

Respostes Pràctica 2

December 23, 2020

1 Dataset

El dataset emprat en aquesta pràctica és el generat a la PRA1 de l'assignatura, on vam recollir les dades de la web de Fotocasa dels pisos en lloguer a la ciutat de Barcelona (a data d'octubre de 2020), mitjançant scraping.

L'objectiu de l'anàlisi de les dades d'aquest dataset és:

- Respondre a la pregunta de si hi ha diferències significatives en els preus de lloguer entre els diferents barris de la ciutat.
- Respondre a la pregunta de si hi ha diferències significatives en els preus de lloguer entre les diferents agències.
- Crear un model de regressió que permeti predir el preu d'un habitatge en base a les seves característiques.
- Crear un model supervisat que ens digui, donades les característiques d'un lloguer i el seu preu, si aquest és un preu inflat o no.
- Crear un model no supervisat que ens permeti identificar clústers d'observacions amb el preu inflat respecte al preu mitjà teòric del seu barri, i plasmar aquests clústers a un mapa de coordenades (latitud, longitud), per tal de veure si es corresponen amb zones específiques de la ciutat.

2 Selecció de dades

Primerament, determinarem quines són les variables del nostre dataset i el seu tipus. Seguidament, eliminarem aquelles que no aportin informació útil de cara a la nostra anàlisi. Finalment, farem una eliminació de files duplicades.

```
[1]: import pandas as pd

apartments = pd.read_csv('../csv/data_initial.csv')
```

2.1 Eliminació de dimensions

```
[2]: apartments.dtypes
```

```
[2]: id                int64
     address            object
     bathrooms          int64
```

```

building_subtype    object
building_type       object
conservation_state  float64
date                object
discount            int64
floor_elevator      int64
is_new_construction bool
latitude            float64
link                object
longitude            float64
price               float64
realestate          object
realestate_id       int64
rooms               int64
sqft_m2             int64
neighbourhood       object
neighb_meanprice    float64
dtype: object

```

Donats els objectius de la nostra anàlisi, hi ha certes columnes del dataset que podem eliminar:

- **id**, que simplement conté l'índex de la fila dintre del dataset, de moment la deixem per si hem de seleccionar files per aquest índex.
- **address**, que conté la direcció del pis. Ja tenim el barri disponible a la columna **neighbourhood** i per tant, aquest camp aporta informació innecessària a la nostra anàlisi.
- **date**, que conté la data en la que es va recollir la mostra. Solament la vam recollir un dia específic i per tant, no aporta cap informació rellevant a l'anàlisi actual.
- **realestate_id**, que conté l'identificador de l'agència que oferta el pis en lloguer. Ens preferim quedar amb el nom ja que ens permetrà visualitzar la informació de manera més clara

```
[3]: apartments = apartments.drop(columns=['address', 'date', 'realestate_id'])
```

2.2 Eliminació de files duplicades

L'eliminació de files duplicades la portarem a terme en base a la variable **link**. Si tenim dues o més observacions del mateix pis de la web de Fotocasa, ens quedarem únicament amb la primera:

```
[4]: apartments.loc[apartments.duplicated('link')]
```

```
[4]:
```

	id	bathrooms	building_subtype	building_type	conservation_state	\
39	39	1	Apartment	Flat	NaN	
40	40	1	Apartment	Flat	NaN	
41	41	1	Apartment	Flat	NaN	
42	42	1	Apartment	Flat	NaN	
43	43	1	Apartment	Flat	NaN	
...	
11803	11803	2	Attic	Flat	1.0	
11804	11804	2	Attic	Flat	1.0	

12007	12007	5	Flat	Flat	2.0
12061	12061	2	Apartment	Flat	NaN
12116	12116	7	House_Chalet	Flat	2.0

	discount	floor_elevator	is_new_construction	latitude	\
39	0	0	False	41.38071	
40	0	0	False	41.38071	
41	0	0	False	41.38071	
42	0	0	False	41.38071	
43	0	0	False	41.38071	
...	
11803	200	1	False	41.39394	
11804	200	1	False	41.39394	
12007	0	1	False	41.39514	
12061	594	0	False	41.39498	
12116	0	0	False	41.39722	

	link	longitude	price	\
39	/es/alquiler/vivienda/barcelona-capital/el-rav...	2.168707	525.0	
40	/es/alquiler/vivienda/barcelona-capital/el-rav...	2.168707	525.0	
41	/es/alquiler/vivienda/barcelona-capital/el-rav...	2.168707	525.0	
42	/es/alquiler/vivienda/barcelona-capital/el-rav...	2.168707	525.0	
43	/es/alquiler/vivienda/barcelona-capital/el-rav...	2.168707	525.0	
...	
11803	/es/alquiler/vivienda/barcelona-capital/aire-a...	2.121359	3700.0	
11804	/es/alquiler/vivienda/barcelona-capital/aire-a...	2.121359	3700.0	
12007	/es/alquiler/vivienda/barcelona-capital/aire-a...	2.141838	5300.0	
12061	/es/alquiler/vivienda/barcelona-capital/aire-a...	2.147670	6073.0	
12116	/es/alquiler/vivienda/barcelona-capital/aire-a...	2.137248	9000.0	

	realestate	rooms	sqft_m2	\
39	INVERSIONES INMOBILIARIAS Y FINANCIERAS SANT A...	0	35	
40	INVERSIONES INMOBILIARIAS Y FINANCIERAS SANT A...	0	35	
41	INVERSIONES INMOBILIARIAS Y FINANCIERAS SANT A...	0	35	
42	INVERSIONES INMOBILIARIAS Y FINANCIERAS SANT A...	0	35	
43	INVERSIONES INMOBILIARIAS Y FINANCIERAS SANT A...	0	35	
...	
11803	Stipek Ferrer Premium Properties	4	280	
11804	Stipek Ferrer Premium Properties	4	280	
12007	Max Ricart	6	280	
12061	Homelike	4	100	
12116	FINQUES CANO PUJOL	8	560	

	neighbourhood	neighb_meanprice
39	el raval	14.323614
40	el raval	14.323614
41	el raval	14.323614

```

42          el raval          14.323614
43          el raval          14.323614
...
11803       sarrià            16.866645
11804       sarrià            16.866645
12007  sant gervasi - galvany  15.275452
12061  sant gervasi - galvany  15.275452
12116  sant gervasi - galvany  15.275452

[2268 rows x 17 columns]

```

```
[5]: apartments = apartments.drop_duplicates('link')
```

2.3 Tipus de dades

Si observem els tipus de dades anteriors, es pot veure com **conservation_state**, **building_subtype** y **building_type** poden ser variables categòriques, o com **floor_elevator** és en realitat una variable booleana. Podem aplicar la conversió sobre aquestes variables amb l'objectiu de millorar l'anàlisi:

```
[6]: apartments = apartments.astype({
    'conservation_state': 'category',
    'building_subtype': 'category',
    'building_type': 'category',
    'floor_elevator': 'bool',
    'neighbourhood': 'category'
})
```

3 Neteja de dades

3.1 Valors buits

Anem a determinar quins són els valors buits de cadascuna de les columnes:

```
[7]: apartments.isna().sum()
```

```
[7]: id          0
     bathrooms   0
     building_subtype  0
     building_type  0
     conservation_state  5340
     discount     0
     floor_elevator  0
     is_new_construction  0
     latitude     0
     link         0
     longitude    0
```

```

price                4
realestate           106
rooms                0
sqft_m2              0
neighbourhood        1
neighb_meanprice      1
dtype: int64

```

Veiem que molts habitatges no tenen informació sobre l'estat de conservació. Omplirem amb -1 aquells que no en tinguin:

```

[8]: apartments['conservation_state'] = apartments['conservation_state'].cat.
      ↪add_categories(-1)

apartments['conservation_state'] = apartments['conservation_state'].fillna(-1)

```

Donat que volem respondre a la pregunta de si hi ha diferències de preu entre les diferents agències, eliminem les observacions que no tenen agència:

```

[9]: apartments = apartments.dropna(subset=['realestate'])
     apartments.isna().sum()

```

```

[9]: id                0
     bathrooms          0
     building_subtype   0
     building_type       0
     conservation_state  0
     discount            0
     floor_elevator      0
     is_new_construction 0
     latitude            0
     link                0
     longitude           0
     price               4
     realestate          0
     rooms              0
     sqft_m2             0
     neighbourhood      1
     neighb_meanprice    1
     dtype: int64

```

Pel que fa al preu, es tracta de poques observacions que tenen un valor buit per aquesta variable. Com que l'objectiu d'aquesta anàlisi gira entorn al preu, decidim inserir la informació de forma manual consultant la web de Fotocasa:

```

[10]: apartments[apartments.price.isna()]['link']

```

```
[10]: 12154    /es/alquiler/vivienda/barcelona-capital/calefa...
      12155    /es/alquiler/vivienda/barcelona-capital/ascens...
      12156    /es/alquiler/vivienda/barcelona-capital/aire-a...
      12157    /es/alquiler/vivienda/barcelona-capital/aire-a...
      Name: link, dtype: object
```

Les quatre observacions, un cop consultat el link al web de Fotocasa, corresponen a habitatges que no tenen preu, indicant-ho al web com pre “a consultar”. Com necessitem treballar amb els preus, procedirem a eliminar aquestes dades.

```
[11]: apartments = apartments.dropna(subset=['price'])
```

Finalment, ens queda omplir manualment les observacions que no tenen barri, de nou, mitjançant l'enllaç de Fotocasa per veure si podem determinar aquesta informació:

```
[12]: apartments[apartments.neighbourhood.isna()]
```

```
[12]:      id  bathrooms  building_subtype  building_type  conservation_state  \
1384  1384          1                Flat            Flat                -1.0

      discount  floor_elevator  is_new_construction  latitude  \
1384          0             False                False  41.35395

                                     link longitude price  \
1384  /es/alquiler/vivienda/barcelona-capital/la-mar...  2.148532  750.0

      realestate  rooms  sqft_m2  neighbourhood  neighb_meanprice
1384  FINCAS LAFONT      3       70            NaN                NaN
```

Veiem que es correspon amb el barri de “La marina de Port”, així que li assignem el preu mitjà d'aquest barri.

```
[13]: mean_price_marina = float(apartments[apartments['neighbourhood']=='la marina de_
    ↳port'].head(1)['neighb_meanprice'])

apartments.loc[apartments.id ==1384, 'neighbourhood'] = 'la marina de port'
apartments.loc[apartments.id ==1384, 'neighb_meanprice'] = mean_price_marina
```

Podem observar que ara el dataset no té valors buits:

```
[14]: apartments.isna().sum()
```

```
[14]: id                0
      bathrooms         0
      building_subtype  0
      building_type     0
      conservation_state 0
      discount          0
```

```
floor_elevator      0
is_new_construction  0
latitude            0
link               0
longitude           0
price              0
realestate          0
rooms              0
sqft_m2            0
neighbourhood       0
neighb_meanprice    0
dtype: int64
```

Podem eliminar l'id:

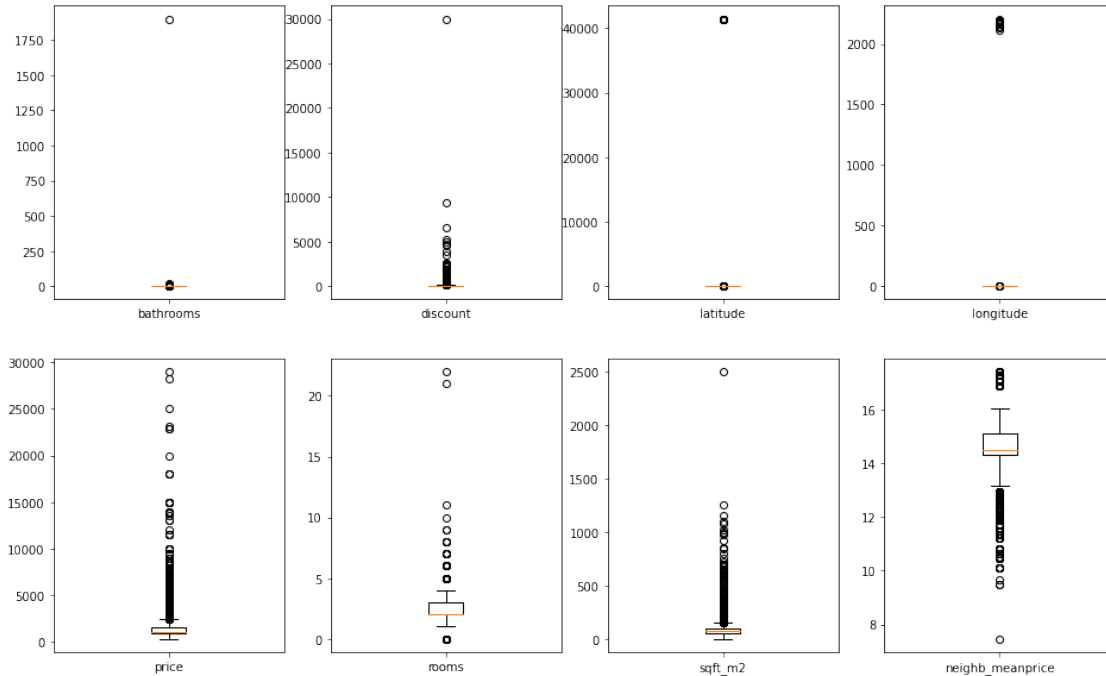
```
[15]: apartments = apartments.drop(columns=['id'])
```

3.2 Outliers

El que primer farem, serà visualitzar a gràfics de tipus boxplot cadascuna de les variables quantitatives per tal de determinar si tenim o no outliers:

```
[16]: import matplotlib.pyplot as plt
import matplotlib.colors as colors
import numpy as np
import math

fig = plt.figure(figsize=(16,10))
numeric_attributes = apartments.select_dtypes(include=np.number).columns
for idx, column in enumerate(numeric_attributes):
    ax = fig.add_subplot(2, math.ceil(len(numeric_attributes)/2), idx+1)
    ax.boxplot(apartments[[column]], labels=[column])
```



Sembla evident que tenim prou valors extrems al dataset per a cadascuna de les variables quantitatives. El que cal és determinar si aquests valors són reals i per tant els hem de tenir en compte a l'anàlisi, o d'altra banda podem eliminar aquells que considerem outliers.

3.2.1 Price

Si seleccionam els 5 preus més alts del dataset, podem comprovar com es tracta de valors vàlids quan accedim al detall a la pàgina de Fotocasa:

```
[17]: apartments.sort_values('price', ascending=False).head()[['link', 'rooms', 'bathrooms', 'realestate', 'sqft_m2', 'price']]
```

```
[17]:
```

	link	rooms	bathrooms	\
12153	/es/alquiler/vivienda/barcelona-capital/aire-a...	4	4	
12152	/es/alquiler/vivienda/barcelona-capital/aire-a...	8	7	
12151	/es/alquiler/vivienda/barcelona-capital/aire-a...	21	21	
12150	/es/alquiler/vivienda/barcelona-capital/aire-a...	3	2	
12149	/es/alquiler/vivienda/barcelona-capital/aire-a...	2	1	

	realestate	sqft_m2	price
12153	VIP PISOS	600	29000.0
12152	Homelike	550	28279.0
12151	ENGEL & VOELKERS	2500	25000.0
12150	Homelike	85	23092.0
12149	Homelike	54	22839.0

Sembla per tant que existeixen uns pocs pisos al dataset amb preus molt elevants, i característiques com nombre d'habitacions, metres quadrats o nombre de banys amb valors que tot i semblar extrems, són vàlids. Per aquest motiu, decidim **mantenir** aquestes observacions per tal de dur a terme l'anàlisi de dades.

3.2.2 Latitude i Longitude

Si s'examinen les variables latitude i longitude, hi veiem valors sospitosos. La latitud és un valor que oscil·la entre -90 i 90, mentre que la longitud ho fa entre -180 i 180. Per tant, qualsevol observació d'aquestes variables fora d'aquest rang es podria considerar un outlier. D'altra banda, esperem que tots els habitatges es trobin a la ciutat de Barcelona, per tant, que tinguin coordenades molt semblants.

```
[18]: apartments[(apartments['latitude'] > 90) | (apartments['latitude'] < -90)].
      ↪sort_values('latitude', ascending=False).head(100)[['link', 'latitude',
      ↪'longitude', 'neighbourhood', 'price']]
```

```
[18]:
```

	link	latitude	longitude	\
3979	/es/alquiler/vivienda/barcelona-capital/aire-a...	41436.0	2.173259	
4177	/es/alquiler/vivienda/barcelona-capital/aire-a...	41428.0	2.189229	
1266	/es/alquiler/vivienda/barcelona-capital/ascens...	41424.0	2.151020	
3694	/es/alquiler/vivienda/barcelona-capital/aire-a...	41422.0	2.188695	
6027	/es/alquiler/vivienda/barcelona-capital/calefa...	41418.0	2.171966	
...	
4181	/es/alquiler/vivienda/barcelona-capital/aire-a...	41373.0	2.157460	
6920	/es/alquiler/vivienda/barcelona-capital/aire-a...	41372.0	2.138111	
7003	/es/alquiler/vivienda/barcelona-capital/aire-a...	41372.0	2.127435	
2567	/es/alquiler/vivienda/barcelona-capital/aire-a...	41372.0	2.147444	
171	/es/alquiler/vivienda/barcelona-capital/amuebl...	41371.0	2.148023	

	neighbourhood	price
3979	porta	950.0
4177	la sagrera	955.0
1266	la teixonera	750.0
3694	sant antoni	910.0
6027	el guinardó	1100.0
...
4181	el poble sec - aei parc montjuïc	955.0
6920	la bordeta	1200.0
7003	sants	1200.0
2567	la font de la guatl·la	850.0
171	la font de la guatl·la	580.0

[87 rows x 5 columns]

Pel que fa a **latitude**, sembla que existeix un error a l'hora d'interpretar els decimals. Les observacions que tenim pels pisos de Barcelona, tenen una latitud que oscil·la entre els següents valors:

```
[19]: apartments[(apartments['latitude']) < 90]['latitude'].max()
```

```
[19]: 41.46262
```

```
[20]: apartments[(apartments['latitude']) < 90]['latitude'].min()
```

```
[20]: 41.35395
```

Podem corregir fàcilment els valors incorrectes, ja que tots són superiors a 41370, dividint el valor de **latitude** entre 1000 per a aquestes observacions:

```
[21]: apartments.loc[apartments['latitude'] > 41370, 'latitude'] =  
    ↪ apartments['latitude']/1000  
apartments.sort_values('latitude', ascending=False).head(10)[['link',  
    ↪ 'latitude', 'longitude', 'neighbourhood', 'price']]
```

```
[21]:
```

		link	latitude	longitude	\
228	/es/alquiler/vivienda/barcelona-capital/terraz...	41.46262	2.180195		
15	/es/alquiler/vivienda/barcelona-capital/terraz...	41.45524	2.174660		
609	/es/alquiler/vivienda/barcelona-capital/terraz...	41.45345	2.190106		
978	/es/alquiler/vivienda/barcelona-capital/amuebl...	41.45231	2.191007		
509	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.45208	2.192924		
578	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.45131	2.190549		
296	/es/alquiler/vivienda/barcelona-capital/terraz...	41.45129	2.189722		
222	/es/alquiler/vivienda/barcelona-capital/terraz...	41.45118	2.189822		
1344	/es/alquiler/vivienda/barcelona-capital/no-amu...	41.45047	2.191933		
7797	/es/alquiler/vivienda/barcelona-capital/calefa...	41.44968	2.190399		

	neighbourhood	price
228	ciutat meridiana	600.0
15	torre baró	490.0
609	la trinitat vella	675.0
978	la trinitat vella	715.0
509	la trinitat vella	650.0
578	la trinitat vella	664.0
296	la trinitat vella	647.0
222	la trinitat vella	600.0
1344	la trinitat vella	750.0
7797	la trinitat vella	1250.0

Podem aplicar el mateix procés per al camp **longitude**:

```
[22]: apartments[(apartments['longitude'] > 180) | (apartments['longitude'] < -180)].  
    ↪ sort_values('longitude', ascending=False).head(100)[['link', 'latitude',  
    ↪ 'longitude', 'neighbourhood', 'price']]
```

```
[22]:
```

		link	latitude	longitude	\
4023	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.39861	2206.0	
1430	/es/alquiler/vivienda/barcelona-capital/ascens...		41.41937	2188.0	
3229	/es/alquiler/vivienda/barcelona-capital/calefa...		41.38230	2188.0	
2042	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.43369	2186.0	
411	/es/alquiler/vivienda/barcelona-capital/sant-p...		41.38697	2177.0	
4265	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.38697	2177.0	
5685	/es/alquiler/vivienda/barcelona-capital/calefa...		41.40808	2169.0	
5683	/es/alquiler/vivienda/barcelona-capital/calefa...		41.40816	2169.0	
2899	/es/alquiler/vivienda/barcelona-capital/ascens...		41.37688	2168.0	
896	/es/alquiler/vivienda/barcelona-capital/can-ba...		41.41630	2164.0	
8972	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.39114	2164.0	
1140	/es/alquiler/vivienda/barcelona-capital/ascens...		41.38768	2161.0	
8145	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.39986	2159.0	
7232	/es/alquiler/vivienda/barcelona-capital/terraz...		41.40638	2146.0	
7437	/es/alquiler/vivienda/barcelona-capital/calefa...		41.40638	2146.0	
10100	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.40403	2142.0	
9729	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.39605	2137.0	
10196	/es/alquiler/vivienda/barcelona-capital/aire-a...		41.38006	2111.0	

	neighbourhood	price
4023	el poblenou	950.0
1430	navas	750.0
3229	la barceloneta	900.0
2042	sant andreu	800.0
411	sant pere, santa caterina i la ribera	650.0
4265	sant pere, santa caterina i la ribera	975.0
5685	el baix guinardó	1090.0
5683	el baix guinardó	1090.0
2899	el raval	855.0
896	can baró	700.0
8972	la drete de l'eixample	1490.0
1140	l'antiga esquerra de l'eixample	738.0
8145	la vila de gràcia	1300.0
7232	el putxet i el farró	1200.0
7437	el putxet i el farró	1200.0
10100	el putxet i el farró	1750.0
9729	sant gervasi - galvany	1650.0
10196	la maternitat i sant ramon	1800.0

Les observacions pels pisos de Barcelona tenen una longitud que oscil·la entre els següents valors:

```
[23]: apartments[(apartments['longitude']) < 180]['longitude'].max()
```

```
[23]: 2.224126
```

```
[24]: apartments[(apartments['longitude']) < 180]['longitude'].min()
```

```
[24]: 2.0915220000000003
```

De nou, si dividim els valors de **longitude** superiors a 2110 entre 1000, corregirem els valors d'aquestes observacions:

```
[25]: apartments.loc[apartments['longitude'] > 2110, 'longitude'] =  
    ↪apartments['longitude']/1000  
apartments.sort_values('longitude', ascending=False).head(10)[['link',  
    ↪'latitude', 'longitude', 'neighbourhood', 'price']]
```

```
[25]:
```

	link	latitude	longitude	\
8502	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41129	2.224126	
10272	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41315	2.222797	
11252	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41155	2.221454	
10433	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41163	2.220950	
10232	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41353	2.220937	
9594	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41261	2.220891	
8157	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41361	2.220773	
11048	/es/alquiler/vivienda/barcelona-capital/calefa...	41.41253	2.220563	
8611	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.40981	2.220171	
11098	/es/alquiler/vivienda/barcelona-capital/aire-a...	41.41261	2.220109	

	neighbourhood	price
8502	el besòs i el maresme	1350.0
10272	el besòs i el maresme	1800.0
11252	el besòs i el maresme	2600.0
10433	el besòs i el maresme	1900.0
10232	el besòs i el maresme	1800.0
9594	el besòs i el maresme	1600.0
8157	el besòs i el maresme	1300.0
11048	el besòs i el maresme	2400.0
8611	diagonal mar i el front marítim del poblenou	1400.0
11098	el besòs i el maresme	2500.0

3.2.3 Bathrooms

Per a aquest camp, tot i que hem comentat que hi ha valors alts que considerem vàlids, si observem el boxplot veiem que hi ha una observació que té un valor molt extrem:

```
[26]: apartments[apartments['bathrooms'] > 100][['link', 'bathrooms', 'rooms',  
    ↪'sqft_m2', 'price']]
```

```
[26]:
```

	link	bathrooms	rooms	\
10875	/es/alquiler/vivienda/barcelona-capital/aire-a...	1901	3	

	sqft_m2	price
10875	135	2200.0

A la web de Fotocasa, s'indica també aquest valor. Tot i això, no és lògic tenir 1901 banys a un pis de 135 metres quadrats de 3 habitacions i per tant, decidim eliminar aquesta observació:

```
[27]: apartments = apartments.drop(index=10875)
```

```
[28]: apartments.bathrooms.max()
```

```
[28]: 21
```

3.2.4 Superfície

```
[29]: apartments = apartments[apartments['sqft_m2'] != 0]
```

3.3 Transformació de dades

Afegirem una nova variable que calculi la diferència de cada habitatge sobre el preu mig del barri:

```
[30]: apartments['diff_mitjana'] = ((apartments['price']/(apartments.  
    ↳neighb_meanprice*apartments.sqft_m2))-1)*100
```

3.4 Fitxer csv amb les dades finals a analitzar

```
[31]: apartments.to_csv('../csv/data_clean.csv')
```

4 Anàlisi

4.1 Grups a comparar

- Es vol comparar si hi ha diferències de preus estadísticament significatives entre els diferents barris. Cada barri serà, per tant, un grup diferent.
- Es vol comparar si hi ha diferències de preus estadísticament significatives entre les diferents agències. Cada agència serà, per tant, un grup diferent.

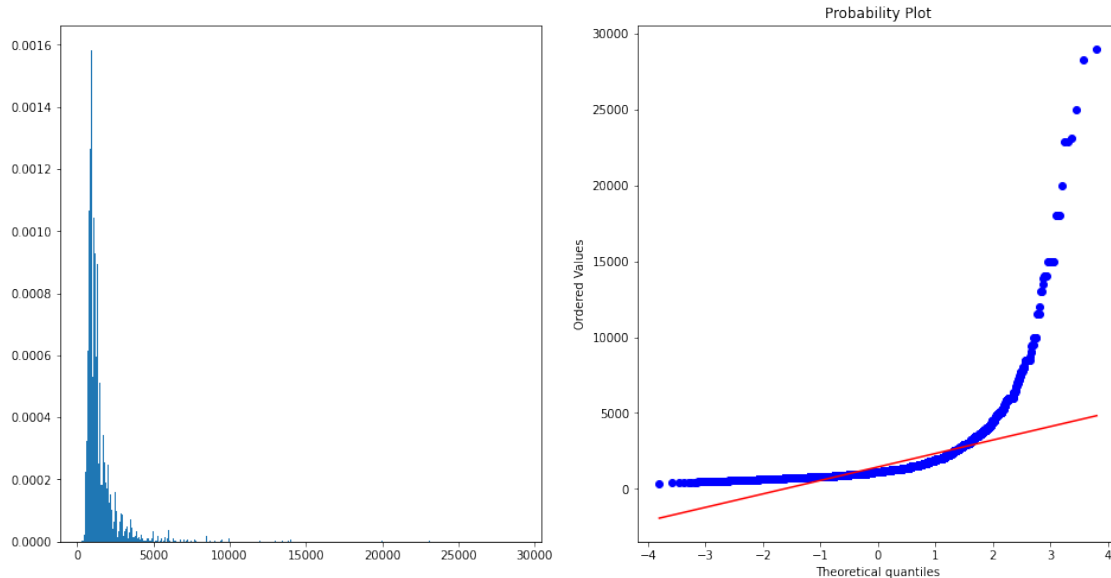
4.2 Comprovació de la normalitat de la variable price

Atès que les anàlisi d'estadística inferencial que portarem a terme impliquen en tots els casos la variable **price**, serà necessari comprovar prèviament la normalitat d'aquesta variable per tal de saber si hem d'aplicar proves paramètriques o no paramètriques.

Anem a visualitzar l'histograma i el gràfic Q-Q de la variable price per tenir una orientació visual de si aquesta variable segueix o no una distribució normal:

```
[32]: from scipy import stats  
  
fig = plt.figure(figsize=(16,8))  
  
ax = fig.add_subplot(1, 2, 1)  
ax.hist(apartments.price, 500, density=True)
```

```
ax = fig.add_subplot(1, 2, 2)
stats.probplot(apartments.price, dist='norm', plot=plt)
pass
```



A simple vista, no sembla que la variable **price** segueixi una distribució normal. Per tal de comprovar-ho, portarem a terme la prova de Kolmogorov-Smirnov. El motiu de portar a terme aquesta prova i no la de Shapiro-Wilk, és perquè amb la implementació de Scipy, per un nombre de mostres superior a 5000, el p-value pot no ésser precís.

```
[33]: stats.kstest(apartments.price, 'norm', args=(apartments.price.mean(),
↪ apartments.price.std()))
```

```
[33]: KstestResult(statistic=0.2449490400964482, pvalue=0.0)
```

Podem veure que el p-value és 0 en el nostre cas, més petit que el valor de significació escollit: 0.05. Per tant podem rebutjar la hipòtesi nul·la i concloure que la variable **price no segueix una distribució normal**.

4.3 Comprovació de l'homoscedasticitat de la variable price per als diferents grups

Per comprovar l'homoscedasticitat de la variable **price** per als diferents grups que volem comparar, emprarem el test de Fligner-Killeen, ja que prèviament hem observat que aquesta variable no segueix una distribució normal.

4.3.1 Homoscedasticitat dels preus dels barris

```
[34]: neighbourhood_grouped_samples = apartments.groupby('neighbourhood')['price'].  
      ↪apply(pd.Series.tolist).tolist()  
      stats.fligner(*neighbourhood_grouped_samples)
```

```
[34]: FlignerResult(statistic=2104.733108385882, pvalue=0.0)
```

Donat el p-value és < 0.05 (el valor de significació escollit), podem concloure doncs que **price** presenta variàncies estadísticament diferents entre els barris de Barcelona.

4.3.2 Homoscedasticitat dels preus de les agències

```
[35]: realestate_grouped_samples = apartments.groupby('realestate')['price'].apply(pd.  
      ↪Series.tolist).tolist()  
      stats.fligner(*realestate_grouped_samples)
```

```
[35]: FlignerResult(statistic=3493.643125045974, pvalue=1.0719141814502517e-260)
```

Donat que el p-value és < 0.05 (el valor de significació escollit), podem concloure que **price** presenta variàncies estadísticament diferents entre les agències immoniliaries.

4.4 Comprovació de mitjanes de la variable price per als diferents grups

Ja hem vist com la variable **price** no segueix una distribució normal, i com per als grups que es volen analitzar, la variància de **price** entre ells presenta diferències estadísticament significants. Donat tot això, haurem d'emprar proves no paramètriques, en concret el test de Kruskal-Wallis, per tal de saber si la mitjana entre els diferents grups (que són més de dos tant en el cas dels barris com en el cas de les agències) presenta diferències estadísticament significants.

4.4.1 Comparació de mitjanes de preus entre els diferents barris

```
[36]: stats.kruskal(*neighbourhood_grouped_samples)
```

```
[36]: KruskalResult(statistic=2637.6757596495586, pvalue=0.0)
```

Donat que el p-value és < 0.05 (el valor de significació escollit), podem concloure que hi ha difències de preu estadísticament significatives entre els barris de Barcelona.

4.4.2 Comparació de mitjanes de preus entre les diferents agències

```
[37]: stats.kruskal(*realestate_grouped_samples)
```

```
[37]: KruskalResult(statistic=4887.593854046478, pvalue=0.0)
```

Donat que el p-value és < 0.05 (el valor de significació escollit), podem concloure que hi ha diferències de preu estadísticament significatives entre les ofertes de les diferents agències.

4.5 Model de regressió

Anem a construir un mode de regressió lineal que ens permeti fer prediccions sobre el preu dels habitatges. Per fer això, primer haure de fer unes transformacions a les dades per que siguin manejables.

4.5.1 Codificació de les variables quantitatives

En un model de regressió no podem treballar directament amb dades categòriques. Hem de codificar-les per poder-les incloure en el nostre model. Optarem per One-Hot-Encoding, que ens permet introduir variables dummy per cada una de les categories d'una variable.

```
[38]: apartments.realestate.value_counts()
```

```
[38]: ENGEL & VOELKERS                836
      Homelike                        760
      APROPERTIES REAL ESTATE Nº Aicat 6388  481
      SH BARCELONA                     381
      AREA CASA                       332
      ...
      SELLARES GESTIO DE PATRIMONIS IMMOBILIARIS SL  1
      JAMES & ASSOCIATES DESIGN COMPANY SL        1
      FINCAS ASENSIO                             1
      LOFTSANTFELIU                             1
      ISABEL PRIU                                1
      Name: realestate, Length: 1046, dtype: int64
```

Veiem que per la variable realestate tenim moltes categories. Per reduir-ne el número, les classificarem segons el nombre d'habitatges que posen en lloguer. Anem a veure com es distribuïx la quantitat d'habitatges que té cada immobiliària.

```
[39]: counts_realestate = apartments.realestate.value_counts().to_frame().
      ↪reset_index()
      counts_realestate.columns = ['realestate', 'counts']

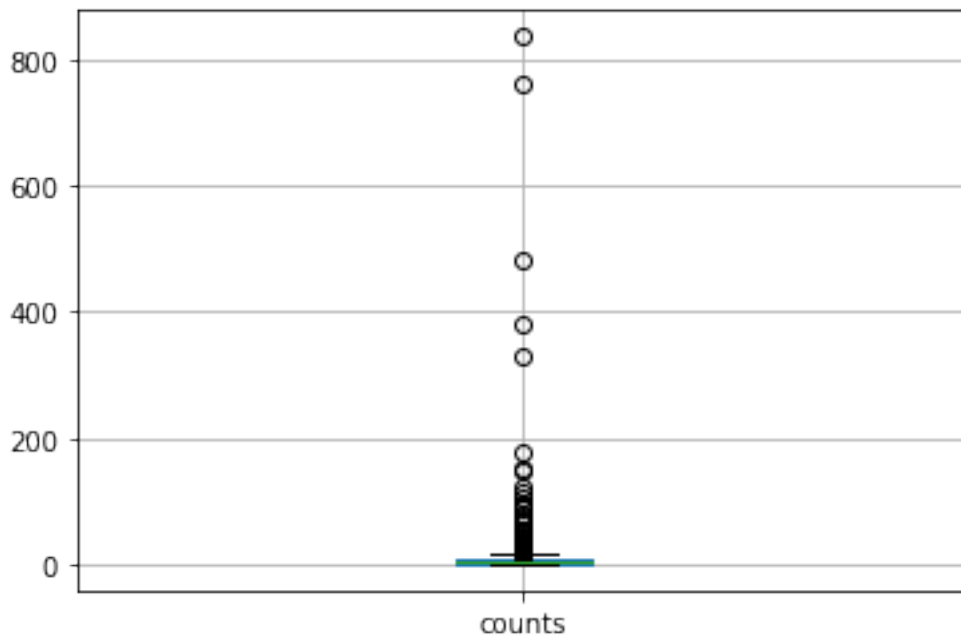
      counts_realestate.describe()
```

```
[39]:          counts
count  1046.000000
mean     9.334608
std     42.954420
min      1.000000
25%      1.000000
50%      3.000000
75%      7.000000
max     836.000000
```

```
[40]: counts_realestate.boxplot()
```



```
[40]: <AxesSubplot:>
```



Explorant les dades hem vist que TECNOCASA apareix disgregada en diferents oficines. Comencem per assignar-li la mateixa immobiliària

```
[41]: counts_realestate[counts_realestate.realestate.astype(str).str.  
      ↪contains('TECNOCASA')]
```

```
[41]:
```

	realestate	counts
60	TECNOCASA RIERA ALTA MAR	24
120	TECNOCASA - ESTUDI AVINGUDA DE ROMA	13
193	TECNOCASA	9
370	TECNOCASA-ESTUDI GRAN DE SANT ANDREU	4
411	TECNOCASA - ESTUDI MARINA 311	4
466	TECNOCASA AVDA. MISTRAL	3
480	TECNOCASA - HOME GUINARDO	3
515	TECNOCASA - PISOS BORNE MAR	3
532	TECNOCASA - PROJECTE LA PROSPERITAT	3
564	TECNOCASA - ESTUDI PROJECTE SANT ANDREU	2
653	TECNOCASA - ESTUDI CAP MARAGALL	2
673	TECNOCASA ESTUDI MERIDIANA	2
758	TECNOCASA - ESTUDIO PROYECTO MARAGALL 2004	1
844	TECNOCASA - ESTUDI NAPOLS	1
946	TECNOCASA - ESTUDI NOU GAUDI	1

```
[42]: apartments.loc[apartments.realestate.astype(str).str.
      ↪contains('TECNOCASA'),'realestate'] = 'TECNOCASA'
counts_realestate = apartments.realestate.value_counts().to_frame()
      ↪reset_index()
counts_realestate.columns = ['realestate', 'counts']
```

Farem la suposició on tots aquells valors de realstate que només tinguin un habitatge els etiquetarem com a propietaris particulars.

```
[43]: particulars = counts_realestate[counts_realestate.counts==1].realestate
apartments.loc[apartments.realestate.isin(particulars),"realestate_size"] =
      ↪'Particular'
```

Entre 2 i 10, els classificarem com a petits propietaris.

```
[44]: petits_propietaris = counts_realestate[np.logical_and(counts_realestate.
      ↪counts>1, counts_realestate.counts<=10)].realestate
apartments.loc[apartments.realestate.
      ↪isin(petits_propietaris),"realestate_size"] = 'Petit'
```

Entre 11 i 50 els classificarem com a propietaris mitjans.

```
[45]: mitja = counts_realestate[np.logical_and(counts_realestate.counts>10,
      ↪counts_realestate.counts<=50)].realestate
apartments.loc[apartments.realestate.isin(mitja),"realestate_size"] = 'Mitja'
```

Més de 50 i menys de 300 com a grans propietaris

```
[46]: grans = counts_realestate[np.logical_and(counts_realestate.counts>50,
      ↪counts_realestate.counts<=300)].realestate
apartments.loc[apartments.realestate.isin(grans),"realestate_size"] = 'Grans'
```

Més de 300, com a molt grans propietaris.

```
[47]: molt_grans = counts_realestate[counts_realestate.counts>300].realestate
apartments.loc[apartments.realestate.isin(molt_grans),"realestate_size"] =
      ↪'Molt grans'
```

Hem simplificat notablement la informació sobre les immobiliàries.

```
[48]: apartments.realestate_size.astype('category')
apartments.realestate_size.value_counts()
```

```
[48]: Molt grans      2790
      Mitja          2619
      Petit          2276
      Grans          1721
      Particular      358
      Name: realestate_size, dtype: int64
```

Ara codifiquem les variables:

```
[49]: from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

apartments_regression = apartments.
    ↳ drop(columns=['realestate', 'building_type', 'link', 'neighb_meanprice', 'diff_mitjana'])
cat_labels = apartments_regression.select_dtypes(include=['object',
    ↳ 'category', 'bool']).columns.to_list()
numeric_labels = apartments_regression.select_dtypes(include=['float64',
    ↳ 'int']).columns.to_list()

preprocessor = ColumnTransformer([('encoder', OneHotEncoder(), cat_labels)],
    remainder='passthrough'
)
apartments_regression_cod = preprocessor.fit_transform(apartments_regression)
cat_labels_encoded = preprocessor.named_transformers_['encoder'].
    ↳ get_feature_names(cat_labels)
new_labels = np.concatenate([cat_labels_encoded, numeric_labels])

apartments_regression_cod = pd.DataFrame(apartments_regression_cod.todense(),
    ↳ columns=new_labels)
apartments_regression_cod
```

```
[49]:
```

	building_subtype_Apartment	building_subtype_Attic \
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	1.0	0.0
...
9759	1.0	0.0
9760	1.0	0.0
9761	0.0	0.0
9762	1.0	0.0
9763	0.0	0.0

	building_subtype_CountryHouse	building_subtype_Duplex \
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0
...
9759	0.0	0.0
9760	0.0	0.0
9761	1.0	0.0

9762	0.0	0.0
9763	0.0	0.0

	building_subtype_Flat	building_subtype_GroundFloorWithGarden	\
0	0.0		0.0
1	1.0		0.0
2	0.0		0.0
3	1.0		0.0
4	0.0		0.0
...
9759	0.0		0.0
9760	0.0		0.0
9761	0.0		0.0
9762	0.0		0.0
9763	1.0		0.0

	building_subtype_House_Chalet	building_subtype_Loft	\
0	0.0	0.0	
1	0.0	0.0	
2	0.0	0.0	
3	0.0	0.0	
4	0.0	0.0	
...	
9759	0.0	0.0	
9760	0.0	0.0	
9761	0.0	0.0	
9762	0.0	0.0	
9763	0.0	0.0	

	building_subtype_SemiDetached	building_subtype_SemidetachedHouse	...	\
0	0.0		0.0	...
1	0.0		0.0	...
2	0.0		0.0	...
3	0.0		0.0	...
4	0.0		0.0	...
...
9759	0.0		0.0	...
9760	0.0		0.0	...
9761	0.0		0.0	...
9762	0.0		0.0	...
9763	0.0		0.0	...

	realestate_size_Molt	grans	realestate_size_Particular	\
0	0.0		0.0	
1	0.0		0.0	
2	0.0		0.0	
3	0.0		0.0	

4	0.0	0.0
...
9759	1.0	0.0
9760	1.0	0.0
9761	1.0	0.0
9762	1.0	0.0
9763	0.0	0.0

	realestate_size_Petit	bathrooms	discount	latitude	longitude	\
0	1.0	1.0	0.0	41.37723	2.160380	
1	1.0	1.0	0.0	41.43157	2.169108	
2	1.0	1.0	0.0	41.37141	2.144235	
3	1.0	2.0	0.0	41.36364	2.139371	
4	1.0	1.0	0.0	41.38046	2.160540	
...	
9759	0.0	1.0	0.0	41.37645	2.142625	
9760	0.0	2.0	0.0	41.38043	2.152265	
9761	0.0	21.0	30000.0	41.42890	2.167005	
9762	0.0	7.0	0.0	41.42745	2.167871	
9763	1.0	4.0	0.0	41.39231	2.145066	

	price	rooms	sqft_m2
0	320.0	0.0	15.0
1	400.0	2.0	63.0
2	430.0	0.0	17.0
3	450.0	1.0	29.0
4	450.0	0.0	25.0
...
9759	22839.0	2.0	54.0
9760	23092.0	3.0	85.0
9761	25000.0	21.0	2500.0
9762	28279.0	8.0	550.0
9763	29000.0	4.0	600.0

[9764 rows x 99 columns]

```
[50]: from sklearn.linear_model import LinearRegression
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import StandardScaler

from statsmodels.api import OLS
import statsmodels.api as sm

X = apartments_regression_cod.drop(columns = 'price')
y = apartments_regression_cod['price']
reg = OLS(y,X).fit()
```

```
reg.summary()
```

```
[50]: <class 'statsmodels.iolib.summary.Summary'>
```

```
"""
                                OLS Regression Results
=====
Dep. Variable:                price    R-squared:                0.595
Model:                        OLS      Adj. R-squared:           0.591
Method:                       Least Squares    F-statistic:            154.6
Date:                         Wed, 23 Dec 2020    Prob (F-statistic):      0.00
Time:                         16:42:10    Log-Likelihood:         -79351.
No. Observations:             9764    AIC:                    1.589e+05
Df Residuals:                 9671    BIC:                    1.596e+05
Df Model:                      92
Covariance Type:              nonrobust
=====
=====
                                coef      std err
t      P>|t|      [0.025      0.975]
-----
building_subtype_Apartment                4211.8233    5729.218
0.735      0.462    -7018.643    1.54e+04
building_subtype_Attic                    4117.7463    5729.957
0.719      0.472    -7114.168    1.53e+04
building_subtype_CountryHouse             -1.604e+04    5912.692
-2.713      0.007    -2.76e+04    -4450.380
building_subtype_Duplex                    3822.2003    5731.142
0.667      0.505    -7412.038    1.51e+04
building_subtype_Flat                      3788.4442    5730.031
0.661      0.509    -7443.617    1.5e+04
building_subtype_GroundFloorWithGarden     3884.5898    5726.958
0.678      0.498    -7341.446    1.51e+04
building_subtype_House_Chalet              4368.3338    5737.173
0.761      0.446    -6877.726    1.56e+04
building_subtype_Loft                     3989.0087    5728.314
0.696      0.486    -7239.687    1.52e+04
building_subtype_SemiDetached              4825.7286    5742.887
0.840      0.401    -6431.531    1.61e+04
building_subtype_SemidetachedHouse         3256.8293    5737.739
0.568      0.570    -7990.340    1.45e+04
building_subtype_Study                    3812.6296    5730.310
0.665      0.506    -7419.977    1.5e+04
conservation_state_-1.0                   4066.8750    1.05e+04
0.387      0.699    -1.65e+04    2.47e+04
conservation_state_1.0                    4099.3003    1.05e+04
0.390      0.696    -1.65e+04    2.47e+04
=====
```

conservation_state_2.0				4055.9893	1.05e+04
0.386	0.699	-1.65e+04	2.47e+04		
conservation_state_3.0				4016.3879	1.05e+04
0.382	0.702	-1.66e+04	2.46e+04		
conservation_state_4.0				3716.2390	1.05e+04
0.353	0.724	-1.69e+04	2.43e+04		
conservation_state_8.0				4082.0489	1.05e+04
0.388	0.698	-1.65e+04	2.47e+04		
floor_elevator_False				1.204e+04	3.15e+04
0.382	0.703	-4.97e+04	7.38e+04		
floor_elevator_True				1.2e+04	3.15e+04
0.381	0.703	-4.98e+04	7.38e+04		
is_new_construction_False				1.135e+04	3.15e+04
0.360	0.719	-5.04e+04	7.31e+04		
is_new_construction_True				1.269e+04	3.15e+04
0.402	0.687	-4.91e+04	7.45e+04		
neighbourhood_can baró				164.3599	986.007
0.167	0.868	-1768.420	2097.139		
neighbourhood_can peguera				404.1565	1292.205
0.313	0.754	-2128.836	2937.149		
neighbourhood_canyelles				133.2780	1175.588
0.113	0.910	-2171.121	2437.677		
neighbourhood_ciutat meridiana				326.0711	1343.437
0.243	0.808	-2307.346	2959.488		
neighbourhood_diagonal mar i el front marítim del poblenou				1387.0103	953.552
1.455	0.146	-482.151	3256.172		
neighbourhood_el baix guinardó				318.8832	959.239
0.332	0.740	-1561.426	2199.193		
neighbourhood_el barri gòtic				373.1920	892.265
0.418	0.676	-1375.834	2122.218		
neighbourhood_el besòs i el maresme				684.5552	978.451
0.700	0.484	-1233.413	2602.523		
neighbourhood_el bon pastor				548.7602	1118.618
0.491	0.624	-1643.966	2741.486		
neighbourhood_el camp d'en grassot i gràcia nova				278.4870	946.572
0.294	0.769	-1576.993	2133.967		
neighbourhood_el camp de l'arpa del clot				363.9740	957.908
0.380	0.704	-1513.726	2241.674		
neighbourhood_el carmel				149.7152	988.664
0.151	0.880	-1788.273	2087.704		
neighbourhood_el clot				405.9271	959.511
0.423	0.672	-1474.915	2286.770		
neighbourhood_el coll				137.1961	976.566
0.140	0.888	-1777.078	2051.470		
neighbourhood_el congrés i els indians				362.0409	993.041
0.365	0.715	-1584.528	2308.609		
neighbourhood_el fort pienc				370.1739	927.200

0.399	0.690	-1447.332	2187.679		
neighbourhood_el	guinardó			282.7792	976.880
0.289	0.772	-1632.110	2197.668		
neighbourhood_el	parc i la llacuna del poblenou			449.0932	925.325
0.485	0.627	-1364.738	2262.925		
neighbourhood_el	poble sec - aei parc montjuïc			270.9089	872.471
0.311	0.756	-1439.317	1981.135		
neighbourhood_el	poblenou			612.5381	938.870
0.652	0.514	-1227.843	2452.919		
neighbourhood_el	putxet i el farró			145.7862	947.153
0.154	0.878	-1710.831	2002.404		
neighbourhood_el	raval			230.3642	888.162
0.259	0.795	-1510.619	1971.348		
neighbourhood_el	turó de la peira			351.7656	1012.835
0.347	0.728	-1633.603	2337.134		
neighbourhood_horta				286.5606	1011.921
0.283	0.777	-1697.017	2270.138		
neighbourhood_hostafrancs				827.3325	881.660
0.938	0.348	-900.907	2555.572		
neighbourhood_l'	antiga esquerra de l'eixample			367.6466	907.478
0.405	0.685	-1411.200	2146.493		
neighbourhood_la	barceloneta			539.8316	888.808
0.607	0.544	-1202.418	2282.081		
neighbourhood_la	bordeta			55.4799	871.401
0.064	0.949	-1652.649	1763.609		
neighbourhood_la	dreta de l'eixample			615.9862	916.977
0.672	0.502	-1181.480	2413.452		
neighbourhood_la	font d'en fargues			2581.6406	1015.508
2.542	0.011	591.033	4572.248		
neighbourhood_la	font de la guatlla			139.9752	875.859
0.160	0.873	-1576.893	1856.843		
neighbourhood_la	guineueta			341.2280	1064.731
0.320	0.749	-1745.867	2428.323		
neighbourhood_la	marina de port			-131.3133	856.279
-0.153	0.878	-1809.799	1547.172		
neighbourhood_la	maternitat i sant ramon			-57.8130	889.849
-0.065	0.948	-1802.103	1686.477		
neighbourhood_la	nova esquerra de l'eixample			264.6756	895.029
0.296	0.767	-1489.768	2019.119		
neighbourhood_la	prosperitat			284.2654	1045.616
0.272	0.786	-1765.361	2333.892		
neighbourhood_la	sagrada família			403.4895	943.072
0.428	0.669	-1445.129	2252.108		
neighbourhood_la	sagrera			421.6366	994.712
0.424	0.672	-1528.208	2371.481		
neighbourhood_la	salut			213.5273	960.671
0.222	0.824	-1669.590	2096.645		

neighbourhood_la teixonera	-295.2657	993.267
-0.297 0.766 -2242.277 1651.745		
neighbourhood_la trinitat nova	218.3446	1114.169
0.196 0.845 -1965.661 2402.350		
neighbourhood_la trinitat vella	342.9484	1081.029
0.317 0.751 -1776.095 2461.992		
neighbourhood_la vall d'hebron	-65.8449	1154.439
-0.057 0.955 -2328.788 2197.098		
neighbourhood_la verneda i la pau	468.2804	1001.635
0.468 0.640 -1495.133 2431.694		
neighbourhood_la vila de gràcia	398.9947	936.240
0.426 0.670 -1436.232 2234.221		
neighbourhood_les corts	198.8403	904.250
0.220 0.826 -1573.679 1971.360		
neighbourhood_les roquetes	306.0658	1090.569
0.281 0.779 -1831.677 2443.809		
neighbourhood_les tres torres	223.3326	931.064
0.240 0.810 -1601.748 2048.413		
neighbourhood_montbau	253.4808	1065.395
0.238 0.812 -1834.916 2341.877		
neighbourhood_navas	429.9618	973.374
0.442 0.659 -1478.055 2337.979		
neighbourhood_pedralbes	379.3253	916.557
0.414 0.679 -1417.318 2175.969		
neighbourhood_porta	389.7852	1020.063
0.382 0.702 -1609.752 2389.322		
neighbourhood_sant andreu	361.1188	1015.544
0.356 0.722 -1629.560 2351.798		
neighbourhood_sant antoni	259.7658	893.451
0.291 0.771 -1491.585 2011.116		
neighbourhood_sant genís dels agudells	747.4315	1046.185
0.714 0.475 -1303.310 2798.173		
neighbourhood_sant gervasi - galvany	423.1587	926.455
0.457 0.648 -1392.888 2239.205		
neighbourhood_sant gervasi - la bonanova	418.7284	948.482
0.441 0.659 -1440.494 2277.951		
neighbourhood_sant martí de provençals	425.9120	979.172
0.435 0.664 -1493.470 2345.294		
neighbourhood_sant pere, santa caterina i la ribera	424.8387	901.497
0.471 0.637 -1342.284 2191.962		
neighbourhood_sants	37.0572	881.130
0.042 0.966 -1690.143 1764.257		
neighbourhood_sarrià	377.6531	932.658
0.405 0.686 -1450.553 2205.859		
neighbourhood_torre baró	310.9440	1334.257
0.233 0.816 -2304.478 2926.366		
neighbourhood_vallcarca i els penitents	224.6084	964.374

0.233	0.816	-1665.766	2114.982		
neighbourhood_vallvidrera, el tibidabo i les planes				56.0155	987.799
0.057	0.955	-1880.277	1992.308		
neighbourhood_verdun				507.7932	1128.550
0.450	0.653	-1704.401	2719.987		
neighbourhood_vilapicina i la torre llobeta				308.4005	1003.745
0.307	0.759	-1659.149	2275.950		
realestate_size_Grans				4898.8523	1.26e+04
0.389	0.698	-1.98e+04	2.96e+04		
realestate_size_Mitja				4729.0675	1.26e+04
0.375	0.708	-2e+04	2.94e+04		
realestate_size_Molt grans				5029.4246	1.26e+04
0.399	0.690	-1.97e+04	2.97e+04		
realestate_size_Particular				4674.8737	1.26e+04
0.371	0.711	-2e+04	2.94e+04		
realestate_size_Petit				4704.6224	1.26e+04
0.373	0.709	-2e+04	2.94e+04		
bathrooms				295.9737	17.234
17.173	0.000	262.191	329.757		
discount				0.3119	0.040
7.813	0.000	0.234	0.390		
latitude				-575.7537	2244.014
-0.257	0.798	-4974.490	3822.983		
longitude				-5804.8852	2067.011
-2.808	0.005	-9856.660	-1753.110		
rooms				-44.2087	10.246
-4.315	0.000	-64.293	-24.125		
sqft_m2				11.0921	0.220
50.319	0.000	10.660	11.524		
=====					
Omnibus:		14896.749	Durbin-Watson:		0.944
Prob(Omnibus):		0.000	Jarque-Bera (JB):		21815890.159
Skew:		9.066	Prob(JB):		0.00
Kurtosis:		233.857	Cond. No.		1.29e+16
=====					

Notes:

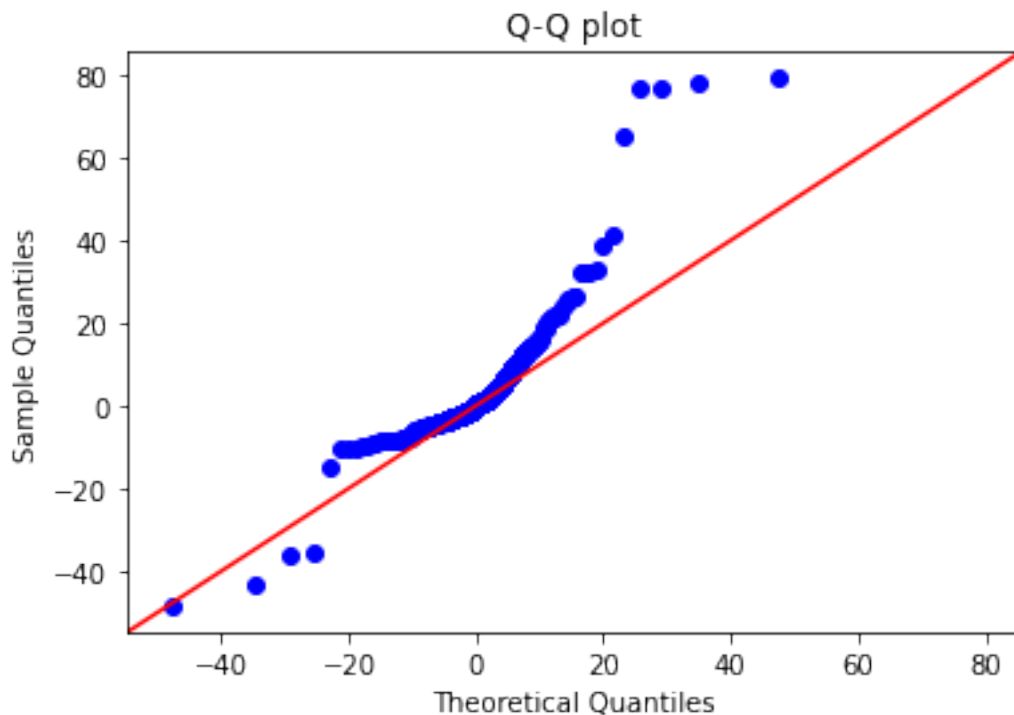
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The smallest eigenvalue is 8.34e-24. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

"""

```
[51]: import scipy.stats as stats
res = reg.resid # residuals
fig = sm.qqplot(res, stats.t, fit=True, line="45")
```

```
plt.title('Q-Q plot')
plt.show()
```



Veiem que el model de regressió lineal no ens dona una capacitat predictiva gaire elevada, amb un R^2 proper al 0,6. Al resum del model, observem que moltes de les variables categòriques codificades no són especialment rellevants estadísticament. S'ha de mencionar, que el model de regressió lineal està enfocat a variables contínues que originin una resposta lineal, així que no es sorprenen aquests resultats. Al Q-Q plot veiem que no es compleixen les condicions d'homoscedasticitat, fet que podríem imaginar després de l'anàlisi de la variable price.

A continuació, provarem amb un model no lineal.

4.6 Regressió amb arbre

```
[52]: from sklearn.model_selection import train_test_split
      from sklearn.ensemble import GradientBoostingRegressor
      import seaborn as sb;
      from sklearn.ensemble import RandomForestClassifier

X = apartments_regression_cod.drop(columns = 'price')
y = apartments_regression_cod['price']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
↳random_state=42)
model = GradientBoostingRegressor( max_depth=3, min_samples_split=20)

model.fit(X_train, y_train)
```

```
[52]: GradientBoostingRegressor(min_samples_split=20)
```

```
[53]: accuracy = model.score(X_test, y_test)

print("El model té una precisió pel conjunt de test de:
↳"+str(round(accuracy*100,2))+"%")
```

El model té una precisió pel conjunt de test de: 66.19%

Sembla que aquest model millora la capacitat predictiva de la regressió lineal. La capacitat de treballar amb atributs categòrics i numèrics i la natura geomètrica del algoritme, són punts a favor d'aquest model que es traslladen en una millor predicció.

4.7 Classificació sobre preu inflat

Anem a construir un altre model, en aquest cas no per predir el preu, sinó per saber si aquest està inflat respecte a la mitjana del barri o no.

```
[54]: apartments_classification = apartments.
↳drop(columns=['realestate', 'building_type', 'link', 'neighb_meanprice'])
cat_labels = apartments_classification.select_dtypes(include=['object',
↳'category', 'bool']).columns.to_list()
numeric_labels = apartments_classification.select_dtypes(include=['float64',
↳'int']).columns.to_list()

preprocessor = ColumnTransformer([('encoder', OneHotEncoder(), cat_labels)],
↳remainder='passthrough'
)
apartments_classification = preprocessor.
↳fit_transform(apartments_classification)
cat_labels_encoded = preprocessor.named_transformers_['encoder'].
↳get_feature_names(cat_labels)
new_labels = np.concatenate([cat_labels_encoded, numeric_labels])

apartments_classification = pd.DataFrame(apartments_classification.todense(),
↳columns=new_labels)
apartments_classification['inflat'] = apartments_classification['diff_mitjana']
↳> 0
apartments_classification = apartments_classification.
↳drop(columns=['diff_mitjana'])
```

```
[55]: from sklearn.metrics import accuracy_score

X = apartments_classification.drop(columns = 'inflat')
y = apartments_classification['inflat']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
↳random_state=42)

rf_tree = RandomForestClassifier(n_estimators=6, max_depth=10)

rf_tree.fit(X_train, y_train)

##CV

accuracy_cv_rf = cross_val_score(rf_tree, X_train, y_train)
print("El model té una precisió de validació creudada: "+str(round(np.
↳mean(accuracy_cv_rf)*100,2))+"%")

### Precisió conjunt de test

y_pred = rf_tree.predict(X = X_test)
accuracy_rd = accuracy_score(y_test, y_pred)
print("El model té una precisió pel conjunt de test de:
↳"+str(round(accuracy_rd*100,2))+"%")
```

El model té una precisió de validació creudada: 78.61%

El model té una precisió pel conjunt de test de: 80.11%

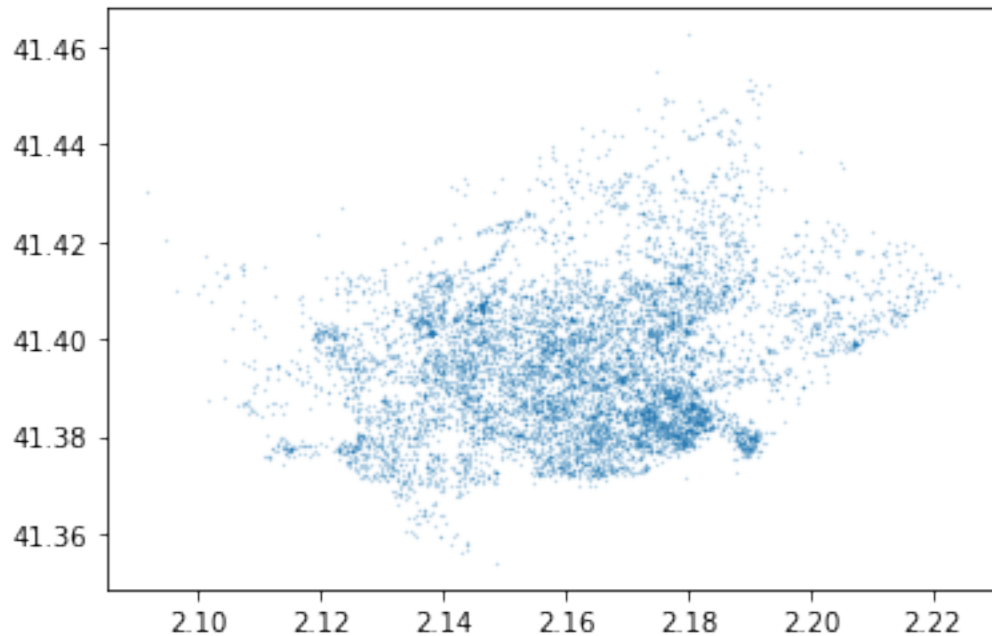
Obtenim bons resultats per aquest classificador, tant en el conjunt de test com el de validació creuada.

4.8 Detecció de zones tensionades

A continuació, aprofitem que tenim les dades geogràfiques per veure si en podem treure alguna conclusió.

```
[56]: plt.scatter(apartments.longitude,apartments.latitude, s=0.1, alpha=0.4,
↳cmap='viridis')
```

```
[56]: <matplotlib.collections.PathCollection at 0x7f53d9bedd00>
```



Ralitzarem un agrupament segons la diferència sobre la mitjana i la posició i visualitzarem el resultat amb les coordenades espacials, per veure si observem algún patró.

Utilitzarem DBSCAN ja que ens permet detecció d'outliers i és sensible a estructures per densitat. L'objectiu de l'agrupament és trobar punts calents a la ciutat on el preu estigui inflat. Per això, treballarem només amb les dades que tinguin el preu per sobre la mitjana, agruparem segons la posició, i balancejarem els punts segons la diferència sobre la mitjana.

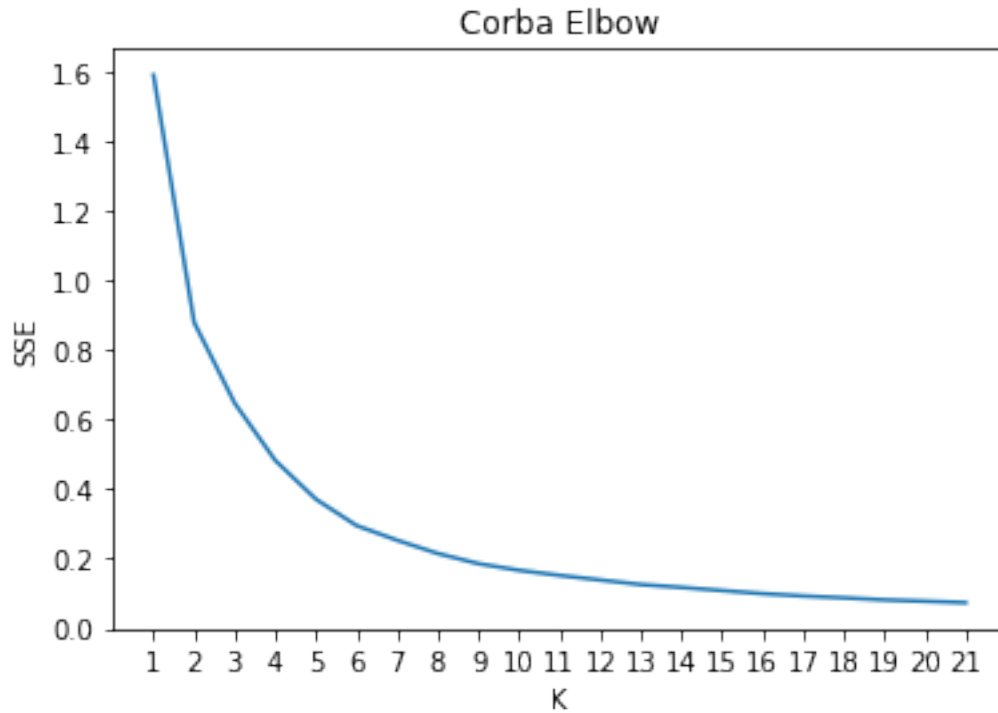
Per començar, fem un estudi de quin és el nombre òptim de clusters:

```
[57]: from sklearn import cluster
      from sklearn.preprocessing import scale

      apartments_cl = apartments[apartments['diff_mitjana'] > 0].copy()
      apartments_cl['diff_mitjana_positive'] = (apartments_cl['price']/
      →(apartments_cl['neighb_meanprice']*apartments_cl['sqft_m2']))-1
      points = apartments_cl[["longitude", "latitude"]].values
      X = apartments_cl[['longitude', 'latitude']].values

      Nc = range(1, 22)
      kmeans = [cluster.KMeans(n_clusters=i) for i in Nc]
      kmeans
      score = [kmeans[i].fit(X, sample_weight=apartments_cl.diff_mitjana_positive).
      →inertia_ for i in range(len(kmeans))]
      score
      plt.plot(Nc, score)
      plt.xlabel('K')
```

```
plt.ylabel('SSE')
plt.title('Corba Elbow')
plt.xticks(Nc)
plt.show()
```



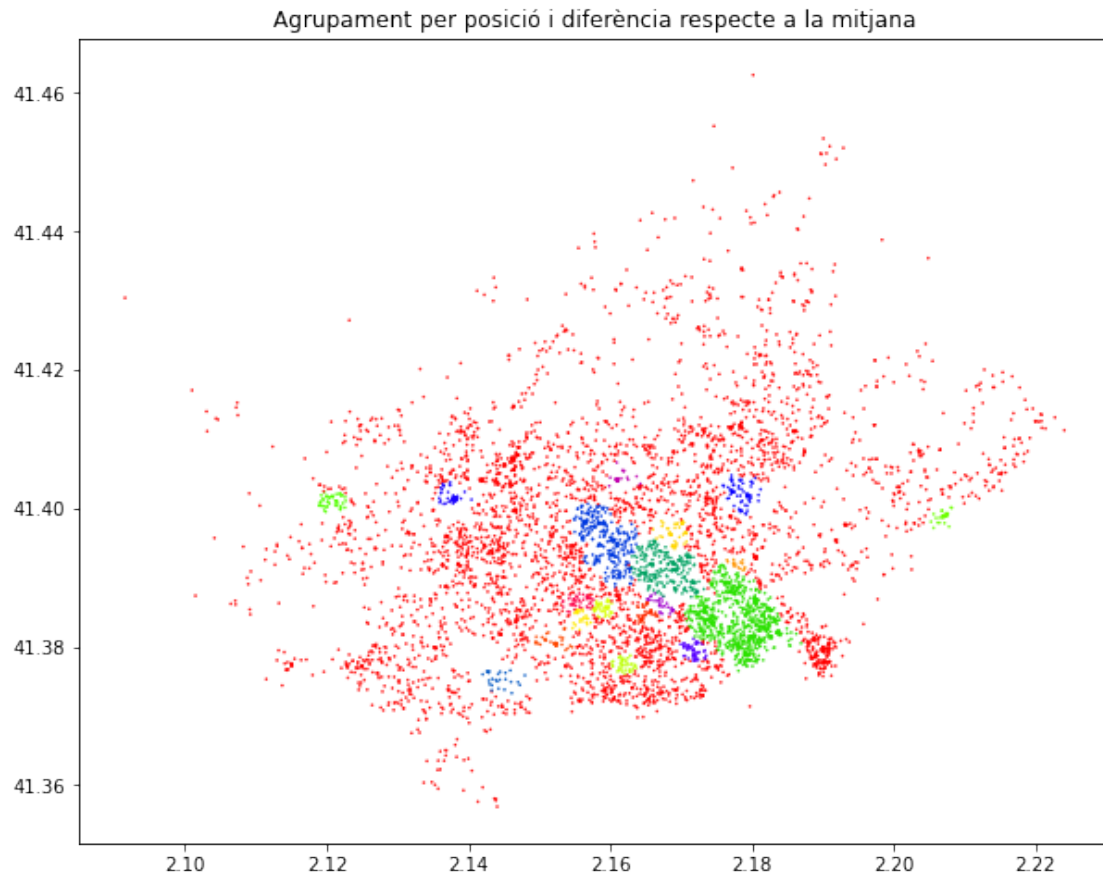
```
[58]: from sklearn import cluster
from sklearn.preprocessing import scale
clustering = cluster.DBSCAN(eps=0.0015, min_samples=18).
    ↳ fit(X, sample_weight=apartments_cl['diff_mitjana_positive'])

y_kmeans = clustering.
    ↳ fit_predict(X, sample_weight=apartments_cl['diff_mitjana_positive'])
plt.figure(figsize=(10, 8))

apartments_cl['clusters'] = y_kmeans

plt.scatter(points[:, 0], points[:, 1], c=y_kmeans, alpha=1, s=0.3, cmap='prism')
plt.title("Agrupament per posició i diferència respecte a la mitjana")
```

```
[58]: Text(0.5, 1.0, 'Agrupament per posició i diferència respecte a la mitjana')
```

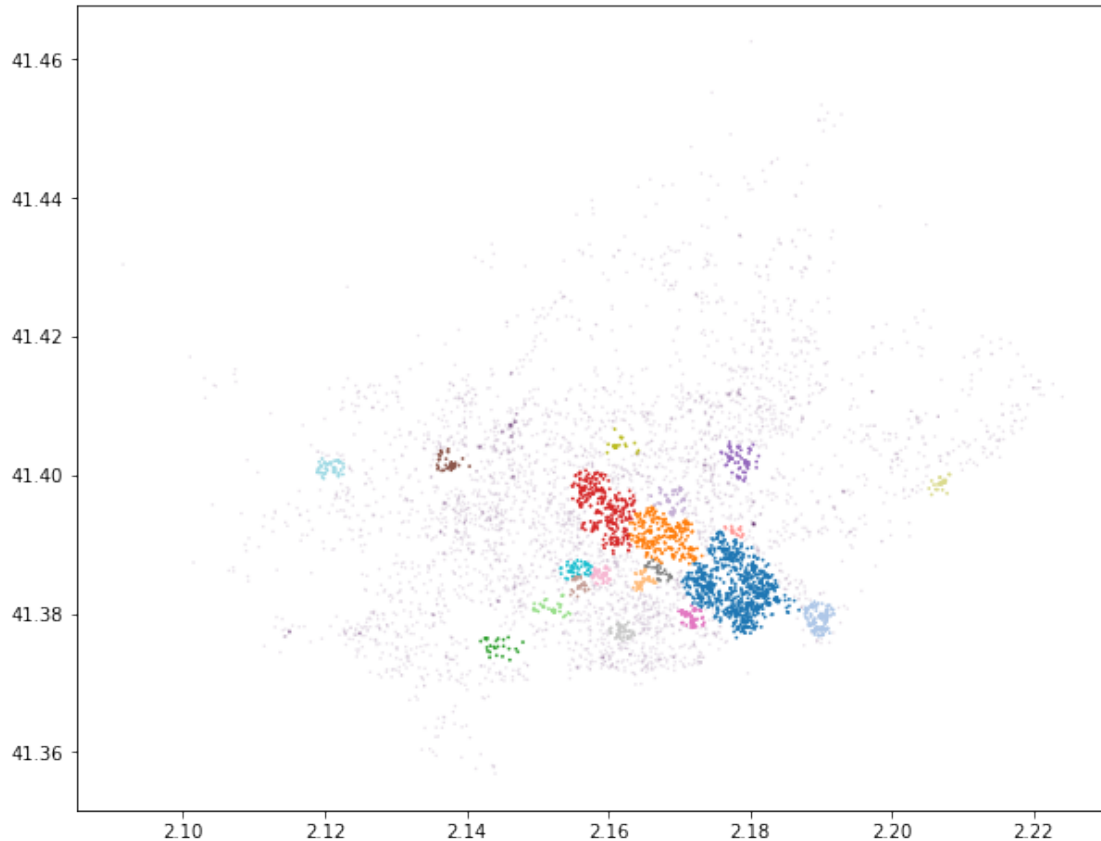


```
[59]: fig = plt.figure(figsize=(10, 8))

ax1 = fig.add_subplot(111)
apartments_nclustered = apartments_cl[apartments_cl['clusters'] == -1]
ax1.scatter(apartments_nclustered.longitude, apartments_nclustered.latitude,
            ↪c=apartments_nclustered.clusters,alpha=0.1, s=0.3)

apartments_clustered = apartments_cl[apartments_cl['clusters'] != -1]
ax1.scatter(apartments_clustered.longitude, apartments_clustered.latitude,
            ↪c=apartments_clustered.clusters,alpha=1, s=0.6, cmap='tab20')
```

```
[59]: <matplotlib.collections.PathCollection at 0x7f5421127610>
```

L'algorisme DBSCAN ens permet detectar outliers, que en aquest cas són elements que no s'han pogut relacionar amb cap clúster. Al segon gràfic els veiem en un color claret, mentre que els clústers en un color més fort i definit. Sembla que l'agrupament resultant respón a zones turístiques de la ciutat, que acostumen a ser zones tensionades.

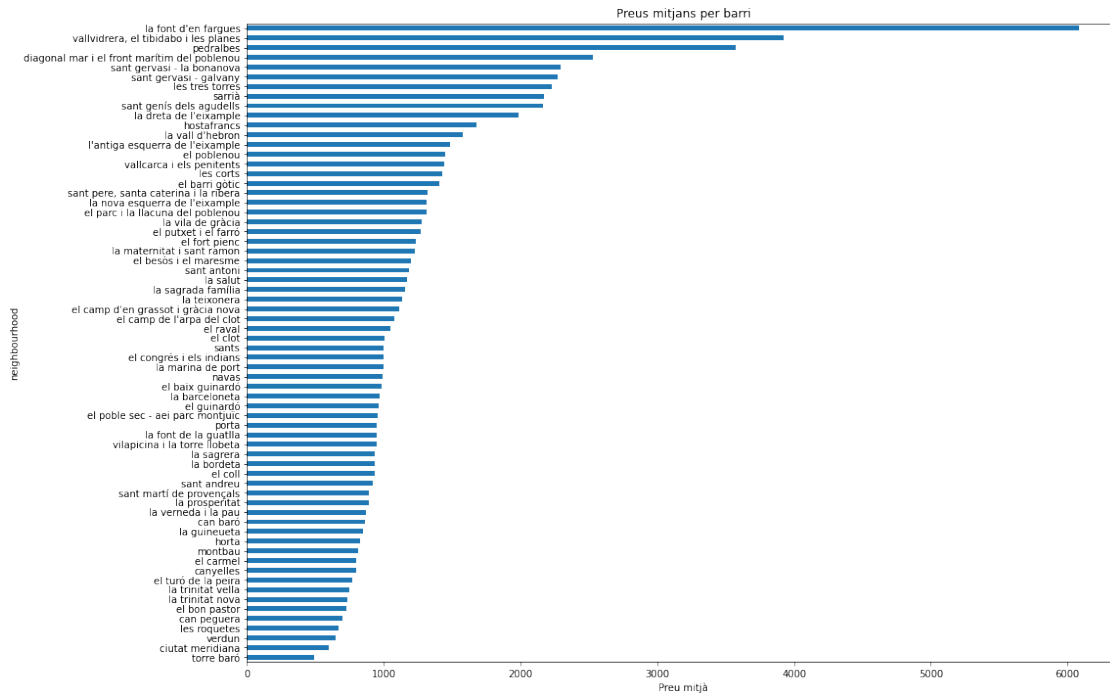
5 Conclusions

Després de portar a terme l'anàlisi de les dades, podem presentar amb confiança les respostes als diferents objectius plantejats:

- Podem confirmar que existeixen diferències estadísticament significants entre els preus de lloguer dels diferents barris de la ciutat de Barcelona. Sembla un fet lògic, ja que a la majoria de ciutats existeixen barris on el preu és elevat, mentre que d'altres tenen preus mes moderats (zones menys turístiques) o fins i tot preus sorprenentment baixos (zones conflictives).

```
[60]: neighbourhood_mean_prices = apartments[['neighbourhood', 'price']].
      ↳groupby('neighbourhood')['price'].mean().sort_values()
fig = plt.figure(figsize=(16,12))
plt.xlabel('Preu mitjà')
neighbourhood_mean_prices.plot.barh(title='Preus mitjans per barri')
```

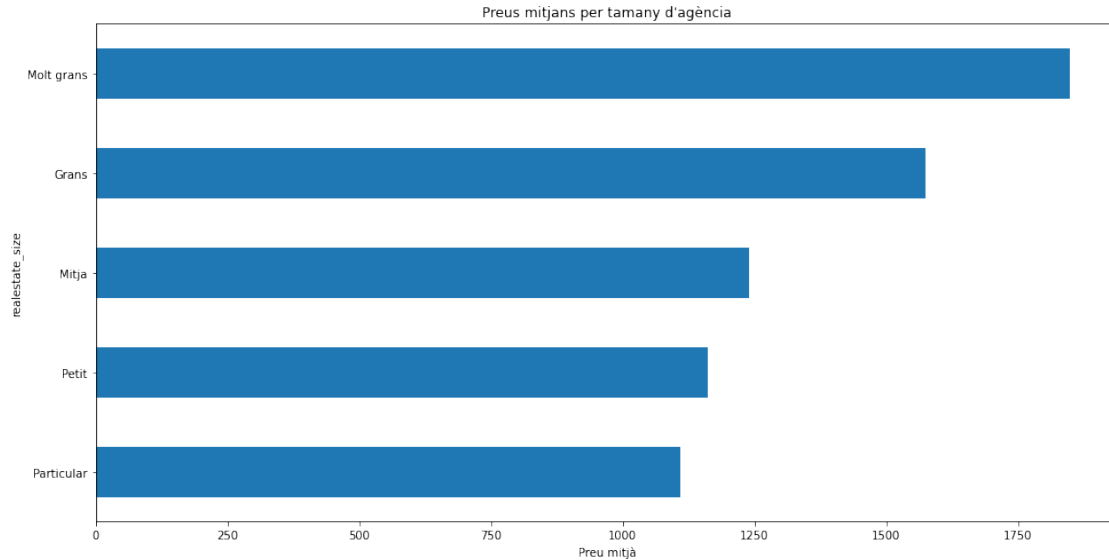
```
[60]: <AxesSubplot:title={'center':'Preus mitjans per barri'}, xlabel='Preu mitjà',
ylabel='neighbourhood'>
```



- Podem confirmar també que existeixen diferències estadísticament significants entre els preus de lloguer que ofereixen les diferents agències a la ciutat de Barcelona. Podem explicar aquestes diferències degut a que sovint grans immobiliàries tenen una gran quota de mercat, especialment de pisos amb preus elevats, respecte a d'altres més petites i els particulars, per tant estan en una posició en la que poden jugar amb els preus més lliurement. En part podem observar aquest fet visualment si mirem els preus mitjans segons el tamany d'agència:

```
[61]: realestate_size_mean_price = apartments[['realestate_size', 'price']].
      ↳groupby('realestate_size')['price'].mean().sort_values()
fig = plt.figure(figsize=(16, 8))
plt.xlabel('Preu mitjà')
realestate_size_mean_price.plot.barh(title="Preus mitjans per tamany d'agència")
```

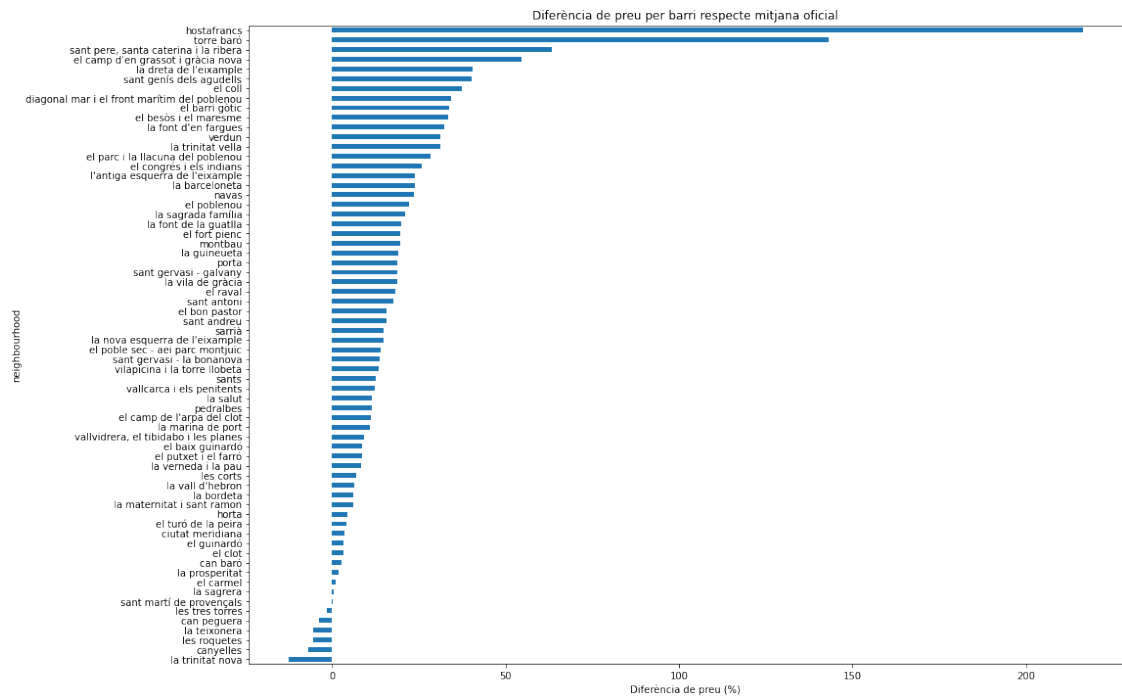
```
[61]: <AxesSubplot:title={'center':"Preus mitjans per tamany d'agència"}, xlabel='Preu
mitjà', ylabel='realestate_size'>
```



- Es poden construir **models de regressió** per intentar predir el preu de lloguer d'un habitatge. Tot i això, els models resultants que hem construït a aquesta pràctica (regressió lineal i regressió amb arbre) tenen una **precisió per sota del 70%** en ambdós casos. Com a alternativa, podríem estudiar l'efecte dels valors extrems i proporcionar un model més precís per un rang de preus més acotat.
- El model de classificació supervisat que hem entrenat ens permet saber amb una precisió notable (~80%) si el preu d'un allotjament està inflat o no. Això pot ser útil per a evaluar, donades les característiques específiques d'un pis, si el preu és massa elevat. A priori, però, es tracta d'un model que s'hauria d'anar actualitzant sovint, ja que les mitjanes de preus dels organismes oficials canvien al llarg del temps, i són la base per la qual hem decidit si un preu està o no inflat.
- El model de classificació no supervisat que hem creat mitjançant l'algorisme DBSCAN (a priori no coneixem el nombre de clústers que podríem formar), ens ha permès confirmar que hi ha zones específiques de la ciutat on el preu està molt per sobre del preu mitjà que aporten els organismes oficials. Com s'ha comentat, sembla que correspon amb zones turístiques, per tant no sorprenen aquests resultats. Tot i això, existeix la **possibilitat que els organismes oficials emprin un altre criteri per determinar els preus mitjans dels barris**, ja que a la majoria d'ells el preu mitjà de les observacions està per sobre del preu mitjà proporcionat pels organismes oficials:

```
[62]: overpriced_neighbourhoods = apartments.groupby('neighbourhood',
↳observed=True)['diff_mitjana'].mean().sort_values()
fig = plt.figure(figsize=(16, 12))
plt.xlabel('Diferència de preu (%)')
overpriced_neighbourhoods.plot.barh(title='Diferència de preu per barri',
↳respecte mitjana oficial')
```

```
[62]: <AxesSubplot:title={'center':'Diferència de preu per barri respecte mitjana oficial'}, xlabel='Diferència de preu (%)', ylabel='neighbourhood'>
```



6 Taula de contribucions

Contribucions	Firma
Investigació prèvia	JMOB, PBL
Redacció de les respostes	JMOB, PBL
Desenvolupament codi	JMOB, PBL