Cristina Cosma

DATA SCIENCE

Sprint 3 : Programació numèrica, dataframes i anàlisi estadístic

Tasca M3 T02 - Exercicis amb Dataframes

- Realitzar anàlisis exploratòria de les dades
- Introducció a la llibreria Pandas, als Dataframes i els mètodes

---

Lliurament: Enviar l'URL a un repositori anomenat estructures_Dataframe que contingui la solució.

EXERCICI 1

1.1 Descarrega el data set Airlines Delay: Airline on-time statistics and delay causes. Carrega'l a un Pandas Dataframe i explora les dades que conté.

1.2 Explica breument quines variables hi ha.

1.3 Queda't únicament amb les columnes que consideris rellevants, justificant la teva elecció.

1.4 Redueix la dimensió del dataset de manera aleatòria **per tal d'obtenir un dataset de només 200.000 registres. Tots els exercicis s'han de fer amb aquest dataset reduït.**

In [2]:
```
import pandas as pd
import numpy as np
from scipy import stats
```

1.1 Descarrega el data set Airlines Delay: *Airline on-time statistics and delay causes*.

El descarreguem a un Pandas Dataframe i explorem les dades que conté :

In [3]:
```
# Descarregant el Dataset en un Pandas Dataframe desde la ruta local :
data = pd.read_csv('C:/Users/Buba/Documents/CURSOS-PROGRAMACION/IT-Academy/IT-DATA-SCIENCE/DS_Sprint3-Pandas-Numpy/CSV_AirplaneDelays/AirplaneDelays/DelayedFlights.csv')
```

In [4]:
```
# Explorando el nombre de cada columna y el tipo de datos que se encuentran en ellas.
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1936758 entries, 0 to 1936757
Data columns (total 30 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Unnamed: 0         int64  
 1   Year               int64  
 2   Month              int64  
 3   DayofMonth         int64  
 4   DayOfWeek          int64  
 5   DepTime            float64
 6   CRSDepTime         int64  
 7   ArrTime            float64
 8   CRSArrTime         int64  
 9   UniqueCarrier      object 
 10  FlightNum          int64  
 11  TailNum            object 
 12  ActualElapsedTime  float64
 13  CRSElapsedTime     float64
 14  AirTime            float64
 15  ArrDelay           float64
 16  DepDelay           float64
 17  Origin             object 
 18  Dest               object 
 19  Distance           int64  
 20  TaxiIn             float64
 21  TaxiOut            float64
 22  Cancelled          int64  
 23  CancellationCode   object 
 24  Diverted           int64  
 25  CarrierDelay       float64
 26  WeatherDelay       float64
 27  NASDelay           float64
 28  SecurityDelay      float64
 29  LateAircraftDelay  float64
dtypes: float64(14), int64(11), object(5)
memory usage: 443.3+ MB
```

In [5]:
```
data
```

Out[5]:

	Unnamed: 0	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	...	TaxiIn	TaxiOut	Cancelled	CancellationCode	Diverted	CarrierDelay	Weathe
0	0	2008	1	3	4	2003.0	1955	2211.0	2225	WN	...	4.0	8.0	0	N	0	NaN	
1	1	2008	1	3	4	754.0	735	1002.0	1000	WN	...	5.0	10.0	0	N	0	NaN	
2	2	2008	1	3	4	628.0	620	804.0	750	WN	...	3.0	17.0	0	N	0	NaN	
3	4	2008	1	3	4	1829.0	1755	1959.0	1925	WN	...	3.0	10.0	0	N	0	2.0	
4	5	2008	1	3	4	1940.0	1915	2121.0	2110	WN	...	4.0	10.0	0	N	0	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1936753	7009710	2008	12	13	6	1250.0	1220	1617.0	1552	DL	...	9.0	18.0	0	N	0	3.0	
1936754	7009717	2008	12	13	6	657.0	600	904.0	749	DL	...	15.0	34.0	0	N	0	0.0	
1936755	7009718	2008	12	13	6	1007.0	847	1149.0	1010	DL	...	8.0	32.0	0	N	0	1.0	
1936756	7009726	2008	12	13	6	1251.0	1240	1446.0	1437	DL	...	13.0	13.0	0	N	0	NaN	
1936757	7009727	2008	12	13	6	1110.0	1103	1413.0	1418	DL	...	8.0	11.0	0	N	0	NaN	

1936758 rows × 30 columns

In [443…

```
# visualitzar les columnes ocultades al mig, les de [10 a 20], que no podem veure a la taula general de dalt.
data.iloc[:, 10:20]
```

Out[443]:

	FlightNum	TailNum	ActualElapsedTime	CRSElapsedTime	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance
0	335	N712SW	128.0	150.0	116.0	-14.0	8.0	IAD	TPA	810
1	3231	N772SW	128.0	145.0	113.0	2.0	19.0	IAD	TPA	810
2	448	N428WN	96.0	90.0	76.0	14.0	8.0	IND	BWI	515
3	3920	N464WN	90.0	90.0	77.0	34.0	34.0	IND	BWI	515
4	378	N726SW	101.0	115.0	87.0	11.0	25.0	IND	JAX	688
...	...	...	...	...	...	...	...	...	...	...
1936753	1621	N938DL	147.0	152.0	120.0	25.0	30.0	MSP	ATL	906
1936754	1631	N3743H	127.0	109.0	78.0	75.0	57.0	RIC	ATL	481
1936755	1631	N909DA	162.0	143.0	122.0	99.0	80.0	ATL	IAH	689
1936756	1639	N646DL	115.0	117.0	89.0	9.0	11.0	IAD	ATL	533
1936757	1641	N908DL	123.0	135.0	104.0	-5.0	7.0	SAT	ATL	874

1936758 rows × 10 columns

1.2 Explica breument quines variables hi ha.

Descripció general de les columnes del dataset i el tipus de dada (Dtype) per cada columna :

- hi ha 30 columnes de, [0] a [29]
- hi ha 1.936.758 rangs (entrades)
- les dades sont totes recopilades de vols de l'any 2008 als Estats-Units
- hi han valors només dels següents : float64 (14), int64 (11), object (5)
- a les columnes Year [1], Month [2], DayofMonth [3], DayOf Week [4], es troben la data per any, mes i dia dels vols
- a les columnes DepTime [5], CRSDepTime [6], ArrTime [7], CRSArrTime [8], es troben les hores de començament i fi dels vols, amb l'indice "CRS" indiquant l'hora planificada i l'altra variable es la real.
- la columna UniqueCarrier [9] té com a tipus de dades "objectes" en forma del letres, que son el símbol de la companya aéria (aerolinia)
- la columna TailNum [11] conté el número de matrícula o "registration number" únic a cada aeronau
- Les columnes [15] a [16] indiquen els temps de duració dels viatge, sient AirTime [14] el temps en minuts passats en el aire
- CRSElapsedTime [13] es el temps total del viatge (terra+aire) previst en minuts i ActualElapsedTime [12] el el temps real total del vol.
- la columna Origin [17] i Dest [18] indiquen aeroports d'inici i destinació
- la columna Distance [19] és la distancia en milles del recorregut
- les columnes de TaxiIn [21] i TaxiOut [22] són gestions que potser que no ens interessin per al nostre anàlisi
- les columnes [23 i 24] indiquen si els viatges ha sigut cancel·lats o desviats
- les columnes [25 a 29] mostren dades en minuts i per cada motiu de retard ; aquí n'hi han moltes dades manquants o imputades de 2 maneres diferents (exemples : NaN, 0.0, 32.0, etc).
- Info sobre NASDelays [27]: " delays or cancellations coded "NAS" are the type of weather delays that could be reduced with corrective action by the airports or the Federal Aviation Administration".

Aquí podem veure quantes dates úniques n'hi han per columna que ens ajuden a decidir quines columnes mantindre i quines esborrar. Destaquem :

- hi ha 1 sol valor a la columna any, que es el 2008
- hi han 20 valors corresponents als codis d'aerolínies
- hi ha 303 eroports d'origin i 304 d'arribada
- les columnes *Cancelled* i *CancellationCode* no són equivalents donat que la primera mostra 2 valors únics i la segona, 4, aixi doncs no esborrarem cap d'elles de moment.

In [7]:

```
# veure quantes dates úniques n'hi han per columna.
data.nunique()
```

Out[7]:
```
Unnamed: 0           1936758
Year                       1
Month                     12
DayofMonth                31
DayOfWeek                  7
DepTime                 1438
CRSDepTime              1207
ArrTime                 1440
CRSArrTime              1364
UniqueCarrier             20
FlightNum               7499
TailNum                 5366
ActualElapsedTime        673
CRSElapsedTime           515
AirTime                  650
ArrDelay                1128
DepDelay                1058
Origin                   303
Dest                     304
Distance                1419
TaxiIn                   180
TaxiOut                  332
Cancelled                  2
CancellationCode           4
Diverted                   2
CarrierDelay             983
WeatherDelay             599
NASDelay                 574
SecurityDelay            156
LateAircraftDelay        564
dtype: int64
```

1.3 Queda't únicament amb les columnes que consideris rellevants, justificant la teva elecció.

- Quitar del df original las columnas que no son necesarias: *Unnamed: 0, Year, AirTime, TaxiIn, TaxiOut*, porque no influyen en los cálculos posteriores sobre tiempos de retraso, compaías o momentos clave. El año es siempre 2008.
    - Se mantienen las variables que puedan ayudar a detectar momentos y lugares clave, patrones en las aerolineas, trayectos, distancias, incluso el número de matrícula de cada aeronave.
    - En lo que respeta las de los diferentes tipos de *Delays* (causas de retraso), *CancellationCode* y *Diverted*, de momento dejamos algunas para su análisis posterior.
    - Observamos que las columnas ArrDelay, DepDelay y ActualElapsedTime ya contienen los tiempos de las variables *Delays* (causas), y usaremos éstas en muchos cálculos
    - les columnes *Cancelled* i *CancellationCode* no són equivalents donat que la primera mostra 2 valors únics i la segona, 4, aixi doncs les mantenim de moment.

In [406…
```
# Esborrem algunes columnes deixant les que ens ajudaràn a respondre a les preguntes i a les hipòtesis que plantejarem:
new_df=data.drop(columns=["Unnamed: 0", "Year", "AirTime","TaxiIn", "TaxiOut", "SecurityDelay", "LateAircraftDelay"], axis=1)
```

In [407…
```
# veiem de nou els tipus de valors i els noms de cada columna
new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1936758 entries, 0 to 1936757
Data columns (total 23 columns):
 #   Column             Dtype  
---  ------             -----  
 0   Month              int64  
 1   DayofMonth         int64  
 2   DayOfWeek          int64  
 3   DepTime            float64
 4   CRSDepTime         int64  
 5   ArrTime            float64
 6   CRSArrTime         int64  
 7   UniqueCarrier      object 
 8   FlightNum          int64  
 9   TailNum            object 
 10  ActualElapsedTime  float64
 11  CRSElapsedTime     float64
 12  ArrDelay           float64
 13  DepDelay           float64
 14  Origin             object 
 15  Dest               object 
 16  Distance           int64  
 17  Cancelled          int64  
 18  CancellationCode   object 
 19  Diverted           int64  
 20  CarrierDelay       float64
 21  WeatherDelay       float64
 22  NASDelay           float64
dtypes: float64(9), int64(9), object(5)
memory usage: 339.9+ MB
```

- Quan la columna és completa, es mostren 1936758 entrades
- recompte dels valors no nuls ens mostra on hi ha valors manquants :
- ArrTime" [5], TailNum [9], ActualElapsedTime [10], CRSElapsedTime [11], ArrDelay [12], CarrierDelay [20], WeatherDelay [21], NASDelay [22], SecurityDelay [23], LateAircraftDelay [24]

Canvi de *FlightNum* amb el número de vol i les inicials de l'aerolinia tot junt en lloc de només els números:

In [447…
```
new_df['FlightNum'] = new_df['UniqueCarrier'].astype(str) + ' ' + new_df['FlightNum'].astype(str)
```

In [448…
```
# Recompte dels valors "non-NA" o no nuls a les columnes del nou dataframe
new_df.count()
```

Out[448]:
```
Month                1936758
DayofMonth           1936758
DayOfWeek            1936758
DepTime              1936758
CRSDepTime           1936758
ArrTime              1929648
CRSArrTime           1936758
UniqueCarrier        1936758
FlightNum            1936758
TailNum              1936753
ActualElapsedTime    1928371
CRSElapsedTime       1936560
ArrDelay             1928371
DepDelay             1936758
Origin               1936758
Dest                 1936758
Distance             1936758
Cancelled            1936758
CancellationCode     1936758
Diverted             1936758
CarrierDelay         1247488
WeatherDelay         1247488
NASDelay             1247488
dtype: int64
```

1.4 Redueix la dimensió del dataset de manera aleatòria per tal d'obtenir un dataset de només 200.000 registres. Tots els exercicis s'han de fer amb aquest dataset reduït.

In [449…
```
# reducció del dataset
data_reduced = new_df.sample(n=200000)
```

EXERCICI 2 : Fes un informe complet del dataset:

2.1 Resumeix estadísticament el dataset i les columnes d'interès. Fes una anàlisi estadístic del que consideris rellevant.

2.2 Troba quantes dades faltants hi ha per columna.

2.3 Crea columnes noves (velocitat mitjana del vol, si ha arribat tard o no...).

2.4 Fes una taula de les aerolínies amb més endarreriments acumulats.

2.5 Quins són els vols més llargs? I els més endarrerits? Busca les rutes més llargues i les que acumulen més retards.

2.6 Aporta allò que consideris rellevant.

2.1 Resumeix estadísticament el dataset i les columnes d'interès. Fes una anàlisi estadístic del que consideris rellevant.

- Queden 25 columnes amb 200.000 rangs i la següent inforació :

In [450…
```
data_reduced.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 200000 entries, 389857 to 1841885
Data columns (total 23 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   Month              200000 non-null  int64
 1   DayofMonth         200000 non-null  int64
 2   DayOfWeek          200000 non-null  int64
 3   DepTime            200000 non-null  float64
 4   CRSDepTime         200000 non-null  int64
 5   ArrTime            199303 non-null  float64
 6   CRSArrTime         200000 non-null  int64
 7   UniqueCarrier      200000 non-null  object
 8   FlightNum          200000 non-null  object
 9   TailNum            200000 non-null  object
 10  ActualElapsedTime  199166 non-null  float64
 11  CRSElapsedTime     199971 non-null  float64
 12  ArrDelay           199166 non-null  float64
 13  DepDelay           200000 non-null  float64
 14  Origin             200000 non-null  object
 15  Dest               200000 non-null  object
 16  Distance           200000 non-null  int64
 17  Cancelled          200000 non-null  int64
 18  CancellationCode   200000 non-null  object
 19  Diverted           200000 non-null  int64
 20  CarrierDelay       128644 non-null  float64
 21  WeatherDelay       128644 non-null  float64
 22  NASDelay           128644 non-null  float64
dtypes: float64(9), int64(8), object(6)
memory usage: 36.6+ MB
```

Per facilitar la vizualització i de proporcions a la pantalla, separem les vstes en 2 trams de dimensions similars

In [451…
```
data_reduced.iloc[:, 0:11]
```

Out[451]:

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	TailNum	ActualElapsedTime
389857	3	16	7	1756.0	1705	1859.0	1820	WN	WN 535	N235WN	63.0
103123	1	3	4	1800.0	1745	1947.0	1935	EV	EV 4408	N829AS	107.0
949448	6	15	7	1819.0	1800	1904.0	1851	OO	OO 5454	N564SW	45.0
1490492	9	16	2	714.0	655	821.0	810	MQ	MQ 3534	N520DC	67.0
1611124	10	28	2	1855.0	1800	2202.0	2125	AA	AA 1744	N615AA	127.0
...	...	...	...	...	...	...	...	...	...	...	...
604382	4	14	1	915.0	900	1012.0	1005	XE	XE 2548	N16961	57.0
1699150	11	22	6	820.0	810	1113.0	1110	MQ	MQ 3719	N622MQ	113.0
1026398	6	4	3	730.0	705	1123.0	1037	NW	NW 688	N376NW	173.0
1627406	10	13	1	2136.0	2130	2318.0	2306	DL	DL 1024	N920DL	102.0
1841885	12	26	5	1043.0	950	1243.0	1127	DL	DL 1917	N909DA	180.0

200000 rows × 11 columns

In [441…
```
data_reduced.iloc[:, 11:23]
```

Out[441]:

	CRSElapsedTime	ArrDelay	DepDelay	Origin	Dest	Distance	Cancelled	CancellationCode	Diverted	CarrierDelay	WeatherDelay	NASDelay
684212	126.0	23.0	23.0	MSP	BIL	748	0	N	0	23.0	0.0	0.0
1856686	94.0	51.0	55.0	MCO	ATL	403	0	N	0	51.0	0.0	0.0
1141550	47.0	57.0	68.0	SFO	SMF	86	0	N	0	57.0	0.0	0.0
459826	130.0	-10.0	9.0	MCO	IAD	758	0	N	0	NaN	NaN	NaN
1276706	60.0	13.0	13.0	HRL	HOU	276	0	N	0	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
171294	85.0	73.0	23.0	BUF	JFK	301	0	N	0	0.0	7.0	50.0
1134884	186.0	45.0	50.0	BDL	FLL	1173	0	N	0	45.0	0.0	0.0
330887	75.0	6.0	6.0	MEM	TYS	342	0	N	0	NaN	NaN	NaN
1262067	108.0	2.0	16.0	ATL	DCA	547	0	N	0	NaN	NaN	NaN
854043	150.0	-4.0	16.0	IAH	MIA	964	0	N	0	NaN	NaN	NaN

200000 rows × 12 columns

Estadístiques bàsiques amb la funció describe() :

- El análisis amb la funció bàsica "describe", podem veure que aquestes estadístques son rellevants en les dades en dies de l'any, o en minuts de demores i temps de duració, peró per veure infromació de les columnes d'objectes, no ens aporten.
- Ens adonem doncs que de que en molts casos es irrellevant fer un "count", sumar i fer mitjanes simples.
- El valor mínim (min) en algunes columnes crida l'atenció ; poden ser errors, vol en proves, vols cancel·lats, avions privats, altres casos d'excepció. *FlightNum :1, ActualElapsedTime :16, CRSElapsedTime : 1, ArrDelay :-68*.
- Es interessant contrastar la mitjana simple "mean" amb el 50% percentil, ja que en principi donen dades més o menys properes a les columnes *Month, DayofMonth, DayOfWeek, DepTime, CRSDepTime, ArrTime, CRSArrTime* i entre Cancelled i Diverted, però a partir de *ActualElapsedTime, CRSElapsedTime, ArrDelay, DepDelay, Distance* divergen molt i més encara quan observem les temps (min) per cada causa de retards amb *CarrierDelay, WeatherDelay, NASDelay, SecurityDelay, LateAircraftDelay*
- Sobre això :

```
  -> la mitjana simple (mean) és sensible als valors extrems, en canvi, els 50 dona una idea d'on es troben el 50% o la meitat dels valors i presenta
números molt més baixos que la mitjana simple.
  -> en el cas dels retards per cada causa, podem observar com la mitjana simple (mean) s'aproxima més al 75 percentil que al 50, volant dir que pocs
valors excepcionalment grans influencien les estadístiques del dataset.
```

In [452…
```
data_reduced.describe()
```

Out[452]:

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	ActualElapsedTime	CRSElapsedTime	ArrDelay	DepDelay	Distance	Can
count	200000.000000	200000.000000	200000.000000	200000.000000	200000.000000	199303.000000	200000.000000	199166.000000	199971.000000	199166.000000	200000.000000	200000.000000	200000.0
mean	6.108675	15.734130	3.981870	1517.858690	1466.615270	1610.232219	1633.935265	133.364339	134.368463	42.030321	43.008020	766.091130	0.0
std	3.485435	8.772513	1.992341	450.333037	424.796318	547.788308	464.636301	72.042475	71.355824	56.766096	53.396196	573.964249	0.0
min	1.000000	1.000000	1.000000	1.000000	0.000000	1.000000	0.000000	15.000000	1.000000	-109.000000	6.000000	31.000000	0.0
25%	3.000000	8.000000	2.000000	1203.000000	1135.000000	1315.000000	1325.000000	80.000000	81.000000	9.000000	12.000000	338.000000	0.0
50%	6.000000	16.000000	4.000000	1545.000000	1509.000000	1715.000000	1705.000000	116.000000	117.000000	24.000000	24.000000	610.000000	0.0
75%	9.000000	23.000000	6.000000	1900.000000	1815.000000	2031.000000	2014.000000	165.000000	165.000000	56.000000	53.000000	998.000000	0.0
max	12.000000	31.000000	7.000000	2400.000000	2359.000000	2400.000000	2400.000000	711.000000	660.000000	1951.000000	1952.000000	4962.000000	1.0

A més d'aquestes estadístiques básiques, provem d'analitzar el dataset amb la funció estadística del valor de MODA ("mode"), on veiem quin es cas més freqüent per cada variable

- Per el dataset que ens interessa es una informació rellevant donat que ens mostra dades recurrents, moments de l'any i del dia on es concentren els vols, els retards, vols, amb quina companya, quin es el dia i l'hora més habituals (més circulació d'avions als aeroports) ; també ens dona una idea de si lo habitual es que els avions surtin a temps, o al contrari, que tinguin retards (concretament a les últimes columnes)
- data.mode() ens dona tots els resultats a la fila de dalt (0) i deixa les altres buides. Per tant, li demanem "data.mode().head(1)"

In [453…
```
data_reduced.iloc[:, 0:11].mode().head(1)
```

Out[453]:

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	TailNum	ActualElapsedTime
0	12	22	5	1900.0	1800	2055.0	1930	WN	WN 50	N676SW	81.0

In [454…
```
data_reduced.iloc[:, 11:25].mode().head(1)
```

Out[454]:

	CRSElapsedTime	ArrDelay	DepDelay	Origin	Dest	Distance	Cancelled	CancellationCode	Diverted	CarrierDelay	WeatherDelay	NASDelay
0	75.0	10.0	6.0	ATL	ORD	337	0	N	0	0.0	0.0	0.0

- **Primeres observacions bàsiques però útils amb MODE :**

  - l'hora d'arribada real a destinació (ArrTime) que més surt és 20:30 ArrTime (amb el dataset original el mateix cálcul donava 21:00)
  - la duració més habitual d'aquesta llista de vols és de 75 minuts (amb el dataset original el mateix cálcul donava 80 min)
  - lo més habitual es que els vols no es calcel·lin, ni que siguin desviats
  - la companya aéria més utilitzada per a aquest dataset és WN (Southwest Airlines)
  - l'aeroport de surtida que més vegades apareix és Atlanta (ATL) i el de destí, Orlando (ORD)
  - el vol número 16 (molt probablement WN16) es el que més vols efectua i apareix més vegades al dataset
  - les el temps de retard més habitual sol ser de 10 minuts a l'arribada i de 6 minuts a la sortida dels vols
  - la distància més freqüent es de 337 (km o miles)
  - lo més habitual es que els avions surtin bastant a temps i com previst, tenint en compte que si el retard es menor a 15 minuts, no compta com a retard

```
- MODE també ens dona pistes sobre el tràfic a les pistes i les portes dels diferents aeroports:

          - el mes de l'any on hi han més vols es desembre (mes 12)
          - el dia del mes amb més sortides ("departures") sol ser el 22,i el de la setmana, el divendres (dia 5)
          - l'hora de sortida de més afluencia de vols (no necessariament de persones ja que depén del tamany de cada aeronau) tant la prevista = real
de és a les 18:00
          - l'hora d'arribada planificada més freqüentment per les aerolínies es a les 19:30 del vespre, mentres que la realitat s'aproxima més cap a
les 21:00 de la nit.
          Això podria indicar que els viatgers que planifiquen volar pel vespre tenen bastants probabilitats d'arribar tard al seu destí.
```

2.2 Troba quantes dades faltants hi ha per columna.

In [427…
```
# Aquí veiem quants valors nuls hi ha a les columnes del nou dataframe, des de les columnes més complertes a les que ho són menys.

null_counts = data_reduced.isnull().sum()
null_counts[null_counts > 0].sort_values(ascending=True)
```

Out[427]:
```
CRSElapsedTime          25
ArrTime                742
ActualElapsedTime      884
ArrDelay               884
CarrierDelay         71170
WeatherDelay         71170
NASDelay             71170
dtype: int64
```

2.3 Crea columnes noves (velocitat mitjana del vol, si ha arribat tard o no...).

Creació d'una nova columna *AverageSpeed*, o velocitat mitjana del vol :

In [459…
```
# Creació de la columna de velocitat mitjana AverageSpeed per cada viatge.
data_reduced['AverageSpeed'] = (data_reduced['Distance']/data_reduced['ActualElapsedTime']*60)
data_reduced['AverageSpeed'] = data_reduced['AverageSpeed'].round(1)
```

Creació de la nova columna *LateCode* :

La velocitat mitjana del vol es calcula en x miles/hour => tindrém que fer una conversió -> Distance / ActualElapsedTime són miles/minut

- -> passar de miles per minut a miles per hora -> resultat *60 min

In [460…
```
# Creació de la columna del nivells d'enderririment "LateCode" per cada viatge.
# crearem un codi que digui si ha arribat a temps (0), o en cas d'arribar tard, de quina magnitud es la demora (del 1 al 3)

LateCode=[]
for x in data_reduced['ArrDelay']:
    if x > 60 :
        LateCode.append(3)
    elif x > 30 :
        LateCode.append(2)
    elif x > 15 :
        LateCode.append(1)
    else:
        LateCode.append(0)

data_reduced['LateCode']=LateCode

# "AverageSpeed", "LateCode" i "Routes" es poden veure a la part dreta del dataset "data_reduced"
```

In [461…
```
# les dades de les columnes per comprobació:
data_reduced['AverageSpeed'].head(5), data_reduced['LateCode'].head(5)
```

Out[461]:
```
(389857     243.8
 103123     347.1
 949448     145.3
 1490492    272.2
 1611124    464.9
 Name: AverageSpeed, dtype: float64,
 389857     2
 103123     0
 949448     0
 1490492    0
 1611124    2
 Name: LateCode, dtype: int64)
```

Creació d'una nova columna *Route*:

In [456…
```
data_reduced['Route'] = data_reduced['Origin'].astype(str) + ' - ' + data_reduced['Dest'].astype(str)
data_reduced['Route']
```

Out[456]:
```
389857     LAS - PHX
103123     ATL - MDT
949448     LAX - BFL
1490492    LIT - DFW
1611124    DFW - MCO
               ...
604382     DAL - IAH
1699150    DFW - TYS
1026398    MSP - PHL
1627406    ATL - IND
1841885    ATL - DFW
Name: Route, Length: 200000, dtype: object
```

In [462…
```
data_reduced
```

Out[462]:

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	TailNum	...	Distance	Cancelled	CancellationCode	Diverted	CarrierDelay	WeatherD
389857	3	16	7	1756.0	1705	1859.0	1820	WN	WN 535	N235WN	...	256	0	N	0	0.0	
103123	1	3	4	1800.0	1745	1947.0	1935	EV	EV 4408	N829AS	...	619	0	N	0	NaN	
949448	6	15	7	1819.0	1800	1904.0	1851	OO	OO 5454	N564SW	...	109	0	N	0	NaN	
1490492	9	16	2	714.0	655	821.0	810	MQ	MQ 3534	N520DC	...	304	0	N	0	NaN	
1611124	10	28	2	1855.0	1800	2202.0	2125	AA	AA 1744	N615AA	...	984	0	N	0	30.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
604382	4	14	1	915.0	900	1012.0	1005	XE	XE 2548	N16961	...	217	0	N	0	NaN	
1699150	11	22	6	820.0	810	1113.0	1110	MQ	MQ 3719	N622MQ	...	772	0	N	0	NaN	
1026398	6	4	3	730.0	705	1123.0	1037	NW	NW 688	N376NW	...	980	0	N	0	0.0	
1627406	10	13	1	2136.0	2130	2318.0	2306	DL	DL 1024	N920DL	...	432	0	N	0	NaN	
1841885	12	26	5	1043.0	950	1243.0	1127	DL	DL 1917	N909DA	...	732	0	N	0	0.0	

200000 rows × 26 columns

2.4 Fes una taula de les aerolínies amb més endarreriments acumulats.

Primer obtenim només els vols del dataset que no tinguin enderreriments "oficials" (no més de 15 min)

-> ho fem filtrant només els vols amb codi LateCode que no sigui 0, (on 0 es que ha arribat a l'hora)

In [463…
```
VuelosRetrasados = data_reduced[(data_reduced.LateCode > 0)]
```

En resum, les aerolinies menys puntuals en numero de vols (indepentdentment del seu tamany), de més a menys

-> l'aerolinia amb més retards és Sowthwest Airlines (WN) amb 20308 registres, la que també va apareixer en les estadístiques inicials amb "mode".

-> la segona és American Airlines (AA) i la tercera, Envoy Air (MQ).

In [464…
```
CarrierDelayed=VuelosRetrasados[['UniqueCarrier','LateCode']].groupby(['UniqueCarrier']).count()
CarrierDelayed.sort_values(by=['LateCode'], ascending=False)
```

Out[464]:

UniqueCarrier	LateCode
WN	20313
AA	13279
MQ	9883
UA	9583
OO	8828
XE	7325
DL	7230
US	5969
CO	5951
EV	5818
NW	5407
YV	5035
FL	4661
OH	3957
B6	3852
9E	3630
AS	2443
F9	1586
HA	409
AQ	38

2.5 Quins són els vols més endarrerits? Els més llargs? Busca les rutes més llargues i les que acumulen més retards.

Els vols que MÉS SOVINT s'han endarerrit són els vols número WN50, WN40, AS64, WN48, AS61.

In [465…
```
Vols=VuelosRetrasados[['FlightNum','LateCode']].groupby(['FlightNum']).count()
Vols.sort_values(by=['LateCode'], ascending=False)
```

Out[465]:

FlightNum	LateCode
WN 50	57
WN 40	51
AS 64	45
WN 48	42
AS 61	40
...	...
B6 713	1
HA 160	1
WN 3074	1
B6 715	1
AS 811	1

23328 rows × 1 columns

En canvi, nos son els mateixos els vols que MÉS MINUTS DE RETARD CUMULEN.

In [469…
```
VolsMinDelay=VuelosRetrasados[['FlightNum','ArrDelay']].groupby(['FlightNum']).sum()
VolsMinDelay.sort_values(by=['ArrDelay'], ascending=False)
```

Out[469]:

FlightNum	ArrDelay
XE 2362	3941.0
AA 1555	3617.0
MQ 4096	3565.0
B6 547	3377.0
AA 510	3333.0
...	...
OH 5782	16.0
WN 2921	16.0
WN 2917	16.0
CO 1061	16.0
US 119	16.0

23328 rows × 1 columns

Els vols més llargs (en milles -> distancia) son els següents, prenent el trajecte que ha sigut més llarg de tots els dies que un vol va operar amb un mateix codi:

In [500…
```
Dist=data_reduced[['FlightNum','Distance']].groupby(['FlightNum']).max()
Dist.sort_values(by=['Distance'], ascending=False)
```

Out[500]:

FlightNum	Distance
CO 14	4962
CO 15	4962
DL 1561	4502
DL 851	4502
DL 1282	4502
...	...
OH 6280	64
OO 5458	49
OO 5457	49
OO 5456	49
OO 5514	49

24938 rows × 1 columns

Els vols que han sigut més llargs en MINUTS de duració real, son els següents, prenent el trajecte que ha sigut més llarg de tots els dies que un vol va operar amb un mateix codi :

In [499…
```
LlargTemps=data_reduced[['FlightNum','ActualElapsedTime']].groupby(['FlightNum']).max()
LlargTemps.sort_values(by=['ActualElapsedTime'], ascending=False)
```

Out[499]:

FlightNum	ActualElapsedTime
CO 15	711.0
UA 29	679.0
AA 73	625.0
DL 1273	609.0
DL 851	605.0
...	...
OO 6178	NaN
OO 6780	NaN
US 1478	NaN
WN 3718	NaN
XE 2090	NaN

24938 rows × 1 columns

Les rutes que més retards acumulen, en número D'HORES, son les següents:

-> Chicago O'hare-Laguardia ; Laguardia-Chicago O'hare ; Atlanta-Newark ; Los Angeles-San Francisco ; Chicago O'hare-Newark, etc.
-> Podem constatar que en molts casos sortend d'aeroports newyorquins especialitzats amb vols "charters/barats" regionals, mentres que JFK que és més internacional, cumula menys minuts de retard

In [502…

```
data_reduced['Route']
VolsMinDelay=((VuelosRetrasados[['Route','ArrDelay']].groupby(['Route']).sum())/60).round(1)  # /60 per convertir minuts en hores
VolsMinDelay.sort_values(by=['ArrDelay'], ascending=False).head(20)
```

Out[502]:

Route	ArrDelay
ORD - LGA	472.5
LGA - ORD	371.6
ATL - EWR	359.2
LAX - SFO	344.2
ORD - EWR	324.5
CLT - EWR	303.6
EWR - ORD	298.9
SFO - LAX	290.4
DFW - ORD	289.8
ATL - LGA	289.6
PHL - ORD	277.8
LGA - ATL	263.9
JFK - BOS	248.5
ORD - PHL	242.6
ATL - ORD	235.8
ORD - MSP	234.9
ORD - BOS	234.2
MSP - ORD	232.4
BOS - JFK	229.7
JFK - LAX	217.7

Les rutes que més llargues en MILLES, son les que comuniquen ciutats com Nova York, Atlanta, Chicago o Minneapolis amb aeroports de Hawai:

-> Newark-Honolulu ; Atlanta-Honolulu ; Onizuka Kona-Chicago O'Hare, Houston-Honolulu, i vice-versa.

In [504…

```
RoutesDist=data_reduced[['Route','Distance']].groupby(['Route']).max()
RoutesDist.sort_values(by=['Distance'], ascending=False).head(15)
```

Out[504]:

Route	Distance
EWR - HNL	4962
HNL - EWR	4962
ATL - HNL	4502
HNL - ATL	4502
HNL - ORD	4243
ORD - HNL	4243
KOA - ORD	4213
ORD - OGG	4184
MSP - HNL	3972
HNL - MSP	3972
IAH - HNL	3904
HNL - IAH	3904
HNL - DFW	3784
DFW - HNL	3784
OGG - DFW	3711

2.6 Aporta allò que consideris rellevant.

Aqui veiem quants viatges han sigut puntuals (codi 0), i quands han tingut endarreriments, per codi de retard (1, 2, 3)

-> la taula mostra que sobre 200.000 viatges en total, hi ha hagut més viatges enderrerits (uns 125.000) que no pas a l'hora (74.692)

-> aprox. 75.000, encara que 'a l'hora' sigui el valor amb més freqüencia dels tres (com ens anava anuniant la funció "mode" al principi)

-> observem també que sobre 200.000 viatges en total, uns 85.000 van sortir al menys 30 minuts més tard de l'hora prevista.

In [505…
```
RetrasoVuelosCode=data_reduced[['UniqueCarrier','LateCode']].groupby(['LateCode']).count()
RetrasoVuelosCode
```

Out[505]:

LateCode	UniqueCarrier
0	74803
1	39831
2	40139
3	45227

Les aerolinies amb més cancel·lacions : *Envoy Air* (MQ) és la que més vols ha tingut que cancel·lar

In [506…
```
CarrierCancelations=Cancelations[['UniqueCarrier','Cancelled']].groupby(['UniqueCarrier']).count()
CarrierCancelations.sort_values(by=['Cancelled'], ascending=False)
```

Out[506]:

UniqueCarrier	Cancelled
MQ	12
OO	10
YV	7
EV	6
XE	5
9E	4
CO	4
UA	4
AA	3
OH	3
B6	2
DL	2
NW	2
US	2
WN	2

Els mesos de l'any amb més cancel·lacions: desembre, novembre i octubre son els únics mesos on hi ha hagut vols cancel·lats

In [507…
```
Cancel=Cancelations[['Month','Cancelled']].groupby(['Month']).count()
Cancel.sort_values(by=['Cancelled'], ascending=False)
```

Out[507]:

Month	Cancelled
12	50
11	12
10	6

En canvi, NO sempre son els mateixos que els mesos amb més HORES ACUMULADES d'endarreriments, que són : desembre, juny, març, febrer i gener.

-> el primers són els mesos que corresponen amb les festes de Nadal i els viatges d'estiu, del "March Break dels estudiants" i els dies lliures de l'Any Nou.

In [508…
```
RetardsMesos=VuelosRetrasados[['Month','DepDelay']].groupby(['Month']).count()
Result=RetardsMesos.sort_values(by=['DepDelay'], ascending=False)
(Result/60).round(1)
```

Out[508]:

Month	DepDelay
12	237.3
6	231.8
2	219.1
3	214.8
1	202.7
7	197.7
8	166.4
4	163.5
5	158.1
11	107.4
10	94.3
9	93.4

Les aeronaus que van recorre més distància EN MILLES -> interessant per el seu manteniment ("TailNum" es la matrícula única a cada avió)

In [509…
```
Aeronaus_temps=data_reduced[['TailNum','Distance']].groupby(['TailNum']).sum()
Aeronaus_temps.sort_values(by=["Distance"], ascending=False)
```

Out[509]:

TailNum	Distance
N338AA	91585
N56859	89699
N167US	82889
N829MH	82407
N454UA	82091
...	...
N253MQ	109
N324MQ	109
N317MQ	109
N828AL	100
89439E	74

5265 rows × 1 columns

Quants vols han sortit de cada aeroport :

-> els més ocupats -> Atlanta, Chicago O'Hare, Dallas Fort Worth, Denver, Los Angeles
-> els menys : Pierre Regional(South Dakota), Yellowstone, el Pueblo, Chicago/Rockford, Tri-State/Milton

In [ ]:
```
Vols_per_aeroport= (
  data_reduced
  .groupby(['Origin'])
  .apply(lambda x: pd.Series({
      'nb_flights': len(x['FlightNum'])
  }))
)
Vols_per_aeroport.sort_values(by=['nb_flights'], ascending=False)
```

Out[ ]:

Origin	nb_flights
ATL	13604
ORD	12828
DFW	9872
DEN	7536
LAX	5989
...	...
HTS	2
RFD	1
PUB	1
WYS	1
PIR	1

299 rows × 1 columns

Els aeroports d'on els avions triguen més en sortir:

-> aeroports que presenten **més codis de retard (independentment de la magnitud)**:

-> son : Chicago O'hare, Atlanta, Dallas Fort Worth, Denver, Newark, que son dels aeroports que ja veiem en el càlcul anterior

In [511…
```
AeroportsRetras=VuelosRetrasados[['Origin','LateCode']].groupby(['Origin']).count()
AeroportsRetras.sort_values(by=['LateCode'], ascending=False)
```

Out[511]:

Origin	LateCode
ORD	9016
ATL	8779
DFW	6223
DEN	4608
EWR	3718
...	...
YKM	2
SLE	1
RFD	1
PUB	1
PIR	1

297 rows × 1 columns

Rutes amb més i menys retards per RAONS CLIMÀTIQUES :

-> més (entre ciutats del nord): Detroit-Laguardia ; Philadelphia-Boston ; Chicago O'hare - Cincinnati ; Detroit - Newark ; Kennedy - Boston

-> menys (entre ciutats del sud): Las Vegas - Jacksonville Fl. ; Las Vegas - Long Beach Ca. ; Las Vegas - Clinton Az. ; Las Vegas - Midland Tx. ; Yuma Az. - Salt Lake City

In [512…
```
Weather=VuelosRetrasados[['Route','WeatherDelay']].groupby(['Route']).sum()
Weather.sort_values(by=['WeatherDelay'], ascending=False)
```

Out[512]:

Route	WeatherDelay
DTW - LGA	2726.0
PHL - BOS	2015.0
ORD - CVG	1911.0
DTW - EWR	1712.0
JFK - BOS	1559.0
...	...
LAS - JAX	0.0
LAS - LGB	0.0
LAS - LIT	0.0
LAS - MAF	0.0
YUM - SLC	0.0

4711 rows × 1 columns

Exercici 3. Exporta el dataset a Excel

In [247…
```
data_reduced.to_excel("data_reduced_delays_flights.xlsx")
```