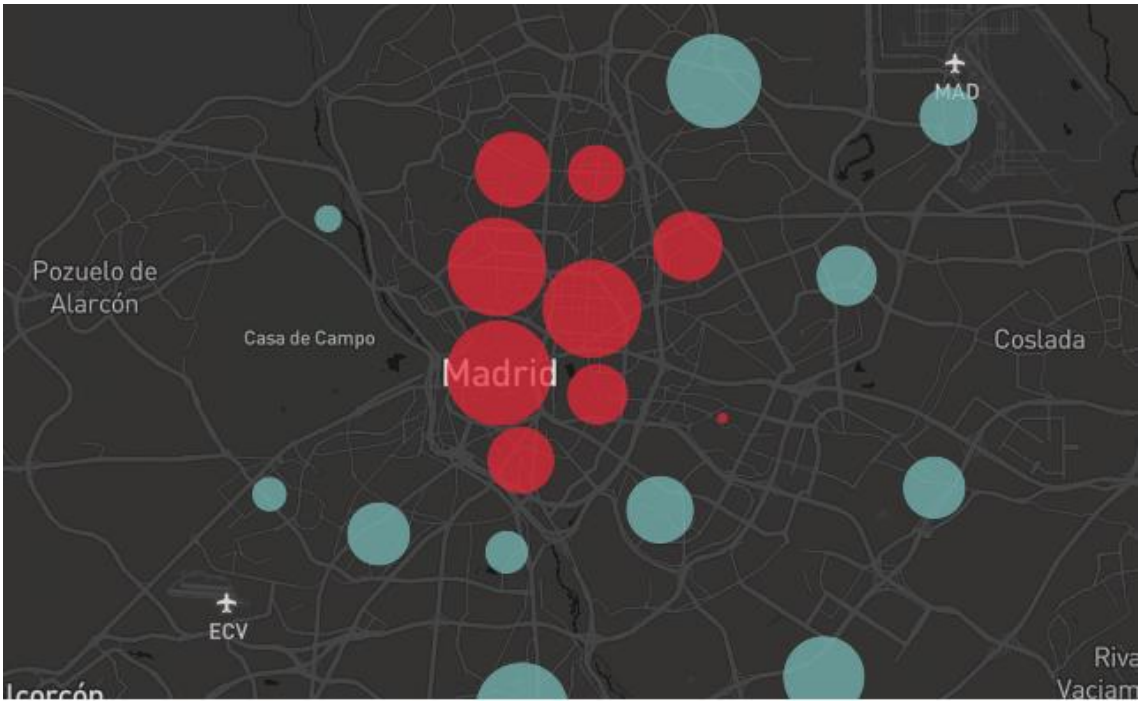
The Bridge

Bootcamp Data Science

Análisis exploratorio de datos

AirBNB y los efectos de los alojamientos turísticos en Madrid



Madrid a 3 de noviembre 2020

Kerstin Kleinert

Memoria

1. Obtención de datos:

Fuentes de datos:

Inside AirBNB (principal)

Idealista

Padrón Municipal (Madrid)

El dataset principal se obtuvo en la página de AirBNB mediante descarga de un csv. Lo mismo con diversos datasets del portal de datos abiertos del muncipio de Madrid.

Los datos del Idealista se obtuvieron de forma “scraping manual” (copiar – pegar), ya que el Idealista tiene su página protegida del scraping y el acceso a su API no incluye datos históricos, que son los que me interesaban.

2. Limpieza de datos

El dataset de AirBNB consistía de 20.226 entradas en 74 columnas. Después de la primera limpieza se quedó en 17 columnas, y después de borrar las entradas sin reseña (5.523 entradas) mediante ‘dropna()’, quedaron 14.971 entradas.

Se borraron las entradas sin reseña, para utilizar solamente alojamientos que realmente han sido activos en los últimos 10 años.

Para poder comparar los datos de todos los alojamientos, antes del ‘dropna’, se guardó un dataframe con todas las entradas y las fechas de alta del propietario por entrada.

3. Los dataframes

a.) El dataframe principal es el de AirBNB, con los alojamientos como entradas e index por ID de alojamiento. Para poder efectuar una analítica histórica, hay que definir una fecha como alta del alojamiento en la base de datos de AirBNB, ya que el propio dataset no trae este dato..

Tenemos 2 columnas de fechas:

1. Fecha de primera reseña

A tener en cuenta:

- un 26% de las entradas no tienen reseña

- La primera reseña de un apartamento no coincide con la fecha de alta

2. Fecha de alta de propietario

A tener en cuenta:

- un 22% de propietarios ofrecen 2 o más alojamientos. A partir del 2º alojamiento no tenemos fecha de alta. Serían 11.157 alojamientos sin fecha asignada

b) El segundo dataframe es el dataframe que combina los datos del Idealista, los datos del Padron de varios distritos de Madrid, el PIB de la Comunidad de Madrid con los datos de AirBNB del primer Dataframe, relativo a fechas y distritos.

Los datos del Idealista etc. tienen el intervalo de fechas por meses. Los datos de AirBNB son fechas de intervalo de día. Para poder unificar los dataframes, se transformaron todas las fechas a format datetime con intervalo de principio de mes (MS).

(Al principio lo hice de manera complicada con .dt.year, dt.month y .astype(str) + '/01/' + df["year"].astype(str), pero en la segunda conversión para otro dataframe lo hice mediante df_host.resample('YS')….)

c) El tercer dataframe es un histórico de saldo de Padrón entre los distritos municipales obtenida mediante descarga del portal de datos abiertos del municipio de Madrid. Son movimientos intramunicipales , quiere decir el saldo total en el municipio es 0, de los últimos 10 años.

d) El cuarto dataframe se creó mediante conexión con la API de OpenStreetMaps y la descarga de coordenadas de equipamientos y servicios para distritos elegidos y para Madrid central, la parte del municipo situado dentro de la M-30.

4. El desarrollo

a) Como anteriormente mencionado, el primer dataframe se redujo a 17 columnas y 14.971 filas.

Las entradas con NaNs resultantes de la columna “first_review” se borraron.

La columna “license” tb tenía muchísimos NaNs, pero en este caso se reemplazaron con un “no”, indicando que el alojamiento no cuenta con licencia.

El siguiente paso fue convertir la columna de “first_review” a datetime object, como base para la analítica histórica de datos.

Luego mediante “groupby”, agregaciones y la definición de máscaras, se crean las series y dataframes para los gráficos.

b) Por ejemplo se aíslan series de distritos del dataframe de AirBNB, para poder integrarles mediante “join” en el dataframe df_dis con la información económica y poblacional obtenida en el portal de datos abiertos de Madrid.

c) El dataframe de equipamiento y servicios (Notebook aparte) consiste de los resultados de 4 queries a la API de OpenStreetMaps, que se juntaron mediante “append” en un dataframe. Los límites para las queries se definieron con 2 puntos en

el mapa, formando un rectángulo. Habría sido más elegante utilizar las “subareas” de los distritos de Madrid definidos en el OpenStreetMap, pero después de investigar, he llegado a la conclusión que la API no permite definir las subareas como limites.

5. Los gráficos

Quería utilizar módulos distintos para la creación de los gráficos, Matplotlib, Plotly, Seaborn,… Los gráficos no llevan títulos en el Notebook, porque he preferido poner el título en el programa final de montaje para que sean iguales todos.

Por regla general los gráficos de líneas o barras se han creado con Matplotlib y Seaborn, y los mapas se han creado con Plotly Express Mapbox.

Como los dataframes tenían muchas dimensiones, he creado 2 mapas que además de las dimensiones de x, y, color y tamaño, incluyen la dimensión temporal en forma de animación.

Con Plotly Express solamente hace falta añadir una fila con la columna de secuencia temporal, y el resultado es un mapa animado.

EXTRA:

Adjunto tb un Notebook con una matrix de correlación que he utilizado para crear un gráfico tipo “Chord” para grafiar los movimientos de población entre los distritos.

El chord quedó muy bonito, pero desgraciadamente no se puede llegar a ninguna conclusión, porque el gráfico no deja claras las relaciones. Lección aprendida.

6. La presentación

Me habría gustado incrustar de alguna manera los gráficos interactivos en la presentación, pero después de varios intentos he llegado a la conclusión de que en un proyecto real, la manera más elegante sería montar una página web.

Por eso he elegido el formato vertical en la presentación, que es lo que más se parece a una página web, aunque sin su interactividad.

Los gráficos interactivos se presentarán en el navegador por separado.