Informe técnico housing_BCN

Indice

Introducción

Para este proyecto, se nos dio a analizar una base de datos extraída de la página de la empresa Fotocasas, una empresa dedicada al sector de la vivienda. Esta base de datos contenía la información de 8188 viviendas distintas. En la base de datos se mostraban distintos tipos de información, como el precio del alquiler de la vivienda, el número de habitaciones, el número de baños, si tenía ascensor y terraza, los metros cuadrados, el tipo de vivienda, el barrio en el que estaba situada y el precio del metro cuadrado. Con estos datos, se nos pidió hacer un análisis del mercado buscando información relevante dentro de la base de datos.

Para realizar esta búsqueda de datos, usamos la librería de Python 'Pandas', enfocada en el análisis de datos, en la plataforma de Google Colab o Visual Studio. Subimos los avances que realizamos a un repositorio de GitHub para el control de versiones.

Iniciamos el análisis limpiando los datos de la base de datos, ya que había muchos valores nulos o errores, como valores anormales (por ejemplo, más de 10 baños o precios exageradamente altos). Después de terminar la limpieza de datos, pasamos al análisis. En esta etapa, buscamos los datos más relevantes y realizamos comparaciones para representarlos en gráficos de forma clara y visual. Finalmente, creamos un mapa para visualizar algunos datos, generando un archivo GeoJSON con los parámetros de los barrios de Barcelona, que cruzamos con la base de datos mediante distintas librerías como geopandas, folium y folium.features de la librería de GeoJsonTooltip.

Inicio del proyecto

Antes de comenzar el análisis, llevamos a cabo una serie de pasos previos. Primero, nos creamos una cuenta en GitHub en caso de no contar con una, ya que subimos a esta plataforma todos los cambios y avances del análisis. Posteriormente, creamos un nuevo repositorio donde guardamos los datos. En este repositorio subimos la base de datos y los archivos utilizados para el análisis. Añadimos a los otros integrantes del equipo y a los profesores para que pudieran mantener un control sobre los datos y su distribución. Por último, realizamos un primer commit para iniciar el control de versiones y generar lo que se conoce como rama main. No realizamos cambios directamente en esta rama, ya que podría complicar la recuperación de versiones anteriores en caso de fallos. Por ello, creamos varias ramas, una para cada integrante, donde realizamos cambios antes de subirlos a la rama main.

Posteriormente, abrimos un nuevo documento en la plataforma elegida (Colab o Visual Studio) y clonamos el repositorio de GitHub para obtener en la plataforma los datos que se encuentren en el repositorio con el comando:

- ! git clone https://github.com/arantafall/housing_BCN.git

Importamos las distintas librerías necesarias para el análisis. Para esta parte de búsqueda de errores, utilizamos Pandas con el código:

- Import pandas as pd

Lo que hará que la librería panda sea importada a nuestro documento con el nombre de pd, para analizar la base de datos.

Y, por último, cargar la base de datos al archivo mediante el código

- data=pd.read_csv("./housing_BCN/data/Barcelona_Fotocasa_HousingPrices.csv")

Este código busca el archivo dentro del repositorio que importamos con el primer código

Preparación de los datos

Una vez hecha la preparación. viene el turno de la limpieza de los datos. Para esto realizamos un análisis preliminar y superficial de estos datos, para poder ver los valores que son incorrectos y que no deberían estar en esta base de datos, para esto podemos usar el código, que nos mostrara en que columnas hay que tipo de información como los valores no nulos y el tipo datos que contiene:

- Data.info()

Y este otro:

- data.isnull().sum()
- (data.isnull().sum() / len(data)) * 100

Este código nos muestra el porcentaje de casillas nulas que hay en cada columna. En esta base de datos, un 3,27 % aproximadamente de los datos de la categoría de tipo de vivienda. Al sr un porcentaje tan reducido, la mejor opción era eliminar las filas que contenían estos datos para así poder trabajar mejor con la base de datos. para el borrado de las filas que contienen valores nulos, podemos ver los tipos de valor que contiene con el código:

- print(data['real_state'].unique())

Lo que devuelve todos los valores únicos que contiene la columna, en este caso la columna property_type, ya que es la columna que contenía los valores nulos. Y nos devuelve esta información: ['flat' 'attic' nan 'apartment' 'study']. Lo que significa que además de los valores flat, attic, apartament y study, cuenta también con valores nan. Para eliminarlos de la base de datos usaremos este otro código:

- data = data.dropna(subset=['real_state'])
- data.reset_index(drop=True, inplace=True)

Este código buscara los valores nulos de esta columna y eliminara la fila que los contenga, después reseteara el índice, ya que si no habrá algunos índices que falten ya que habrán sido eliminados.

Una vez hecho esto, podemos ver los cambios si ponemos el codigo:

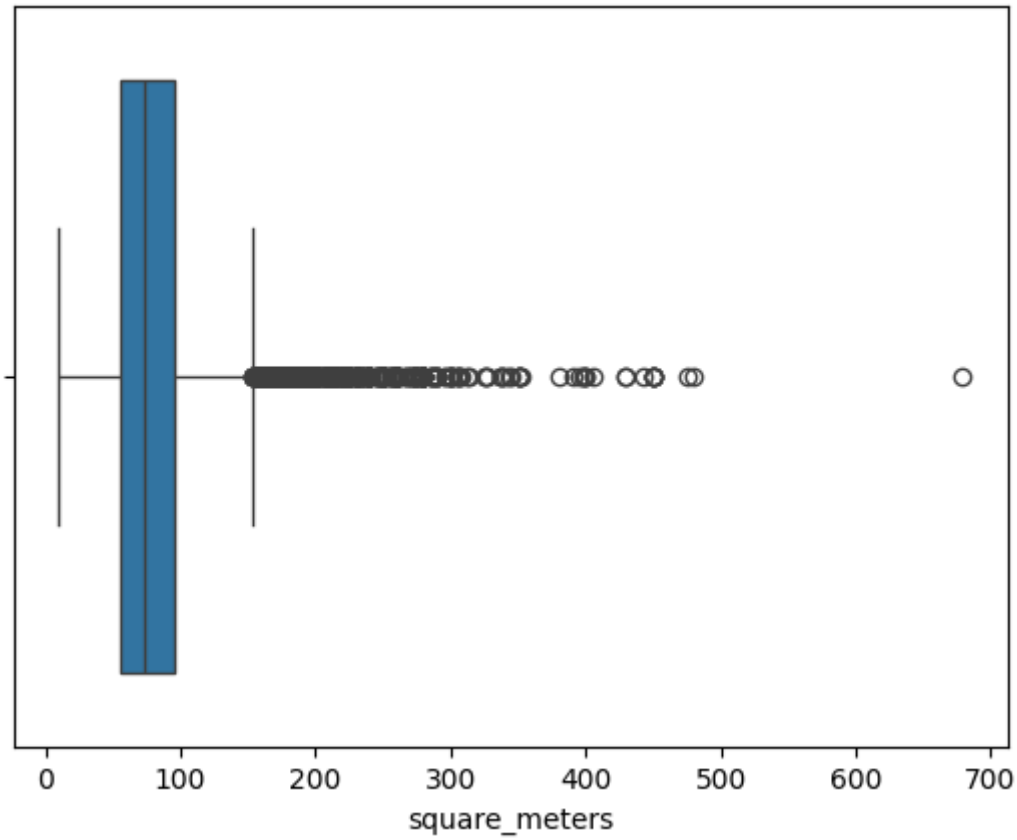
- Data.shape()

Con esto podremos ver como el número total de filas que contiene la base de datos disminuyó de 8188 a 7920 Y mediate el data.index() vemos como ya no quedan más valores nulos.

Después de esto fue el turno de examinar los valores anormales. para el tratamiento de valores aberrantes u *outliers*, decidimos visualizar de manera gráfica algunos datos para ver rápidamente en un vistazo si había algun valor que se alejaba de la media. La librería de *Seaborn* que utilizamos nosperitió esto mediante:

- sns.boxplot(x=data['square_meters'])

para ver todos los valores de 'square_meters' en una gráfica horizontal



Al ver que había un valor por encima de 600 metros cuadrados, muy alejado de los demás, decidimos visualizarlo en formato tabla mediante:

- data[data['square_meters'] > 600]

para encontrar todas las viviendas con mas de 600 metros cuadrados, solo había una y la eliminamos con:

- data.drop(data[data.square_meters > 600].index, inplace=True)
- data.reset_index(drop=True, inplace=True)

Repetimos lo mismo con otras variables numéricas como el precio o el precio por metro cuadrado

Para las variables categóricas, decidimos buscar viviendas con un número elevado de baños lo hicimos mediante:

- data[data['bathroom'] > 6]

lo que nos devolvió un resultado de solamente 3 viviendas con 7 u 8 baños, las cuales decidimos eliminar también con:

- data.drop(data[data['bathroom'] > 6].index, inplace=True)
- data.reset_index(drop=True, inplace=True)

Una vez terminada la limpieza, podemos hacer unos últimos cambios como son el cambio de nombre de la columna real_state, a property_type, ya que con este nuevo nombre se puede entender de forma más clara el tipo de Información que contiene la columna con el codigo:

- data.rename(columns={'real_state': 'property_type'}, inplace=True)

Finalmente, pasamos la base de datos limpia a un nuevo archivo csv con el nombre "base_dades_neta":

- data.to_csv('../data/base_dades_neta.csv', index=False)

Y cargaremos todos los cambios a una nueva data que se llama base_dades_neta.csv .

Análisis y creación de gráficos

Para el análisis, usamos la misma librería de pandas, ya que es la más útil en cuanto a analizar bases de datos. Y para generar los gráficos, podemos utilizar distintas herramientas, como los histogramas y otros gráficos que puede hacer Python o usar la herramienta Power BI que sirve para mostrar datos de forma visual mediante distintos tipos de entorno y crear representaciones gráficas mucho más amplias que los que nos ofrece Python.

El análisis tiene como objetivo identificar patrones y relaciones en los datos del mercado inmobiliario para comprender las diferencias de precios o tamaños entre otras cosas que puede haber entre barrios y sus características clave.

Primero usamos el comando .head() para ver qué datos tenemos en la base de datos, y poder ver los datos más interesantes para el análisis, en este caso los datos que nos ofrece la base de datos son los de precio de la vivienda, número de habitaciones, numero de baños, si el edificio tiene ascensor, si tiene terraza, los metros cuadrados que mide la casa, el tipo de vivienda que es, el barrio en el que está y el precio por metro cuadrado.

Posteriormente, usamos el código:

- data.describe().round(2)

este código genera una tabla donde se nos muestran distintos datos relevantes únicamente de las columnas numéricas, ya que hace la media de estos valores y los valores no numéricos no tienen esta cualidad

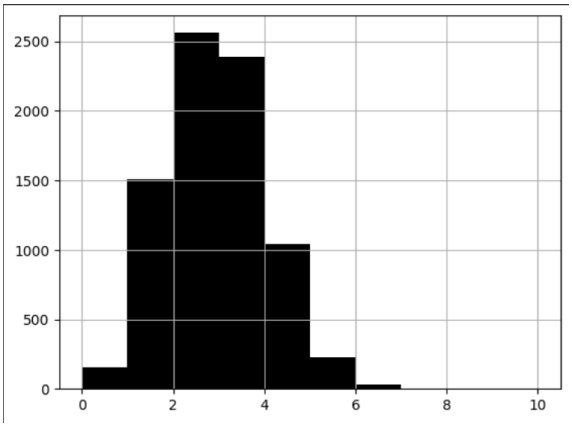
	Unnamed: 0	price	rooms	bathroom	square_meters	square_meters_price
count	7907.00	7907.00	7907.00	7907.00	7907.00	7907.00
mean	4101.10	1416.83	2.44	1.50	84.08	17.58
std	2361.84	1013.36	1.12	0.72	45.78	8.80
min	0.00	320.00	0.00	1.00	10.00	5.56
25%	2052.50	875.00	2.00	1.00	56.00	12.78
50%	4097.00	1100.00	2.00	1.00	73.00	15.22
75%	6153.50	1500.00	3.00	2.00	95.00	19.29
max	8187.00	13478.00	10.00	6.00	480.00	144.50

En la tabla se muestran el número de datos con los que cuenta la base de datos, la media de los valores de cada columna, los valores máximos y mínimos y los percentiles.

Análisis univariante

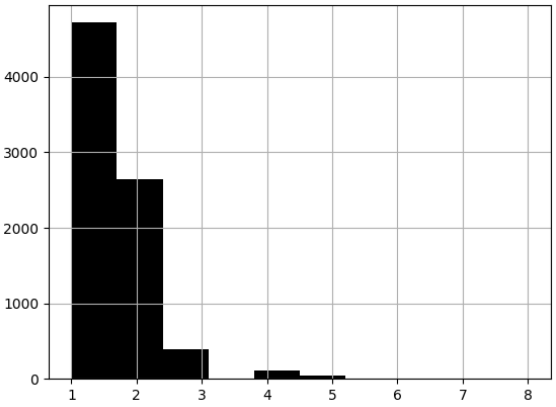
El análisis univariante, se basa en ver los datos de columnas por separado, este análisis es más simple que el análisis bivariante, así que podemos trabajarlo de forma sencilla con python. Para este análisis hemos visto:

1. Media de dormitorios



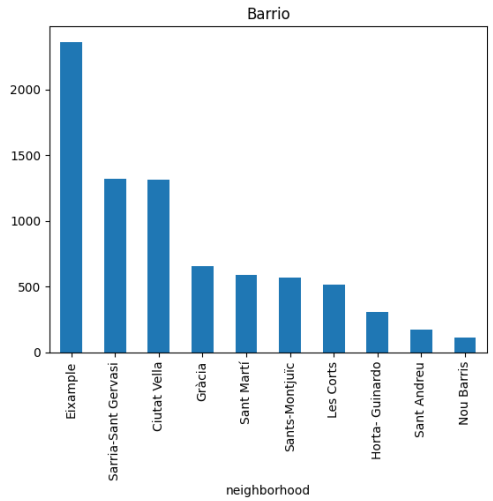
En el gráfico de arriba podemos observar que la mayoría de las viviendas tiene 2 o 3 habitaciones.

2. Cantidad de baños promedio



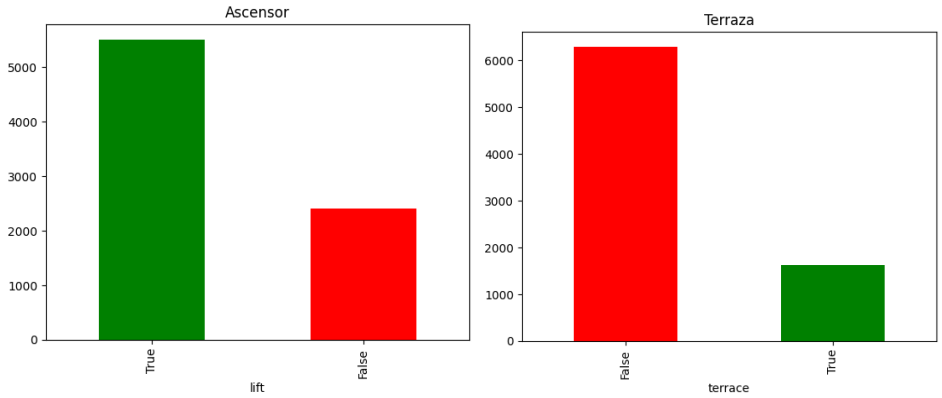
En el gráfico de arriba se puede observar que la gran mayoría de viviendas tiene 1 solo baño

3. Cantidad de viviendas en alquiler según el barrio



Con este grafico podemos ver como el barrio en el que más viviendas alquilables hay es en l'Eixample y donde menos en Nou Barris.

Por último, para terminar con el análisis univariable, vamos a comparar la cantidad de viviendas que hay con ascensor y también la cantidad que hay con terraza



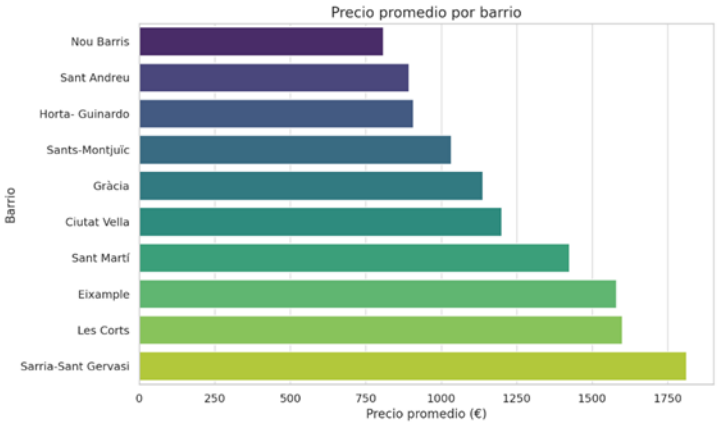
Estos dos gráficos nos muestran que la mayor parte de las viviendas que hay en el mercado, cuentan con ascensor, pero también nos muestra que la gran mayoría no cuenta con terraza.

Con esto terminamos el análisis univariable y empezamos con el bivariable.

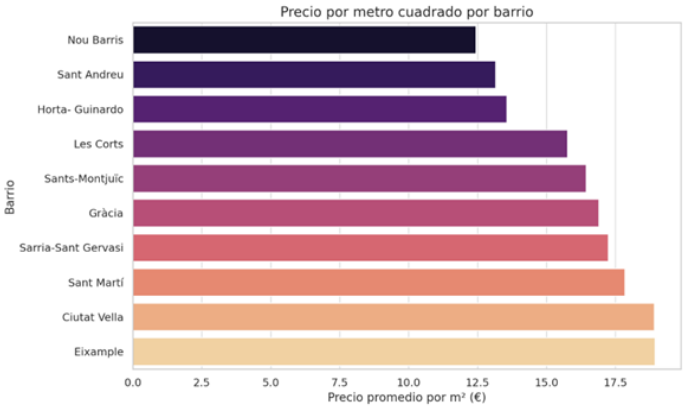
Este tipo de análisis es más complejo ya que se trata de unir dos columnas e datos y compararlas. Para hacerlo más visual, y no trabajar únicamente con filas y columnas, importamos la base de datos a Power BI, y debemos transformar los tipos de datos, ya que no estarán correctamente importados. Durante la transformación lo que vamos a hacer va a ser adaptar los datos a Power BI para que los entienda de mejor manera. Los datos que contengan precios, los tenemos que pasar a moneda, y los valores numéricos, normalmente la página hace de forma automática un sumatorio de los valores, por los que deveras cambiar esa fórmula a acumulación, ya que con el sumatorio los datos que te devolverá serán falsos. Una vez cambiados estos parámetros ya podemos empezar con el análisis.

Para este trabajo hemos analizado distintas variables como:

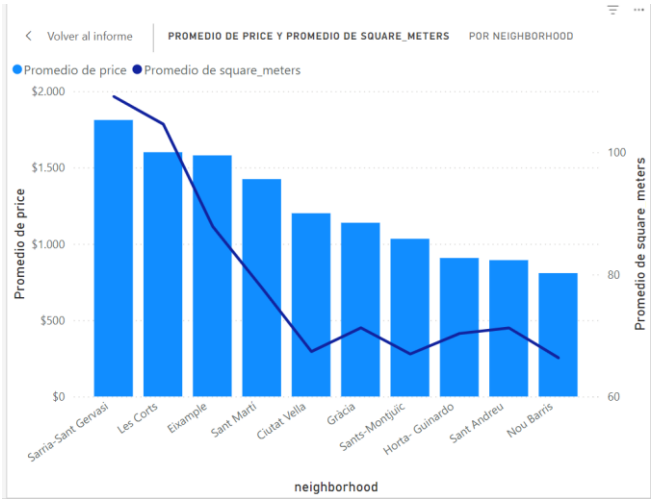
1. Precio promedio por barrio



Con este gráfico podemos observar que los barrios más caros son Sarriá y Les Corts, mientras que los más baratos son Nou Barris y Sant Andreu. Pero, sin embargo, si nos fijamos en este otro gráfico sobre el precio del metro cuadrado según el barrio podemos ver que Nou Barris y Sant Andreu siguen siendo los más baratos, pero los más caros cambian, por Eixample y Ciutat Vella



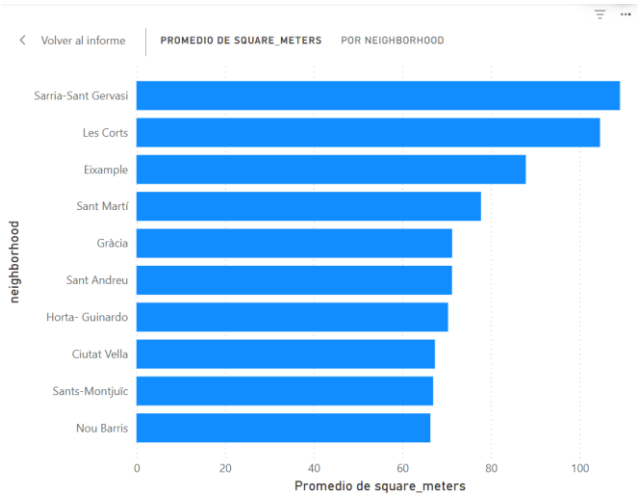
2. Precio promedio total y por metro cuadrado por barrio



En este grafico Se comparan el precio total promedio y el precio promedio por metro cuadrado de las propiedades en cada barrio.

Lo que podemos ver con el grafico es como los barrios más exclusivos mostraron precios por metro cuadrado significativamente más altos, mientras que los barrios menos exclusivos tienen precios más asequibles para gente con un ingreso medio. con lo que podemos deducir que existe una correlación entre el precio total y el precio por metro cuadrado, pero no es perfecta, lo que quiere decir que otros factores influyen en el precio total.

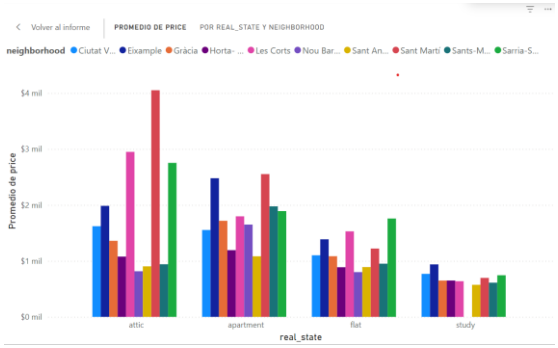
3. Tamaño promedio por barrio



En este gráfico, lo que podemos observar el tamaño de las propiedades según el barrio en el que se encuentran. Podemos observar cómo los barrios familiares tienden a tener

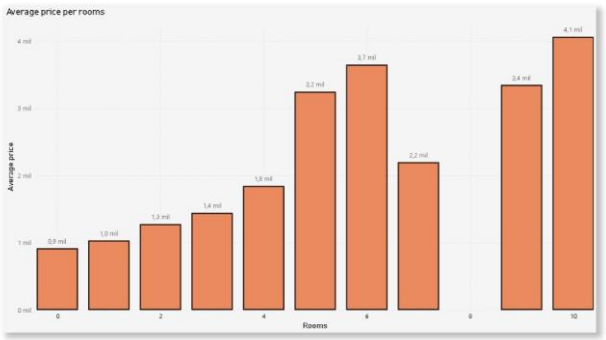
propiedades más grandes (100-120 m^2 en promedio), mientras que los barrios urbanos y de alta densidad tienen propiedades significativamente más pequeñas (50-70 m^2).

4. Distribución de tipo de propiedad por barrio



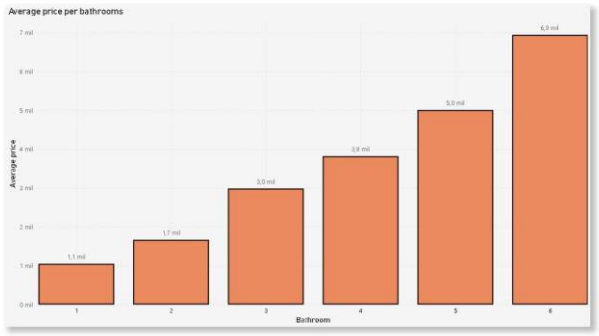
Este grafico nos muestra que tipo de vivienda hay más en cada barrio, mostrándonos por ejemplo como el barrio en el que más áticos se venden es Sant Martí o como lo que menos se vende son apartamentos.

5. Comparación de precio según el número de habitaciones



En este grafico podemos observar que el número de habitaciones sí que suele influir en el precio de la casa, pero no es lo único que influye ya que hay casas más con menos habitaciones que cuestan más que algunas que cuentan con mayor número de habitaciones.

6. Comparación de precio según el número de baños



Con este grafico podemos ver claramente la relación entre el aumento de la cantidad de baños y la cantidad de dinero que cuesta.

Creacion del mapa

Para este proyecto se nos pidió que la presentación contase con un mapa, donde poder ver reflejados algunos datos. Para crear dicho mapa tuvimos que usar muchas librerías distintas de python:

```
1 import pandas as pd
2 import geopandas as gpd
3 import folium
4 import json
5 from folium.features import GeoJsonTooltip
6 from shapely.geometry import Polygon
```

obtuvimos la información de los límites de cada barrio mediante la página de open data de Barcelona, después de una búsqueda pudimos encontrar un archivo que nos proporcionaba las coordenadas de las geodatas que necesitamos para separar por barrios el mapa, una vez tuvimos el archivo, lo subimos a python y mediante este código lo transformamos en un archivo geoJson:

```
input_path = "BarcelonaCiutat_Districtes.json"
with open(input_path, "r", encoding="utf-8") as f:
    district_data = json.load(f)

districts = []

for district in district_data["features"]:
    district_name = district["properties"]["nom_districte"]
    coordinates = district["geometry"]["coordinates"][0]
    polygon = Polygon(coordinates)
    districts.append({"nom_districte": district_name, "geometry": polygon})


districts_gdf = gpd.GeoDataFrame(districts, crs="EPSG:4326")

output_path = "distritos_barcelona.geojson"
districts_gdf.to_file(output_path, driver="GeoJSON")
```

Una vez transformado a geoJson, y apodemos unir la base de datos a el, con este otro código, además, generamos el mapa en archivo .html para poder visualizarlo de forma directa en el buscador.

```
geojson_path = "./distritos_barcelona.geojson"

barrios_gdf = gpd.read_file(geojson_path)

data_neta["neighborhood"] = data_neta["neighborhood"].str.strip()
barrios_gdf["nom_districte"] = barrios_gdf["nom_districte"].str.strip()

avg_price_per_neighborhood = data_neta.groupby("neighborhood")["price"].mean().reset_index()
avg_price_per_neighborhood.columns = ["nom_districte", "avg_price"]

barrios_gdf = barrios_gdf.merge(avg_price_per_neighborhood, on="nom_districte", how="left")

barcelona_map = folium.Map(location=[41.3851, 2.1734], zoom_start=12)

folium.Choropleth(
    geo_data=barrios_gdf,
    name="choropleth",
    data=barrios_gdf,
    columns=["nom_districte", "avg_price"],
    key_on="feature.properties.nom_districte",
    fill_color="RdYlGn_r",
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name="Precio medio de vivienda por barrio (€)"
).add_to(barcelona_map)

folium.GeoJson(
    barrios_gdf,
    name="tooltip",
    style_function=lambda x: {
        'fillColor': 'transparent',
        'color': 'transparent'
    },
    tooltip=GeoJsonTooltip(
        fields=["nom_districte", "avg_price"],
        aliases=["Barrio", "Precio medio (€)"],
        localize=True,
        sticky=False,
        labels=True,
        max_width=300
    )
).add_to(barcelona_map)

barcelona_map.save("mapa_precios_barcelona.html")
```

Conclusiones

El análisis realizado sobre la base de datos del mercado inmobiliario en Barcelona reveló patrones significativos en cuanto a precios, características de las viviendas y su distribución geográfica. Los barrios más exclusivos, como Sarrià y Les Corts, presentaron precios elevados, tanto en términos absolutos como por metro cuadrado, aunque Ciutat Vella y Eixample destacan como los más caros por unidad de superficie. En contraste, barrios como Nou Barris y Sant Andreu se identifican como los más asequibles, reflejando una clara segmentación socioeconómica en la ciudad. Además, El precio de las viviendas mostró correlaciones con factores como el número de habitaciones y baños, aunque estas no son estrictamente lineales, lo que sugiere que otros elementos, como la ubicación o el tipo de vivienda, también influyen significativamente. Finalmente, la integración de los datos en un mapa interactivo permite visualizar de forma clara las diferencias de precios entre barrios, proporcionando una herramienta útil para comprender las dinámicas del mercado y apoyar la toma de decisiones informadas en el sector inmobiliario.