Documento Técnico

Teresa Álvarez de Portugal

May 2023

1 Introduction

Este documento contendrá el código fuente Dask empleado para la resolución de cada una de las tareas. El código lo insertarñe como imágenes y con un tamaño que permita leer el texto contenido en las imágenes.

2 Código

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import dask
import dask.dataframe as dd
import os
import graphviz
from IPython.display import display
```

✓ 4.5s

```python
#Cargo el csv usando dask
df = dd.read_csv('air_traffic_data.csv')
```

✓ 0.1s

```python
df.head(10)
```

[3] Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Deplaned	Low Fare	Terminal 1	B	27271	Deplaned	27271	2005	July
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Enplaned	Low Fare	Terminal 1	B	29131	Enplaned	29131	2005	July
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2	10830	2005	July
3	200507	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 1	B	35156	Deplaned	35156	2005	July
4	200507	Air Canada	AC	Air Canada	AC	International	Canada	Enplaned	Other	Terminal 1	B	34090	Enplaned	34090	2005	July
5	200507	Air China	CA	Air China	CA	International	Asia	Deplaned	Other	International	G	6263	Deplaned	6263	2005	July
6	200507	Air China	CA	Air China	CA	International	Asia	Enplaned	Other	International	G	5500	Enplaned	5500	2005	July
7	200507	Air France	AF	Air France	AF	International	Europe	Deplaned	Other	International	A	12050	Deplaned	12050	2005	July
8	200507	Air France	AF	Air France	AF	International	Europe	Enplaned	Other	International	A	11638	Enplaned	11638	2005	July
9	200507	Air New Zealand	NZ	Air New Zealand	NZ	International	Australia / Oceania	Deplaned	Other	International	G	4998	Deplaned	4998	2005	July

```python
len(df)
```

[4] Python

```
15007
```

```python
tipo_datos = df.dtypes
display(tipo_datos.head(16))
```

```
Activity Period                  int64
Operating Airline               object
Operating Airline IATA Code     object
Published Airline               object
Published Airline IATA Code     object
GEO Summary                     object
GEO Region                      object
Activity Type Code              object
Price Category Code             object
Terminal                        object
Boarding Area                   object
Passenger Count                  int64
Adjusted Activity Type Code     object
Adjusted Passenger Count         int64
Year                             int64
Month                           object
dtype: object
```

```python
#contar si hay valores nulos
nulos= df.isna().any().compute()
display(nulos)
```

```
Activity Period                 False
Operating Airline               False
Operating Airline IATA Code      True
Published Airline               False
Published Airline IATA Code      True
GEO Summary                     False
GEO Region                      False
Activity Type Code              False
Price Category Code             False
Terminal                        False
Boarding Area                   False
Passenger Count                 False
```

```python
#Me ha devuelto que hay dos columnas con valores nulos, identifico cuales son las celdas con esos valores
celda_nulos1= df.loc[df['Operating Airline IATA Code'].isna(),'Operating Airline IATA Code'].compute()
display(celda_nulos1)
#Cuento el número de celdas nulas
nceldas_nulas1=df['Operating Airline IATA Code'].isna().sum().compute()
display(nceldas_nulas1)
```

```
148      NaN
6814     NaN
6815     NaN
6925     NaN
6926     NaN
7173     NaN
7174     NaN
7747     NaN
7748     NaN
7972     NaN
7973     NaN
8327     NaN
8328     NaN
8444     NaN
8445     NaN
8562     NaN
8563     NaN
8680     NaN
8793     NaN
8794     NaN
8795     NaN
8796     NaN
9131     NaN
9132     NaN
9357     NaN
...
12011    NaN
12125    NaN
13025    NaN
13026    NaN
Name: Operating Airline IATA Code, dtype: object
```

Cerrar

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings

```python
#Hago lo mismo para la otra columna con nulos
celda_nulos2= df.loc[df['Published Airline IATA Code'].isna(),'Published Airline IATA Code'].compute()
display(celda_nulos2)
nceldas_nulas2=df['Published Airline IATA Code'].isna().sum().compute()
display(nceldas_nulas2)
```

[8]

```
148      NaN
6814     NaN
6815     NaN
6925     NaN
6926     NaN
7173     NaN
7174     NaN
7747     NaN
7748     NaN
7972     NaN
7973     NaN
8327     NaN
8328     NaN
8444     NaN
8445     NaN
8562     NaN
8563     NaN
8680     NaN
8793     NaN
8794     NaN
8795     NaN
8796     NaN
9131     NaN
9132     NaN
9357     NaN
         ...
12011    NaN
12125    NaN
13025    NaN
13026    NaN
Name: Published Airline IATA Code, dtype: object
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
#Consultar los registros que tienen los valores nulos de todo el df
registros_nulos=df.loc[df.isnull().any(axis=1),:].compute()
display(registros_nulos)
```

Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
148	200508	Boeing Company	NaN	Boeing Company	NaN	Domestic	US	Deplaned	Other	Other	Other	18	Deplaned	18	2005	August
6814	201005	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	G	73	Deplaned	73	2010	May
6815	201005	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	G	73	Enplaned	73	2010	May
6925	201006	Pacific Aviation	NaN	Pacific Aviation	NaN	International	Europe	Deplaned	Other	International	A	160	Deplaned	160	2010	June
6926	201006	Pacific Aviation	NaN	Pacific Aviation	NaN	International	Europe	Enplaned	Other	International	A	160	Enplaned	160	2010	June
7173	201008	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	G	118	Deplaned	118	2010	August
7174	201008	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	G	118	Enplaned	118	2010	August
7747	201101	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	40	Deplaned	40	2011	January
7748	201101	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	40	Enplaned	40	2011	January
7972	201103	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	64	Deplaned	64	2011	March
7973	201103	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	64	Enplaned	64	2011	March
8327	201106	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	237	Deplaned	237	2011	June
8328	201106	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	262	Enplaned	262	2011	June
8444	201107	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	72	Deplaned	72	2011	July
8445	201107	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	193	Enplaned	193	2011	July
8562	201108	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	114	Deplaned	114	2011	August
8563	201108	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	98	Enplaned	98	2011	August
8680	201109	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	98	Deplaned	98	2011	September
8793	201110	Servisair	NaN	Servisair	NaN	Domestic	US	Deplaned	Low Fare	International	A	48	Deplaned	48	2011	October
8794	201110	Servisair	NaN	Servisair	NaN	Domestic	US	Enplaned	Low Fare	International	A	48	Enplaned	48	2011	October
8795	201110	Servisair	NaN	Servisair	NaN	International	Europe	Deplaned	Low Fare	International	A	16	Deplaned	16	2011	October

```python
def reemplazar_nulos(fila):
    if pd.isnull(fila['Operating Airline IATA Code']):
        if fila['Published Airline'] == 'Boeing Company':
            return 'BOC'
        elif fila['Published Airline'] == 'Servisair':
            return 'SVA'
        elif fila['Published Airline'] == 'Swissport USA':
            return 'SWU'
        elif fila['Published Airline'] == 'Pacific Aviation':
            return 'PAV'
    else:
        return fila['Operating Airline IATA Code']

# Aplica la función personalizada a través de map_partitions() para reemplazar los nulos
df_lleno = df.map_partitions(lambda df: df.apply(reemplazar_nulos, axis=1), meta=('Operating Airline IATA Code', 'object'))
df['Operating Airline IATA Code'] = df_lleno
```

```python
#Realizo la misma operación para la columna 'Published Airline IATA code'
def reemplazar_nulos2(fila):
    if pd.isnull(fila['Published Airline IATA Code']):
        if fila['Published Airline'] == 'Boeing Company':
            return 'BOC'
        elif fila['Published Airline'] == 'Servisair':
            return 'SVA'
        elif fila['Published Airline'] == 'Swissport USA':
            return 'SWU'
        elif fila['Published Airline'] == 'Pacific Aviation':
            return 'PAV'
    else:
        return fila['Published Airline IATA Code']

# Aplica la función personalizada a través de map_partitions() para reemplazar los nulos
df_lleno = df.map_partitions(lambda df: df.apply(reemplazar_nulos2, axis=1), meta=('Operating Airline IATA Code', 'object'))
df['Published Airline IATA Code'] = df_lleno
```

```python
#Comrpuebo a ver si hay nulos
nulos= df.isna().any().compute()
display(nulos)
```

```
Activity Period                False
Operating Airline              False
Operating Airline IATA Code    False
Published Airline              False
Published Airline IATA Code    False
GEO Summary                    False
GEO Region                     False
Activity Type Code             False
Price Category Code            False
Terminal                       False
Boarding Area                  False
Passenger Count                False
Adjusted Activity Type Code    False
Adjusted Passenger Count       False
Year                           False
Month                          False
dtype: bool
```

Ahora si podemos trabajar con nuestro dataset para resolver las cuestiones propuestas

Ejercicio 2

Ahora resolveré las cuestiones del ejercicio 2 sobre el dataset limpio

- ¿Cuántas compañias diferentes aparecen en el fichero?

```python
#¿Cuantas compañias diferentes aparecen en el fichero?
    #Analizando el fichero se puede comprobar que hay dos columnas que contienen compañias
    #La columna 'Operating Airline' y la columna 'Published Airline'
    #Voy a combinar ambas columnas y sacar los valores unicos de la columna resultante


compañias_combinadas= dd.concat([df['Operating Airline'], df['Published Airline']])
compañias_total= compañias_combinadas.unique().compute().tolist()
display(len(compañias_total))
display(compañias_total)
```

```
77

['ATA Airlines',
 'Air Canada ',
 'Air China',
 'Air France',
 'Air New Zealand',
 'AirTran Airways',
 'Alaska Airlines',
 'All Nippon Airways',
 'American Airlines',
 'American Eagle Airlines',
 'Asiana Airlines',
 'Atlantic Southeast Airlines',
 'BelAir Airlines',
 'British Airways',
 'Cathay Pacific',
 'China Airlines',
 'Delta Air Lines',
```

- ¿Cuántos pasajeros tiene de media los vuelos de cada compañía?

```python
#Observo que hay dos columnas que contienen pasajeros
#Las comparo a ver si son iguales

pasajeros_iguales= df['Passenger Count'] == df['Adjusted Passenger Count']
display(pasajeros_iguales.compute())
```

```
0         True
1         True
2        False
3         True
4         True
         ...
15002     True
15003     True
15004     True
15005     True
15006     True
Length: 15007, dtype: bool
```

```python
#Veo que no son iguales asique saco las filas con datos diferentes e interpreto los datos

filas_diferentes= df.loc[df['Passenger Count'] != df['Adjusted Passenger Count']]
display(filas_diferentes.compute())
```

Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2	10830	2005	July
15	200507	Alaska Airlines	AS	Alaska Airlines	AS	Domestic	US	Thru / Transit	Other	International	A	3678	Thru / Transit * 2	7356	2005	July
20	200507	Alaska Airlines	AS	Alaska Airlines	AS	International	Mexico	Thru / Transit	Other	International	A	2266	Thru / Transit * 2	4532	2005	July
95	200507	United Airlines - Pre 07/01/2013	UA	United Airlines - Pre 07/01/2013	UA	Domestic	US	Thru / Transit	Other	Terminal 3	F	11388	Thru / Transit * 2	22776	2005	July
98	200507	United Airlines - Pre 07/01/2013	UA	United Airlines - Pre 07/01/2013	UA	International	Asia	Thru / Transit	Other	International	G	3953	Thru / Transit * 2	7906	2005	July
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
14860	201602	United Airlines	UA	United Airlines	UA	Domestic	US	Thru / Transit	Other	Terminal 3	F	16	Thru / Transit * 2	32	2016	February
14965	201603	Southwest Airlines	WN	Southwest Airlines	WN	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	29	Thru / Transit * 2	58	2016	March
14978	201603	United Airlines	UA	United Airlines	UA	Domestic	US	Thru / Transit	Other	International	G	169	Thru / Transit * 2	338	2016	March
14981	201603	United Airlines	UA	United Airlines	UA	Domestic	US	Thru / Transit	Other	Terminal 3	E	114	Thru / Transit * 2	228	2016	March
14984	201603	United Airlines	UA	United Airlines	UA	Domestic	US	Thru / Transit	Other	Terminal 3	F	39	Thru / Transit * 2	78	2016	March

920 rows × 16 columns

```python
#Observo que las filas con diferentes datos es pq en la columna de pasajeros ajustados se ha multiplicado el numero de pasajeros por 2
#Esto es pq en la columna de código de tipo de actividad ajustado aparece un *2 en las filasa que se ha multiplicado por dos
#Voy a interpretar que tengo que usar la columna de pasajeros que no esta ajustada
#utilizo la lista creada previamente 'compañias_total' donde esta la combinacion de las dos columnas que contienen las compañias
#Filtro el df para las aerolíneas presentes en esa lista y calculo el número medio de pasajeros por compañia
df_filtrado= df[df['Operating Airline'].isin(compañias_total) | df['Published Airline'].isin(compañias_total)]
media_pasajeros= df_filtrado.groupby(['Operating Airline', 'Published Airline']).mean()
display(media_pasajeros.compute())
```

Operating Airline	Published Airline	Activity Period	Passenger Count	Adjusted Passenger Count	Year
ATA Airlines	ATA Airlines	200586.363636	8744.636364	9661.659091	2005.795455
Aer Lingus	Aer Lingus	201151.469388	4407.183673	4407.183673	2011.448980
Aeromexico	Aeromexico	201207.533333	5463.822222	5463.822222	2012.011111
Air Berlin	Air Berlin	201107.500000	2320.750000	2320.750000	2011.000000
Air Canada	Air Canada	201123.497268	18251.560109	18251.560109	2011.169399
...	...	...	...	...	...
Virgin Atlantic	Virgin Atlantic	201043.744186	9847.104651	9847.104651	2010.372093
WestJet Airlines	WestJet Airlines	201125.844660	5338.155340	5338.155340	2011.184466
World Airways	World Airways	201008.333333	261.666667	261.666667	2010.000000
XL Airways France	XL Airways France	201339.096774	2223.161290	2240.129032	2013.322581
Xtra Airways	Xtra Airways	200608.000000	73.000000	73.000000	2006.000000

82 rows × 4 columns

```python
df_filtrado= df[df['Operating Airline'].isin(compañias_total) | df['Published Airline'].isin(compañias_total)]
media_pasajeros= df_filtrado.groupby(['Operating Airline', 'Published Airline'])['Passenger Count'].mean().reset_index()
lista_media_pasajeros= media_pasajeros.values.compute()
display(len(lista_media_pasajeros))
display(lista_media_pasajeros)
```

```
82

array([['ATA Airlines', 'ATA Airlines', 8744.636363636364],
       ['Aer Lingus', 'Aer Lingus', 4407.183673469388],
       ['Aeromexico', 'Aeromexico', 5463.822222222222],
       ['Air Berlin', 'Air Berlin', 2320.75],
       ['Air Canada ', 'Air Canada ', 18251.560109289618],
       ['Air Canada Jazz', 'Air Canada ', 294.2142857142857],
       ['Air China', 'Air China', 6618.335907335907],
       ['Air France', 'Air France', 11589.077519379845],
       ['Air India Limited', 'Air India Limited', 2834.5],
       ['Air New Zealand', 'Air New Zealand', 7452.339768339768],
       ['AirTran Airways', 'AirTran Airways', 10569.238938053097],
       ['Alaska Airlines', 'Alaska Airlines', 17251.637816245006],
       ['All Nippon Airways', 'All Nippon Airways', 6385.523255813953],
       ['Allegiant Air', 'Allegiant Air', 1516.8125],
       ['American Airlines', 'American Airlines', 127164.38970588235],
       ['American Eagle Airlines', 'American Airlines',
        4006.5283018867926],
       ['Ameriflight', 'Ameriflight', 5.0],
       ['Asiana Airlines', 'Asiana Airlines', 5902.961240310077],
       ['Atlantic Southeast Airlines', 'Delta Air Lines',
        2176.909090909091],
       ['Atlas Air, Inc', 'Atlas Air, Inc', 34.0],
       ['BelAir Airlines', 'BelAir Airlines', 415.3636[illegible]64],
       ['Boeing Company', 'Boeing Company', 18.0],
       ['British Airways', 'British Airways', 17625.124031007752],
...
       ['Virgin Atlantic', 'Virgin Atlantic', 9847.10465116279],
       ['WestJet Airlines', 'WestJet Airlines', 5338.155339805825],
       ['World Airways', 'World Airways', 261.6666666666667],
       ['XL Airways France', 'XL Airways France', 2223.1612903225805],
       ['Xtra Airways', 'Xtra Airways', 73.0]], dtype=object)
```

Cerrar

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

- Eliminar los registros duplicados del campo 'Geo Region', manteniendo unicamente aquel con mayor número de pasajeros

```
#Para esto voy a agrupar los datos de la columna Geo region
#luego calculo el valor mñaximo valor de las columnas de pasajeros para cada grupo de geo region
#Finalmente restablezco el índice del df resultante

df_deduplicado= df.groupby('GEO Region').agg({'Passenger Count':'max', 'Adjusted Passenger Count': 'max'}).reset_index()
display(df_deduplicado.compute())
```

	GEO Region	Passenger Count	Adjusted Passenger Count
0	US	659837	659837
1	Canada	39798	39798
2	Asia	86398	86398
3	Europe	48136	48136
4	Australia / Oceania	12973	12973
5	Mexico	29206	29206
6	Central America	8970	8970
7	Middle East	14769	14769
8	South America	3685	3685

- Volcar datos a un CSV nuevo

```
#Realizo un merge con el df limpio y el df que he realizado en el anterior apartado
df_nuevo= dd.merge(df, df_deduplicado, on=['GEO Region','Passenger Count','Adjusted Passenger Count']).compute()
display(df_nuevo)
```

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200708	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 3	E	39798	Deplaned	39798	2007	August
1	200708	United Airlines - Pre 07/01/2013	UA	United Airlines - Pre 07/01/2013	UA	International	Asia	Deplaned	Other	International	G	86398	Deplaned	86398	2007	August
2	201101	LAN Peru	LP	LAN Peru	LP	International	South America	Deplaned	Other	International	A	3685	Deplaned	3685	2011	January
3	201308	United Airlines	UA	United Airlines	UA	Domestic	US	Deplaned	Other	Terminal 3	F	659837	Deplaned	659837	2013	August
4	201407	United Airlines	UA	United Airlines	UA	International	Mexico	Deplaned	Other	International	G	29206	Deplaned	29206	2014	July
5	201410	TACA	TA	TACA	TA	International	Central America	Deplaned	Other	International	A	8970	Deplaned	8970	2014	October
6	201501	Air New Zealand	NZ	Air New Zealand	NZ	International	Australia / Oceania	Enplaned	Other	International	G	12973	Enplaned	12973	2015	January
7	201507	Emirates	EK	Emirates	EK	International	Middle East	Deplaned	Other	International	A	14769	Deplaned	14769	2015	July
8	201507	United Airlines	UA	United Airlines	UA	International	Europe	Deplaned	Other	International	G	48136	Deplaned	48136	2015	July

```
df_nuevo.to_csv('air_traffic_data_nuevo.csv', index=False)
```

Ejercicio 3. Análisis descriptivo

En este notebook se resolveran las cuestiones pedidas en el ejercicio 3 sobre el primer dataset

Para esto vuelvo a hacer limpieza de los datos nulos para poder trabajar bien

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import dask
import dask.dataframe as dd
from IPython.display import display
```

```
df_clean = dd.read_csv('air_traffic_data_lleno.csv')
display(df_clean.compute())
```

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Deplaned	Low Fare	Terminal 1	B	27271	Deplaned	27271	2005	July
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Enplaned	Low Fare	Terminal 1	B	29131	Enplaned	29131	2005	July
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2	10830	2005	July
3	200507	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 1	B	35156	Deplaned	35156	2005	July
4	200507	Air Canada	AC	Air Canada	AC	International	Canada	Enplaned	Other	Terminal 1	B	34090	Enplaned	34090	2005	July
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
15002	201603	Virgin America	VX	Virgin America	VX	Domestic	US	Enplaned	Low Fare	Terminal 2	D	194636	Enplaned	194636	2016	March
15003	201603	Virgin America	VX	Virgin America	VX	International	Mexico	Deplaned	Low Fare	International	A	4189	Deplaned	4189	2016	March
15004	201603	Virgin America	VX	Virgin America	VX	International	Mexico	Enplaned	Low Fare	Terminal 2	D	4693	Enplaned	4693	2016	March
15005	201603	Virgin Atlantic	VS	Virgin Atlantic	VS	International	Europe	Deplaned	Other	International	A	12313	Deplaned	12313	2016	March
15006	201603	Virgin Atlantic	VS	Virgin Atlantic	VS	International	Europe	Enplaned	Other	International	A	10898	Enplaned	10898	2016	March

15007 rows × 16 columns

```
display(df_clean.info())
```

```
<class 'dask.dataframe.core.DataFrame'>
Columns: 16 entries, Activity Period to Month
dtypes: object(12), int64(4)

None
```

```
resumen= df_clean.describe()
display(resumen.compute())
```

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

Veo que solo hay 4 columnas con valores númericos, asique transformaré el mayor número posible de columnas categóricas en númericas para poder trabajar

Para realizar el proceso de codificación podría usar tanto la técnica de Label Encoding o One-Hot Encoding, ya que ambas son usadas comunmente en el procesamiento de datos antes de realizar análisis descriptivos como el que tengo que realizar

Para este caso en el que tengo que calcular la media y la desviación estandar de cada elemento del conjunto de datos voy a usar Label Encoding. Este asignará valores numéricos únicos a cada categoria, y así hará mas facil el calculo de la media y la desviación estándar.

```python
display(df_clean.compute())

moda = df_clean.mode().compute()
display(moda)
```

Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Deplaned	Low Fare	Terminal 1	B	27271	Deplaned	27271	2005	July
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Enplaned	Low Fare	Terminal 1	B	29131	Enplaned	29131	2005	July
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	Domestic	US	Thru / Transit	Low Fare	Terminal 1	B	5415	Thru / Transit * 2	10830	2005	July
3	200507	Air Canada	AC	Air Canada	AC	International	Canada	Deplaned	Other	Terminal 1	B	35156	Deplaned	35156	2005	July
4	200507	Air Canada	AC	Air Canada	AC	International	Canada	Enplaned	Other	Terminal 1	B	34090	Enplaned	34090	2005	July
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
15002	201603	Virgin America	VX	Virgin America	VX	Domestic	US	Enplaned	Low Fare	Terminal 2	D	194636	Enplaned	194636	2016	March
15003	201603	Virgin America	VX	Virgin America	VX	International	Mexico	Deplaned	Low Fare	International	A	4189	Deplaned	4189	2016	March
15004	201603	Virgin America	VX	Virgin America	VX	International	Mexico	Enplaned	Low Fare	Terminal 2	D	4693	Enplaned	4693	2016	March
15005	201603	Virgin Atlantic	VS	Virgin Atlantic	VS	International	Europe	Deplaned	Other	International	A	12313	Deplaned	12313	2016	March
15006	201603	Virgin Atlantic	VS	Virgin Atlantic	VS	International	Europe	Enplaned	Other	International	A	10898	Enplaned	10898	2016	March

15007 rows × 16 columns

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200807	United Airlines - Pre 07/01/2013	UA	United Airlines - Pre 07/01/2013	UA	International	US	Deplaned	Other	International	A	2	Deplaned	2	2015	August

```python
from dask_ml.preprocessing import LabelEncoder
columnas_categoricas = [ 'GEO Summary', 'GEO Region',
                        'Activity Type Code', 'Price Category Code', 'Terminal', 'Boarding Area',
                        'Adjusted Activity Type Code', 'Month']

mapeo_valores = {}

for columna in columnas_categoricas:
    le = LabelEncoder()
    le.fit(df_clean[columna].astype(str))
    df_clean[columna] = le.transform(df_clean[columna].astype(str))
    mapeo_valores[columna] = {valor: codificacion for valor, codificacion in zip(le.classes_, le.transform(le.classes_))}

display(df_clean.compute())

for columna, mapeo in mapeo_valores.items():
    print(f'Valores para la columna {columna}:')
    print(mapeo)
    print()
```

Python

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200507	ATA Airlines	TZ	ATA Airlines	TZ	0	8	0	0	2	1	27271	0	27271	2005	5
1	200507	ATA Airlines	TZ	ATA Airlines	TZ	0	8	1	0	2	1	29131	1	29131	2005	5
2	200507	ATA Airlines	TZ	ATA Airlines	TZ	0	8	2	0	2	1	5415	2	10830	2005	5
3	200507	Air Canada	AC	Air Canada	AC	1	2	0	1	2	1	35156	0	35156	2005	5
4	200507	Air Canada	AC	Air Canada	AC	1	2	1	1	2	1	34090	1	34090	2005	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
15002	201603	Virgin America	VX	Virgin America	VX	0	8	1	0	3	3	194636	1	194636	2016	7
15003	201603	Virgin America	VX	Virgin America	VX	1	5	0	0	0	0	4189	0	4189	2016	7

15007 rows × 16 columns

```
Valores para la columna GEO Summary:
{'Domestic': 0, 'International': 1}

Valores para la columna GEO Region:
{'Asia': 0, 'Australia / Oceania': 1, 'Canada': 2, 'Central America': 3, 'Europe': 4, 'Mexico': 5, 'Middle East': 6, 'South America': 7, 'US': 8}

Valores para la columna Activity Type Code:
{'Deplaned': 0, 'Enplaned': 1, 'Thru / Transit': 2}

Valores para la columna Price Category Code:
{'Low Fare': 0, 'Other': 1}

Valores para la columna Terminal:
{'International': 0, 'Other': 1, 'Terminal 1': 2, 'Terminal 2': 3, 'Terminal 3': 4}

Valores para la columna Boarding Area:
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'Other': 7}

Valores para la columna Adjusted Activity Type Code:
{'Deplaned': 0, 'Enplaned': 1, 'Thru / Transit * 2': 2}

Valores para la columna Month:
{'April': 0, 'August': 1, 'December': 2, 'February': 3, 'January': 4, 'July': 5, 'June': 6, 'March': 7, 'May': 8, 'November': 9, 'October': 10, 'September': 11}
```

Las columnas que contienen las areolíneas y sus codigos he decidido no codificarlas ya que contienen una gran cantidad de variables y no creo que sea necesario.

```python
#Ahora realizo la media y la desviación tipica con los datos codificados
resumen= df_clean.describe()
moda= df_clean.mode()
display(resumen.compute())
display(moda.compute())
```

Python

	Activity Period	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
count	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000	15007.000000
mean	201045.073366	0.613714	4.438995	0.590125	0.872060	1.089691	2.652829	29240.521090	0.590125	29331.917105	2010.385220	5.531352
std	313.336196	0.486914	3.240626	0.603748	0.334034	1.495813	2.544096	58319.509284	0.603748	58284.182219	3.137589	3.454595
min	200507.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	1.000000	2005.000000	0.000000
25%	200803.000000	0.000000	1.000000	0.000000	1.000000	0.000000	0.000000	5373.500000	0.000000	5495.500000	2008.000000	3.000000
50%	201011.000000	1.000000	4.000000	1.000000	1.000000	0.000000	2.000000	9210.000000	1.000000	9354.000000	2010.000000	5.000000
75%	201308.000000	1.000000	8.000000	1.000000	1.000000	2.000000	6.000000	21158.500000	1.000000	21182.000000	2013.000000	9.000000
max	201603.000000	1.000000	8.000000	2.000000	1.000000	4.000000	7.000000	659837.000000	2.000000	659837.000000	2016.000000	11.000000

	Activity Period	Operating Airline	Operating Airline IATA Code	Published Airline	Published Airline IATA Code	GEO Summary	GEO Region	Activity Type Code	Price Category Code	Terminal	Boarding Area	Passenger Count	Adjusted Activity Type Code	Adjusted Passenger Count	Year	Month
0	200807	United Airlines - Pre 07/01/2013	UA	United Airlines - Pre 07/01/2013	UA	1	8	0	1	0	0	2	0	2	2015	1

0 200807 70 65 61 59 1 8 0 1 0 0

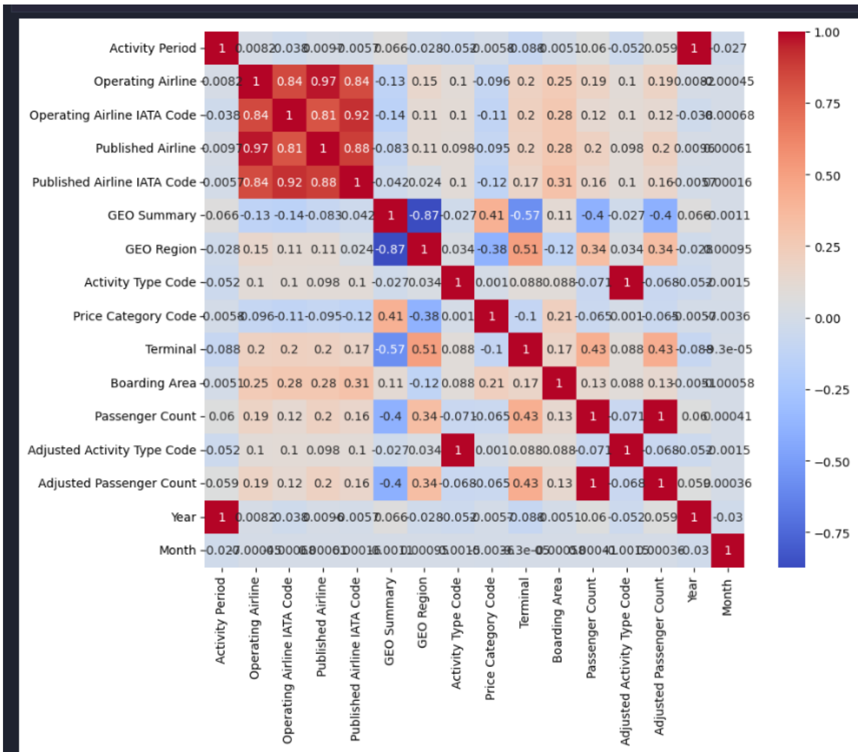
Interpretación de los datos

	Media	Desviación Estandar	Moda	Valor
Activity Period	201045.07	313.34	200807	
Operating Airline	43.74	23.67	United Airline-Pre 07/01/2013	
"IATA Code	42.79	21.79	UA	
Published Airline	37.619	21.17	United Airline-Pre 07/01/2013	
"IATA Code	38.54	20.35	UA	
GEO Summary	0.614	0.487	1	0:Domestic 1:International
GEO Region	4.439	3.24	8	8:US
Activity type code	0.59	0.604	0	0:Desembarcado 1:Embarcado 2:En transito
Price Category Code	0.879	0.334	1	0:Tarifa baja 1: Otro
Terminal	1.0897	1.49	0	0:Internacional
Boarding Area	2.65	2.54	0	0:A
Passenger count	29240.52	58319.51	2	
Adjusted Activity type code	0.5901	0.604	0	0:Desembarcado 1:Embarcado 2:En transito
Adjusted passenger count	29331.92	58284.18	2	
Year	2010.38	3.137	2015	
Month	5.53	3.45	1	1:Agosto

Matriz de correlación

```
matriz_correlacion = df_clean.corr().compute()

plt.figure(figsize=(10,8))
sns.heatmap(matriz_correlacion, annot=True, cmap='coolwarm')
plt.show()
```



Algoritmo

En esta parte del trabajo voy a seleccionar un algoritmo de programación paralela y distribuida visto en clase para aplicar a mi dataset. Tengo que basar mi elección en lo aprendido y las características del dataset.

En función a como es mi dataset y las conclusiones que he sacado de trabajar con el, creo que la mejor opción es usar Apache Spark como framework de procesamiento paralelo y distribuido. Es una buena opción gracias a la capacidad que tiene para procesar grandes volúmenes de datos de manera escalable. Esto significa que puede servir bien con el dataset de gran dimension que tenemos.

Tambien porque proporciona una amplia gama de bibliotecas útiles para aplicar algoritmos y análisis específicos al dataset y porque se integra con python a travez de la biblioteca PySpark.

```
import dask.dataframe as dd
from pyspark.sql import SparkSession
```

[2] ✓ 1.7s Python