Tasca6

February 1, 2021

1 Visualització gràfica d'un dataset

[1]:

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

1.0.1 Exercici 1

Resumeix gràficament el data set DelayedFlights.csv

Crea almenys una visualització per:

- Una variable categòrica (UniqueCarrier)
- Una variable numèrica (ArrDelay)
- Una variable numèrica i una categòrica (ArrDelay i UniqueCarrier)
- Dues variables numèriques (ArrDelay i DepDelay)
- Tres variables (ArrDelay, DepDelay i UniqueCarrier)
- Més de tres variables (ArrDelay, DepDelay, AirTime i UniqueCarrier).

1.0.2 Exercici 2

Exporta els gràfics com imatges o com html.

1.0.3 Exercici 3

Exporta el data set net i amb les noves columnes a Excel.

1.0.4 Exercici 4

Integra les visualitzacions gràfiques, en la tasca 5, del Sprint 3.

2 Data:

Airlines Delay: Airline on-time statistics and delay causes

[2]:

```
## Import dataset
df_raw = pd.read_csv("archive/DelayedFlights.csv", index_col = 0)
```

```
/Users/luis/opt/anaconda3/lib/python3.8/site-
packages/numpy/lib/arraysetops.py:580: FutureWarning: elementwise comparison
failed; returning scalar instead, but in the future will perform elementwise
comparison
  mask |= (ar1 == a)
```

[3]:

```
## Columns and Data types
df_raw.info(show_counts = True)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1936758 entries, 0 to 7009727
Data columns (total 29 columns):
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   Year               1936758 non-null  int64
 1   Month              1936758 non-null  int64
 2   DayofMonth         1936758 non-null  int64
 3   DayOfWeek          1936758 non-null  int64
 4   DepTime            1936758 non-null  float64
 5   CRSDepTime         1936758 non-null  int64
 6   ArrTime            1929648 non-null  float64
 7   CRSArrTime         1936758 non-null  int64
 8   UniqueCarrier      1936758 non-null  object
 9   FlightNum          1936758 non-null  int64
 10  TailNum            1936753 non-null  object
 11  ActualElapsedTime  1928371 non-null  float64
 12  CRSElapsedTime     1936560 non-null  float64
 13  AirTime            1928371 non-null  float64
 14  ArrDelay           1928371 non-null  float64
 15  DepDelay           1936758 non-null  float64
 16  Origin             1936758 non-null  object
 17  Dest               1936758 non-null  object
 18  Distance           1936758 non-null  int64
 19  TaxiIn             1929648 non-null  float64
 20  TaxiOut            1936303 non-null  float64
 21  Cancelled          1936758 non-null  int64
 22  CancellationCode   1936758 non-null  object
 23  Diverted           1936758 non-null  int64
 24  CarrierDelay       1247488 non-null  float64
 25  WeatherDelay       1247488 non-null  float64
 26  NASDelay           1247488 non-null  float64
 27  SecurityDelay      1247488 non-null  float64
 28  LateAircraftDelay  1247488 non-null  float64
dtypes: float64(14), int64(10), object(5)
memory usage: 443.3+ MB
```

Variable descriptions:

- Year: 1987-2008
- Month: 1-12
- DayofMonth: 1-31
- DayOfWeek: 1 (Monday) - 7 (Sunday)
- DepTime: departure time (local, hhmm)
- CRSDepTime: scheduled departure time (local, hhmm)
- ArrTime: arrival time (local, hhmm)
- CRSArrTime: scheduled arrival time (local, hhmm)
- UniqueCarrier: unique carrier code
- FlightNum: flight number
- TailNum: plane tail number
- ActualElapsedTime: flygth time in minutes (Total)
- CRSElapsedTime: scheduled flygth time in minutes (Total)
- AirTime: time on air in minutes
- ArrDelay: arrival delay in minutes
- DepDelay: departure delay in minutes
- Origin: origin IATA airport code
- Dest: destination IATA airport code
- Distance: distance in miles
- TaxiIn: taxi in time, in minutes (movement on ground)
- TaxiOut: taxi out time, in minutes (movement on ground)
- Cancelled: was the flight cancelled?
- CancellationCode: reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
- Diverted: 1 = yes, 0 = no ("Desviado")
- CarrierDelay: delayed time due to Carrier in minutes
- WeatherDelay: delayed time due to Weather in minutes
- NASDelay: delayed time due to NAS in minutes
- SecurityDelay: delayed time due to security in minuts
- LateAircraftDelay: delayed time due to late aircraft in minutes

[4]:

```
## Dataframe Visualization
pd.set_option('display.max_columns', None)
```

[5]:

```
## Sample
df_raw.sample(10)
```

[5]:

```
         Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime  ArrTime  \
1122251  2008      2           9          6   1716.0        1700   2033.0
2553769  2008      5          27          2    650.0         620    850.0
3099169  2008      6           5          4   1309.0        1130   1700.0
6683150  2008     12          14          7    909.0         857   1233.0
5126330  2008      9          26          5   1350.0        1340   1522.0
263519   2008      1           6          7   1109.0        1051   1640.0
6892136  2008     12           2          2    632.0         625    735.0
3294792  2008      6           1          7   2025.0        1955   2250.0
589819   2008      1           2          3   2012.0        2005   2203.0
4256084  2008      8           9          6   1716.0        1700   2300.0
```

```
         CRSArrTime UniqueCarrier  FlightNum TailNum  ActualElapsedTime  \
1122251        2005            B6         21  N553JB              197.0
2553769         807            OH       5121  N548CA              120.0
3099169        1510            XE       2846  N13913              171.0
6683150        1219            UA        910  N593UA              144.0
5126330        1512            DL        138  N757AT               92.0
263519         1635            US        149  N821AW              211.0
6892136         730            9E       2954  85329E               63.0
3294792        2210            WN        904  N306SW              145.0
589819         2143            DL       1226  N951DL              111.0
4256084        2255            WN       1966  N288WN              224.0

         CRSElapsedTime  AirTime  ArrDelay  DepDelay Origin Dest  Distance  \
1122251           185.0    165.0      28.0      16.0    JFK  TPA      1005
2553769           107.0     80.0      43.0      30.0    PHL  CVG       507
3099169           160.0    146.0     110.0      99.0    IAH  RDU      1043
6683150           142.0    106.0      14.0      12.0    DEN  ORD       888
5126330            92.0     64.0      10.0      10.0    TPA  ATL       406
263519            224.0    187.0       5.0      18.0    PHX  CMH      1671
6892136            65.0     36.0       5.0       7.0    SBN  DTW       157
3294792           135.0    117.0      40.0      30.0    MCO  PIT       834
589819             98.0     72.0      20.0       7.0    ATL  PHF       508
4256084           235.0    204.0       5.0      16.0    SEA  MDW      1733

         TaxiIn  TaxiOut  Cancelled CancellationCode  Diverted  CarrierDelay  \
1122251     7.0     25.0          0                N         0           0.0
2553769    10.0     30.0          0                N         0          30.0
3099169     6.0     19.0          0                N         0          99.0
6683150     4.0     34.0          0                N         0           NaN
5126330    10.0     18.0          0                N         0           NaN
263519      7.0     17.0          0                N         0           NaN
6892136     8.0     19.0          0                N         0           NaN
3294792     6.0     22.0          0                N         0           0.0
589819      5.0     34.0          0                N         0           0.0
4256084     8.0     12.0          0                N         0           NaN

         WeatherDelay  NASDelay  SecurityDelay  LateAircraftDelay
1122251           0.0      12.0            0.0               16.0
2553769           0.0      13.0            0.0                0.0
3099169           0.0      11.0            0.0                0.0
6683150           NaN       NaN            NaN                NaN
5126330           NaN       NaN            NaN                NaN
263519            NaN       NaN            NaN                NaN
6892136           NaN       NaN            NaN                NaN
3294792          30.0      10.0            0.0                0.0
589819            0.0      13.0            0.0                7.0
```

```
4256084            NaN        NaN              NaN                  NaN
```

3 Data Transformation:

[6]:

```
## Copy DF
df_trans = df_raw.copy()
```

Duplicates:

[7]:

```
## Search Duplicates
print ("Duplicates: ", df_trans.duplicated().sum())
df_trans[df_trans.duplicated(keep = False)]
```

```
Duplicates:  2
```

[7]:

```
        Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime  ArrTime  \
938224  2008      2          28          4   1854.0        1807   1946.0
938225  2008      2          28          4   1854.0        1807   1946.0
938226  2008      2          28          4   2027.0        1942   2314.0
938227  2008      2          28          4   2027.0        1942   2314.0

        CRSArrTime UniqueCarrier  FlightNum TailNum  ActualElapsedTime  \
938224        1902            F9        773  N201FR              112.0
938225        1902            F9        773  N201FR              112.0
938226        2229            F9        780  N201FR              107.0
938227        2229            F9        780  N201FR              107.0

        CRSElapsedTime  AirTime  ArrDelay  DepDelay Origin Dest  Distance  \
938224           115.0     91.0      44.0      47.0    DEN  LAS       629
938225           115.0     91.0      44.0      47.0    DEN  LAS       629
938226           107.0     84.0      45.0      45.0    LAS  DEN       629
938227           107.0     84.0      45.0      45.0    LAS  DEN       629

        TaxiIn  TaxiOut  Cancelled CancellationCode  Diverted  CarrierDelay  \
938224     8.0     13.0          0                N         0          44.0
938225     8.0     13.0          0                N         0          44.0
938226    10.0     13.0          0                N         0           1.0
938227    10.0     13.0          0                N         0           1.0

        WeatherDelay  NASDelay  SecurityDelay  LateAircraftDelay
938224           0.0       0.0            0.0                0.0
938225           0.0       0.0            0.0                0.0
938226           0.0      44.0            0.0                0.0
938227           0.0      44.0            0.0                0.0
```

[8]:

```
## Drop Duplicates
df_trans.drop_duplicates(inplace = True)
```

Null Values:

[9]:

```
## Null Values %
df_trans.isnull().mean()*100
```

[9]:

```
Year                  0.000000
Month                 0.000000
DayofMonth            0.000000
DayOfWeek             0.000000
DepTime               0.000000
CRSDepTime            0.000000
ArrTime               0.367109
CRSArrTime            0.000000
UniqueCarrier         0.000000
FlightNum             0.000000
TailNum               0.000258
ActualElapsedTime     0.433044
CRSElapsedTime        0.010223
AirTime               0.433044
ArrDelay              0.433044
DepDelay              0.000000
Origin                0.000000
Dest                  0.000000
Distance              0.000000
TaxiIn                0.367109
TaxiOut               0.023493
Cancelled             0.000000
CancellationCode      0.000000
Diverted              0.000000
CarrierDelay         35.588892
WeatherDelay         35.588892
NASDelay             35.588892
SecurityDelay        35.588892
LateAircraftDelay    35.588892
dtype: float64
```

[10]:

```
## Columns with low percentage of nulls (less than 2% in total)
subset = ["ArrTime", "TailNum", "ActualElapsedTime", "CRSElapsedTime",␣
 ↪"AirTime",
                           "ArrDelay", "TaxiIn", "TaxiOut"]
```

[11]:

```
## Save rows with low percentage of nulls before drop
df_null = df_trans[df_trans[subset].isnull().any(axis=1)]
```

[12]:

```
## Drop rows with low percentage of nulls
df_trans = df_trans.dropna(subset=subset)
```

We could set the nulls in the **Delay** columns equal to the median or equal to zero. If equal to

zero, assuming that nulls in these columns correspond to absence of delay, we should filter those rows before extracting information to avoid bias. However, I'm not sure that nulls mean zero delay, maybe are just unknown information, which will explain why they have identical null percentage if all these observations come front the same font. We could set the nulls equal to the **median**, but in this case I rather prefer leaving the dataframe as it is right now, since nulls do not interfer with the statistics we are going to infer.

Clean Time:

[13]:
```
## Transform DepTime and ArrTime to a more consistent notation (hh:mm)
df_trans["DepTime"] = df_trans["DepTime"].astype(int).apply(lambda x: str(x).
 ↪zfill(4)).apply(lambda x: x[0:2] + ":" + x[2:])
df_trans["CRSDepTime"] = df_trans["CRSDepTime"].astype(int).apply(lambda x:␣
 ↪str(x).zfill(4)).apply(lambda x: x[0:2] + ":" + x[2:])
df_trans["ArrTime"] = df_trans["ArrTime"].astype(int).apply(lambda x: str(x).
 ↪zfill(4)).apply(lambda x: x[0:2] + ":" + x[2:])
df_trans["CRSArrTime"] = df_trans["CRSArrTime"].astype(int).apply(lambda x:␣
 ↪str(x).zfill(4)).apply(lambda x: x[0:2] + ":" + x[2:])
```

Describe:

[14]:
```
## Change dtypes
df_trans["FlightNum"] = df_trans["FlightNum"].astype(str)
df_trans["Cancelled"] = df_trans["Cancelled"].astype(str)
df_trans["Diverted"] = df_trans["Diverted"].astype(str)
```

[15]:
```
## Divide into numerical and categorical
df_num = df_trans.select_dtypes(include = ["int64", "float64"])
df_cat = df_trans.select_dtypes(exclude = ["int64", "float64"])
```

[16]:
```
## Describe num
df_num.describe().round(2)
```

[16]:
```
            Year       Month  DayofMonth   DayOfWeek  ActualElapsedTime  \
count  1928366.0  1928366.00  1928366.00  1928366.00         1928366.00
mean      2008.0        6.11       15.75        3.98             133.31
std          0.0        3.48        8.78        2.00              72.06
min       2008.0        1.00        1.00        1.00              14.00
25%       2008.0        3.00        8.00        2.00              80.00
50%       2008.0        6.00       16.00        4.00             116.00
75%       2008.0        9.00       23.00        6.00             165.00
max       2008.0       12.00       31.00        7.00            1114.00

       CRSElapsedTime     AirTime    ArrDelay    DepDelay    Distance  \
count      1928366.00  1928366.00  1928366.00  1928366.00  1928366.00
mean           134.20      108.28       42.20       43.09      764.95
std             71.23       68.64       56.78       53.27      573.89
min            -21.00        0.00     -109.00        6.00       11.00
```

```
25%             82.00        58.00        9.00        12.00       338.00
50%            116.00        90.00       24.00        24.00       606.00
75%            165.00       137.00       56.00        53.00       997.00
max            660.00      1091.00     2461.00      2467.00      4962.00

          TaxiIn     TaxiOut  CarrierDelay  WeatherDelay    NASDelay  \
count 1928366.00  1928366.00    1247484.00    1247484.00  1247484.00
mean        6.81       18.22         19.18          3.70       15.02
std         5.27       14.31         43.55         21.49       33.83
min         0.00        0.00          0.00          0.00        0.00
25%         4.00       10.00          0.00          0.00        0.00
50%         6.00       14.00          2.00          0.00        2.00
75%         8.00       21.00         21.00          0.00       15.00
max       240.00      422.00       2436.00       1352.00     1357.00

      SecurityDelay  LateAircraftDelay
count    1247484.00         1247484.00
mean           0.09              25.30
std            2.02              42.05
min            0.00               0.00
25%            0.00               0.00
50%            0.00               8.00
75%            0.00              33.00
max          392.00            1316.00
```

- **std of Year = 0.** All flygths are in 2008. Does not give any information.

[17]:
```
## Drop Year
df_trans.drop(columns = "Year", inplace = True)
```

[18]:
```
## Describe cat
df_cat.describe()
```

[18]:
```
        DepTime CRSDepTime  ArrTime CRSArrTime UniqueCarrier FlightNum  \
count   1928366    1928366  1928366    1928366       1928366   1928366
unique     1438       1193     1440       1361            20      7498
top       18:00      18:00    21:00      19:30            WN        16
freq       3176      13867     2981       9148        376201      1575

        TailNum   Origin     Dest Cancelled CancellationCode Diverted
count   1928366  1928366  1928366   1928366          1928366  1928366
unique     5360      303      302         1                1        1
top      N325SW      ATL      ORD         0                N        0
freq        961   131213   108265   1928366          1928366  1928366
```

- **unique of *Cancelled*, *CancellationCode* and *Diverted* are equal to 1.** These columns do not give any information. Any flygth has been cancelled ore diverted.

```
[19]: ## Drop Cancelled, CancellationCode and Diverted
      df_trans.drop(columns = ["Cancelled", "CancellationCode", "Diverted"], inplace␣
       ↪= True)
```

4 Feature Engineering:

```
[20]: ## Categorical column with delay > 15 min (1 = Yes, 0 = No)
      df_trans["DelayCat"] = df_trans["ArrDelay"].apply(lambda x: 1 if x > 15 else 0)
```

```
[21]: ## Mean Velocity columns in miles/min
      df_trans["Velocity"] = df_trans["Distance"] / df_trans["AirTime"]
```

```
[22]: ## Origin-Destination Columns
      df_trans["Fligth"] = df_trans["Origin"] + "-" + df_trans["Dest"]
```

5 Save Data:

```
[23]: ## Save Final Dataframe
      df_trans.to_csv("df_clean.csv")
```

6 Plots:

```
[24]: ## Import Clean data
      df = pd.read_csv("df_clean.csv", index_col = 0)
```

```
/Users/luis/opt/anaconda3/lib/python3.8/site-
packages/numpy/lib/arraysetops.py:580: FutureWarning: elementwise comparison
failed; returning scalar instead, but in the future will perform elementwise
comparison
  mask |= (ar1 == a)
```

```
[25]: ## Sample
      df.sample(10)
```

```
[25]:          Month  DayofMonth  DayOfWeek DepTime CRSDepTime ArrTime CRSArrTime  \
      3574071      6          14          6   17:09      16:40   21:23      19:58
      6051740     11           2          7   08:07      08:00   09:16      09:12
      6949764     12          18          4   22:18      21:31   00:35      23:48
      4198208      7          20          7   17:55      17:15   20:44      20:19
      5396649     10           6          1   11:19      11:05   13:00      12:45
      4758019      8           8          5   20:09      18:10   22:02      20:10
      2810960      5          12          1   18:50      17:10   20:45      19:00
      4403316      8          29          5   19:00      18:35   20:20      19:51
      4060700      7           3          4   12:56      12:00   18:10      17:18
```

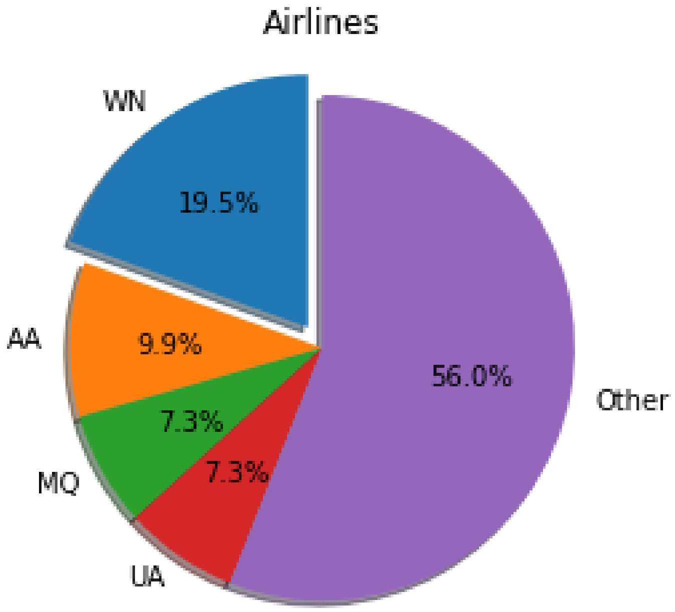
```
840356       2          15           5   11:07      09:45   12:26      11:15

        UniqueCarrier  FlightNum TailNum  ActualElapsedTime  CRSElapsedTime  \
3574071            CO       1820  N14609              194.0           138.0
6051740            XE       2349  N13913               69.0            72.0
6949764            AS        621  N514AS              137.0           137.0
4198208            CO       1693  N17139              169.0           184.0
5396649            WN       1174  N505SW              101.0           100.0
4758019            AA       1973  N5FGAA              173.0           180.0
2810960            MQ       4918  N704PG              115.0           110.0
4403316            OH       5665  N528CA               80.0            76.0
4060700            NW        780  N360NB              194.0           198.0
840356             UA       1197  N848UA              139.0           150.0

         AirTime  ArrDelay  DepDelay Origin Dest  Distance  TaxiIn  TaxiOut  \
3574071    107.0      85.0      29.0    IAH  ATL       689    58.0     29.0
6051740     53.0       4.0       7.0    BTR  IAH       253     7.0      9.0
6949764    116.0      47.0      47.0    LAS  PDX       762     3.0     18.0
4198208    146.0      25.0      40.0    MCO  EWR       938    10.0     13.0
5396649     88.0      15.0      14.0    STL  DAL       546     4.0      9.0
4758019    150.0     112.0     119.0    MIA  DFW      1121     6.0     17.0
2810960     86.0     105.0     100.0    CMH  LGA       478    10.0     19.0
4403316     53.0      29.0      25.0    PIT  CVG       256     5.0     22.0
4060700    160.0      52.0      56.0    LAS  MSP      1300     9.0     25.0
840356     115.0      71.0      82.0    DEN  LAX       862    13.0     11.0

         CarrierDelay  WeatherDelay  NASDelay  SecurityDelay  \
3574071           0.0           0.0      56.0            0.0
6051740           NaN           NaN       NaN            NaN
6949764           0.0           0.0       0.0            0.0
4198208           0.0           0.0      25.0            0.0
5396649           0.0           0.0       1.0            0.0
4758019          53.0           0.0       0.0            0.0
2810960          93.0           0.0       5.0            0.0
4403316           0.0          25.0       4.0            0.0
4060700           0.0           0.0       0.0            0.0
840356            5.0           0.0       0.0            0.0

         LateAircraftDelay  DelayCat  Velocity   Fligth
3574071               29.0         1  6.439252  IAH-ATL
6051740                NaN         0  4.773585  BTR-IAH
6949764               47.0         1  6.568966  LAS-PDX
4198208                0.0         1  6.424658  MCO-EWR
5396649               14.0         0  6.204545  STL-DAL
4758019               59.0         1  7.473333  MIA-DFW
2810960                7.0         1  5.558140  CMH-LGA
4403316                0.0         1  4.830189  PIT-CVG
```

```
4060700                 52.0          1   8.125000  LAS-MSP
840356                  66.0          1   7.495652  DEN-LAX
```

Pie Chart:

[26]:

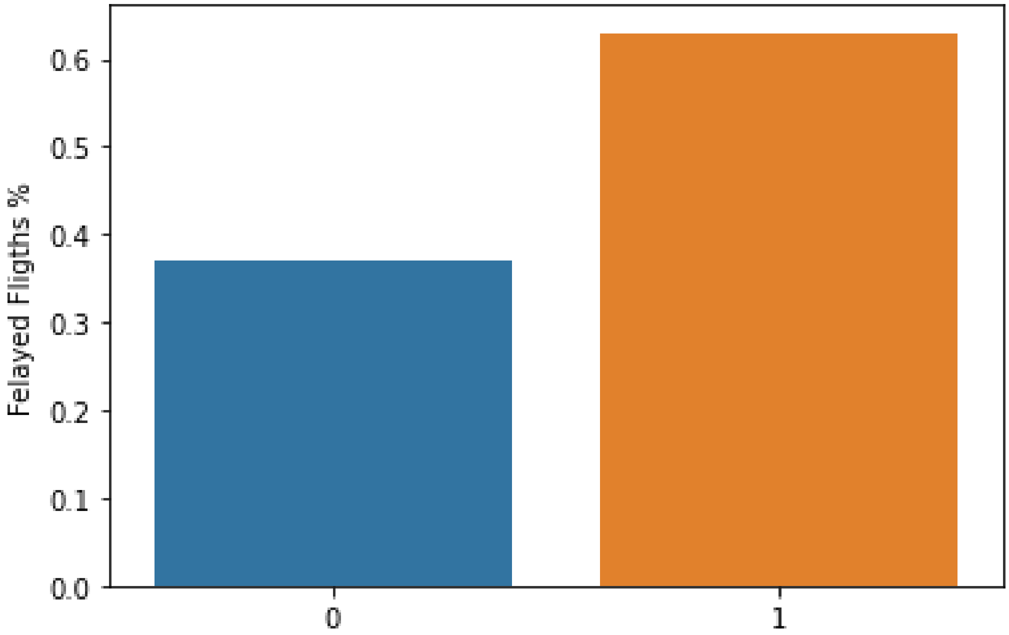
```
## Pie chart of UniqueCarrier using Matplotlib

main_sizes = list(df["UniqueCarrier"].value_counts()[:4])
other_sizes = df["UniqueCarrier"].value_counts()[4:].sum()
main_sizes.append(other_sizes)

main_labels = list(df["UniqueCarrier"].value_counts().index[:4])
labels = main_labels.append("Other")

fig1, ax1 = plt.subplots()
plt.title("Airlines")

ax1.pie(main_sizes, labels = main_labels, explode = [0.1, 0, 0, 0, 0],␣
 ↪autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.savefig('PieChart.png')
```



Bar Plot:

[27]:

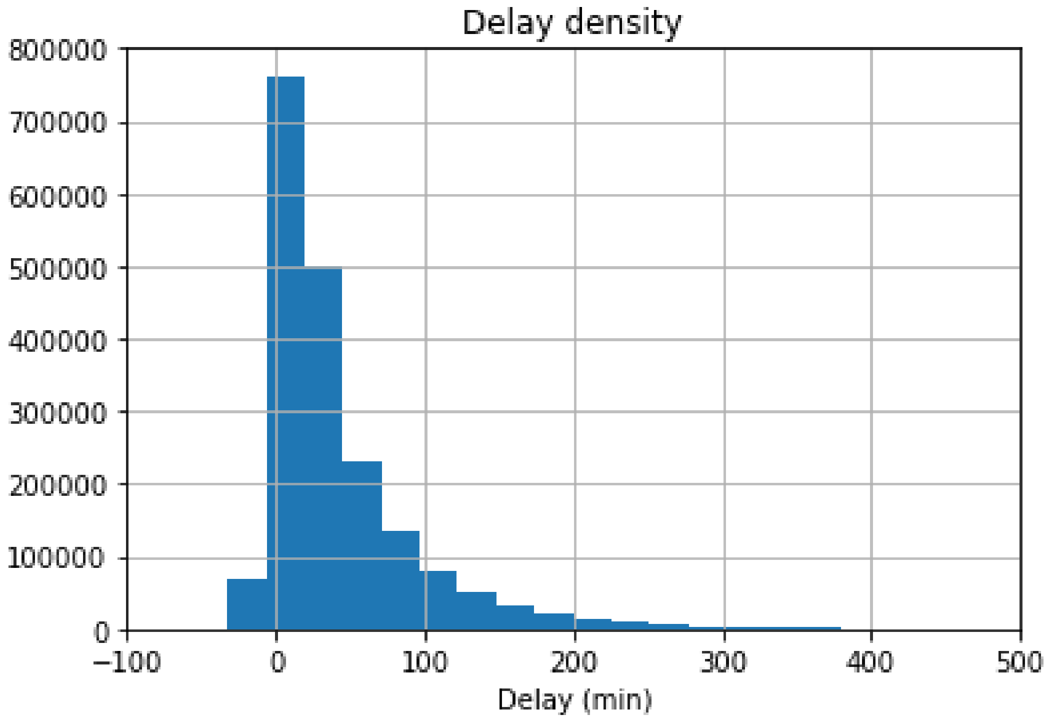
```
## BarPlot with Seaborn %
percentage = df["DelayCat"].value_counts()/df["DelayCat"].value_counts().sum()
ax = sns.barplot(x = df["DelayCat"].value_counts().index, y = percentage);
ax.set_ylabel("Felayed Fligths %");
```



Histogram:

[28]:

```
## Histogram with pandas
ax = df.ArrDelay.hist(bins = 100);
ax.set(xlim=(-100, 500));
ax.set_xlabel("Delay (min)");
ax.set_title("Delay density");
```



Box Plot:

[29]:

```
## Filter Data of the 4 biggest airlines
main_airlines = df["UniqueCarrier"].value_counts().index[:4]
df_main_airlines = df[df["UniqueCarrier"].isin(main_airlines)]
```

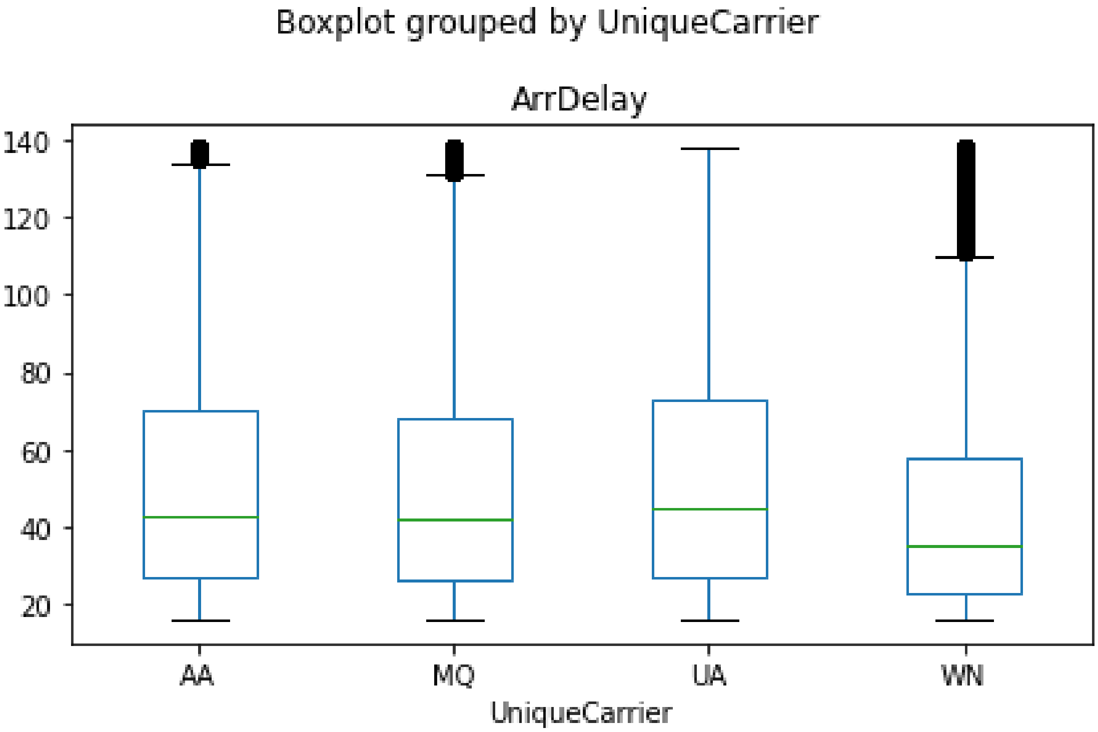
[30]:

```
## Filter Outliers (quantile 0.9)
quant = df_main_airlines["ArrDelay"].quantile(0.95)
df_main_airlines = df_main_airlines[df["ArrDelay"] < quant]
```

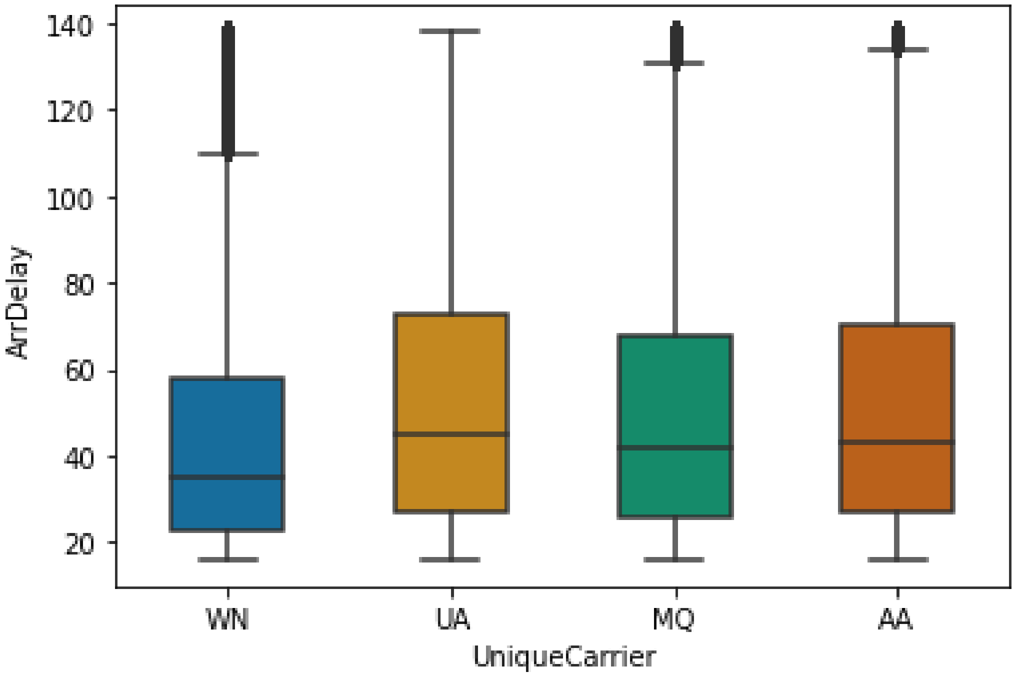
```
<ipython-input-30-b39595a63a30>:3: UserWarning: Boolean Series key will be
reindexed to match DataFrame index.
  df_main_airlines = df_main_airlines[df["ArrDelay"] < quant]
```

[31]:

```
## BoxPlot using Pandas
df_main_airlines[df_main_airlines["DelayCat"] == 1].boxplot(by='UniqueCarrier',␣
 ↪column=['ArrDelay'], grid = False);
plt.tight_layout()
```



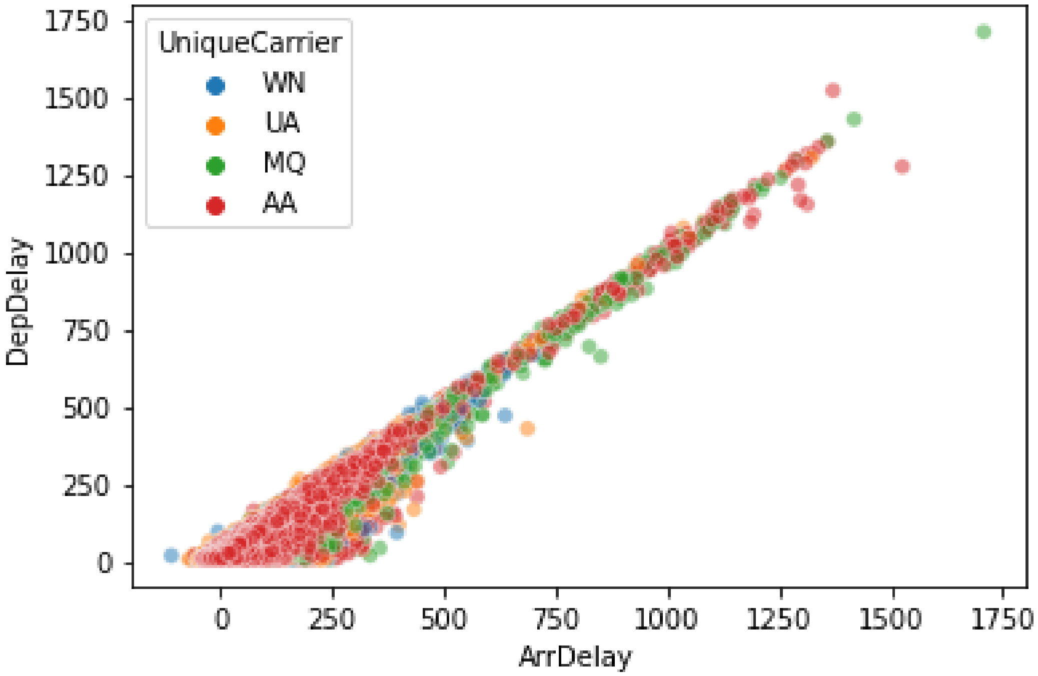
```
[32]: ## BoxPlot using Seaborn
      bplot = sns.boxplot(y='ArrDelay', x='UniqueCarrier',␣
       ↪data=df_main_airlines[df_main_airlines["DelayCat"] == 1],
                        width=0.5, palette="colorblind")
      plt.savefig('BoxPlot.png')
```



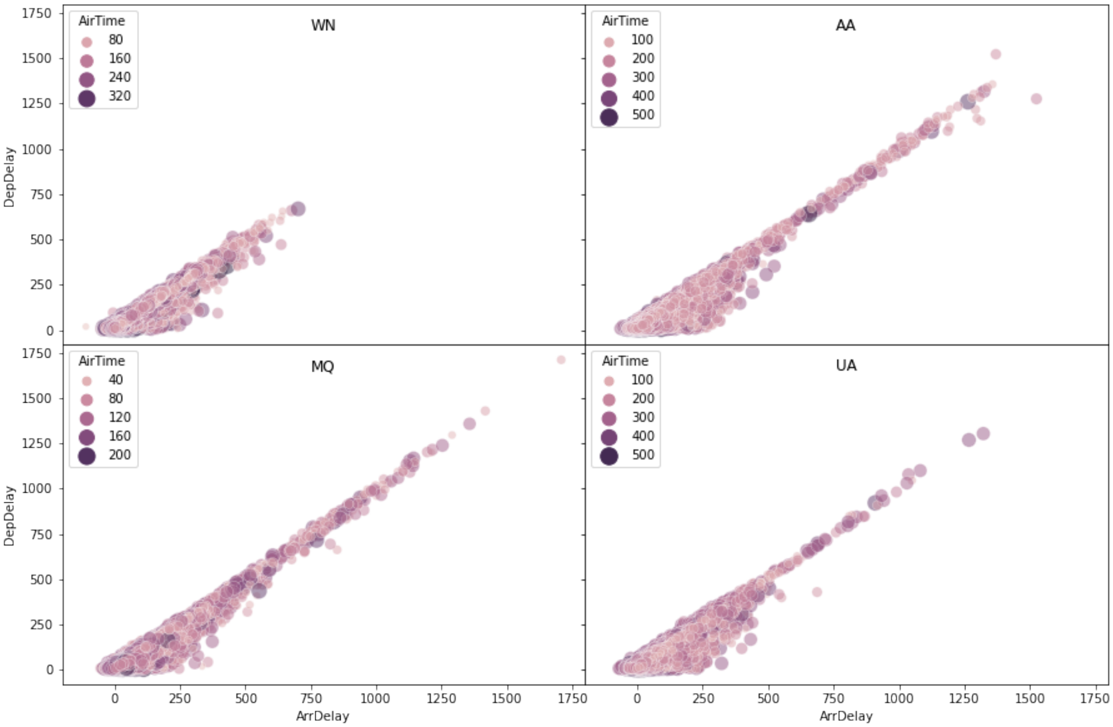
Scatter Plot:

[33]:
```
## Filter Data of the 2 biggest airlines
main_airlines = df["UniqueCarrier"].value_counts().index[:4]
df_main_airlines = df[df["UniqueCarrier"].isin(main_airlines)]
```

[34]:
```
## Scatter plot using Seaborn
ArrDelay = df_main_airlines.ArrDelay
DepDelay = df_main_airlines.DepDelay
UniqueCarrier = df_main_airlines.UniqueCarrier

sns.scatterplot(x = 'ArrDelay', y = 'DepDelay', data=df_main_airlines, alpha=0.
 ↪5, hue='UniqueCarrier');
```



[35]:
```python
## Subplots
fig, axes = plt.subplots(2, 2, sharex=True, sharey=True, figsize=(15, 10));
color = sns.color_palette("pastel", 4)

flag = 0
for i in range(2):
    for j in range(2):
        df_scatter = df[df["UniqueCarrier"] == main_airlines[flag]]
        ax = sns.scatterplot(ax=axes[i, j], x = 'ArrDelay', y = 'DepDelay',␣
 ↪data=df_scatter,
                        hue = "AirTime", color = color[flag], size = "AirTime",␣
 ↪sizes=(20, 200),
                        alpha=0.5, legend="brief");
        ax.set_title(main_airlines[flag], x=0.5, y=0.9)
        ax.legend(loc = 'upper left', title = "AirTime")
        flag += 1

plt.subplots_adjust(wspace=0, hspace=0);
plt.savefig('ScatterPlot.png')
```


WN
AA
MQ
UA
AirTime
80
160
240
320
AirTime
100
200
300
400
500
AirTime
40
80
120
160
200
AirTime
100
200
300
400
500
DepDelay
DepDelay
ArrDelay
ArrDelay
0
250
500
750
1000
1250
1500
1750

```
[ ]:
```