & Clinic Heliport	

	latitude_deg	longitude_deg	elevation_ft	continent	iso_country	iso_region	\
0	40.070801	-74.933601	11.0	NaN	US	US-PA	
1	38.704022	-101.473911	3435.0	NaN	US	US-KS	
2	59.947733	-151.692524	450.0	NaN	US	US-AK	

```
3     34.864799     -86.770302         820.0       NaN          US      US-AL
4     35.608700     -91.254898         237.0       NaN          US      US-AR

   … le_longitude_deg le_elevation_ft le_heading_degT  \
0  …              NaN             NaN             NaN
1  …              NaN             NaN             NaN
2  …              NaN             NaN             NaN
3  …              NaN             NaN             NaN
4  …              NaN             NaN             NaN

  le_displaced_threshold_ft he_ident he_latitude_deg he_longitude_deg  \
0                       NaN      NaN             NaN              NaN
1                       NaN      NaN             NaN              NaN
2                       NaN        S             NaN              NaN
3                       NaN       19             NaN              NaN
4                       NaN       H1             NaN              NaN

  he_elevation_ft  he_heading_degT  he_displaced_threshold_ft
0             NaN              NaN                        NaN
1             NaN              NaN                        NaN
2             NaN              NaN                        NaN
3             NaN              NaN                        NaN
4             NaN              NaN                        NaN

[5 rows x 38 columns]
```

[14]:
```
air_runw.columns
```

[14]:
```
Index(['id_x', 'ident', 'type', 'name', 'latitude_deg', 'longitude_deg',
       'elevation_ft', 'continent', 'iso_country', 'iso_region',
       'municipality', 'scheduled_service', 'gps_code', 'iata_code',
       'local_code', 'home_link', 'wikipedia_link', 'keywords', 'id_y',
       'airport_ref', 'airport_ident', 'length_ft', 'width_ft', 'surface',
       'lighted', 'closed', 'le_ident', 'le_latitude_deg', 'le_longitude_deg',
       'le_elevation_ft', 'le_heading_degT', 'le_displaced_threshold_ft',
       'he_ident', 'he_latitude_deg', 'he_longitude_deg', 'he_elevation_ft',
       'he_heading_degT', 'he_displaced_threshold_ft'],
      dtype='object')
```

[15]:
```
count_runway = air_runw.groupby('airport_ident')[['id_y']].count().
 ↪sort_values(by = 'id_y', ascending = False).reset_index()
count_runway.head()
```

[15]:
```
  airport_ident  id_y
0          KORD    11
1          KNHU    10
2           JRA     9
```

```
3          TA12     8
4           SXS     8
```

[18]:
```
count_runway.tail()
```

[18]:
```
      airport_ident  id_y
37260           H43     1
37261           H28     1
37262          33LS     1
37263           H11     1
37264          rjns     1
```

[16]:
```
air_run = pd.merge(airports, count_runway, how = 'left', left_on = 'ident',␣
↪right_on = 'airport_ident')[['iata_code', 'type', 'elevation_ft','id_y']]
air_run.rename(columns = {'id_y': 'runway_count'}, inplace = True)
```

[17]:
```
air_run.head()
```

[17]:
```
  iata_code           type  elevation_ft  runway_count
0       NaN       heliport          11.0           1.0
1       NaN  small_airport        3435.0           NaN
2       NaN  small_airport         450.0           1.0
3       NaN  small_airport         820.0           1.0
4       NaN         closed         237.0           1.0
```

[18]:
```
air_run.dropna().to_csv('run_2.csv', index = False)
```

[19]:
```
combined_data = pd.merge(airlines, air_run, how = 'left', left_on =␣
↪'AirportFrom', right_on = 'iata_code')

new_names = list(combined_data[air_run.columns].columns + '_source_airport')
old_names = list(combined_data[air_run.columns].columns)

combined_data.rename(columns = {old:new for old,new in zip(old_names,␣
↪new_names)}, inplace = True)
combined_data.head(2)
```

[19]:
```
   id Airline  Flight AirportFrom AirportTo  DayOfWeek  Time  Length  Delay  \
0   1      CO     269         SFO       IAH          3    15     205      1
1   2      US    1558         PHX       CLT          3    15     222      1

  iata_code_source_airport type_source_airport  elevation_ft_source_airport  \
0                      SFO       large_airport                         13.0
1                      PHX       large_airport                       1135.0

   runway_count_source_airport
0                          4.0
```

```
1                          3.0
```

[20]:
```
combined_data.columns
```

[20]:
```
Index(['id', 'Airline', 'Flight', 'AirportFrom', 'AirportTo', 'DayOfWeek',
       'Time', 'Length', 'Delay', 'iata_code_source_airport',
       'type_source_airport', 'elevation_ft_source_airport',
       'runway_count_source_airport'],
      dtype='object')
```

[21]:
```
combined_data = pd.merge(combined_data, air_run, how = 'left', left_on =␣
↪'AirportTo', right_on = 'iata_code')
```

[22]:
```
new_names = list(combined_data[air_run.columns].columns + '_dest_airport')
old_names = list(combined_data[air_run.columns].columns)
combined_data.rename(columns = {old:new for old,new in zip(old_names,␣
↪new_names)}, inplace = True)
combined_data.head(2)
```

[22]:
```
   id Airline  Flight AirportFrom AirportTo  DayOfWeek  Time  Length  Delay  \
0   1      CO     269         SFO       IAH          3    15     205      1
1   2      US    1558         PHX       CLT          3    15     222      1

  iata_code_source_airport type_source_airport  elevation_ft_source_airport  \
0                      SFO       large_airport                         13.0
1                      PHX       large_airport                       1135.0

   runway_count_source_airport iata_code_dest_airport type_dest_airport  \
0                          4.0                   IAH     large_airport
1                          3.0                   CLT     large_airport

   elevation_ft_dest_airport  runway_count_dest_airport
0                       97.0                       5.0
1                      748.0                       4.0
```

[23]:
```
combined_data.drop(columns =  list(combined_data.columns[combined_data.columns.
↪str.startswith('iata_code')]), inplace = True)
```

[24]:
```
import requests
from bs4 import BeautifulSoup
```

[25]:
```
website = requests.get('https://en.wikipedia.org/wiki/
↪List_of_airlines_of_the_United_States').text
```

[26]:
```
soup = BeautifulSoup(website,'lxml')
```

[27]:
```
My_table = soup.findAll("table",{"class":"wikitable"})
```

[28]:
```
len(My_table)
```

[28]: 7

[29]:
```
airlines_wiki_list = []
for tab in My_table:
    temp = pd.read_html(str(tab))
    temp = pd.DataFrame(temp[0])
    airlines_wiki_list.append(temp)
```

[30]:
```
airlines_wiki = pd.concat(airlines_wiki_list)
```

[31]:
```
combined_data.head(2)
```

[31]:

	id	Airline	Flight	AirportFrom	AirportTo	DayOfWeek	Time	Length	Delay	\
0	1	CO	269	SFO	IAH	3	15	205	1	
1	2	US	1558	PHX	CLT	3	15	222	1	

	type_source_airport	elevation_ft_source_airport	\
0	large_airport	13.0	
1	large_airport	1135.0	

	runway_count_source_airport	type_dest_airport	elevation_ft_dest_airport	\
0	4.0	large_airport	97.0	
1	3.0	large_airport	748.0	

	runway_count_dest_airport
0	5.0
1	4.0

[32]:
```
airlines_founded = pd.merge(combined_data[['Airline']].
↪drop_duplicates(),airlines_wiki[['IATA', 'Founded']].drop_duplicates(),
        how = 'left', left_on = 'Airline', right_on = 'IATA')
```

[33]:
```
airlines_founded
```

[33]:

	Airline	IATA	Founded
0	CO	NaN	NaN
1	US	NaN	NaN
2	AA	AA	1926.0
3	AS	AS	1932.0
4	DL	DL	1924.0
5	B6	B6	1998.0
6	HA	HA	1929.0
7	OO	OO	1972.0
8	9E	9E	1985.0
9	OH	OH	1979.0

```
10      EV  NaN      NaN
11      XE   XE   2016.0
12      YV   YV   1980.0
13      UA   UA   1926.0
14      MQ   MQ   1984.0
15      F9   F9   1994.0
16      WN   WN   1967.0
```

[34]:
```
website_url = requests.get('https://en.wikipedia.org/wiki/
 ↪List_of_the_busiest_airports_in_the_United_States').text
soup = BeautifulSoup(website_url,'lxml')
My_table = soup.findAll("table",{"class":"wikitable"})
```

[35]:
```
hub_data = {}
i = 0
for tab in My_table:
    hub_data[i] = pd.read_html(str(tab))
    hub_data[i] = pd.DataFrame(hub_data[i][0])
    i +=1
```

[36]:
```
large_hub = hub_data[0].copy()
med_hub = hub_data[1].copy()
```

[37]:
```
large_hub.insert(loc =1, column= 'Hub_type', value = 'large')
med_hub.insert(loc =1, column= 'Hub_type', value = 'medium')
```

[38]:
```
column_temp = large_hub.columns.str.split('[([]').str[0].str.strip().str.
 ↪lower().str.replace(' ','_').values
column_temp[list(map( lambda x : x.isnumeric(), column_temp))]  = 'data_' +␣
 ↪column_temp[list(map( lambda x : x.isnumeric(), column_temp))]
large_hub.columns = column_temp
large_hub.columns
```

[38]:
```
Index(['rank', 'hub_type', 'airports', 'iatacode', 'major_cities_served',
       'state', 'data_2021', 'data_2020', 'data_2019', 'data_2018',
       'data_2017', 'data_2016', 'data_2015', 'data_2014', 'data_2013',
       'data_2012'],
      dtype='object')
```

[39]:
```
column_temp = med_hub.columns.str.split('[([]').str[0].str.strip().str.lower().
 ↪str.replace(' ','_').values
column_temp[list(map( lambda x : x.isnumeric(), column_temp))]  = 'data_' +␣
 ↪column_temp[list(map( lambda x : x.isnumeric(), column_temp))]
med_hub.columns = column_temp
med_hub.columns
```

[39]:
```
Index(['rank', 'hub_type', 'airports', 'iatacode', 'city_served', 'state',
       'data_2021', 'data_2020', 'data_2019', 'data_2018', 'data_2017',
       'data_2016', 'data_2015', 'data_2014', 'data_2013', 'data_2012'],
      dtype='object')
```

[40]:
```
large_hub.rename(columns = {'major_cities_served':'city_served'}, inplace =␣
↪True)
```

[41]:
```
final_hub_data = pd.concat([large_hub, med_hub])
```

[42]:
```
final_hub_data.head(2)
```

[42]:

	rank	hub_type	airports	iatacode	city_served	state	data_2021	data_2020	data_2019	data_2018	data_2017	data_2016	data_2015	data_2014	data_2013	data_2012
0	1	large	Hartsfield-Jackson Atlanta International Airport	ATL	Atlanta	GA	36676010	20559866	53505795	51865797	50251964	50501858	49340732	46604273	45308407	45798928
1	2	large	Dallas/Fort Worth International Airport	DFW	Dallas & Fort Worth	TX	30005266	18593421	35778573	32821799	31816933	31283579	31589839	30804567	29038128	28022904

[43]:
```
final_hub_data.data_2019.isnull().sum()
```

[43]:
```
0
```

[44]:
```
combined_data_pax = pd.merge(combined_data, final_hub_data[['iatacode',␣
↪'data_2019']],how = 'left' , left_on = 'AirportFrom', right_on = 'iatacode')
```

[45]:
```
combined_data_pax.rename(columns = {'iatacode': 'iatacode_source' ,'data_2019':␣
↪'data_2019_source_airport'}, inplace = True)
```

[46]:
```
combined_data_pax = pd.merge(combined_data_pax, final_hub_data[['iatacode',␣
↪'data_2019']],how = 'left' , left_on = 'AirportTo', right_on = 'iatacode')
```

[47]:
```
combined_data_pax.rename(columns = {'iatacode': 'iatacode_dest' ,'data_2019':␣
↪'data_2019_dest_airport'}, inplace = True)
```

[48]:
```
combined_data_pax = combined_data_pax.loc[:,~combined_data_pax.columns.str.
↪startswith('iatacode')].copy()
```

[49]:
```
combined_data_pax
```

[49]:

	id	Airline	Flight	AirportFrom	AirportTo	DayOfWeek	Time	Length
0	1	CO	269	SFO	IAH	3	15	205

```
1            2      US    1558         PHX       CLT          3    15     222
2            3      AA    2400         LAX       DFW          3    20     165
3            4      AA    2466         SFO       DFW          3    20     195
4            5      AS     108         ANC       SEA          3    30     202
…          …     …     …          …       …        …   …     …
518551  539377      B6     717         JFK       SJU          5  1439     220
518552  539378      B6     739         JFK       PSE          5  1439     223
518553  539379      CO     178         OGG       SNA          5  1439     326
518554  539382      UA      78         HNL       SFO          5  1439     313
518555  539383      US    1442         LAX       PHL          5  1439     301

        Delay type_source_airport  elevation_ft_source_airport  \
0           1       large_airport                         13.0
1           1       large_airport                       1135.0
2           1       large_airport                        125.0
3           1       large_airport                         13.0
4           0       large_airport                        152.0
…         …                 …                          …
518551      1       large_airport                         13.0
518552      1       large_airport                         13.0
518553      0      medium_airport                         54.0
518554      1       large_airport                         13.0
518555      1       large_airport                        125.0

        runway_count_source_airport type_dest_airport  \
0                               4.0     large_airport
1                               3.0     large_airport
2                               4.0     large_airport
3                               4.0     large_airport
4                               3.0     large_airport
…                             …              …
518551                          4.0     large_airport
518552                          4.0    medium_airport
518553                          2.0     large_airport
518554                          6.0     large_airport
518555                          4.0     large_airport

        elevation_ft_dest_airport  runway_count_dest_airport  \
0                           97.0                        5.0
1                          748.0                        4.0
2                          607.0                        7.0
3                          607.0                        7.0
4                          433.0                        4.0
…                          …                          …
518551                       9.0                        2.0
518552                      29.0                        1.0
518553                      56.0                        2.0
```

```
518554                        13.0                         4.0
518555                        36.0                         4.0

        data_2019_source_airport  data_2019_dest_airport
0                     27779230.0              21905309.0
1                     22433552.0              24199688.0
2                     42939104.0              35778573.0
3                     27779230.0              35778573.0
4                      2713843.0              25001762.0
…                            …                       …
518551                31036655.0               4590117.0
518552                31036655.0                     NaN
518553                 3791807.0               5153276.0
518554                 9988678.0              27779230.0
518555                42939104.0              16006389.0

[518556 rows x 17 columns]
```

[50]:
```
airlines_founded
```

[50]:
```
   Airline IATA  Founded
0       CO  NaN      NaN
1       US  NaN      NaN
2       AA   AA   1926.0
3       AS   AS   1932.0
4       DL   DL   1924.0
5       B6   B6   1998.0
6       HA   HA   1929.0
7       OO   OO   1972.0
8       9E   9E   1985.0
9       OH   OH   1979.0
10      EV  NaN      NaN
11      XE   XE   2016.0
12      YV   YV   1980.0
13      UA   UA   1926.0
14      MQ   MQ   1984.0
15      F9   F9   1994.0
16      WN   WN   1967.0
```

[51]:
```
combined_data_pax = pd.merge(combined_data_pax, airlines_founded[['Airline',␣
↪'Founded']], on = 'Airline')
```

[52]:
```
combined_data_pax.head(2)
```

[52]:
```
   id Airline  Flight AirportFrom AirportTo  DayOfWeek  Time  Length  Delay  \
0   1      CO     269         SFO       IAH          3    15     205      1
1   6      CO    1094         LAX       IAH          3    30     181      1
```

```
  type_source_airport  elevation_ft_source_airport  \
0       large_airport                         13.0
1       large_airport                        125.0

   runway_count_source_airport type_dest_airport  elevation_ft_dest_airport  \
0                          4.0     large_airport                       97.0
1                          4.0     large_airport                       97.0

   runway_count_dest_airport  data_2019_source_airport  \
0                        5.0                27779230.0
1                        5.0                42939104.0

   data_2019_dest_airport  Founded
0              21905309.0      NaN
1              21905309.0      NaN
```

[53]:
```
combined_data_pax.isna().sum().sort_values(ascending = False)
```

[53]:
```
Founded                        83601
data_2019_source_airport       83582
data_2019_dest_airport         83531
runway_count_dest_airport         31
elevation_ft_dest_airport         31
type_dest_airport                 31
runway_count_source_airport       31
elevation_ft_source_airport       31
type_source_airport               31
AirportTo                          0
Airline                            0
Flight                             0
AirportFrom                        0
Delay                              0
DayOfWeek                          0
Time                               0
Length                             0
id                                 0
dtype: int64
```

[54]:
```
combined_data_pax[combined_data_pax.type_source_airport.isna()].AirportFrom.
↪unique()
```

[54]:
```
array(['CYS'], dtype=object)
```

[55]:
```
airport_dict = pd.read_excel('Data Dictionary.xlsx', sheet_name =␣
↪'airlines',header = 29)
```

[56]:
```
airport_dict.head()
```

[56]:

	Aiport ID	Description	Unnamed: 2
0	ABE	RAF Calveley	NaN
1	ABE	Bisho Airport	NaN
2	ABE	Beica Airport	NaN
3	ABE	Lehigh Valley International Airport	NaN
4	ABE	Bethel Airport	NaN

[57]:
```
airport_dict = pd.read_excel('Data Dictionary.xlsx', sheet_name =␣
↪'airlines',header = 29, usecols = [0,1])
airport_dict.head(2)
```

[57]:

	Aiport ID	Description
0	ABE	RAF Calveley
1	ABE	Bisho Airport

[58]:
```
name = airport_dict[airport_dict['Aiport ID'] == 'CYS'].Description.values[0]
name.lower()
```

[58]:
```
'cheyenne regional jerry olson field'
```

[59]:
```
air_miss = airports.loc[name.lower() == airports.name.str.lower(), ['ident',␣
↪'name', 'iata_code', 'type', 'elevation_ft']]
```

[60]:
```
air_miss_comb = pd.merge(air_miss, runways[['airport_ident', 'id']], how =␣
↪'left', left_on = 'ident', right_on = 'airport_ident')
runway_count_miss = air_miss_comb.groupby('ident')[['id']].count().
↪sort_values(by = 'id', ascending = False).reset_index()
runway_count_miss
```

[60]:

	ident	id
0	KCYS	2

[61]:
```
air_miss_data = pd.merge(air_miss,runway_count_miss ).rename(columns = {'id' :␣
↪'runway_count'})[['iata_code', 'type', 'elevation_ft', 'runway_count']]
```

[62]:
```
combined_data_pax.loc[combined_data_pax.AirportFrom == 'CYS',␣
↪'type_source_airport'] = air_miss_data.type.values[0]
combined_data_pax.loc[combined_data_pax.AirportFrom == 'CYS',␣
↪'elevation_ft_source_airport'] = air_miss_data.elevation_ft.values[0]
combined_data_pax.loc[combined_data_pax.AirportFrom == 'CYS',␣
↪'runway_count_source_airport'] = air_miss_data.runway_count.values[0]
```

[63]:
```
combined_data_pax.isna().sum().sort_values(ascending = False)
```

[63]:
```
Founded                          83601
data_2019_source_airport         83582
data_2019_dest_airport           83531
runway_count_dest_airport           31
elevation_ft_dest_airport           31
type_dest_airport                   31
DayOfWeek                            0
Airline                              0
Flight                               0
AirportFrom                          0
AirportTo                            0
Delay                                0
Time                                 0
Length                               0
type_source_airport                  0
elevation_ft_source_airport          0
runway_count_source_airport          0
id                                   0
dtype: int64
```

[64]:
```
airline_dict = pd.read_excel('Data Dictionary.xlsx', sheet_name =␣
 ↪'airlines',header = 10, usecols = [0,1])
airline_dict.head(2)
```

[64]:

	Airlines ID	Description
0	WN	Southwest
1	DL	Delta

[65]:
```
miss_founded = combined_data_pax[combined_data_pax.Founded.isna()].Airline.
 ↪unique()
print(airline_dict[airline_dict['Airlines ID'].isin( ['EV', 'CO', 'US'])])
```

```
  Airlines ID                          Description
5          US  PSA (initially US Airway Express)
7          EV                           ExpressJet
9          CO     United Airlines (initially CO)
```

[66]:
```
miss_val = {'US' : 1967, 'CO' : 1934, 'EV' : 1986}
for aline in miss_founded:
    combined_data_pax.loc[(combined_data_pax.Founded.isna()) &
                          (combined_data_pax.Airline == aline), 'Founded'] =␣
 ↪miss_val[aline]
```

[67]:
```
(combined_data_pax.isna().sum().sort_values(ascending = False)/
 ↪combined_data_pax.shape[0])*100
```

[67]:
```
data_2019_source_airport       16.118221
data_2019_dest_airport         16.108386
runway_count_dest_airport       0.005978
elevation_ft_dest_airport       0.005978
type_dest_airport               0.005978
Founded                         0.000000
DayOfWeek                       0.000000
Airline                         0.000000
Flight                          0.000000
AirportFrom                     0.000000
AirportTo                       0.000000
Delay                           0.000000
Time                            0.000000
Length                          0.000000
type_source_airport             0.000000
elevation_ft_source_airport     0.000000
runway_count_source_airport     0.000000
id                              0.000000
dtype: float64
```

[68]:
```
combined_data_pax.groupby('type_source_airport')['data_2019_source_airport'].
↪median()
```

[68]:
```
type_source_airport
large_airport     21905309.0
medium_airport     3323614.0
small_airport            NaN
Name: data_2019_source_airport, dtype: float64
```

[69]:
```
med_val = combined_data_pax.
↪groupby('type_source_airport')['data_2019_source_airport'].median()
med_val
```

[69]:
```
type_source_airport
large_airport     21905309.0
medium_airport     3323614.0
small_airport            NaN
Name: data_2019_source_airport, dtype: float64
```

[70]:
```
(combined_data_pax.isna().sum().sort_values(ascending = False)/
↪combined_data_pax.shape[0])*100
```

[70]:
```
data_2019_source_airport       16.118221
data_2019_dest_airport         16.108386
runway_count_dest_airport       0.005978
elevation_ft_dest_airport       0.005978
type_dest_airport               0.005978
```

```
Founded                          0.000000
DayOfWeek                        0.000000
Airline                          0.000000
Flight                           0.000000
AirportFrom                      0.000000
AirportTo                        0.000000
Delay                            0.000000
Time                             0.000000
Length                           0.000000
type_source_airport              0.000000
elevation_ft_source_airport      0.000000
runway_count_source_airport      0.000000
id                               0.000000
dtype: float64
```

[71]:
```
airline_dict = pd.read_excel('Data Dictionary.xlsx', sheet_name =␣
↪'airlines',header = 11)
airline_dict.head()
```

[71]:
```
   WN                          Southwest  \
0  DL                              Delta
1  OO                            Skywest
2  AA                 American Airlines
3  MQ                              Envoy
4  US  PSA (initially US Airway Express)

                                          Unnamed: 2
0                                                NaN
1                                                NaN
2                                                NaN
3                                                NaN
4   There was a US Airways Express, which now has…
```

[72]:
```
id_airline = airline_dict.loc[airline_dict['Description'].str.strip().str.
↪lower() == 'southwest', 'Airlines ID'].values[0]
```

```
      ␣
↪---------------------------------------------------------------------------

       KeyError                                  Traceback (most recent call␣
↪last)

       /usr/local/lib/python3.7/site-packages/pandas/core/indexes/base.py in␣
↪get_loc(self, key, method, tolerance)
      2897             try:
   -> 2898                 return self._engine.get_loc(casted_key)
```

```
      2899             except KeyError as err:

       pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

       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: 'Description'


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

       KeyError                                  Traceback (most recent call␣
↪last)

       <ipython-input-72-83c456917023> in <module>
   ----> 1 id_airline = airline_dict.loc[airline_dict['Description'].str.
↪strip().str.lower() == 'southwest', 'Airlines ID'].values[0]

       /usr/local/lib/python3.7/site-packages/pandas/core/frame.py in␣
↪__getitem__(self, key)
      2904             if self.columns.nlevels > 1:
      2905                 return self._getitem_multilevel(key)
   -> 2906             indexer = self.columns.get_loc(key)
      2907             if is_integer(indexer):
      2908                 indexer = [indexer]

       /usr/local/lib/python3.7/site-packages/pandas/core/indexes/base.py in␣
↪get_loc(self, key, method, tolerance)
      2898                 return self._engine.get_loc(casted_key)
      2899             except KeyError as err:
   -> 2900                 raise KeyError(key) from err
      2901
      2902         if tolerance is not None:
```

```
KeyError: 'Description'
```

[ ]:
```
def percent_Delay(x):
    return round(x.sum()/x.size * 100,2)
```

[ ]:
```
delay_perc = combined_data_pax.groupby('Airline')['Delay'].agg(percent_Delay)
delay_perc = delay_perc.reset_index()
```

[ ]:
```
plot_data = pd.merge(delay_perc, airline_dict, left_on = 'Airline',
                     right_on = 'Airlines ID', how = 'left')['Airline',␣
 ↪'Description', 'Delay']
plot_data
```

[ ]:
```
import matplotlib.pyplot as plt
plt.figure(figsize = (22,5))
plt.bar(plot_data.Description, height =  plot_data.Delay, color = plt.
 ↪get_cmap('Set3').colors)
for v, idx in zip(plot_data.Delay.values,plot_data.index ):
    plt.annotate('{:.1f}%'.format(v), xy = (idx-0.15, v), size = 18, family =␣
 ↪'times')
plt.ylim(1,85)
plt.xticks(size = 13, rotation = 75)
plt.yticks(size = 13)
plt.title('% delays for each airline', size = 25, color = 'midnightblue',␣
 ↪weight = 'heavy', family = 'times')
plt.show()
```

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

[ ]:
```
delay_perc_weekday = combined_data_pax.groupby('DayOfWeek')['Delay'].
 ↪agg(percent_Delay)
delay_perc_weekday
```

[ ]:
```
plt.figure(figsize = (20,5))
plt.bar(delay_perc_weekday.index, height =  delay_perc_weekday.values, color =␣
 ↪plt.get_cmap('Set3').colors)
for v, idx in zip(delay_perc_weekday.values, range(1, len(delay_perc_weekday.
 ↪index)+1)):
    # print(v, idx)
    plt.annotate('{:.1f}%'.format(v), xy = (idx-0.15, v), size = 18, family =␣
 ↪'times')
plt.ylim(1,65)
plt.xticks(size = 13)
plt.yticks(size = 13)
```

```
plt.title('% delays for each airline', size = 25, color = 'midnightblue',␣
 ↪weight = 'heavy', family = 'times')
plt.show()
```

[ ]:
```
duration_data = combined_data_pax[['Airline', 'Length', 'Delay']].copy()
```

[ ]:
```
duration_data['duration'] = pd.cut(duration_data.Length, 3, labels = ['short',␣
 ↪'medium', 'long'])
duration_data_grp = duration_data.groupby(['Airline','duration'])['Delay'].agg(
    percent_Delay).reset_index().pivot(index = 'Airline',
                                       columns = 'duration').fillna(0)['Delay']
duration_data_grp.columns = duration_data_grp.columns.astype(str)
duration_data_grp.reset_index()
```

[ ]:
```
duration_data.index
```

[ ]:
```
airline_dict
```

[ ]:
```
airline_dict.Description = airline_dict.Description.str.strip()
duration_data_grp = pd.merge(duration_data_grp,airline_dict[['Airlines ID',␣
 ↪'Description']],
         left_on = 'Airline', right_on = 'Airlines ID',
         how = 'left')
duration_data_grp
```

[ ]:
```
combined_data_pax.Airline.nunique()
```

[ ]:
```
long = duration_data_grp[duration_data_grp.long == duration_data_grp.long.
 ↪min()].Description.values.tolist()
print('Airlines with no delays for long flights :\n',', '.join(long))
medium = duration_data_grp[duration_data_grp.medium == duration_data_grp.medium.
 ↪min()].Description.values.tolist()
print('\nAirlines with no delays for medium flights :\n', ', '.join(medium))
short = duration_data_grp[duration_data_grp.short == duration_data_grp.short.
 ↪min()].Description.values.tolist()
print('\nAirlines with no delays for short flights :\n',  ', '.join(short)     )
```

[ ]:
```
combined_data_pax['duration'] = pd.cut(combined_data_pax.Length, 3, labels =␣
 ↪['short', 'medium', 'long'])
```

[ ]:
```
combined_data_pax.head(2)
```

[ ]:
```
pd.crosstab(combined_data_pax.Time, combined_data_pax.duration)['long']
```

[ ]:
```
y = pd.crosstab(combined_data_pax.Time, combined_data_pax.duration)['long'].
 ↪index
```

```
x = pd.crosstab(combined_data_pax.Time, combined_data_pax.duration)['long'].
↪values
```

[ ]:
```
filter_data = combined_data_pax.loc[combined_data_pax.duration == 'long',␣
↪['Time', 'duration']]
```

[ ]:
```
filter_data.Time.describe()
```

[ ]:
```
plt.hist(filter_data.Time, bins = 12)
plt.show()
```

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

[ ]:
```
combined_data_pax.groupby('type_source_airport')[['Delay']].agg('sum').plot.
↪bar()
```

[ ]:
```
combined_data_pax.groupby('type_dest_airport')[['Delay']].agg('sum').plot.bar()
```

[ ]:
```
cols = ['iatacode'] + final_hub_data.columns[ final_hub_data.columns.str.
↪startswith('data_')].tolist()
```

[ ]:
```
time_series = final_hub_data.loc[final_hub_data.hub_type == 'large', cols].
↪set_index('iatacode').T
```

[ ]:
```
time_series['ATL']
```

[ ]:
```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,8))
for ser in time_series.columns:
    plt.plot(time_series[ser], label = ser)
    plt.legend()
plt.show(block = True)
```

[ ]:
```
import seaborn as sns
import statsmodels.api as sm
for ser in time_series.columns[:4]:
    series = time_series[ser].copy()
    series.index = pd.to_datetime(series.index.str.replace('data_',''))
    series.sort_index(inplace = True)
    decomposition = sm.tsa.seasonal_decompose(series, model='multiplicative')
    decomposition.plot()
plt.show()
```

[ ]:
```
 from sklearn.metrics import mean_absolute_percentage_error
error = {}
forecast_2022 = {}
f = {}
```

```
wind_min = {}
win_min_mape = {}
for ser in time_series.columns:
    series = time_series[ser].copy()
    series.index = pd.to_datetime(series.index.str.replace('data_',''))
    series.sort_index(inplace = True)
    test = series[-1:]
    train = series[:-1]
    err_temp = {}
    fore_2022 = {}
    for window in range(2,10):
        forecast = series.rolling(window).mean()
        # accuracy
        mape = round(mean_absolute_percentage_error(test, forecast[-1:]),4)
        err_temp.update({window : mape})
        # forecast for 2022
        fore_2022.update({window : series[-window:].mean()})
    err_ser = pd.Series(err_temp)
    min_wind = err_ser[(err_ser == err_ser.min())].index.values[0]
    forecast_2022.update({ser : round(series[-min_wind:].mean(),2)})
    wind_min.update({ser : min_wind})
    win_min_mape.update({ser :err_temp[min_wind] })
    f.update({ser :pd.Series(fore_2022).round(2) })
    error.update({ser : err_ser})
```

[ ]:
```
win_min_mape
```

[ ]:
```
sma_forecast = pd.DataFrame(f)
sma_error = pd.DataFrame(error)
```

[ ]:
```
sma_prediction = pd.DataFrame(forecast_2022.values(), index = forecast_2022.
↪keys(), columns = ['forecast_2022'] )
sma_prediction['window_used'] = wind_min.values()
sma_prediction['mape_at_window'] = win_min_mape.values()
```

[ ]:
```
sma_prediction
```

[ ]:
```
import scipy.stats as stats
from sklearn.linear_model import SGDClassifier
from sklearn.model_selection import StratifiedKFold, RandomizedSearchCV,␣
↪train_test_split
from statsmodels.formula.api import glm
import statsmodels.api as sm
from sklearn.preprocessing import OrdinalEncoder, StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, accuracy_score
from sklearn.tree import DecisionTreeClassifier
```

```
from xgboost import XGBClassifier
H0 : avg elevation for Delayed flights - avg elevation for not Delayed flights␣
 ↪= 0
Ha : avg elevation for Delayed flights - avg elevation for not Delayed flights !
 ↪= 0
```

[73]:
```
combined_data_pax
```

[73]:
```
            id Airline  Flight AirportFrom AirportTo  DayOfWeek  Time  Length  \
0            1      CO     269         SFO       IAH          3    15     205
1            6      CO    1094         LAX       IAH          3    30     181
2           11      CO     223         ANC       SEA          3    49     201
3           18      CO    1496         LAS       IAH          3    60     162
4           20      CO     507         ONT       IAH          3    75     167
…          …      …      …          …        …         …   …      …
518551  538750      WN    2601         LAS       SMF          5  1230      85
518552  538783      WN    1936         SMF       SAN          5  1235      85
518553  538810      WN    2629         LAS       RNO          5  1240      75
518554  538833      WN    1226         SFO       LAX          5  1245      75
518555  538834      WN    2370         LAX       SFO          5  1245      75

        Delay type_source_airport  elevation_ft_source_airport  \
0           1       large_airport                         13.0
1           1       large_airport                        125.0
2           1       large_airport                        152.0
3           0       large_airport                       2181.0
4           0       large_airport                        944.0
…       …                   …                          …
518551      1       large_airport                       2181.0
518552      1       large_airport                         27.0
518553      1       large_airport                       2181.0
518554      1       large_airport                         13.0
518555      1       large_airport                        125.0

        runway_count_source_airport type_dest_airport  \
0                               4.0     large_airport
1                               4.0     large_airport
2                               3.0     large_airport
3                               4.0     large_airport
4                               2.0     large_airport
…                              …              …
518551                          4.0     large_airport
518552                          2.0     large_airport
518553                          4.0     large_airport
518554                          4.0     large_airport
518555                          4.0     large_airport
```

```
        elevation_ft_dest_airport  runway_count_dest_airport  \
0                           97.0                        5.0
1                           97.0                        5.0
2                          433.0                        4.0
3                           97.0                        5.0
4                           97.0                        5.0
…                            …                          …
518551                      27.0                        2.0
518552                      17.0                        1.0
518553                    4415.0                        3.0
518554                     125.0                        4.0
518555                      13.0                        4.0

        data_2019_source_airport  data_2019_dest_airport  Founded
0                     27779230.0              21905309.0   1934.0
1                     42939104.0              21905309.0   1934.0
2                      2713843.0              25001762.0   1934.0
3                     24728361.0              21905309.0   1934.0
4                      2723002.0              21905309.0   1934.0
…                            …                       …      …
518551                24728361.0               6454413.0   1967.0
518552                 6454413.0              12648692.0   1967.0
518553                24728361.0               2162250.0   1967.0
518554                27779230.0              42939104.0   1967.0
518555                42939104.0              27779230.0   1967.0

[518556 rows x 18 columns]
```

[74]:
```
s1 = combined_data_pax[combined_data_pax.Delay == 1].runway_count_source_airport
s2 = combined_data_pax[combined_data_pax.Delay == 0].runway_count_source_airport
```

[76]:
```
import scipy.stats as stats
t, p = stats.ttest_ind(s1, s2)
if p < 0.05:
    result = 'reject null'
else :
    result = 'fail to reject null'
print(result)
```

```
reject null
```

[77]:
```
s1 = combined_data_pax[combined_data_pax.Delay == 1].runway_count_dest_airport
s2 = combined_data_pax[combined_data_pax.Delay == 0].runway_count_dest_airport
```

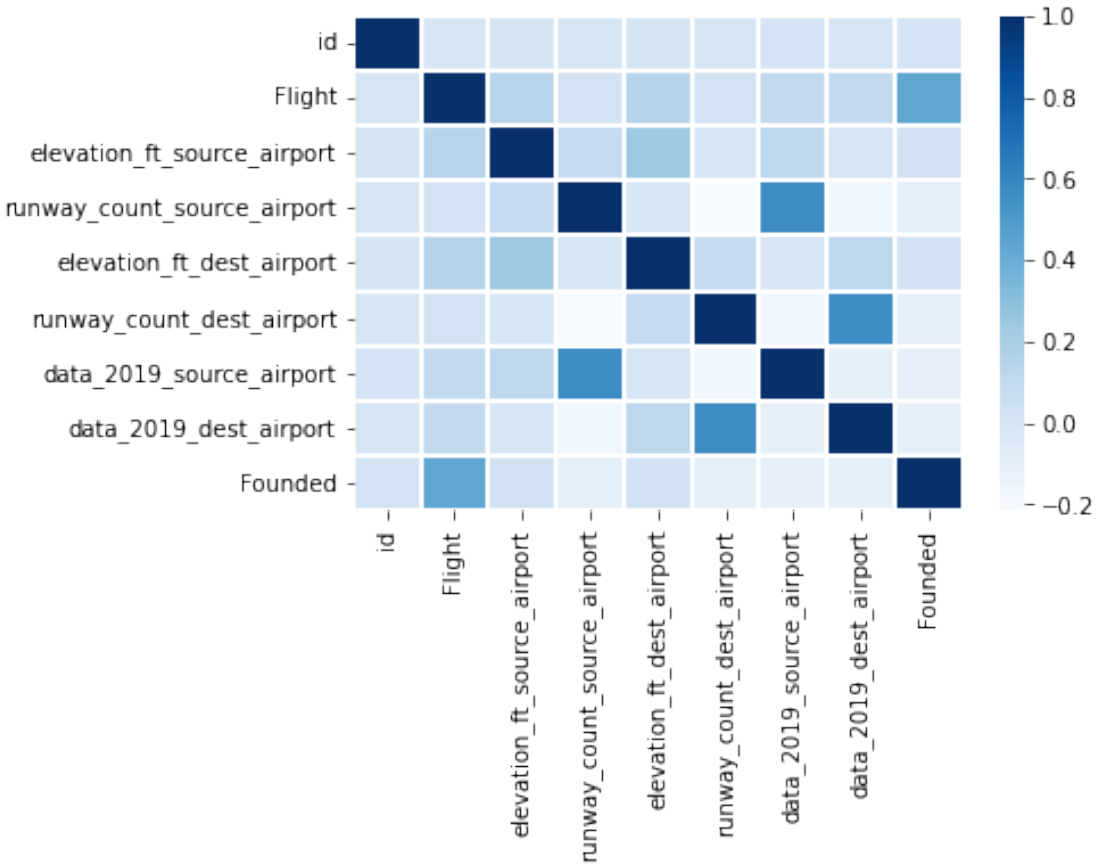
[78]:
```
t, p = stats.ttest_ind(s1, s2)
if p < 0.05:
    result = 'reject null'
```

```
else :
    result = 'fail to reject null'
print(result)
```

```
fail to reject null
```

[80]:
```
import matplotlib.pyplot as plt
correlation_matix = combined_data_pax.drop(columns = ['DayOfWeek', 'Time',␣
↪'Length', 'Delay','type_source_airport','type_dest_airport']).corr()

sns.heatmap(correlation_matix, cmap='Blues',linecolor='white', linewidths=2)
plt.show()
```



[84]:
```
combined_data_pax.isna().sum()
```

[84]:
```
id                  0
Airline             0
Flight              0
AirportFrom         0
```

```
AirportTo                          0
DayOfWeek                          0
Time                               0
Length                             0
Delay                              0
type_source_airport                0
elevation_ft_source_airport        0
runway_count_source_airport        0
type_dest_airport                 31
elevation_ft_dest_airport         31
runway_count_dest_airport         31
data_2019_source_airport       83582
data_2019_dest_airport         83531
Founded                            0
dtype: int64
```

[85]:
```
combined_data_pax.dropna(inplace = True)
```

[88]:
```
combined_data_pax.drop(columns = ['id', 'Flight', ], inplace = True)
```

[89]:
```
combined_data_pax.head(2)
```

[89]:
```
  Airline AirportFrom AirportTo  DayOfWeek  Time  Length  Delay  \
0      CO         SFO       IAH          3    15     205      1
1      CO         LAX       IAH          3    30     181      1

  type_source_airport  elevation_ft_source_airport  \
0       large_airport                         13.0
1       large_airport                        125.0

   runway_count_source_airport type_dest_airport  elevation_ft_dest_airport  \
0                          4.0     large_airport                       97.0
1                          4.0     large_airport                       97.0

   runway_count_dest_airport  data_2019_source_airport  \
0                        5.0               27779230.0
1                        5.0               42939104.0

   data_2019_dest_airport  Founded
0              21905309.0   1934.0
1              21905309.0   1934.0
```

[90]:
```
combined_data_pax.type_dest_airport.unique()
```

[90]:
```
array(['large_airport', 'medium_airport'], dtype=object)
```

[92]:
```
from sklearn.preprocessing import OrdinalEncoder, StandardScaler
ordinal = OrdinalEncoder(categories=[['medium_airport','␣
↪'large_airport'],['medium_airport', 'large_airport']])
ordinal.fit(combined_data_pax[['type_source_airport', 'type_dest_airport']])
```

[92]:
```
OrdinalEncoder(categories=[['medium_airport', 'large_airport'],
                           ['medium_airport', 'large_airport']])
```

[93]:
```
combined_data_pax[['type_source_airport', 'type_dest_airport']] = ordinal.
↪transform(combined_data_pax[['type_source_airport', 'type_dest_airport']])
```

[94]:
```
model_data = combined_data_pax.drop(columns = ['Airline', 'AirportFrom',␣
↪'AirportTo'])
```

[95]:
```
model_data.shape
```

[95]:
```
(352609, 13)
```

[96]:
```
dummy = pd.get_dummies(model_data)
dummy.shape
```

[96]:
```
(352609, 13)
```

[97]:
```
dummy.Founded = 2022 - dummy.Founded
```

[98]:
```
model_data.reset_index(drop = True, inplace = True)
```

[99]:
```
np.random.seed(12)
```

[100]:
```
deploy_idx = np.random.choice(model_data.index, replace = False, size = 5000)
```

[101]:
```
deploy = model_data.loc[deploy_idx]
```

[102]:
```
X_deploy = deploy.drop(columns = 'Delay')
```

[103]:
```
model_dev = model_data.loc[~model_data.index.isin(deploy.index)]
```

[104]:
```
deploy.reset_index(drop = True, inplace = True)
model_dev.reset_index(drop = True, inplace = True)
```

[105]:
```
X = model_dev.drop(columns = 'Delay')
y = model_dev.Delay
```

[110]:
```
from sklearn.linear_model import SGDClassifier

from sklearn.model_selection import StratifiedKFold, RandomizedSearchCV,␣
↪train_test_split
```

```
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, accuracy_score
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier

folds = StratifiedKFold(n_splits=5, shuffle = True, random_state=12)
accuracy_train = {}
accuracy_test = {}
final_predictions_sgd = {}
i = 1
for train_index, test_index in folds.split(X,y):
    print('iter ', i)
    train, test = model_dev.loc[train_index,], model_dev.loc[test_index,]
    sc = StandardScaler()
    sgd = SGDClassifier()

    # define search space

    space = dict()
    space['sgd__penalty'] = ['l1', 'l2', 'elasticnet']
    space['sgd__l1_ratio'] = [0,.1,.2,.8,1]
    space['sgd__alpha'] = [1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100, 1000,10000]
    space['sgd__learning_rate'] = ['constant', 'adaptive']
    space['sgd__eta0']=[1e-5, 1e-4, 1e-3, 1e-2, 1e-1 , 2e-1, 3e-1, 5e-1, 8e-1,␣
 ↪4e-1, 8e-1, 1, 10, 100]

    pipe = Pipeline([('sc',sc), ('sgd', sgd)])

    # define search
    search = RandomizedSearchCV( pipe, space, scoring='accuracy',
                                cv=5, refit=True, return_train_score = True,
                                random_state = 12, n_jobs = -1, n_iter = 2
                           )

    # execute search
    X_train = train.drop(columns = 'Delay')
    y_train = train.Delay

    result = search.fit(X_train, y_train)

    train_pred = result.predict(X_train)

    X_test = test.drop(columns = 'Delay')
    y_test = test.Delay
    test_pred = result.predict(X_test)
    final_predictions_sgd.update({'Fold{}'.format(i):result.predict(X_deploy)})
```

```
    # get rmse for each fold for train data
    accuracy_train.update({'Fold{}'.format(i): round(accuracy_score(y_true =␣
 ↪y_train, y_pred = train_pred)*100,3)})
    accuracy_test.update({'Fold{}'.format(i): round(accuracy_score(y_true =␣
 ↪y_test, y_pred = test_pred) * 100,3)})
    i += 1
```

```
iter  1
iter  2
iter  3
iter  4
iter  5
```

[111]:
```
folds = StratifiedKFold(n_splits=5, shuffle = True, random_state=12)
dt_accuracy_train = {}
dt_accuracy_test = {}
final_predictions_dt = {}
i = 1
for train_index, test_index in folds.split(X,y):
    print('iter ', i)

    train, test = model_dev.loc[train_index,], model_dev.loc[test_index,]

    sc = StandardScaler()
    dt = DecisionTreeClassifier()


    # define search space
    space = dict()
    space['dt__min_samples_split'] = [25000, 30000, 35000, 40000, 45000, 50000,␣
 ↪60000 ]
    space['dt__min_samples_leaf'] = [10000, 15000, 20000]

    pipe = Pipeline([('sc',sc), ('dt', dt)])

    # define search
    search = RandomizedSearchCV( pipe, space, scoring='accuracy',
                                 cv=5, refit=True, return_train_score = True,
                                 random_state = 12, n_jobs = -1, n_iter = 2
                               )
    # execute search
    X_train = train.drop(columns = 'Delay')
    y_train = train.Delay

    result = search.fit(X_train, y_train)

    train_pred = result.predict(X_train)
```

```
    X_test = test.drop(columns = 'Delay')
    y_test = test.Delay
    test_pred = result.predict(X_test)
    final_predictions_dt.update({'Fold{}'.format(i):result.predict(X_deploy)})

    # get rmse for each fold for train data
    dt_accuracy_train.update({'Fold{}'.format(i): round(accuracy_score(y_true =␣
↪y_train, y_pred = train_pred)*100,3)})
    dt_accuracy_test.update({'Fold{}'.format(i): round(accuracy_score(y_true =␣
↪y_test, y_pred = test_pred) * 100,3)})
    i += 1
```

```
iter  1
iter  2
iter  3
iter  4
iter  5
```

[112]:
```
train_results = pd.DataFrame ({'sgd' : accuracy_train.values(), 'dt':␣
 ↪dt_accuracy_train.values() },
                              index = ['Fold {}'.format(i) for i in range(1,6)])
train_results
```

[112]:

	sgd	dt
Fold 1	57.319	61.262
Fold 2	57.424	61.271
Fold 3	57.324	60.789
Fold 4	57.414	61.374
Fold 5	57.392	61.345

[114]:
```
test_results = pd.DataFrame ({'sgd' : accuracy_test.values(), 'dt':␣
 ↪dt_accuracy_test.values() },
                              index = ['Fold {}'.format(i) for i in range(1,6)])
test_results
```

[114]:

	sgd	dt
Fold 1	57.629	61.503
Fold 2	57.042	61.471
Fold 3	57.510	60.844
Fold 4	57.278	61.058
Fold 5	57.386	61.138

[115]:
```
final_predictions_dt
```

[115]:
```
{'Fold1': array([0, 0, 0, …, 1, 0, 0]),
 'Fold2': array([0, 0, 0, …, 1, 0, 0]),
 'Fold3': array([1, 0, 0, …, 1, 0, 0]),
 'Fold4': array([0, 0, 0, …, 1, 0, 0]),
 'Fold5': array([0, 0, 0, …, 1, 0, 0])}
```

[116]:
```
final_predictions_sgd
```

[116]:
```
{'Fold1': array([0, 1, 0, …, 1, 1, 0]),
 'Fold2': array([0, 1, 0, …, 1, 1, 0]),
 'Fold3': array([0, 1, 0, …, 1, 1, 0]),
 'Fold4': array([0, 1, 0, …, 1, 1, 0]),
 'Fold5': array([0, 1, 0, …, 1, 1, 0])}
```

[117]:
```
folds = StratifiedKFold(n_splits=5, shuffle = True, random_state=12)
xgb_accuracy_train = {}
xgb_accuracy_test = {}
final_predictions_xgb = []

i = 1
for train_index, test_index in folds.split(X,y):
    print('iter ', i)
    train, test = model_dev.loc[train_index,], model_dev.loc[test_index,]
    sc = StandardScaler()
    xgb_r = XGBClassifier(random_state = 12, use_label_encoder = False)

    # define search space
    space = dict()
    space['xgb_r__n_estimators'] = [40,50,60]
    space['xgb_r__max_depth'] = [3,4,5]
    space['xgb_r__colsample_bytree']:[0.4,.5,.6]
    space['xgb_r__lambda'] = [.0001,.002,.0004,.0003]
    space['xgb_r__alpha'] = [.01,.02,.1,.4]

    pipe = Pipeline([('sc',sc), ('xgb_r', xgb_r)])

    # define search
    search = RandomizedSearchCV( pipe, space,␣
↪scoring='neg_root_mean_squared_error',
                                cv=5, refit=True, return_train_score = True,
                                random_state = 12, n_jobs = -1, n_iter = 2
                            )

    # execute search
    X_train = train.drop(columns = 'Delay')
    y_train = train.Delay
```

```
    result = search.fit(X_train, y_train)

    train_pred = result.predict(X_train)

    X_test = test.drop(columns = 'Delay')
    y_test = test.Delay
    test_pred = result.predict(X_test)

    final_predictions_xgb.append(result.predict(X_deploy))

    # get rmse for each fold for train data
    xgb_accuracy_train.update({'Fold{}'.format(i): round(accuracy_score(y_true␣
↪= y_train, y_pred = train_pred),3)})
    xgb_accuracy_test.update({'Fold{}'.format(i): round(accuracy_score(y_true =␣
↪y_test, y_pred = test_pred),3)})
    i += 1
```

```
iter  1
```

```
/usr/local/lib/python3.7/site-
packages/joblib/externals/loky/process_executor.py:706: UserWarning: A worker
stopped while some jobs were given to the executor. This can be caused by a too
short worker timeout or by a memory leak.
  "timeout or by a memory leak.", UserWarning
```

```
iter  2
iter  3
iter  4
iter  5
```

[122]:
```
xgb_accuracy_train
```

[122]:
```
{'Fold1': 0.641,
 'Fold2': 0.641,
 'Fold3': 0.648,
 'Fold4': 0.641,
 'Fold5': 0.642}
```

[123]:
```
xgb_accuracy_train.values()
```

[123]:
```
dict_values([0.641, 0.641, 0.648, 0.641, 0.642])
```

[124]:
```
train_results['xgb'] = xgb_accuracy_train.values()
test_results['xgb'] = xgb_accuracy_test.values()
```

[125]:
```
train_results
```

[125]:

	sgd	dt	xgb
Fold 1	57.319	61.262	0.641
Fold 2	57.424	61.271	0.641
Fold 3	57.324	60.789	0.648
Fold 4	57.414	61.374	0.641
Fold 5	57.392	61.345	0.642

[126]:
```
test_results
```

[126]:

	sgd	dt	xgb
Fold 1	57.629	61.503	0.640
Fold 2	57.042	61.471	0.641
Fold 3	57.510	60.844	0.644
Fold 4	57.278	61.058	0.638
Fold 5	57.386	61.138	0.638

[ ]:
```

```