

Jorge González Piedra

PROYECTO FINAL

BIG DATA



Jorge González Piedra

RESPONSABLE DEL DESARROLLO DE LA APLICACIÓN

**Fecha:** 05, FEB, 2023

**DESARROLLO**

1. Un informe científico, en el que se transmitan los resultados de los análisis realizados. Aquí explicaremos paso a paso cada uno de los apartados con las conclusiones correspondientes de las tareas realizadas. Podremos incluir secciones de código si es necesario y por supuesto, los resultados de cada una de las tareas realizadas sobre los datos obtenidos a través de la ejecución del código contenido en el documento técnico.
2. Un documento técnico que tendrá el código fuente (PySpark) empleado para la resolución de cada una de las tareas. El código fuente debe ser insertado como imágenes y con un tamaño que permita leer el texto contenido en las imágenes.
3. Una presentación guardada en formato pdf. Esta presentación nos servirá para mostrar los resultados de cada una de las tareas y no contendrá código fuente, sino que mostrará los resultados obtenidos siguiendo las guías de presentación que hemos visto en el módulo de proyectos *big data* y *storytelling*.
4. **Fuente de datos.**

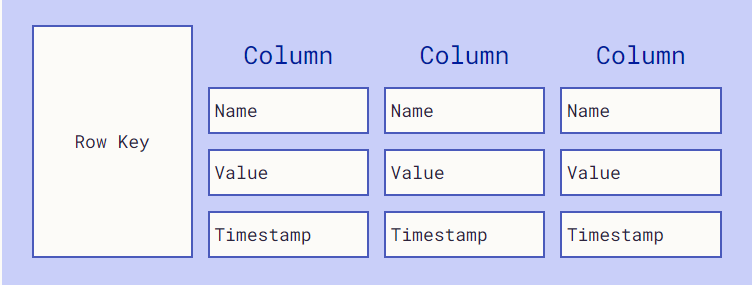
Analizamos la fuente de datos, **Air\_Traffic\_Passenger\_Statistics.csv**.

|  |  |
| --- | --- |
| Nombre del campo | Tipo de dato |
| Activity Period | integer (nullable = true) |
| Operating Airline | string (nullable = true) |
| Operating Airline IATA Code | string (nullable = true) |
| Published Airline | string (nullable = true) |
| Published Airline IATA Code | string (nullable = true) |
| GEO Summary | string (nullable = true) |
| GEO Region | string (nullable = true) |
| Activity Type Code | string (nullable = true) |
| Price Category Code | string (nullable = true) |
| Terminal | string (nullable = true) |
| Boarding Area | string (nullable = true) |
| Passenger Count | integer (nullable = true) |
| Adjusted Activity Type Code | string (nullable = true) |
| Adjusted Passenger Count | integer (nullable = true) |
| Year | integer (nullable = true) |
| Month | string (nullable = true) |

1. **Almacenamiento de datos.**

Para el almacenamiento de los datos utilizaremos la base de datos distribuida NoSQL de tipo columnar **Cassandra**.

Las bases de datos orientadas a familias de columnas son similares a las bases de datos relacionales. La principal diferencia radica en que, a diferencia de las relacionales, que almacenan los datos en filas, en este tipo de bases de datos la información se almacena en columnas.



Mientras una base de datos relacional está optimizada para almacenar filas de datos, normalmente para aplicaciones transaccionales, una base de datos en columnas está optimizada para lograr una recuperación rápida de columnas de datos, normalmente en aplicaciones analíticas.

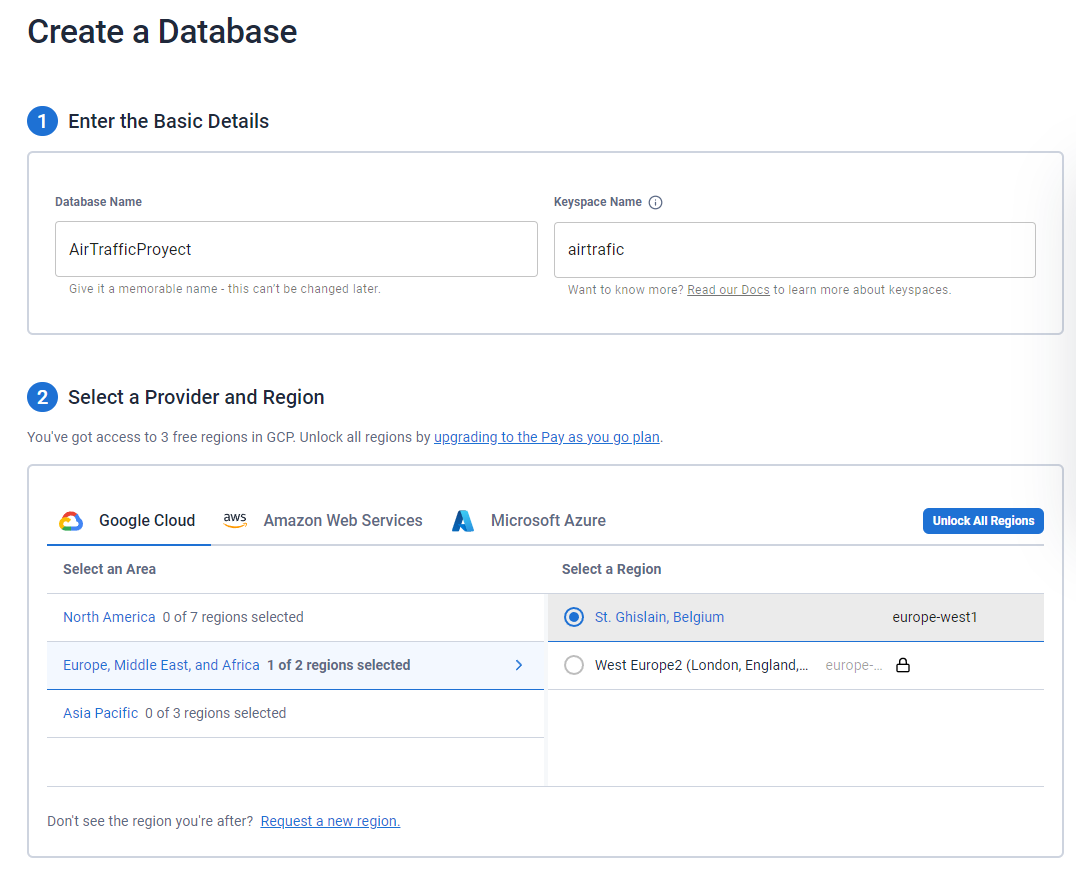
De la misma forma que otras bases de datos NoSQL, las bases de datos columnares están diseñadas para reducir la escala utilizando clústeres distribuidos

* 1. **Creación de la base de datos.**

Para trabajar con Cassandra utilizaremos DataStax.

DataStax es una infraestructura de datos escalable que permite manejar cualquier carga de trabajo en cualquier Cloud. Nos permite crear bases de datos de Apache Cassandra y conectarlas con aplicaciones externas mediante Apis o Drivers.

Se creará una nueva base de datos denominada **AirTrafficProyect**. Esta base de datos utilizará **GoogleCloud** para el almacenamiento de datos en la nube.



* 1. **Creación de tablas.**

Crearemos las tablas necesarias a partir de la fuente de datos. En nuestro caso crearemos la tabla airtraffic:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabla | Nombre del campo | Tipo de dato | Campos Primary Key | Campos Clustering Key |
| airtraffic | activity\_period | integer (nullable = true) | Activity Period | Operating Airline |
| operating\_airline | string (nullable = true) |
| operating\_airline\_IATA\_code | string (nullable = true) |
| published\_airline | string (nullable = true) |
| published\_airline\_IATA\_code | string (nullable = true) |
| GEO\_summary | string (nullable = true) |
| GEO\_region | string (nullable = true) |
| activity\_type\_code | string (nullable = true) |
| price\_category\_code | string (nullable = true) |
| terminal | string (nullable = true) |
| boarding\_area | string (nullable = true) |
| passenger\_count | integer (nullable = true) |
| adjusted\_activity\_type\_code | string (nullable = true) |
| adjusted\_passenger\_count | integer (nullable = true) |
| year | integer (nullable = true) |
| month | string (nullable = true) |

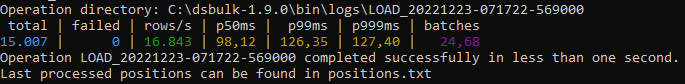
Script de creación de la tabla:



* 1. **Carga de datos.**

Utilizaremos DataStax Bulk Loader que nos permitirá, a través del comando **dsbulk**, cargar todos los datos del CSV a nuestra base de datos creada en DataStax. *(La instalación de DSBULK se encuentra explicada en el ANEXO 1)*





* 1. **Análisis de datos en Cassandra.**

Para comprobar la correcta inserción de los datos llevaremos a cabo las siguientes consultas:

* **Recuperar todos los registros de la aerolínea “Air China”.**



Utilizaremos DataStax Bulk para la extracción de los registros:





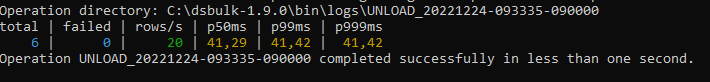
*Los datos recuperados se encuentran en el ANEXO 2*

* **Recuperar todos los vuelos de la compañía “Air Berlín” embarcados por la puerta “G”.**



Utilizaremos DataStax Bulk para la extracción de los registros:





Los datos recuperados se encuentran en el ANEXO 3

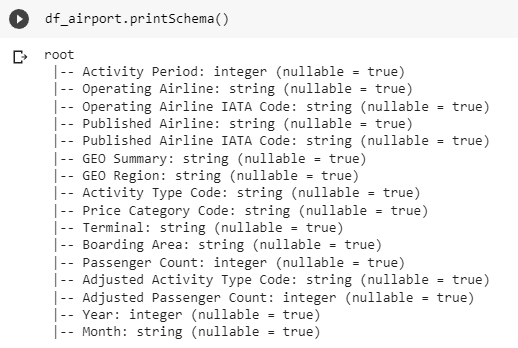
1. **Análisis preliminar de datos.**

Crearemos un Notebook llamado AirTraffic\_Jorge\_Gonzalez\_Piedra que se adjuntará a la entrega.

* 1. **Cargar datos en DataFrame.**

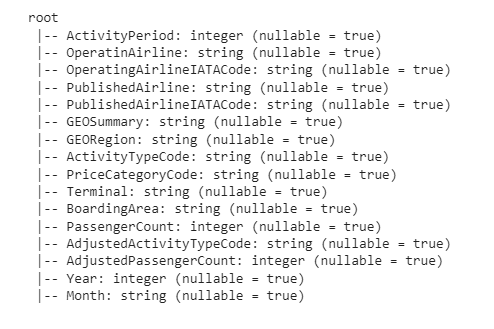
Crearemos un DataFrame donde se cargarán los datos del csv *Air\_Traffic\_Passenger\_Statistics.csv*





Para evitar problemas renombramos las columnas eliminando los espacios en blanco:





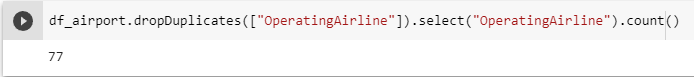
* 1. **Análisis de datos.**

Analizaremos los datos del CSV respondiendo las siguientes preguntas:

* **¿Cuántas compañías diferentes aparecen en el fichero?**

Existen 77 compañías diferentes.





Los datos recuperados se encuentran en el ANEXO 4

* **¿Cuántos pasajeros tienen de media los vuelos de cada compañía?**

Mostraremos la media de los campos PassengerCount y AdjustedPassengerCount:



Los datos recuperados se encuentran en el ANEXO 5.

* 1. **Eliminación de registros duplicados por el campo GEO Region.**

Eliminaremos los registros duplicados por el campo GEORegion manteniendo solo el registro con mayor número de pasajeros (campo PassengerCount).

Para ello crearemos una vista temporal sobre el DataFrame y mediante una consulta SQL agruparemos por GEORegion recuperando el registro cuyo valor de PassengerCount sea más alto:



* 1. **Volcar nuevos datos a CSV.**

Se volcarán los datos del punto anterior a un CSV llamado *airtraffic\_drop\_duplicates\_georegion.csv*. Se adjuntará dicho CSV a la entrega*:*



*Los datos extraídos se encuentran en el ANEXO 6.*

1. **Análisis estadístico**
   1. **Análisis descriptivo**

Estudiaremos las variables descriptivas de cada uno de los datos (media, moda, desviación estándar, mínimo y máximo).

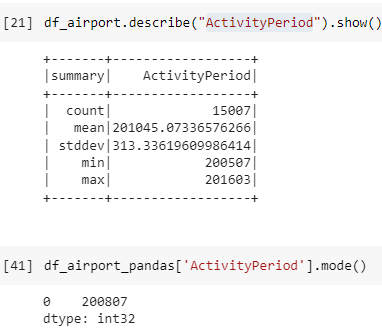
Para los datos categóricos estudiaremos solo la moda.

Para obtener las variables descriptivas utilizaremos la función **describe()** de pyspark, para la obtención de la moda, convertiremos el DataFrame a un DataFrame de pandas y utilizaremos la función **mode().**

* + 1. **Activity period**

Dato categórico que se refiere a un periodo de tiempo mes-año. Por ejemplo, el periodo de actividad 200507 se refiere a Julio de 2005, el 200508 a Agosto de 2005 y así sucesivamente. El valor mínimo, es decir, el primer periodo estudiado es 200507 (Julio 2005) y el valor máximo, el último periodo estudiado es 201603 (Marzo 2016).

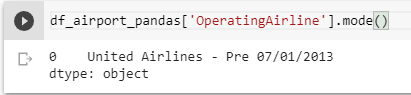
A pesar de ser un dato categórico, utilizaremos el comando **describe()**, pero solo nos fijaremos en los valores máximo y mínimo para entender los periodos que estudiaremos.



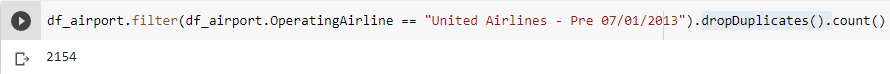
Observamos que el valor más repetido es 200807. Teniendo en cuenta que cada registro de nuestro DataFrame representa un vuelo, podemos afirmar que en este periodo que se corresponde con Julio de 2008, fue el periodo con más vuelos en nuestro aeropuerto.

* + 1. **Operating airline**

Dato que representa el nombre de la aerolínea que realiza cada vuelo.



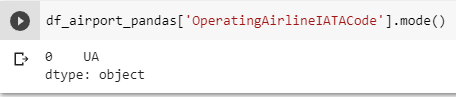
Observamos que el valor más repetido, es decir, la compañía que más vuelo realiza es United Airlines – Pre 07/01/2013.



Observamos que la compañía ha realizado 2154 vuelos.

* + 1. **Operating airline IATA code**

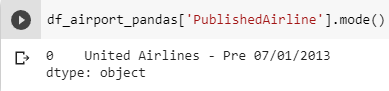
Dato que representa el código de la compañía en IATA (International Air Tranport Association).



El valor más repetido es código asociado a la compañía aérea *United Airlines – Pre 07/01/2013* que hemos visto en el punto anterior.

* + 1. **Published airline**

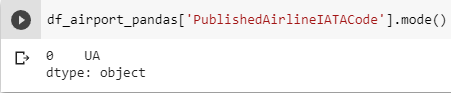
Dato que representa el código de la compañía en IATA (International Air Tranport Association).



La más repetida, de nuevo, es la compañía *United Airlines – Pre 07/01/2013*

* + 1. **Published airline IATA code**

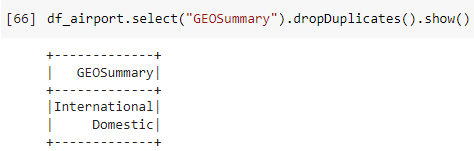
Dato que representa el código de la compañía publicado en IATA (International Air Tranport Association).

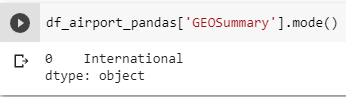


El más repetido, de nuevo, es la asociada con la compañía United Airlines – Pre 07/01/2013.

* + 1. **GEO summary**

Dato que representa el tipo de vuelo. Los valores que puede tomar son International o Domestic:





La moda indica los vuelos suelen ser en mayor parte Internacionales.

* + 1. **GEO region**

Dato que representa la región de destino de nuestro vuelo.



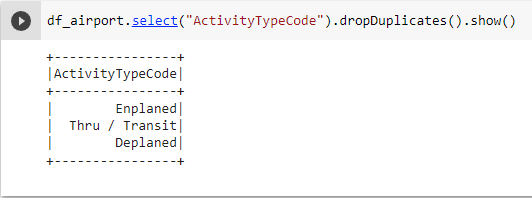
En nuestros datos aparecen 9 regiones posibles:

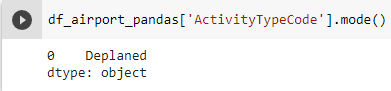


El valor más repetido es US, es decir, la mayoría de los vuelos que pasan por nuestro aeropuerto tienen como región de destino final Estados Unidos.

* + 1. **Activity type code**

Dato que representa el tipo de actividad del vuelo. Los valores posibles son, *Enplaned* (Planificado), *Thru/Transit* (En tránsito), *Deplaned* (Desplanificado).

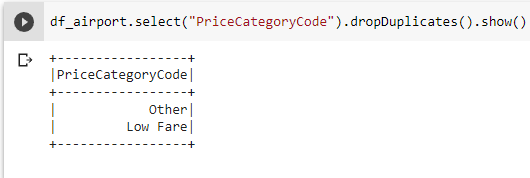


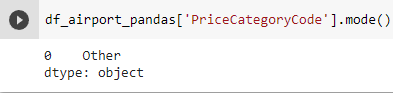


Observamos que el valor más repetido es *Deplaned*, es decir la mayoría de vuelos que estudiaremos serán vuelos desplanificados.

* + 1. **Price category code**

Dato que representa el tipo de tarifa del vuelo. Los posibles valores son *Low Fare* y *Other*. Suponemos que *Other* son tarifas medias y altas y *Low Fare* tarifas bajas.

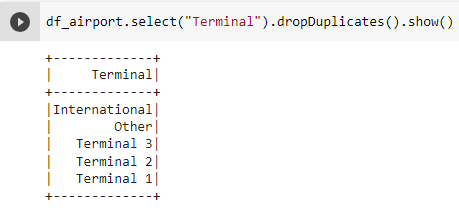


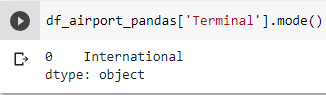


Vemos que el valor más repetido es *Other*. Por lo que la mayor parte de los vuelos, podemos suponer que son tarifas medias y altas.

* + 1. **Terminal**

Dato que representa la terminal del vuelo. Los valores posibles son:

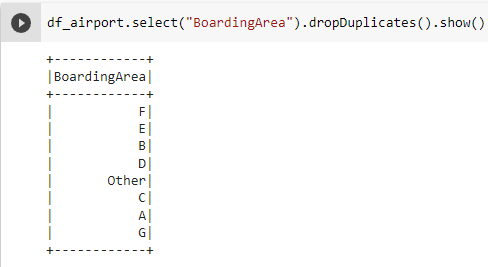


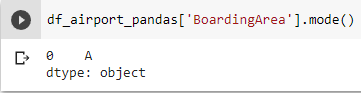


Observamos que el valor más repetido es International. Este dato se corresponde con GEO Summary donde el valor más repetido era también International.

* + 1. **Boarding area**

Dato que representa la terminal del embarque del vuelo. Los valores posibles son:

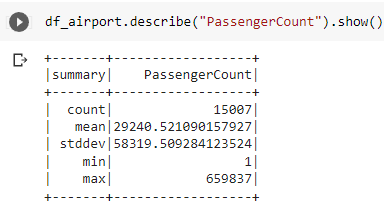




Observamos que la mayoría de vuelos embarcan por la zona A.

* + 1. **Passenger count**

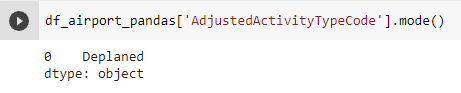
Dato que representa el número de pasajeros de cada vuelo.



Observamos que, de media, cada vuelo lleva a 29240.52 pasajeros.

* + 1. **Adjusted activity type code**

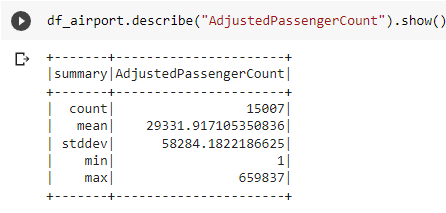
Dato que representa el tipo de actividad, ajustado, suponemos, para los datos nulos o vacíos. De ahora en adelante, si fuera necesario, se utilizará este campo, y no Activity type code, para los cálculos.



Observamos el dato más repetido es *Desplanificado* (Deplaned), coincidiendo con la moda del campo Activity Type Code.

* + 1. **Adjusted passenger count**

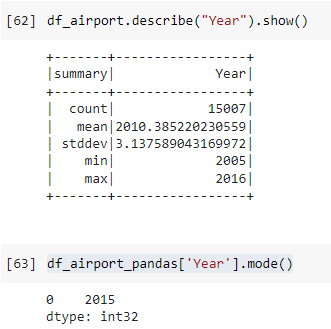
Dato que representa el número de pasajeros, ajustado, suponemos, para los datos nulos o vacíos. De ahora en adelante, si fuera necesario, se utilizará este campo y no Passenger count para los cálculos.



Observamos que la media es muy cercana a la media del dato Passenger count.

* + 1. **Year**

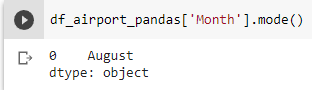
Dato que representa el año del vuelo.



Observamos que el año en el que más vuelos se realizaron fue 2015.

* + 1. **Month**

Dato que representa el mes del vuelo.



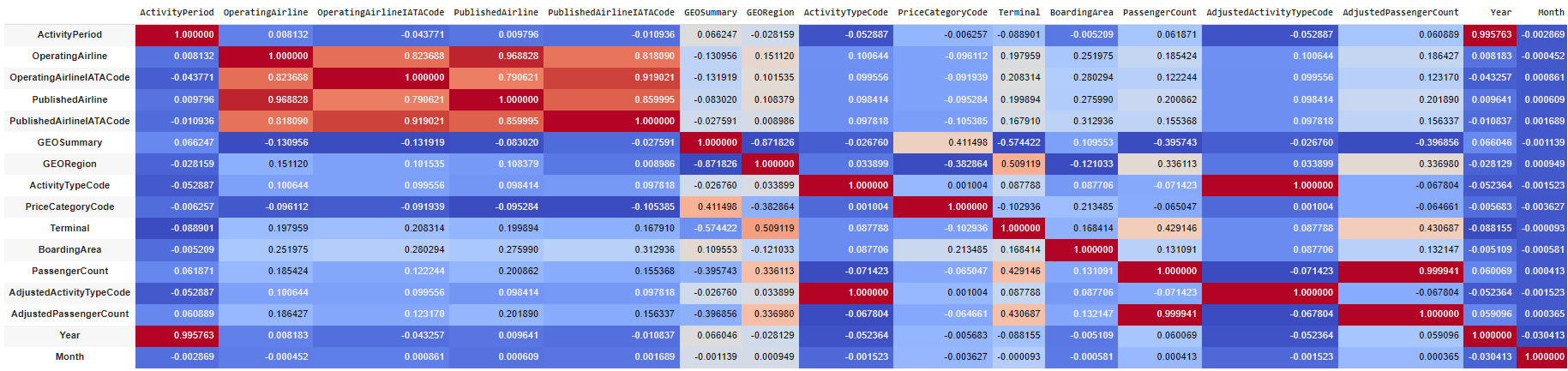
Observamos que el mes en el que más vuelos se realizan es Agosto. Esto puede ser debido al comienzo de las vacaciones de verano.

* 1. **Análisis de correlación**

Realizaremos un análisis de correlación de las variables de nuestro DataFrame. Para ello, convertiremos nuestro DataFrame a uno de Pandas y utilizaremos la función **corr()**, lo que nos devolverá una matriz de correlación de todas las columnas.



Para una mejor visualización de esta matriz, añadiremos un estilo que nos permita diferenciar por colores aquellas correlaciones más fuertes:



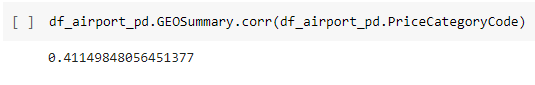
*Esta matriz se encuentra adjunta a la entrega en la carpeta airtraffic\_correlation\_matrix.*

Es importante recalcar que, aunque hemos indexado las variables categóricas y hemos sacado una correlación que estudiaremos a continuación (ya que es técnicamente posible) no tiene por qué haber necesariamente una relación directa entre variables categóricas, aunque la matriz la muestre.

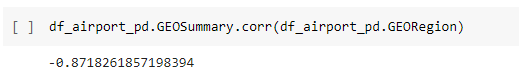
***Nota****: Por ejemplo, en la correlación OperatingAirline – GEOSummary podemos indexar ambas columnas de manera que para Operating Airline la compañía “Air Canada” sea 1 y “Virgin America” sea 2 y para GEOSummary el valor “Domestic” sea 1 e “International” 2. Si Opertaing Airline aumenta al mismo tiempo que GeoSummary esto no tiene que significar realmente que haya una correlación porque ambos son datos categóricos. En nuestro caso el valor 2 representa a “Virgin America”, no es un dato numérico que represente un incremento del valor de la variable, simplemente una categoría distinta.*

Mostramos a continuación las correlaciones más fuertes de esta matriz:

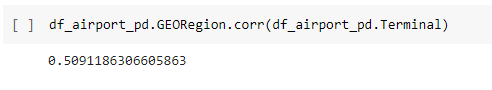
* + 1. **GEO Summary – Price Category Code**



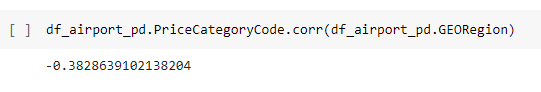
* + 1. **GEO Summary – GEO Region**



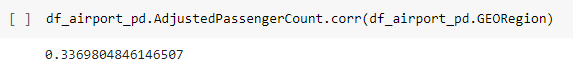
* + 1. **GEO Region – Terminal**



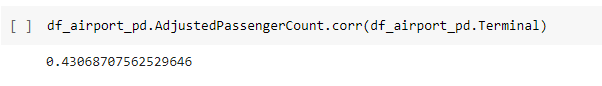
* + 1. **GEO Region – Price Category Code**



* + 1. **Adjusted Passenger Count – GEO Region**

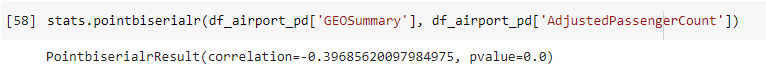


* + 1. **Adjusted Passenger Count – Terminal**

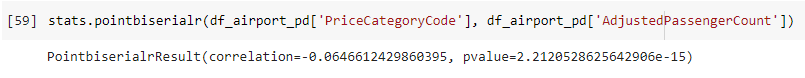


Para estudiar de manera más precisa la correlación entre las variables dicotómicas (aquellas variables categóricas que solo pueden tomar dos posibles valores) GEO Summary *(Domestic, International)* y Price category code *(Low fare, Other)* con el número de pasajeros, utilizaremos el **método** **Point-Biserial.**

* + 1. **GEO Summary– Adjusted passenger count**



* + 1. **Price category code – Adjusted passenger count**

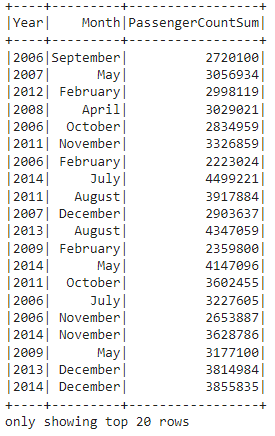


* 1. **Regresión lineal**

Realizaremos una regresión lineal estudiando la cantidad de pasajeros que pasan por nuestro aeropuerto a lo largo del tiempo para intentar predecir la cantidad de pasajeros que tendremos en los próximos años.

Comenzaremos creando un nuevo DataFrame para estudiar el número de pasajeros que hemos tenido a lo largo de los años. Este nuevo DataFrame contendrá las columnas *Year* y *Month*. Además, tendremos la columna PassengerCountSum que será el sumatorio de todos los pasajeros de todos los vuelos del año-mes:

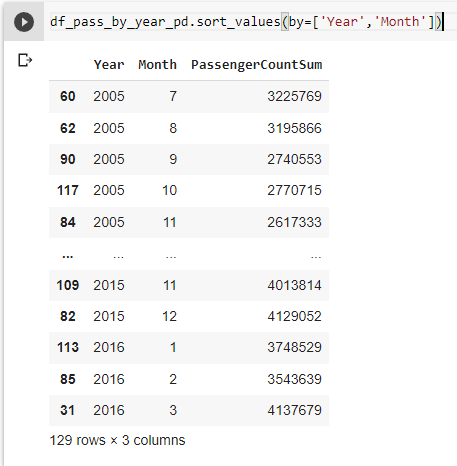




Convertiremos los valores de la columna *Month* a tipo numérico, de manera que January pase a ser 1, February 2 y así sucesivamente. Para ello convertiremos nuestro nuevo DataFrame a uno de Pandas y a través de **map()** y del módulo calendar realizaremos la conversión:

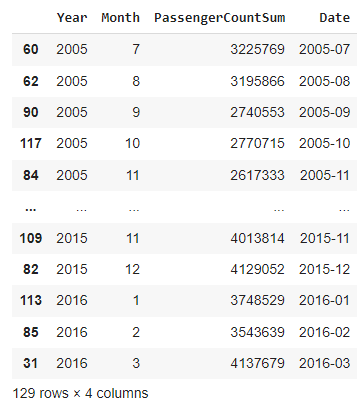


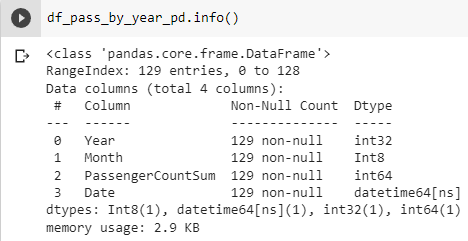




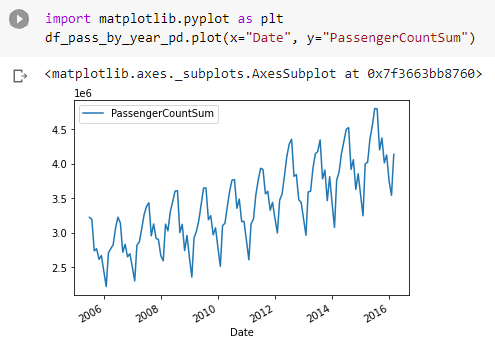
Crearemos una nueva columna para unir Year y Month en un campo de tipo fecha:



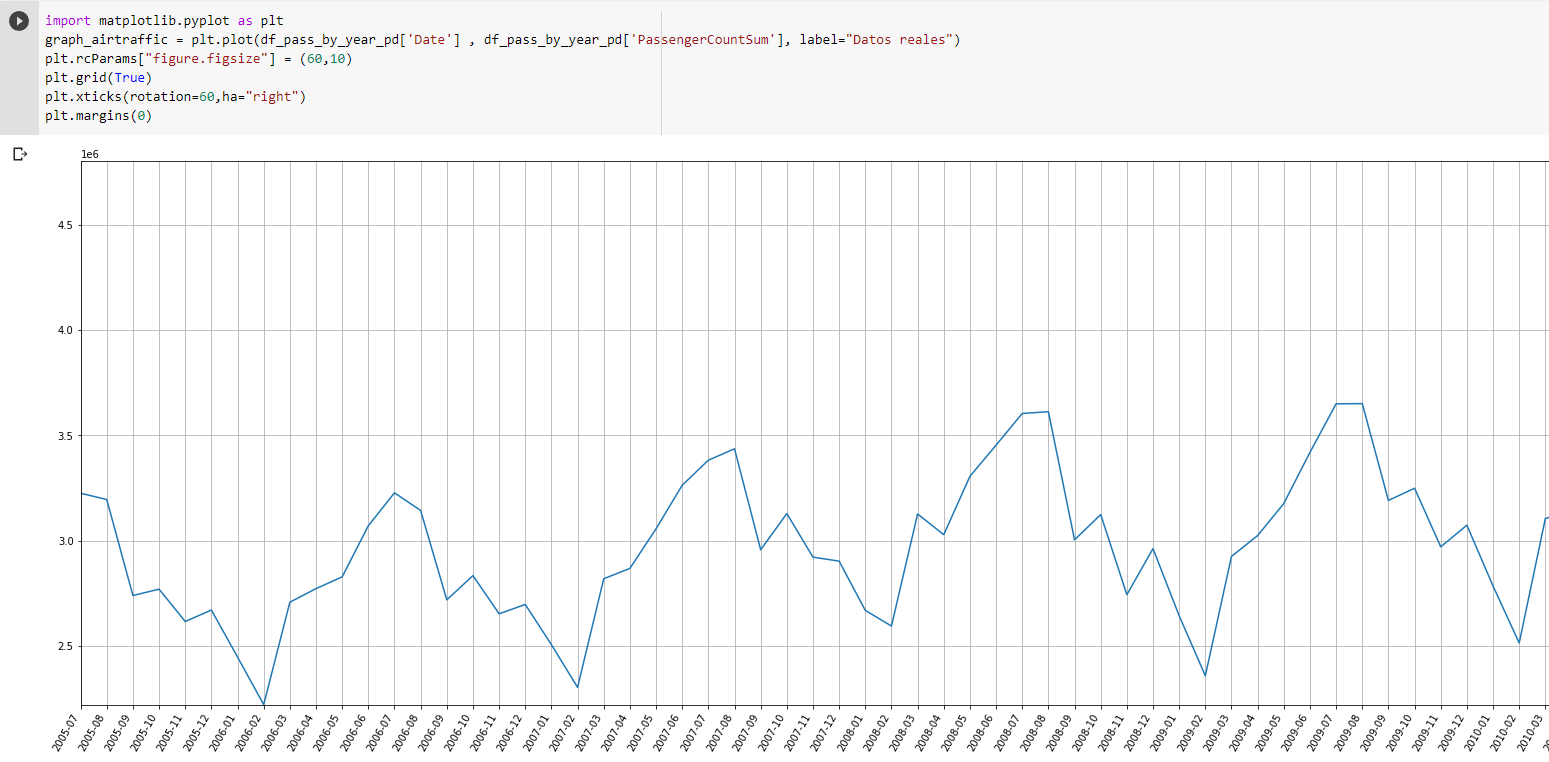




Utilizando la librería matplotlib dibujaremos estos datos en un gráfico para entender un poco mejor los datos:



Observamos un patrón recurrente. El número de pasajeros desciende, y asciende de manera periódica. Para estudiar esto un poco más en detalle hacemos zoom en el gráfico y añadimos un grid:



Observamos que el mes con menos afluencia de pasajeros suele ser Febrero. Los meses con más afluencia de pasajeros son **Julio-Agosto**, coincidiendo con el verano y las vacaciones. A partir de Agosto comienza un descenso del número de pasajeros hasta volver a llegar a **Febrero** que comienza a ascender de nuevo.

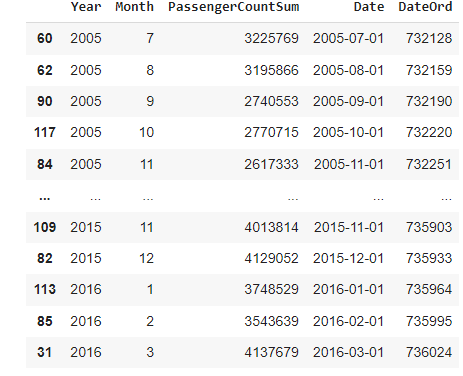
Una vez visto este patrón y estudiado los datos vamos a crear la regresión.

Comenzamos importando el módulo de linearModel de skLearn:

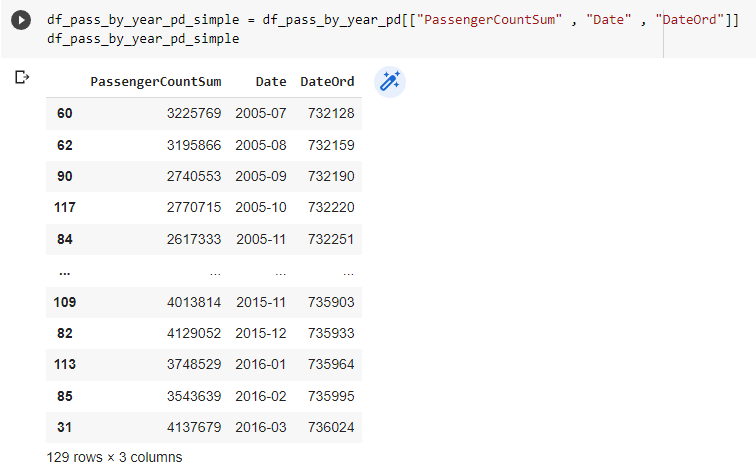


Para realizar la regresión linear no podemos utilizar el campo de tipo fecha por lo que convertiremos el campo Date a un ordinal para seguir manteniendo el orden de los datos:

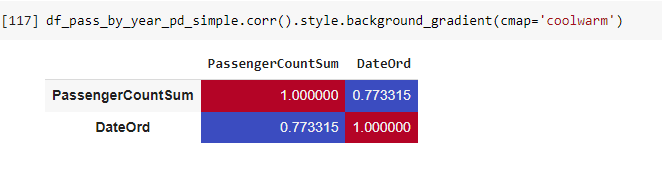




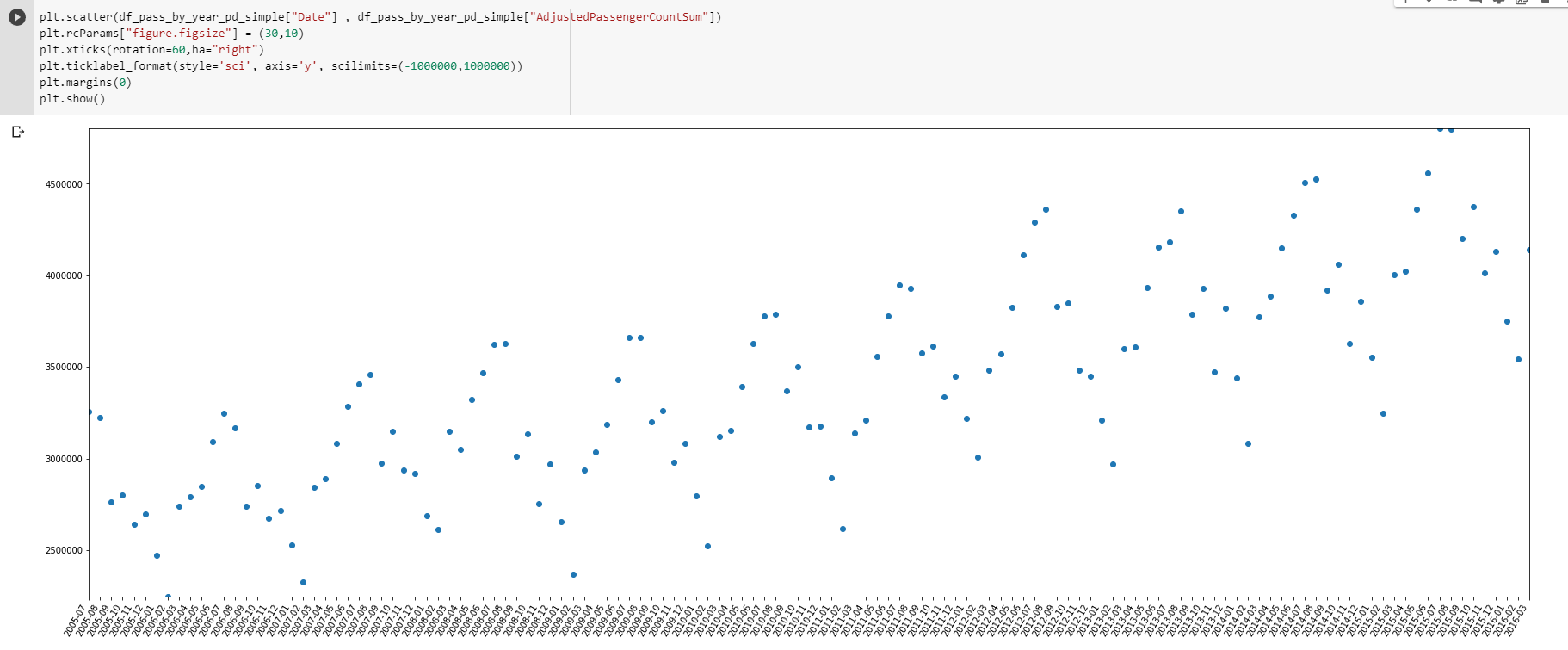
Simplificamos el DataFrame para quedarnos con los datos que realmente estudiaremos, PassengerCountSum, Date y DateOrd:



Observamos la correlación entre las variables de nuestro nuevo DataFrame:



Vemos que existe una correlación positiva fuerte entre el número de pasajeros y el valor ordinal que representa nuestra fecha. Visualizamos esto con un gráfico de dispersión:



Creamos nuestro modelo de regresión lineal utilizando el módulo sklearn:



Creamos dos variables. Una para nuestro eje X y otra para nuestro eje Y:



Ambas variables son arrays de 2 dimensiones que utilizaremos para crear la regresión

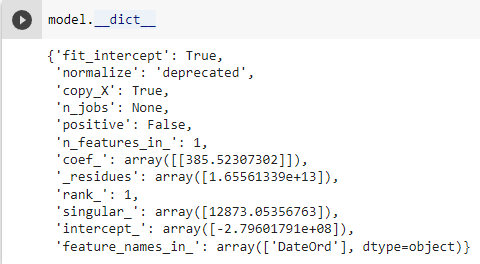
**Explicativas** contiene los ordinales correspondientes a nuestras fechas, es decir, nuestra variable independiente.

**Objetivo** contiene aquello que nosotros queremos predecir, el número de pasajeros.

Utilizando el modelo creado anteriormente, creamos la regresión con la función **fit():**



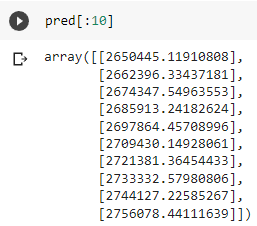
Haciendo uso de \_\_dict\_\_ comprobamos los atributos de nuestro modelo:



Pasamos a realizar la predicción, para lo cual utilizamos el método **predict()** pasándole el valor de nuestras X, es decir nuestras fechas:

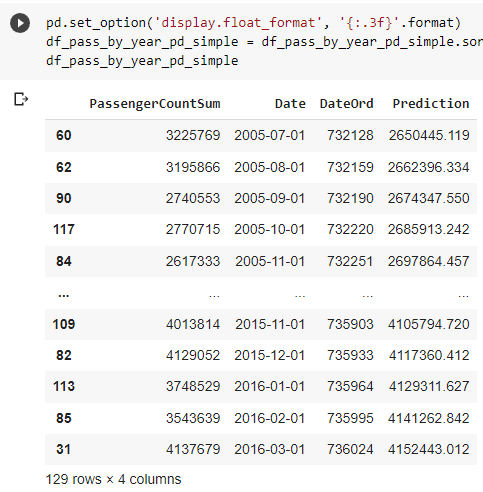


Esto nos devuelve un array con los valores de Y, es decir, nuestro número de pasajeros:

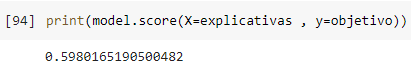


Añadimos estos datos a nuestro DataFrame y lo ordenamos en función del campo DateOrd:

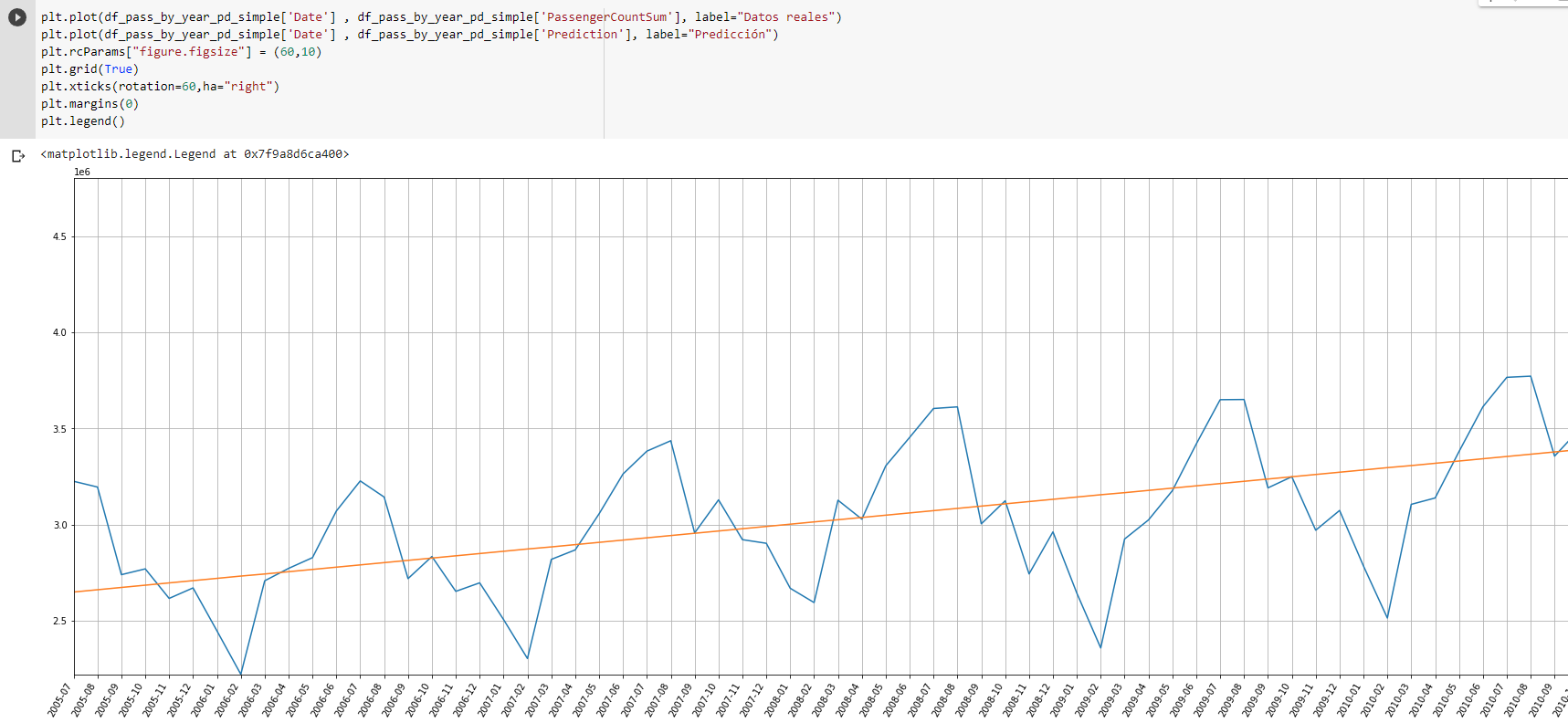




Comprobaremos la precisión nuestro modelo con la función **score():**

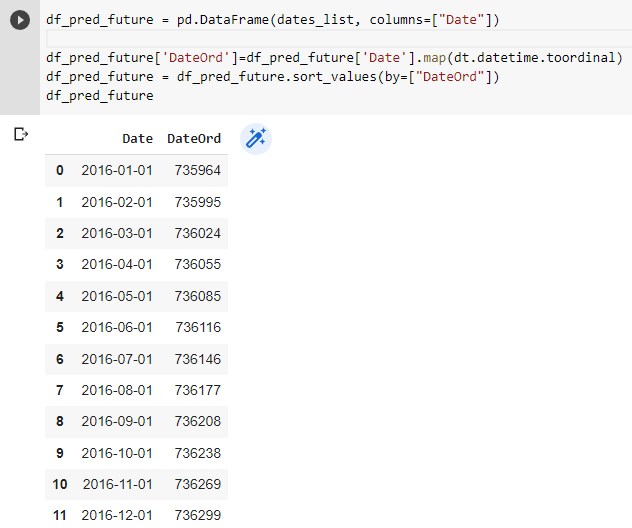


Pasamos nuestras fechas de nuevo a formato AAAA-MM y visualizamos nuestra recta de regresión:



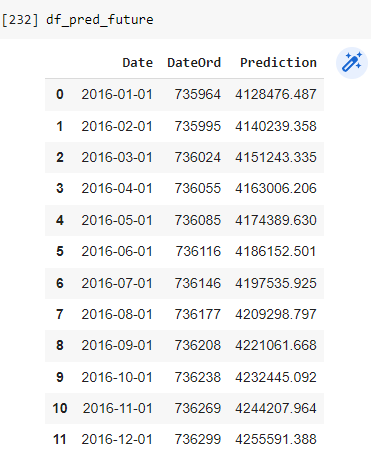
Por último, generamos una lista de fechas a futuro para comprobar las predicciones de nuestro modelo. Estas fechas serán las correspondientes a todo el año 2016:

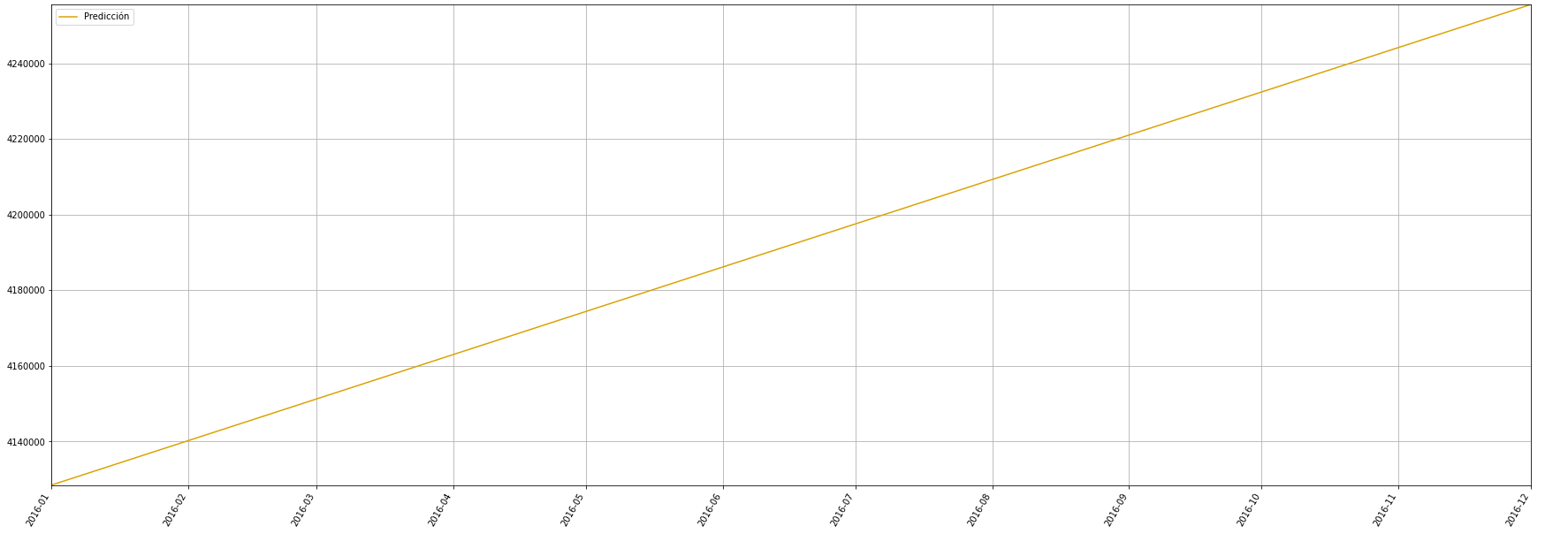




Con estas fechas generamos la predicción con el modelo ya creado:





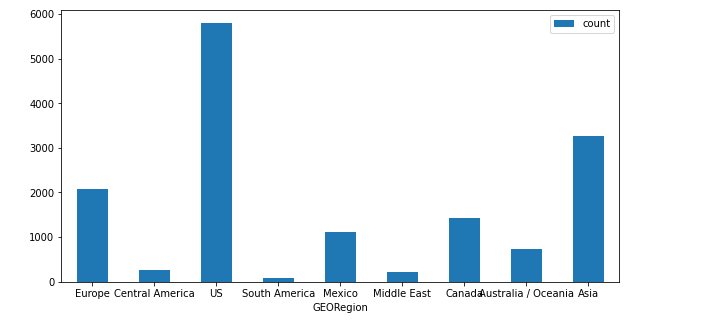


Observamos como la predicción nos indica que el número de pasajeros crecerá de manera lineal durante 2016.

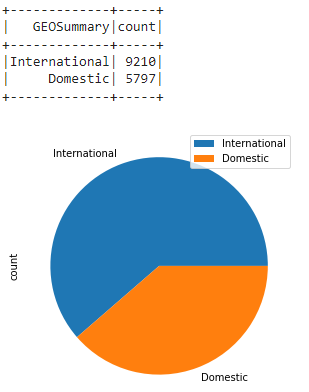
**CONCLUSIONES**

Realizando un análisis descriptivo de los datos hemos visto cómo podemos obtener una visión global de la información que estamos analizando y como, aunque parezca un análisis simple, podemos extraer conclusiones muy valiosas

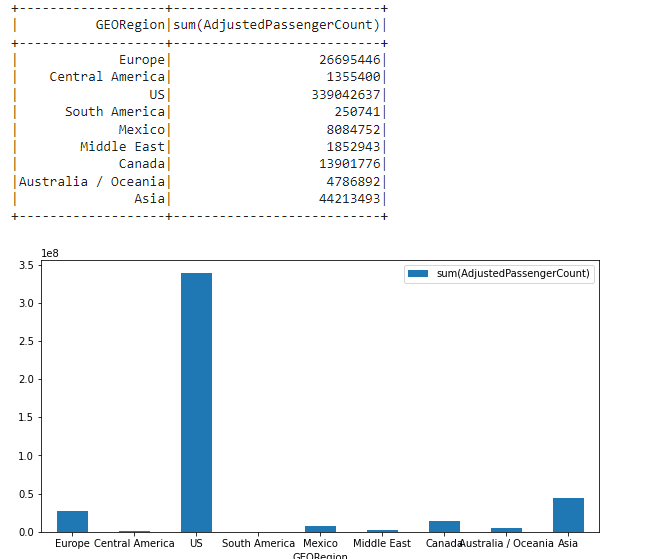
Se puede observar, por ejemplo, que, de todas las regiones, Estados Unidos es la que ha recibido mayor número de vuelos:



Sin embargo, vemos que la mayoría de vuelos son internacionales, es decir, hacia fuera de Estados Unidos.

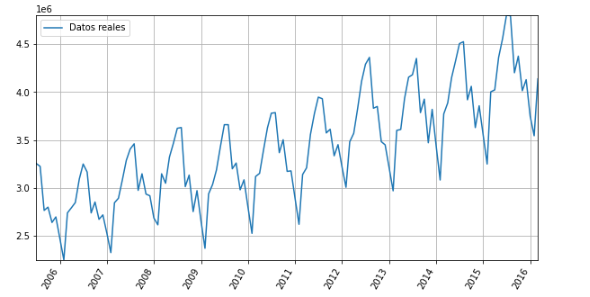


Aun con esto, Estados Unidos es la región con el mayor número de pasajeros, lo que quiere decir, que, aunque la mayoría de vuelos hayan sido internacionales (a otras regiones fueras de Estado Unidos), los vuelos domésticos transportaban a más pasajeros en total.

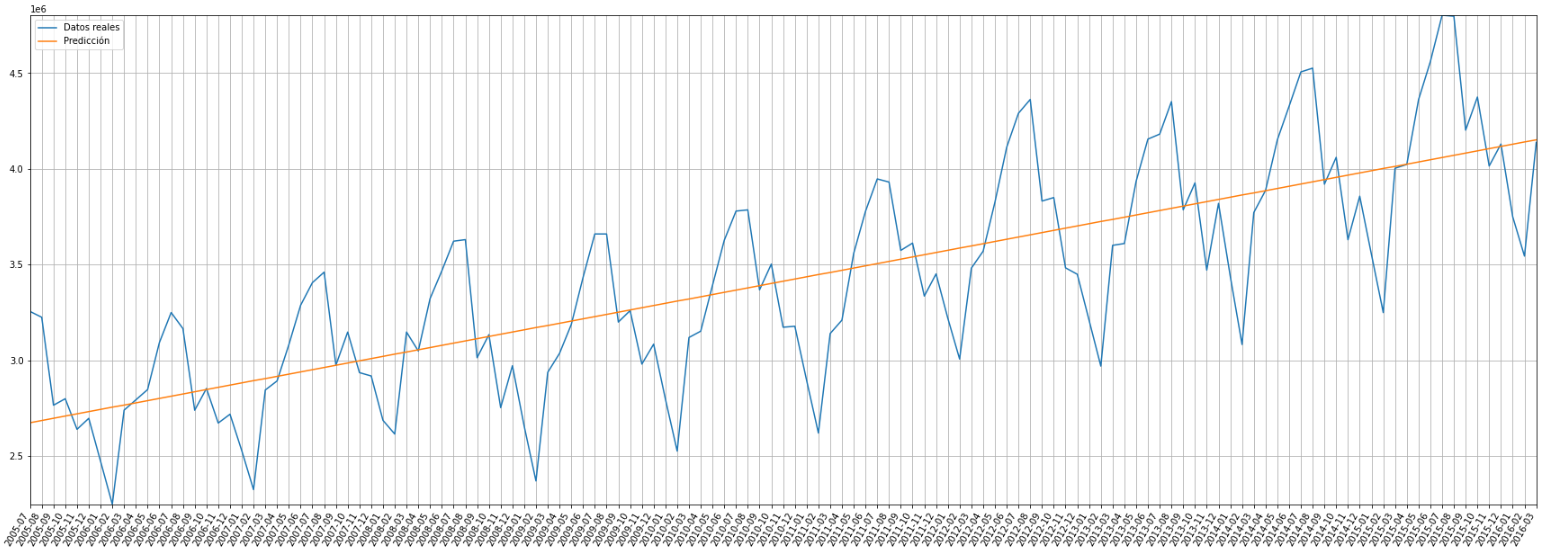


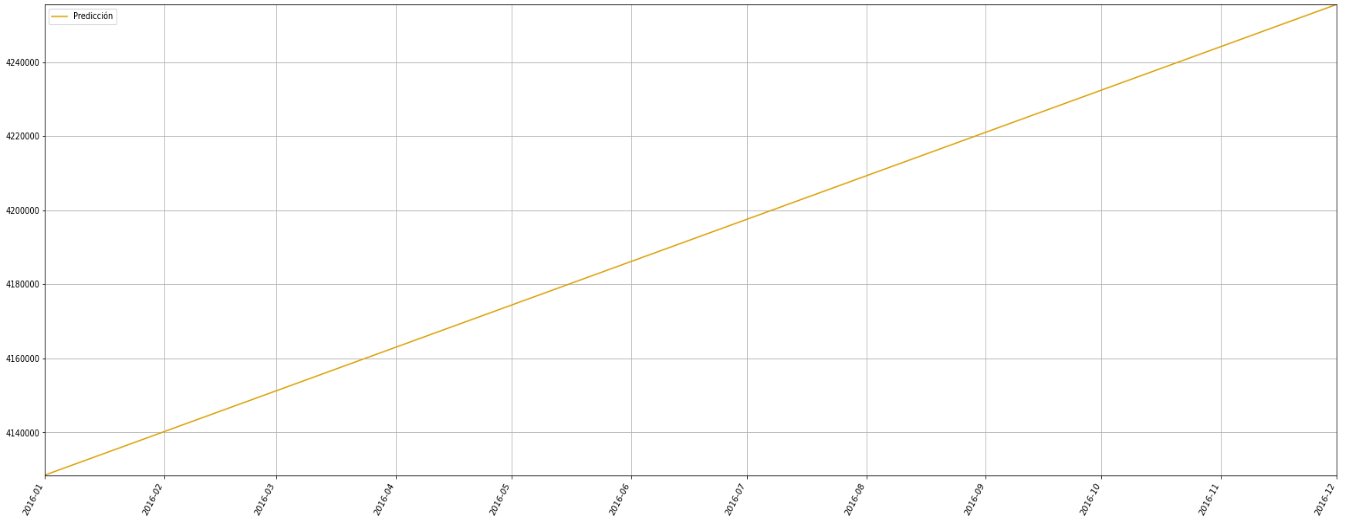
Con esto en cuenta (junto con otros factores) podríamos tomar decisiones, como destinar más recursos a realizar más vuelos domésticos, ya que hemos visto que son los que más pasajeros mueven o, podemos intentar destinar recursos a aumentar el interés de nuestros pasajeros por viajar al extranjero con tarifas más atractivas.

También, estudiado el número total de pasajeros de nuestro aeropuerto hemos observado como este número sigue un patrón, alcanzando su máximo y su mínimo en Agosto y Febrero de cada año respectivamente.



Para predecir como va a evolucionar este número se pueden utilizar diferentes algoritmos. En este estudio se ha realizado una regresión lineal y si solo nos fijásemos en ella, veríamos que este número va a ir en aumento de manera lineal.





Sin embargo, únicamente mirando la regresión lineal no podemos saber en qué momento el número de pasajeros va a descender o ascender bruscamente. Esto, aunque es imposible saberlo con total certeza, lo podemos suponer mirando los datos de años anteriores. De este modo, aunque la regresión lineal no muestre un descenso del número de pasajeros, nosotros podemos suponer, que, en este año, dicho número va a tener una tendencia ascendente hasta Julio/Agosto donde alcanzará un pico y a partir de ahí comenzará descender hasta el mes de Febrero como viene ocurriendo en los años anteriores. En función de esto podemos tomar decisiones.

Como conclusión podemos decir que un análisis descriptivo o una regresión lineal, por sí solos, no nos sirven para tomar decisiones o predecir comportamientos, es necesario analizar bien la información con varias técnicas en conjunto para tomar las mejores decisiones posibles basadas en los datos.

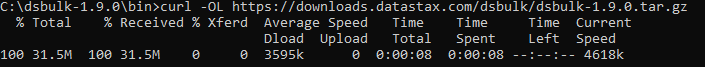
**ANEXOS**

* + - 1. **Instalación DataStax Bulk Loader.**

DataStax Bulk Loader (DSBulk) es una utilidad que permite la carga, descarga y conteo de datos de nuestra base de datos de manera rápida y eficaz desde la consola de comandos. Para su instalación se seguirán los siguientes pasos:

Desde la consola de comandos descargamos el archivo de instalación de DSBulk:





Una vez descargado el archivo de instalación lo descomprimimos:



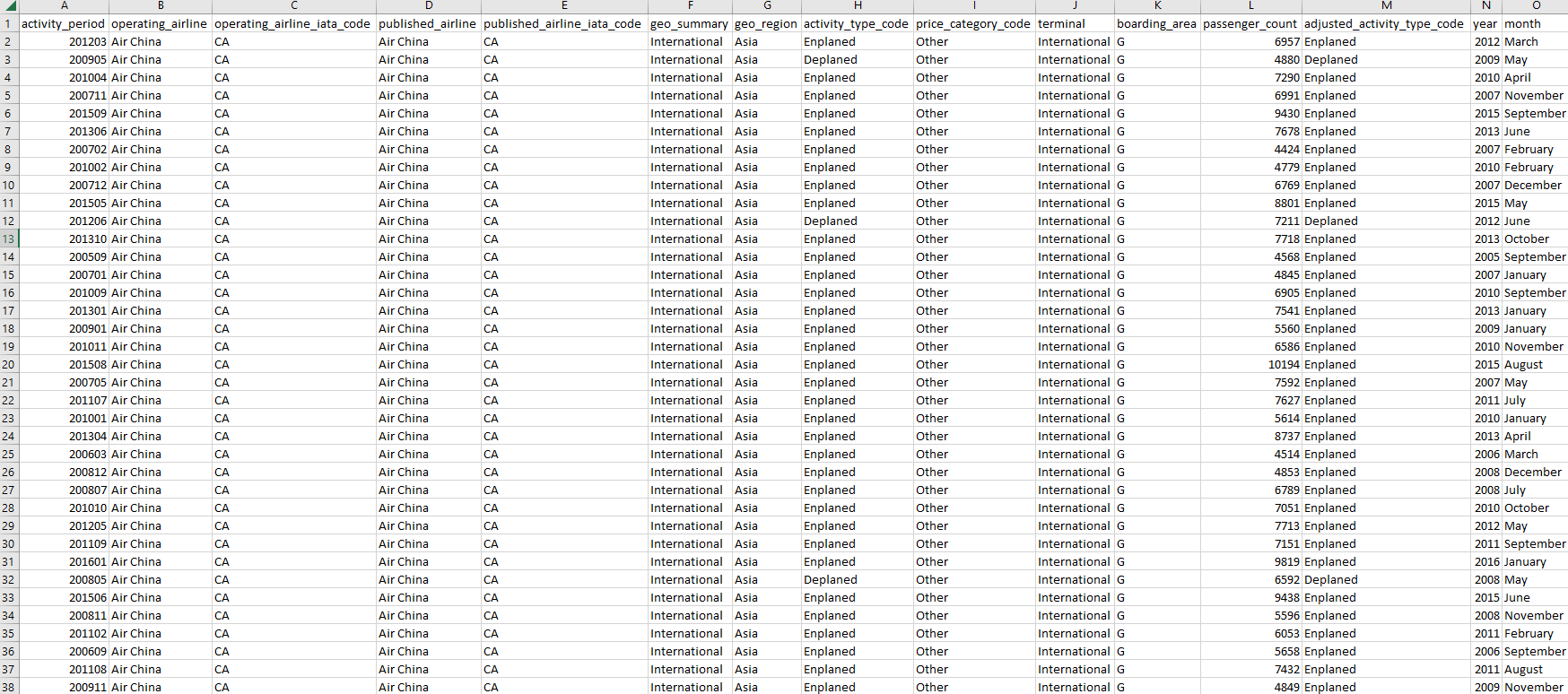
Accedemos a la carpeta bin y desde la consola de comandos comprobamos que todo se ha instalado correctamente comprobando la versión de DSBulk

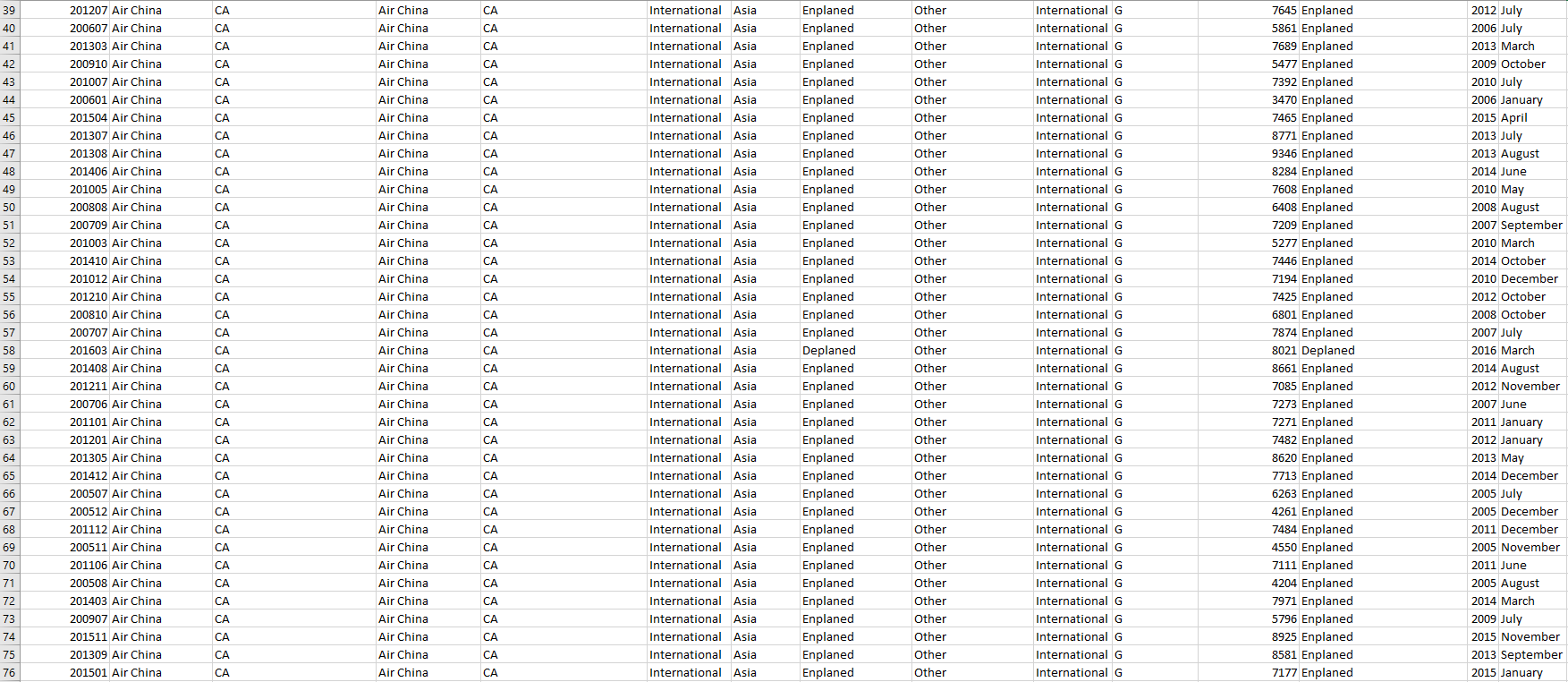


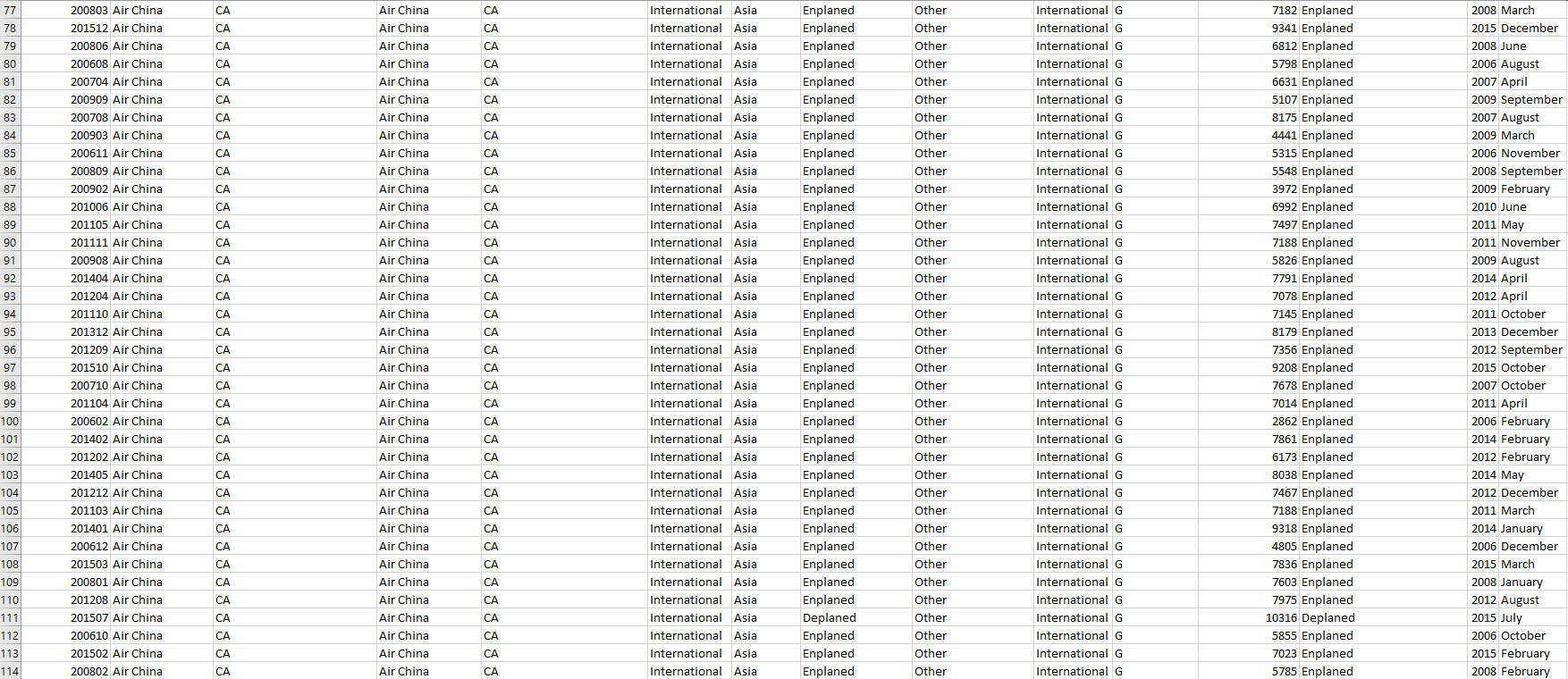


* + - 1. **Extracción de datos. Vuelos de la compañía Air China.**

Se muestra a continuación los datos de los vuelos de la compañía Air China. Se adjunta también a la entrega la carpeta *airchina\_data\_extraction* con el resultado de la extracción:



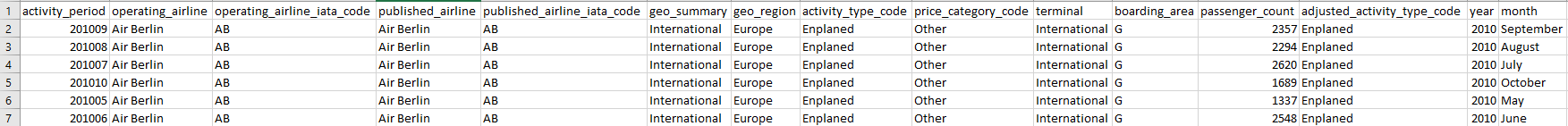






* + - 1. **Extracción de datos. Vuelos de la compañía Air Berlín.**

Se muestra a continuación los datos de los vuelos de la compañía Air Berlin. Se adjunta también a la entrega la carpeta *airberlin\_data\_extraction* con el resultado de la extracción:



* + - 1. **Extracción de datos. Número de compañías diferentes.**

Se muestra a continuación las diferentes compañías existentes. Se adjunta también a la entrega la carpeta *airport\_companies* con el resultado de la extracción:

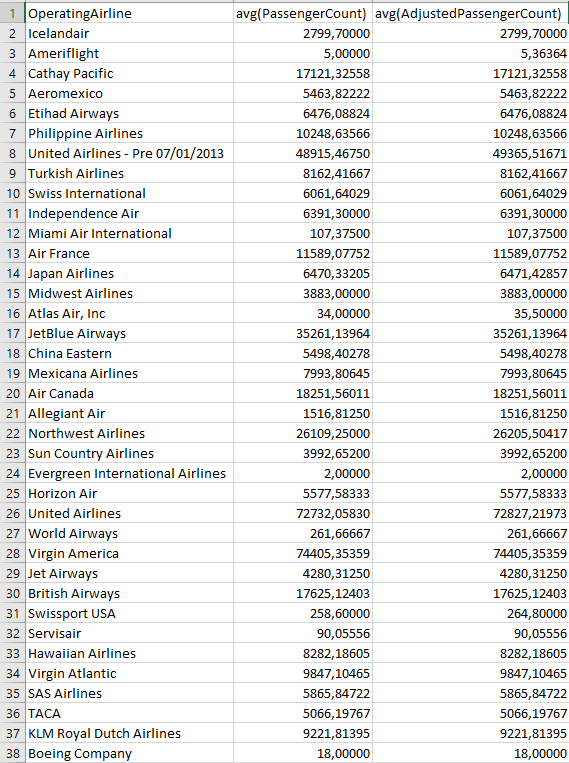


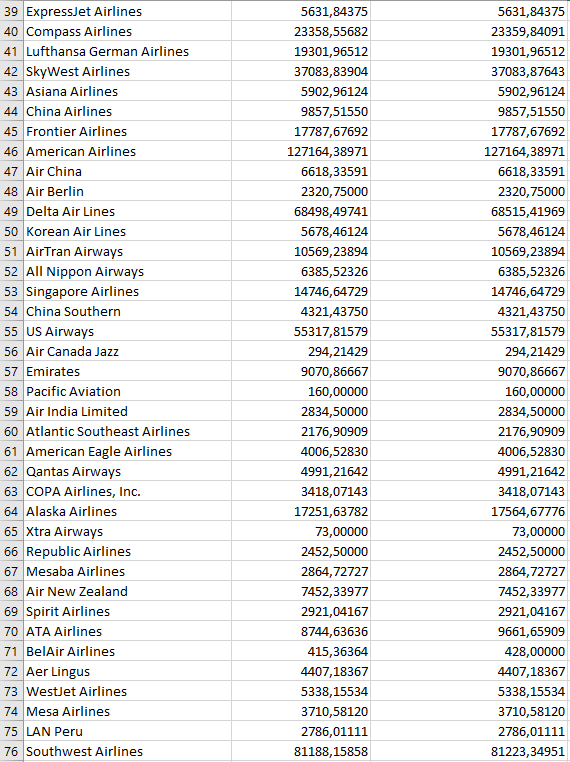




* + - 1. **Extracción de datos. Media de pasajeros por compañía.**

Se muestra a continuación la media de pasajeros por compañía. Se adjunta también a la entrega la carpeta *avg\_flights\_per\_company* con el resultado de la extracción:







* + - 1. **Extracción de datos. Eliminación de duplicados por GEORegion.**

Se muestra a continuación los datos correspondientes a eliminar las duplicidades por el campo GEORegion manteniendo solo aquellos registros con mayor número de pasajeros. Se adjunta también a la entrega la carpeta *airtraffic\_drop\_duplicates\_georegion* con el resultado de la extracción:

