Análisis de datos de viviendas de alquiler de Airbnb en Madrid

Autor: Javier Rodríguez Martín

Las bases de datos utilizadas se pueden obtener en la página oficial de Airbnb aquí.

Se han empleado 3 bases de datos acerca de las viviendas que se alquilan en la plataforma de Airbnb en la ciudad de Madrid durante el año 2018 (*calendar, listings* y *reviews)*

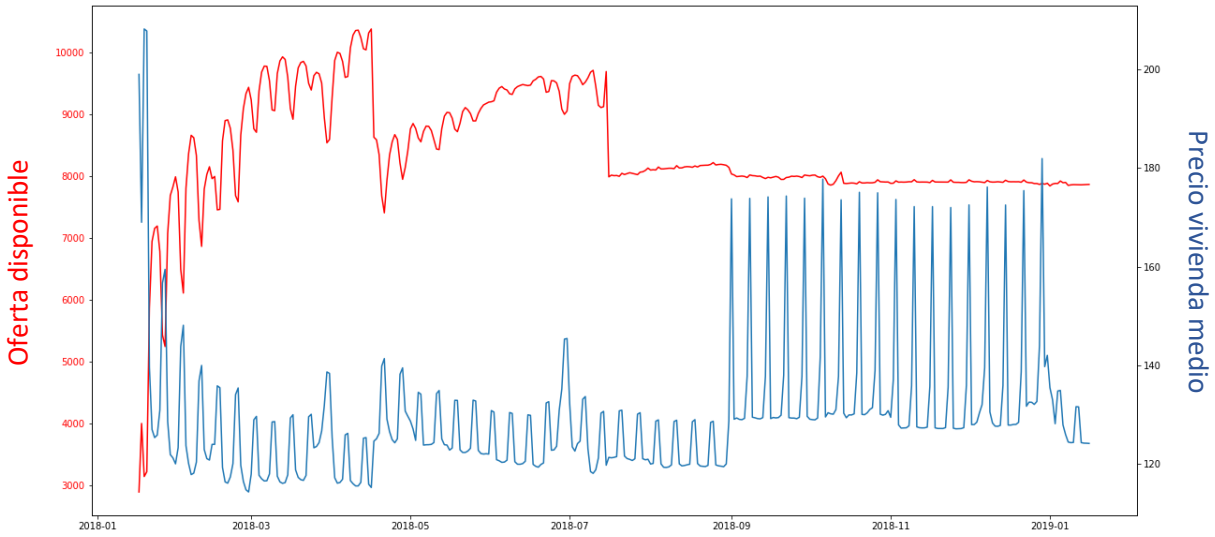
1. *Calendar*: Información sobre la disponibilidad, fechas y precios de las viviendas entre el enero de 2018 hasta enero de 2019.
2. *Listings*: Contiene información detallada de las viviendas y de su perfil en la página web entre enero de 2018 hasta enero de 2019.
3. *Reviews*: Concentra las valoraciones de los usuarios comprendidas entre el 2010 y 2018

El estudio se ha centrado en los siguientes contenidos:

Contenido del análisis

1. Patrón de disponibilidad y precios

Usando la base de datos de *calendar*, se han recogido el número total de viviendas/habitaciones para alquilar y la media de precios de cada día a lo largo del año.



PUNTOS CLAVE:

Oferta disponible:

- En los 3 primeros meses hubo un incremento del 233% en la cantidad de viviendas ofertadas.
- Grandes irregularidades en la primera mitad del año.
- A partir de agosto una caída y estabilización de la demanda hasta fin de año.

Precio de la vivienda medio:

- Caída en el precio a medida que aumenta la oferta en los primeros dos meses.
- Subida repentina de los precios a partir de septiembre, explicada por el bajón de la oferta de viviendas, que se mantienen hasta finales de año.
- Se experimentan constantes subidas y bajadas relacionadas con los ciclos semanales.

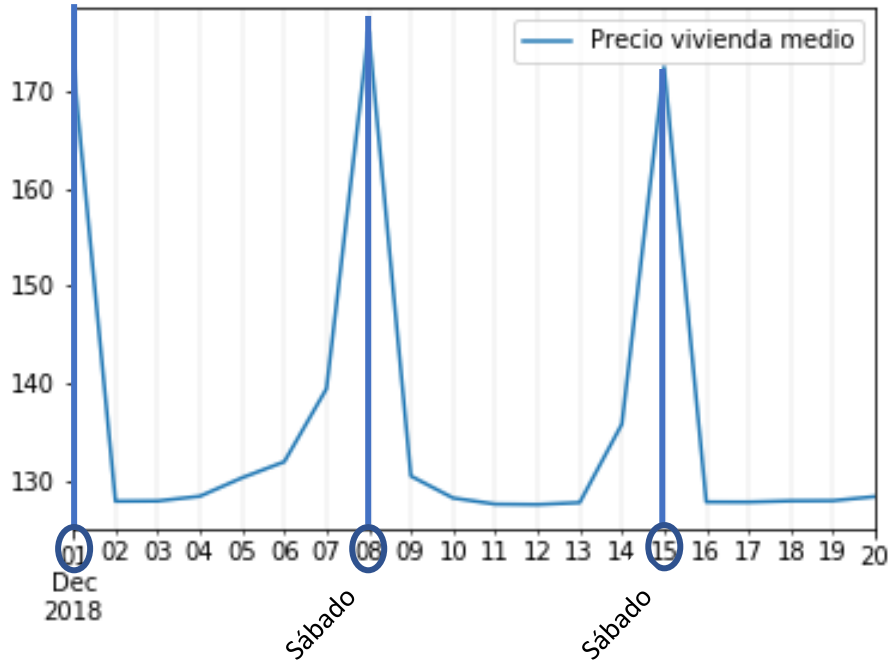
Efecto de los ciclos semanales en el precio medio de la vivienda

Según el día de la semana, se producen grandes cambios en los precios. Los sábados es el día con los precios más caros, la mayoría de gente suele hacer sus reservas para el fin de semana.

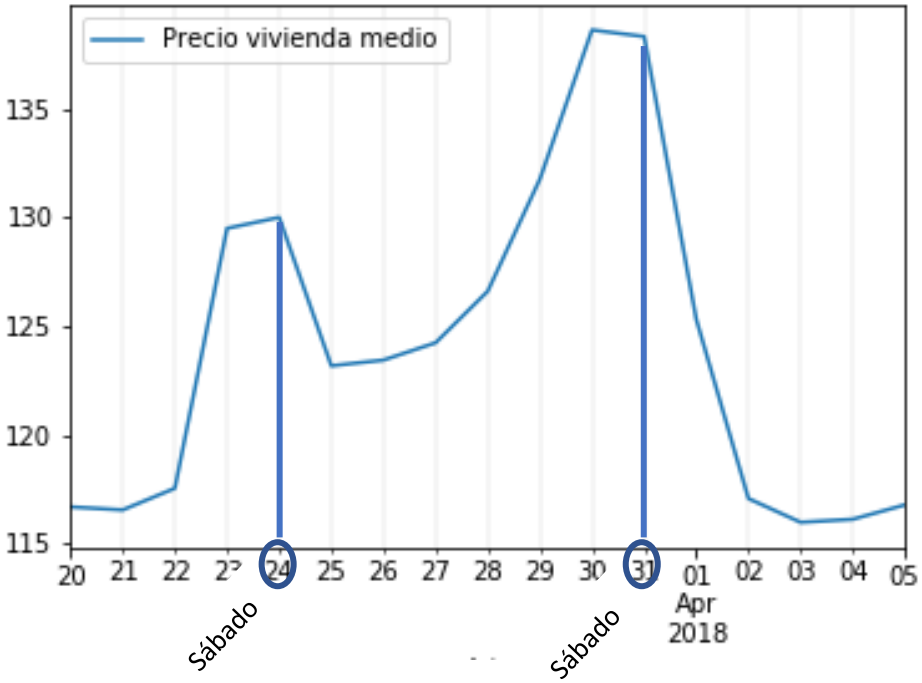
Este gráfico nos dice quién es el tipo de usuario que alquila viviendas en Airbnb.

- El usuario promedio de Airbnb alquila y viaja por turismo, es por ello que la mayor demanda se produce en los fines de semana. Este patrón también existe en los períodos de verano y semana santa.
- Por otra parte, las personas que viajan por negocio no parecen usar mucho Airbnb, los días de diario son los días con menor demanda.

Precio vivienda medio del 1-12-18 al 20-12-18



Precio vivienda medio durante las vacaciones de semana santa



- En el período de semana santa (del 25 de marzo al 1 de abril) los precios en los días de diario aumentaron (aunque se sigue distinguiendo perfectamente donde están los fines de semana)
- Sin embargo, los precios vuelven a la "normalidad" después del 1 de abril que es cuando terminan las fiestas.

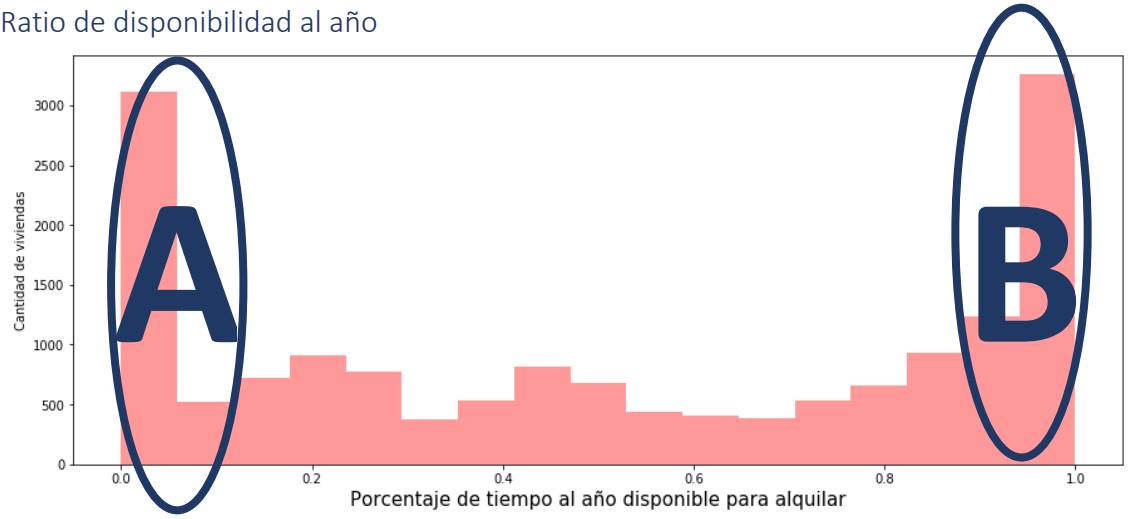
Disminución en la cantidad de oferta disponible. ¿Motivos?

Los motivos no son algo que podamos encontrar en esta fuente de datos, pero sí sabiendo que alrededor de esas fechas se estableció una nueva legislación en el municipio de Madrid con la intención de disminuir y controlar la cantidad de viviendas disponibles en servicios con Airbnb, por lo que podemos observar las consecuencias de dichas medidas.

Disponibilidad de las viviendas a lo largo del año

¿Alquila la gente sus viviendas durante todo el año? ¿O es una actividad casual?

El siguiente gráfico de barras mide cuanta parte del año están las viviendas disponibles.



PUNTOS CLAVE:

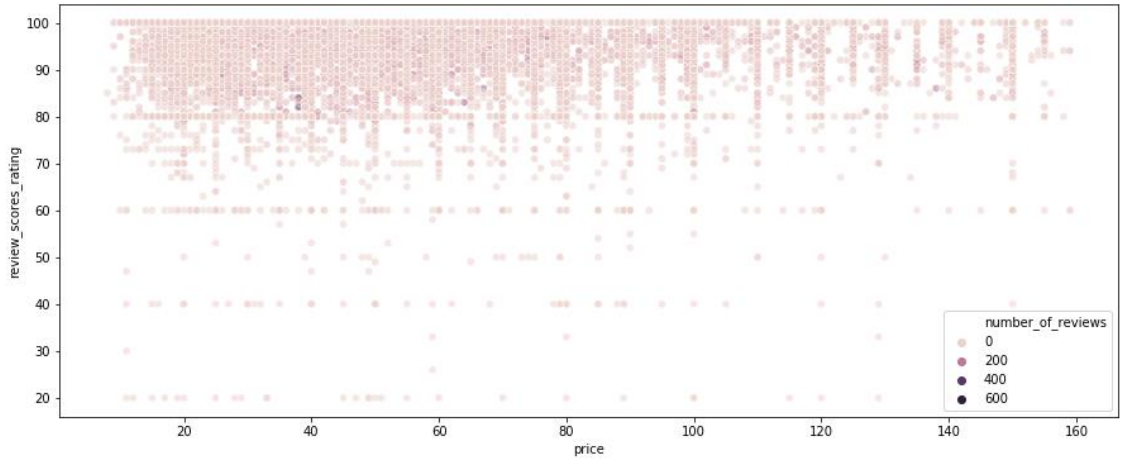
- Concentración en los extremos
- Un extremo **A** en el que una importante cantidad de viviendas solo están disponibles para alquilar un par de semanas al año.
- Otro extremo en el que otra importante cantidad de viviendas tienen disponibilidad todo o casi todo el año.

A	B
Un **20% de toda la oferta** está disponible para alquilar **menos de 1 meses al año** (menos del 10% de todo el año)	Un **25% de toda la oferta** está disponible para **alquiler más de 11 meses al año** (más del 90% de todo el año)

2. Variables con un peso significativo en el precio

Relación entre precio y valoración de los usuarios

Diagrama de dispersión de la relación entre el precio y las valoraciones de los usuarios después de quitar los outliers de alquileres de precios muy altos que alteraban los resultados.



PUNTOS CLAVE:

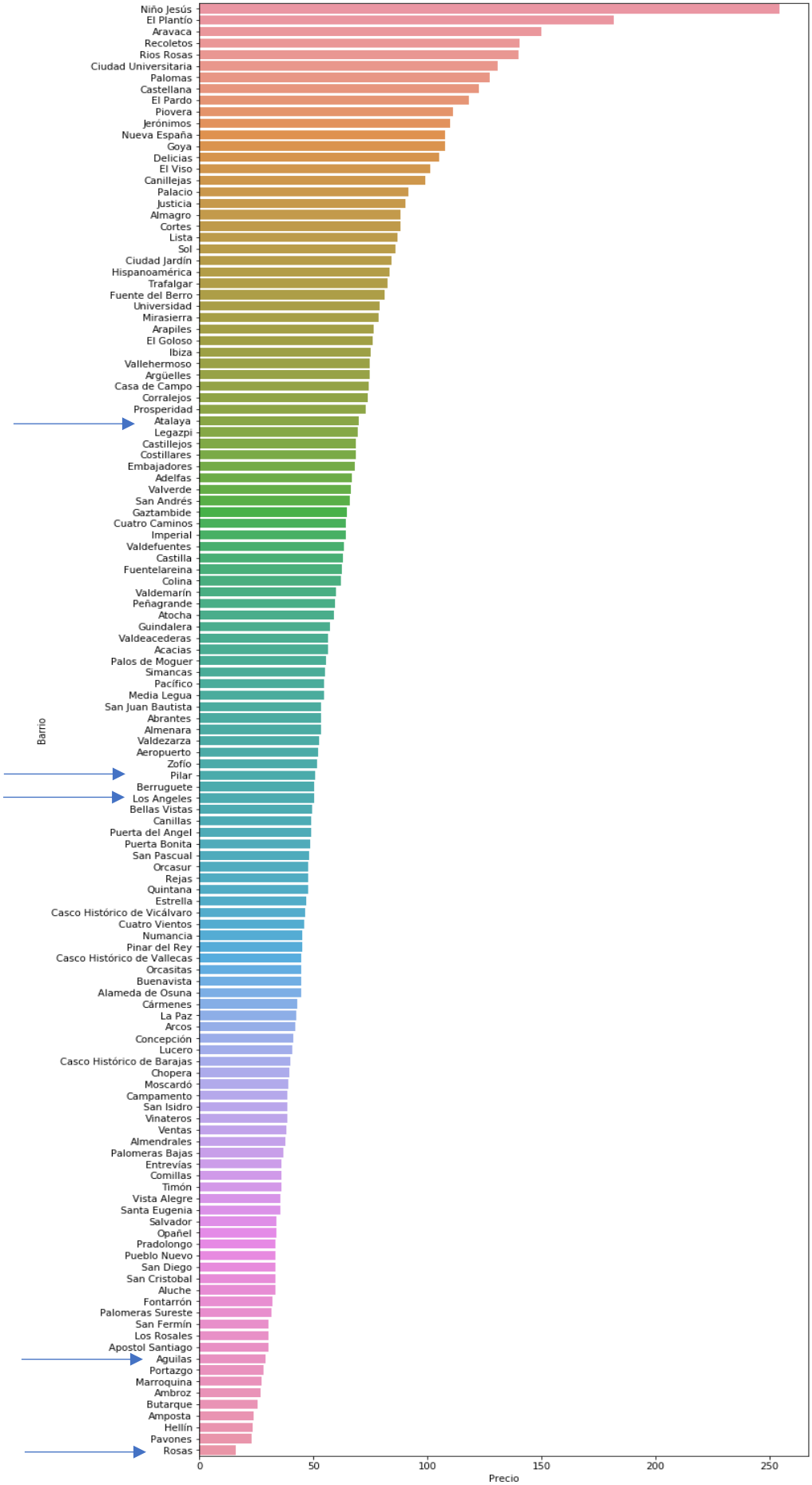
- No hay una gran correlación
- Precios más altos tienen mejores valoraciones.
- Por otra parte, un precio bajo no implica necesariamente bajas valoraciones, pues existe una gran oferta en esos rangos de precio.

No debería ser difícil encontrar pisos disponibles con altas valoraciones, independientemente de tu presupuesto, la media de valoraciones de los usuarios se sitúa en un 92 de 100. Y únicamente un 25% de todas las viviendas que se ofertan tienen una calificación inferior a 90 de 100.

Precios según barrios de Madrid

Entre los factores que tienen un mayor impacto en el precio está sin duda la ubicación. En el siguiente gráfico se distinguen las enormes diferencias entre los distintos barrios de Madrid, lo que será sumamente útil a la hora de intentar ahorrar el máximo.

- Los barrios del centro de Madrid tienden a ser los más caros.
- (Los barrios con una flecha, hablaremos de ellos en la siguiente gráfica)

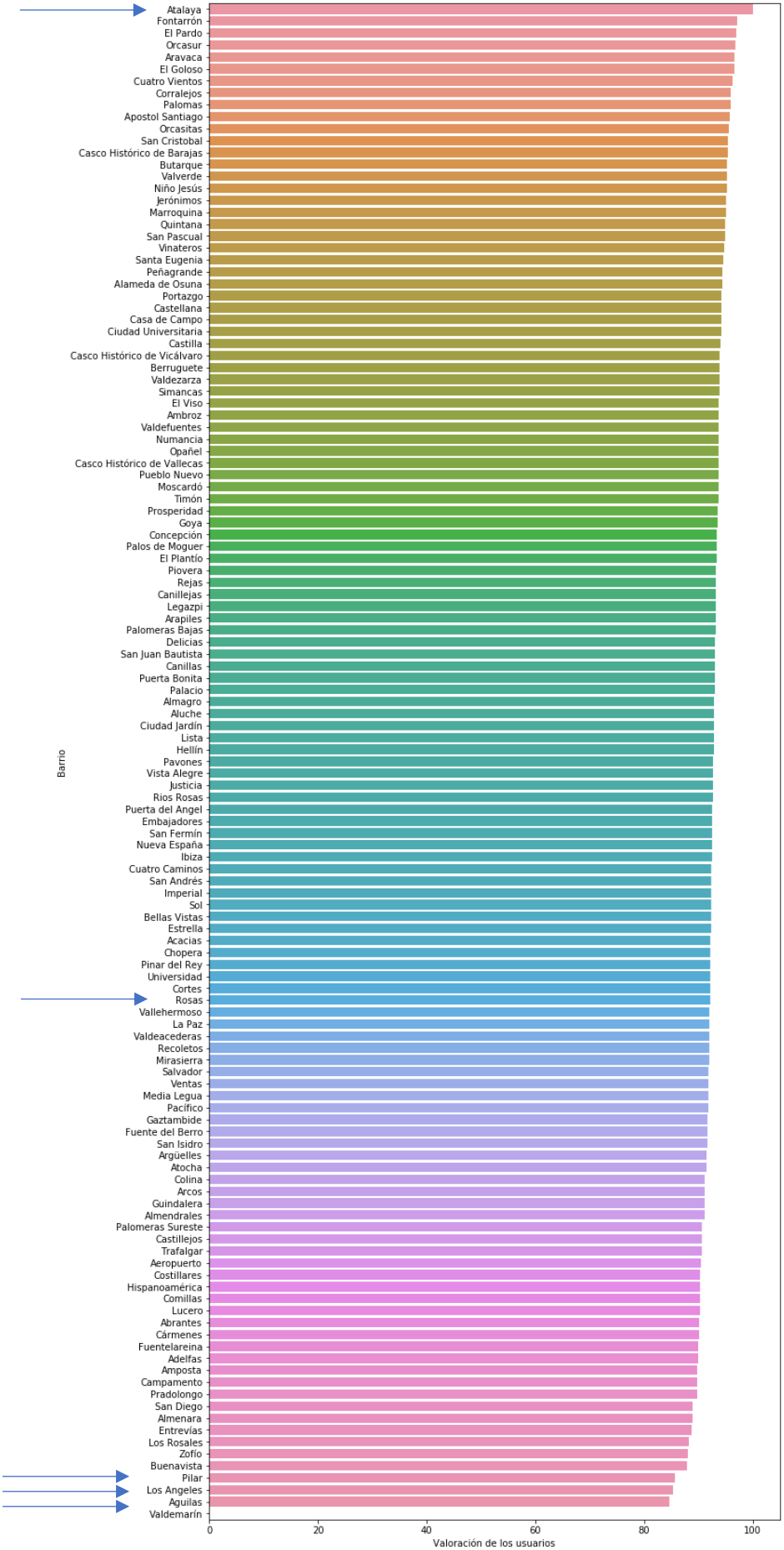

Niño Jesús
El Plantío
Aravaca
Recoletos
Rios Rosas
Ciudad Universitaria
Palomas
Castellana
El Pardo
Piovera
Jerónimos
Nueva España
Goya
Delicias
El Viso
Canillejas
Palacio
Justicia
Almagro
Cortes
Lista
Sol
Ciudad Jardín
Hispanoamérica
Trafalgar
Fuente del Berro
Universidad
Mirasierra
Arapiles
El Goloso
Ibiza
Vallehermoso
Argüelles
Casa de Campo
Corralejos
Prosperidad
Atalaya
Legazpi
Castillejos
Costillares
Embajadores
Adelfas
Valverde
San Andrés
Gaztambide
Cuatro Caminos
Imperial
Valdefuentes
Castilla
Fuentelareina
Colina
Valdemarín
Peñagrande
Atocha
Guindalera
Valdeacederas
Acacias
Palos de Moguer
Simancas
Pacífico
Media Legua
San Juan Bautista
Abrantes
Almenara
Valdezarza
Aeropuerto
Zofío
Pilar
Berruguete
Los Angeles
Bellas Vistas
Canillas
Puerta del Angel
Puerta Bonita
San Pascual
Orcasur
Rejas
Quintana
Estrella
Casco Histórico de Vicálvaro
Cuatro Vientos
Numancia
Pinar del Rey
Casco Histórico de Vallecas
Orcasitas
Buenavista
Alameda de Osuna
Cármenes
La Paz
Arcos
Concepción
Lucero
Casco Histórico de Barajas
Chopera
Moscardó
Campamento
San Isidro
Vinateros
Ventas
Almendrales
Palomeras Bajas
Entrevías
Comillas
Timón
Vista Alegre
Santa Eugenia
Salvador
Opañel
Pradolongo
Pueblo Nuevo
San Diego
San Cristobal
Aluche
Fontarrón
Palomeras Sureste
San Fermín
Los Rosales
Apostol Santiago
Aguilas
Portazgo
Marroquina
Ambroz
Butarque
Amposta
Hellín
Pavones
Rosas
Barrio
0
50
100
150
200
250
Precio

Valoración de los usuarios según barrios de Madrid

Después de ver los precios según la ubicación, sería interesante ver las valoraciones de los usuarios ¿merece la pena pagar coste añadido por la ubicación?

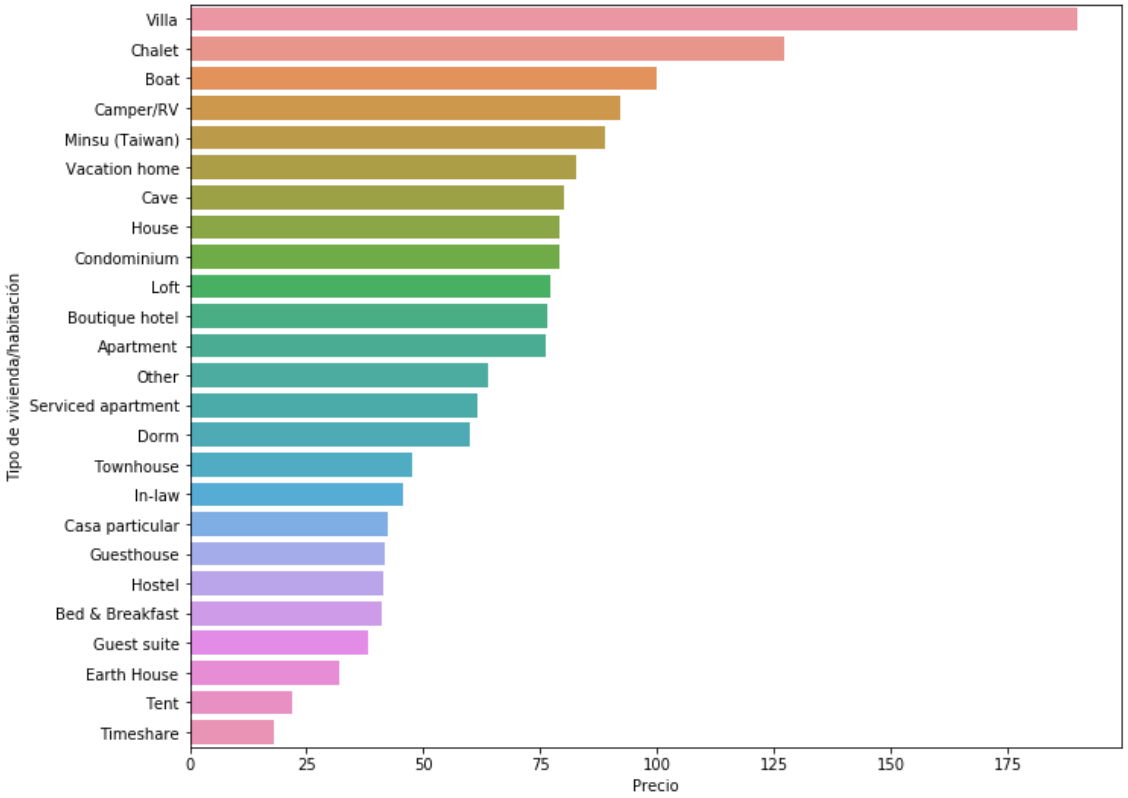
Atendiendo a las valoraciones de los usuarios y dejando de lado otros factores por lo que nos interesaría determinadas ubicaciones (cercanía a puntos de interés, por ejemplo) no hay grandes diferencias entre las valoraciones.

- Barrios del **Pilar**, **Los Ángeles** y **Águilas** tienen las valoraciones más bajas. Son los barrios que menos compensa alquilar atendiendo únicamente a la valoración de los usuarios como factor decisivo.
- **Pilar** y **Águilas** tienen precios bajos, pero cualquier otro barrio de precio humilde tiene mejores valoraciones.
- **Los Ángeles** ni siquiera está entre los más baratos, se encuentra en la media con respecto a precio del alquiler por lo que es el que menos merecería la pena
- Destacan las valoraciones del barrio de **Atalaya**, sin embargo, este es el barrio más pequeño de Madrid, por lo que su escasa oferta y escaso número de valoraciones son responsables de la diferencia con el resto de los barrios.
- El barrio de **Rosas**, que tienen los precios medios más bajos, tiene una valoración más que decente, y que hacen de él una excelente elección atendiendo exclusivamente a la valoración de los usuarios.


Atalaya
Fontarrón
El Pardo
Orcasur
Aravaca
El Goloso
Cuatro Vientos
Corralejos
Palomas
Apostol Santiago
Orcasitas
San Cristobal
Casco Histórico de Barajas
Butarque
Valverde
Niño Jesús
Jerónimos
Marroquina
Quintana
San Pascual
Vinateros
Santa Eugenia
Peñagrande
Alameda de Osuna
Portazgo
Castellana
Casa de Campo
Ciudad Universitaria
Castilla
Casco Histórico de Vicálvaro
Berruguete
Valdezarza
Simancas
El Viso
Ambroz
Valdefuentes
Numancia
Opañel
Casco Histórico de Vallecas
Pueblo Nuevo
Moscardó
Timón
Prosperidad
Goya
Concepción
Palos de Moguer
El Plantío
Piovera
Rejas
Canillejas
Legazpi
Arapiles
Palomeras Bajas
Delicias
San Juan Bautista
Canillas
Puerta Bonita
Palacio
Almagro
Aluche
Ciudad Jardín
Lista
Hellín
Pavones
Vista Alegre
Justicia
Rios Rosas
Puerta del Angel
Embajadores
San Fermín
Nueva España
Ibiza
Cuatro Caminos
San Andrés
Imperial
Sol
Bellas Vistas
Estrella
Acacias
Chopera
Pinar del Rey
Universidad
Cortes
Rosas
Vallehermoso
La Paz
Valdeacederas
Recoletos
Mirasierra
Salvador
Ventas
Media Legua
Pacífico
Gaztambide
Fuente del Berro
San Isidro
Argüelles
Atocha
Colina
Arcos
Guindalera
Almendrales
Palomeras Sureste
Castillejos
Trafalgar
Aeropuerto
Costillares
Hispanoamérica
Comillas
Lucero
Abrantes
Cármenes
Fuentelareina
Adelfas
Amposta
Campamento
Pradolongo
San Diego
Almenara
Entrevías
Los Rosales
Zofío
Buenavista
Pilar
Los Angeles
Aguilas
Valdemarín
Barrio
0
20
40
60
80
100
Valoración de los usuarios

Precios según tipo de vivienda/habitación alquilada

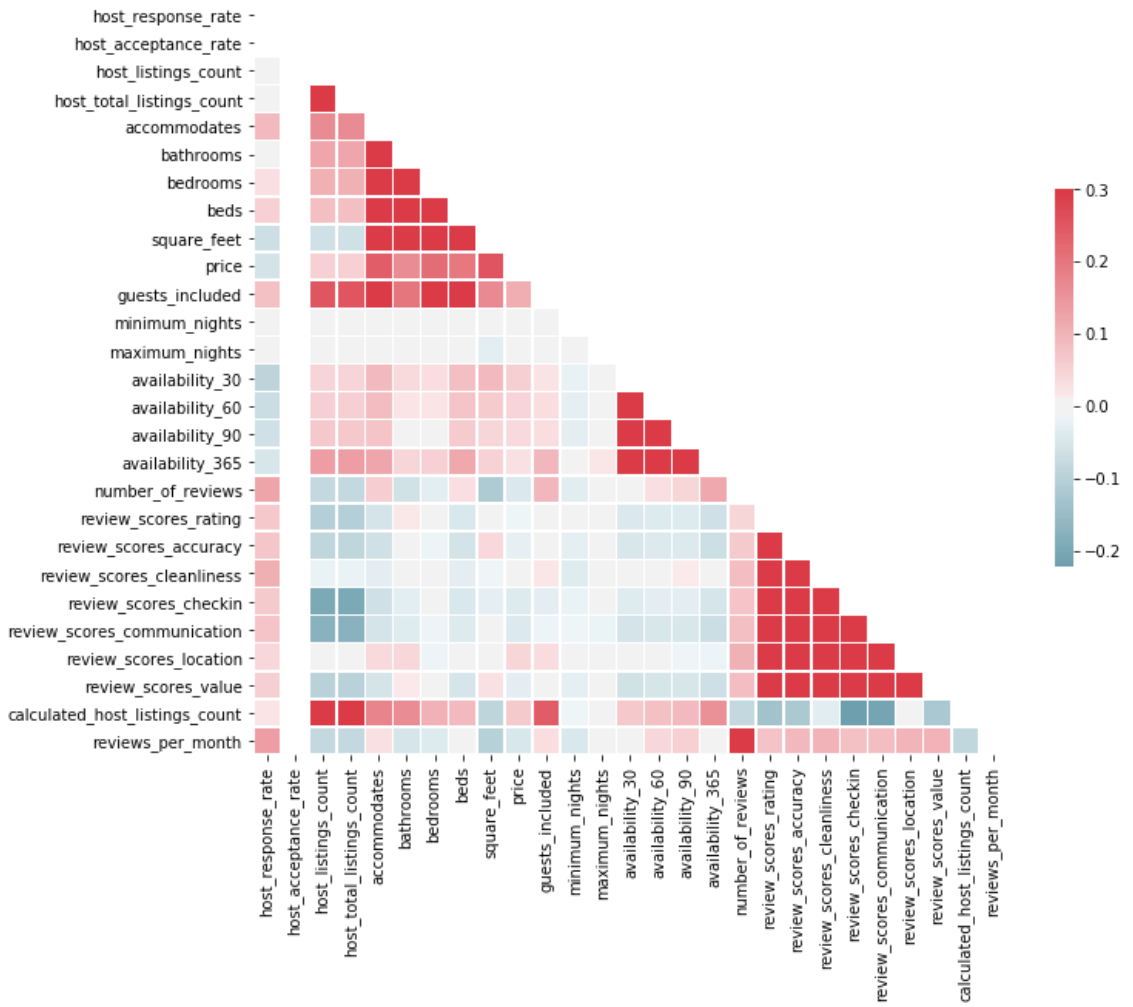
Al igual que con la ubicación, como es lógico el tipo de habitación o vivienda afecta en gran medida al precio. Lo más caro serían las villas, mientras que lo más barato son espacios compartidos.



3. Correlación entre variables numéricas

La correlación de las variables numéricas relevantes presentes en la base de datos de *listings* y mostradas gráficamente en un mapa de calor de correlaciones.

Nos indica aquellas relaciones más fuertes (coloreadas de rojo), y aquellas correlaciones débiles o de ausencia de cualquier tipo de correlación (coloreadas en azul).



Algunos datos interesantes:

- Correlación del precio con características de la vivienda como los metros cuadrados, número de camas, baños.
- Valoración de los usuarios no influye significativamente en el precio, lo que confirma lo que ya habíamos visto en el diagrama de dispersión.
- Sin embargo, paradójicamente la valoración de los usuarios sí se ve condicionada ligeramente por las características del piso como pueden ser el número de baños (que indirectamente suelen encarecer el precio)

4. CODIGO

In [1]:

```
#Importar librerias

import datetime
import pandas as pd
import numpy as np
import seaborn
import matplotlib.pyplot as plt

%matplotlib inline
```

In [2]:

```
#Importar bases de datos
```

In [3]:

```
listings = pd.read_csv(r"C:\Users\Javi\Desktop\proyectoR\AIRBNB MADRID\raw_data\listings.csv")
listings.head()
```

Out[3]:

	id	listing_url	scrape_id	last_scraped	name	sum
0	18628	https://www.airbnb.com/rooms/18628	20180116205114	2018-01-17	Greta Studio Wifi Chueca en Madrid	The ma apa a j
1	19864	https://www.airbnb.com/rooms/19864	20180116205114	2018-01-17	PLAZA MAYOR (wifi, air conditioning)	Cosy studi b clos
2	21512	https://www.airbnb.com/rooms/21512	20180116205114	2018-01-17	Studio in Plaza de España	Studi equ condi vaca
3	21853	https://www.airbnb.com/rooms/21853	20180116205114	2018-01-17	Bright and airy room	We h qui room goo
4	23021	https://www.airbnb.com/rooms/23021	20180116205114	2018-01-17	Elegant Apartment in Spain	

Square

5 rows × 96 columns

In [4]:

```
reviews = pd.read_csv(r"C:\Users\Javi\Desktop\proyectoR\AIRBNB MADRID\raw_data\reviews.csv")
reviews.head()
```

Out[4]:

	listing_id	id	date	reviewer_id	reviewer_name	comments
0	18628	47416	2010-05-29	105188	Angelina	Central location, classic small but comfortabl...
1	18628	125979	2010-10-24	37891	Paul	Gema was charming and helpful. The apartment ...
2	18628	132796	2010-11-03	196963	Melanie	I have nothing negative to say, everything was...
3	18628	2135188	2012-08-29	3001840	Sarah	Small but clean studio at an unbeatable price....
4	18628	2726102	2012-10-26	3698247	Mei Yee	fantastic location. house is small but equipp...

In [5]:

```
calendar = pd.read_csv(r"C:\Users\Javi\Desktop\proyectoR\AIRBNB MADRID\raw_data\calendar.csv")
calendar.head()
```

Out[5]:

	listing_id	date	available	price
0	11200141	2018-01-17	t	$55.00
1	11200141	2018-01-18	t	$55.00
2	11200141	2018-01-19	f	NaN
3	11200141	2018-01-20	f	NaN
4	11200141	2018-01-21	t	$55.00

In [6]:

```
#Cambiar tipos de datos para que podamos operar con ellos que nos interesan
```

In [7]:

```
calendar.date.head()
```

Out[7]:

```
0    2018-01-17
1    2018-01-18
2    2018-01-19
3    2018-01-20
42018-01-21
Name: date, dtype: object
```

In [8]:

```
calendar.dtypes
```

Out[8]:

```
listing_id     int64
date          object
available     object
price         object
dtype: object
```

In [9]:

```
calendar["date"].dtype
```

Out[9]:

```
dtype('O')
```

In [10]:

```
#De str a datetime
calendar["date"] = pd.to_datetime(calendar["date"])
```

In [11]:

```
calendar.date.dtype
```

Out[11]:

```
dtype('<M8[ns]')
```

In [12]:

```
#Borrar signos que impiden hacer operaciones con ciertos numeros
def remove_sign(x, sign):
if type(x) is str:
x= float(x.replace(sign, "").replace(",", ""))
return x
```

In [13]:

```
#borrar símbolo $ del precio y convertir en float
calendar.price = calendar.price.apply(remove_sign, sign="$")
```

In [14]:

```
calendar.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5953880 entries, 0 to 5953879
Data columns (total 4 columns):
listing_id    int64
date          datetime64[ns]
available     object
price         float64
dtypes: datetime64[ns](1), float64(1), int64(1), object(1)
memory usage: 181.7+ MB
```

In [15]:

```
#Hacer lo mismo en listings
```

In [16]:

```
listings.info(verbose=True, null_counts=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16313 entries, 0 to 16312
Data columns (total 96 columns):
id                                16313  non-null int64
listing_url                       16313  non-null object
scrape_id                         16313  non-null int64
last_scraped                      16313  non-null object
name                              16302  non-null object
summary                           15860  non-null object
space                             11798  non-null object
description                       16305  non-null object
experiences_offered               16313  non-null object
neighborhood_overview             10343  non-null object
notes                             6734 non-null object
transit                           10186  non-null object
access                            9057 non-null object
interaction                       9196 non-null object
house_rules                       10189  non-null object
thumbnail_url                     0 non-null float64
medium_url                        0 non-null float64
picture_url                       16313  non-null object
xl_picture_url                    0 non-null float64
host_id                           16313  non-null int64
host_url                          16313  non-null object
host_name                         16284  non-null object
host_since                        16284  non-null object
host_location                     16223  non-null object
host_about                        9965 non-null object
host_response_time                13805  non-null object
host_response_rate                13805  non-null object
host_acceptance_rate              0 non-null float64
host_is_superhost                 16284  non-null object
host_thumbnail_url                16284  non-null object
host_picture_url                  16284  non-null object
host_neighbourhood                11888  non-null object
host_listings_count               16284  non-null float64
host_total_listings_count         16284  non-null float64
host_verifications                16313  non-null object
host_has_profile_pic              16284  non-null object
host_identity_verified            16284  non-null object
street                            16313  non-null object
neighbourhood                     16309  non-null object
neighbourhood_cleansed            16313  non-null object
neighbourhood_group_cleansed      16313  non-null object
city                              16311  non-null object
state                             16230  non-null object
zipcode                           15693  non-null object
market                            16262  non-null object
smart_location                    16313  non-null object
country_code                      16313  non-null object
country                           16312  non-null object
```

```
latitude                          16313   non-null float64
longitude                         16313   non-null float64
is_location_exact                 16313   non-null object
property_type                     16313   non-null object
room_type                         16313   non-null object
accommodates                      16313   non-null int64
bathrooms                         16275   non-null float64
bedrooms                          16308   non-null float64
beds                              16281   non-null float64
bed_type                          16313   non-null object

amenities                         16313   non-null object
square_feet                       446 non-null float64
price                             16313   non-null object
weekly_price                      2879 non-null object
monthly_price                     2703 non-null object
security_deposit                  10095   non-null object
cleaning_fee                      11400   non-null object
guests_included                   16313   non-null int64
extra_people                      16313   non-null object
minimum_nights                    16313   non-null int64
maximum_nights                    16313   non-null int64
calendar_updated                  16313   non-null object
has_availability                  16313   non-null object
availability_30                   16313   non-null int64
availability_60                   16313   non-null int64
availability_90                   16313   non-null int64
availability_365                  16313   non-null int64
calendar_last_scraped             16313   non-null object
number_of_reviews                 16313   non-null int64
first_review                      13261   non-null object
last_review                       13290   non-null object
review_scores_rating              13118   non-null float64
review_scores_accuracy            13105   non-null float64
review_scores_cleanliness         13111   non-null float64
review_scores_checkin             13089   non-null float64
review_scores_communication       13104   non-null float64
review_scores_location            13083   non-null float64
review_scores_value               13082   non-null float64
requires_license                  16313   non-null object
license                           1511 non-null object
jurisdiction_names                0 non-null float64
instant_bookable                  16313   non-null object
is_business_travel_ready          16313   non-null object
cancellation_policy               16313   non-null object
require_guest_profile_picture     16313   non-null object
require_guest_phone_verification  16313   non-null object
calculated_host_listings_count    16313   non-null int64
reviews_per_month                 13261   non-null float64
dtypes: float64(21), int64(13), object(62)
memory usage: 11.9+ MB
```

In [17]:

```
#De str a datetime
listings["host_since"] = pd.to_datetime(listings["host_since"])
```

In [18]:

```
#QuitarSímbolo $ y convertir en float
listings["price"] = listings.price.apply(remove_sign, sign="$")
```

In [19]:

```
#Quitar Símbolo % y convertir en float
listings["host_response_rate"] = listings["host_response_rate"].apply(remove_sign,
sign ="%")
```

In [20]:

```
listings.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16313 entries, 0 to 16312
Data columns (total 96 columns):
id                                16313  non-null int64
listing_url                       16313  non-null object
scrape_id                         16313  non-null int64
last_scraped                      16313  non-null object
name                              16302  non-null object
summary                           15860  non-null object
space                             11798  non-null object
description                       16305  non-null object
experiences_offered               16313  non-null object
neighborhood_overview             10343  non-null object
notes                             6734 non-null object
transit                           10186  non-null object
access                            9057 non-null object
interaction                       9196 non-null object
house_rules                       10189  non-null object
thumbnail_url                     0 non-null float64
medium_url                        0 non-null float64
picture_url                       16313  non-null object
xl_picture_url                    0 non-null float64
host_id                           16313  non-null int64
host_url                          16313  non-null object
host_name                         16284  non-null object
host_since                        16284  non-null datetime64[ns]
host_location                     16223  non-null object
host_about                        9965 non-null object
host_response_time                13805  non-null object
host_response_rate                13805  non-null float64
host_acceptance_rate              0 non-null float64
host_is_superhost                 16284  non-null object
host_thumbnail_url                16284  non-null object
host_picture_url                  16284  non-null object
host_neighbourhood                11888  non-null object
host_listings_count               16284  non-null float64
host_total_listings_count         16284  non-null float64
host_verifications                16313  non-null object
host_has_profile_pic              16284  non-null object
host_identity_verified            16284  non-null object
street                            16313  non-null object
neighbourhood                     16309  non-null object
neighbourhood_cleansed            16313  non-null object
```

```
neighbourhood_group_cleansed      16313  non-null object
city                              16311  non-null object
state                             16230  non-null object
zipcode                           15693  non-null object
market                            16262  non-null object
smart_location                    16313  non-null object
country_code                      16313  non-null object
country                           16312  non-null object
latitude                          16313  non-null float64
longitude                         16313  non-null float64
is_location_exact                 16313  non-null object
property_type                     16313  non-null object
room_type                         16313  non-null object
accommodates                      16313  non-null int64
bathrooms                         16275  non-null float64
bedrooms                          16308  non-null float64
beds                              16281  non-null float64
bed_type                          16313  non-null object

amenities                         16313  non-null object
square_feet                       446 non-null float64
price                             16313  non-null float64
weekly_price                      2879 non-null object
monthly_price                     2703 non-null object
security_deposit                  10095  non-null object
cleaning_fee                      11400  non-null object
guests_included                   16313  non-null int64
extra_people                      16313  non-null object
minimum_nights                    16313  non-null int64
maximum_nights                    16313  non-null int64
calendar_updated                  16313  non-null object
has_availability                  16313  non-null object
availability_30                   16313  non-null int64
availability_60                   16313  non-null int64
availability_90                   16313  non-null int64
availability_365                  16313  non-null int64
calendar_last_scraped             16313  non-null object
number_of_reviews                 16313  non-null int64
first_review                      13261  non-null object
last_review                       13290  non-null object
review_scores_rating              13118  non-null float64
review_scores_accuracy            13105  non-null float64
review_scores_cleanliness         13111  non-null float64
review_scores_checkin             13089  non-null float64
review_scores_communication       13104  non-null float64
review_scores_location            13083  non-null float64
review_scores_value               13082  non-null float64
requires_license                  16313  non-null object
license                           1511 non-null object
jurisdiction_names                0 non-null float64
instant_bookable                  16313  non-null object
is_business_travel_ready          16313  non-null object
cancellation_policy               16313  non-null object
require_guest_profile_picture     16313  non-null object
require_guest_phone_verification  16313  non-null object
calculated_host_listings_count    16313  non-null int64
reviews_per_month                 13261  non-null float64
dtypes: datetime64[ns](1), float64(23), int64(13), object(59)
```

```
memory usage: 11.9+ MB
```

In [21]:

```
#Cambiar el formato de la fecha, del americano al formato europeo
#pd.to_datetime(calendar["date"].dt.strftime("%d-%m-%Y"))
```

In [ ]:

```

```

In [22]:

```
#Patrón de disponibilidad y precios
```

In [23]:

```
#Oferta de casas diaria
oferta_viviendas = calendar.groupby("date").apply(lambda x: x.notnull().sum())[["price"
]]

#cambiar nombre columnas
oferta_viviendas = oferta_viviendas.rename(columns={"price":"Oferta disponible"})
```

In [24]:

```
oferta_viviendas.head()
```

Out[24]:

date	Oferta disponible
2018-01-17	2894
2018-01-18	4003
2018-01-19	3144
2018-01-20	3222
2018-01-21	5873

In [25]:

```
#Precios medios
calendar_notnull= calendar[calendar.price.notnull()]
precio_medio_diario = calendar_notnull.groupby("date").mean()[["price"]]
```

In [26]:

```
#cambiar nombre columnas
precio_medio_diario = precio_medio_diario.rename(columns ={"price":"Precio vivienda med
io"})
```

In [27]:

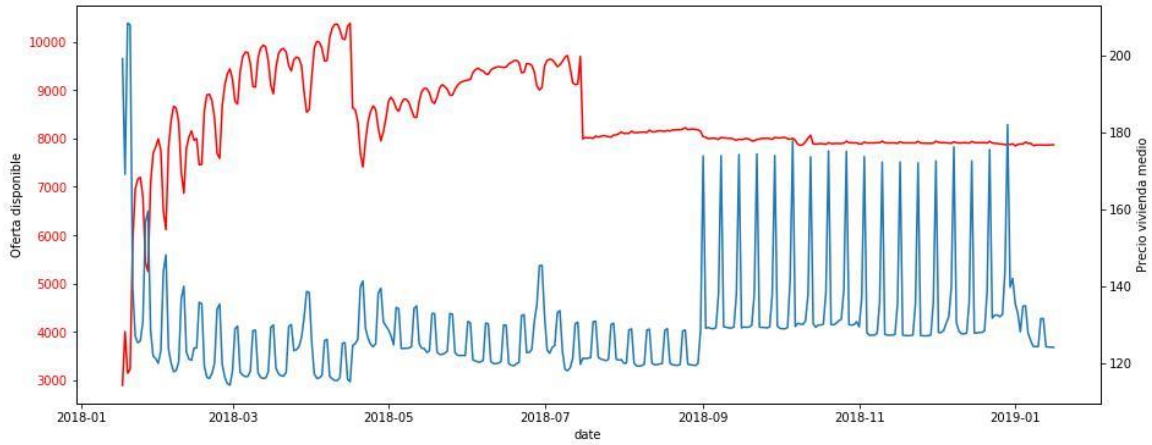
```
precio_medio_diario.head()
```

Out[27]:

date	Precio vivienda medio
2018-01-17	199.054596
2018-01-18	168.999251
2018-01-19	208.280852
2018-01-20	207.864991
2018-01-21	139.568364

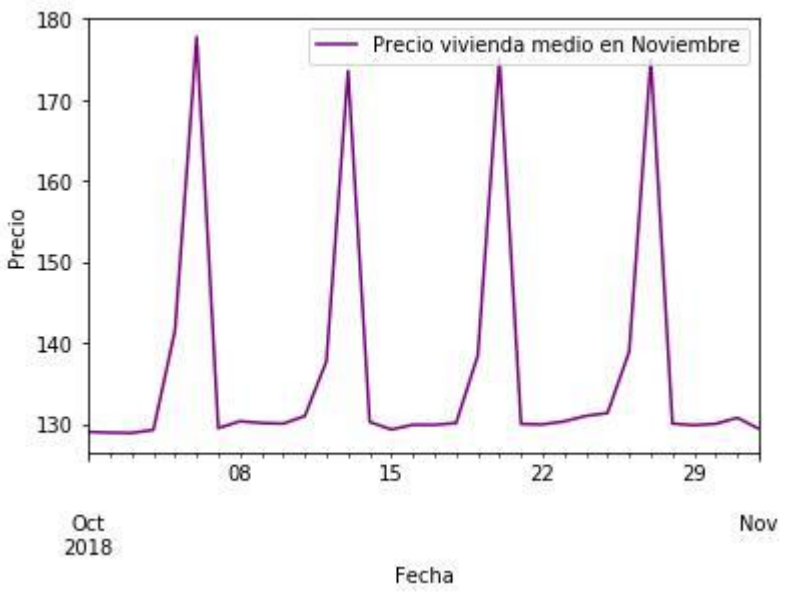
In [29]:

```
#Grafico casas disponibles y precio medio
f, ax =plt.subplots(figsize=(15,6))
plt1 = seaborn.lineplot(x = oferta_viviendas.index, y= "Oferta disponible",
data = oferta_viviendas, color= "r", legend=False)
for tl in ax.get_yticklabels():
tl.set_color("r")

ax2= ax.twinx()
plt2= seaborn.lineplot(x = precio_medio_diario.index,
y= "Precio vivienda medio", data= precio_medio_diario, ax= ax2,
linestyle=":", legend=False)
```



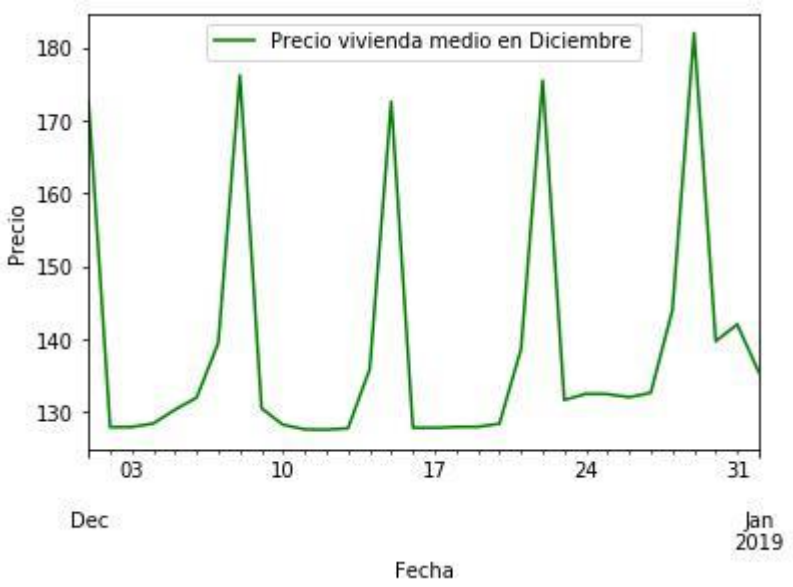
In [30]:

```
#Precio medio del alquiler en octubre de 2018
pmd_oct_18 = precio_medio_diario["2018-10-01":"2018-11-01"]
plot_pmd_oct_18= pmd_oct_18.plot(color="purple")
plot_pmd_oct_18.set_xlabel("Fecha")
plot_pmd_oct_18.set_ylabel("Precio")
plot_pmd_oct_18.legend(["Precio vivienda medio en Noviembre"],loc="upper right")
plt.show()
```



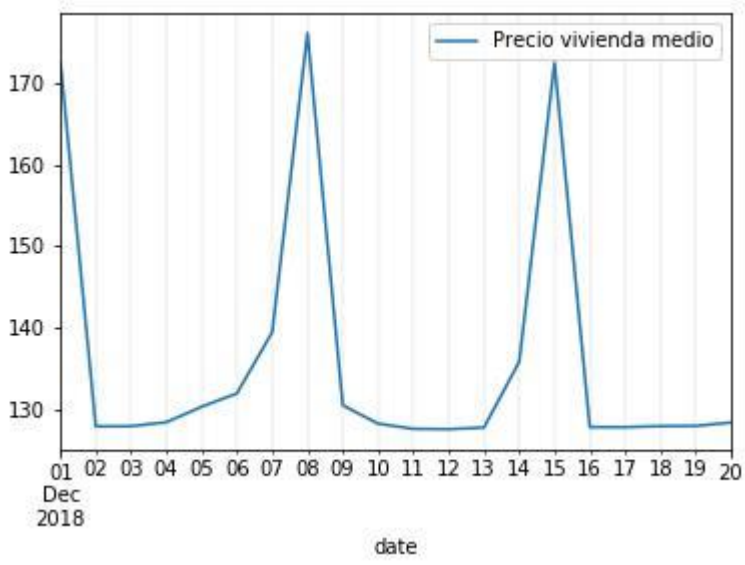
In [31]:

```
#Precio medio del alquiler en diciembre de 2018
pmd_dic_18 = precio_medio_diario["2018-12-01":"2019-01-01"]
plot_pmd_dic_18=pmd_dic_18.plot(color="green")
plot_pmd_dic_18.set_xlabel("Fecha")
plot_pmd_dic_18.set_ylabel("Precio")
plot_pmd_dic_18.legend(["Precio vivienda medio en Diciembre"],loc="upper center")
plt.show()
```



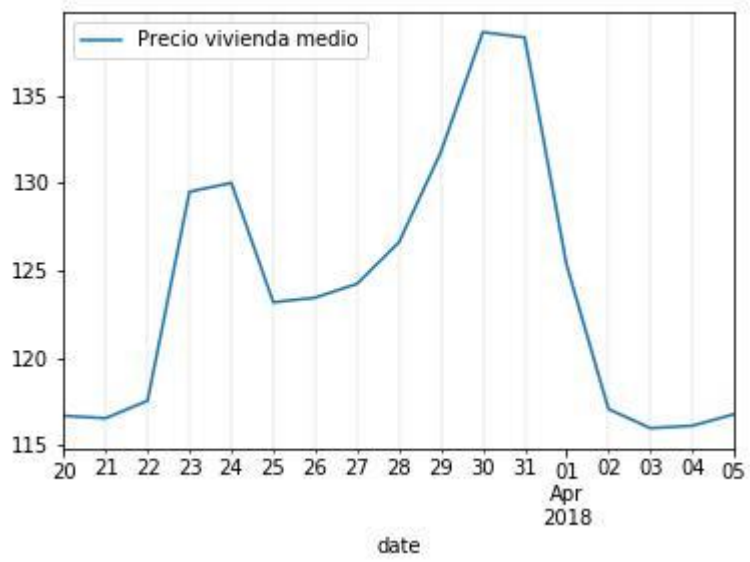
In [32]:

```
#Precio medio en una semana de Diciembre
pmd_dic_finde_2018 = precio_medio_diario["2018-12-01":"2018-12-20"]
plot_pmd_dic_finde_18 =pmd_dic_finde_2018.plot()
plot_pmd_dic_finde_18.xaxis.grid(True, which='minor',
linestyle='-', linewidth=0.25)
```



In [33]:

```
pmd_semanasanta_18 = precio_medio_diario["2018-03-20":"2018-04-05"]
plot_pmd_semanasanta_18 =pmd_semanasanta_18.plot()
plot_pmd_semanasanta_18.xaxis.grid(True, which='minor',
linestyle='-', linewidth=0.25)
```



In [34]:

```
# Tiempo del año en el que están disponible las viviendas
```

In [35]:

```
tiempo_medio_disponible = calendar.groupby("listing_id").apply(lambda x: x.notnull().me
an())[["price"]]
tiempo_medio_disponible = tiempo_medio_disponible.rename({"price":"available_ratio"}, a
xis="columns")
```

In [36]:

```
#Viviendas disponibles menos de un 10% de todo el año
len(tiempo_medio_disponible[tiempo_medio_disponible.available_ratio < 0.10])
```

Out[36]:

3496

In [37]:

```
len(tiempo_medio_disponible)
```

Out[37]:

16312

In [38]:

```
#Viviendas disponibles más de un 90% de todo el año
len(tiempo_medio_disponible[tiempo_medio_disponible.available_ratio > 0.90])
```

Out[38]:

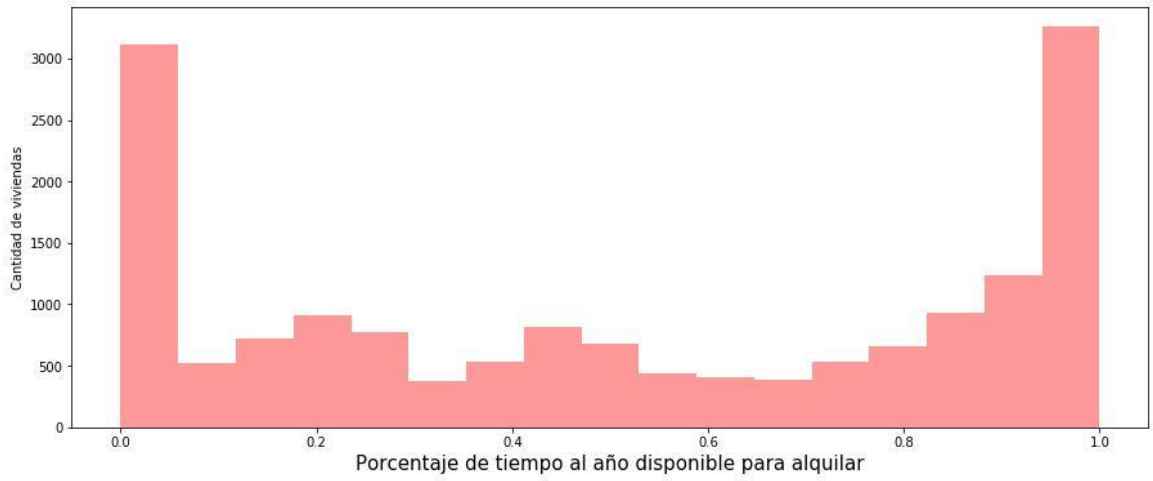
4172

In [39]:

```
f, ax =plt.subplots(figsize=(15,6))
plot_tmd =seaborn.distplot(tiempo_medio_disponible, kde=False, color="red")
plot_tmd.set_xlabel("Porcentaje de tiempo al año disponible para alquilar", fontsize=15
)
plot_tmd.set_ylabel("Cantidad de viviendas", fontsize=10)
```

Out[39]:

```
Text(0, 0.5, 'Cantidad de viviendas')
```



In [40]:

```
#Comprobar si a partir de septiembre existen cambios en la disponibildiad de las vivien
das
```

In [41]:

```
calendar_sept_dic = calendar.loc[(calendar["date"] >= "2018-01-17") & (calendar["date"]
<= "2019-01-01")]
```

In [42]:

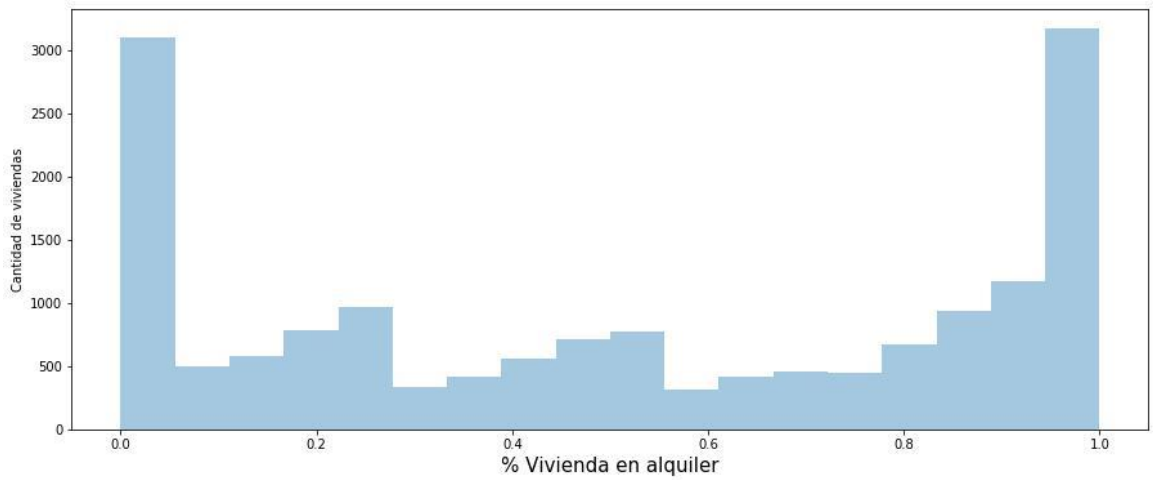
```
tiempo_medio_disponible = calendar_sept_dic.groupby("listing_id").apply(lambda x: x.not
null().mean())[["price"]]
tiempo_medio_disponible = tiempo_medio_disponible.rename({"price":"available_ratio"}, a
xis="columns")
```

In [43]:

```
f, ax =plt.subplots(figsize=(15,6))
ax =seaborn.distplot(tiempo_medio_disponible, kde=False)
ax.set_xlabel("% Vivienda en alquiler", fontsize=15)
ax.set_ylabel("Cantidad de viviendas", fontsize=10)
```

Out[43]:

```
Text(0, 0.5, 'Cantidad de viviendas')
```



In [44]:

```
len(tiempo_medio_disponible[tiempo_medio_disponible.available_ratio < 0.10])
```

Out[44]:

```
3491
```

In [45]:

```
#Crear categorías para los precios
#Tendencia de los precios
calendar_notnull.price.head()
```

Out[45]:

```
0    55.0
1    55.0
4    55.0
5    55.0
655.0
Name: price, dtype: float64
```

In [46]:

```
listings.price.describe()
```

Out[46]:

```
count    16313.000000
mean        75.069270
std        144.492845
min          8.000000
25%         35.000000
50%         59.000000
75%         88.000000
max       9000.000000
Name: price, dtype: float64
```

In [47]:

```
#usar como precio bajo el primer cuartil (25%), usar como precio alto el tercer cuartil
(75%)
def cat_precio(x, precio_bajo=35, precio_alto = 88):
if x<=precio_bajo:
x="Precio Bajo"
elif x>= precio_alto:
x="Precio Alto"

else:
x="Precio Medio"
return x
listings["cat_precio"] = listings.price.apply(cat_precio)
```

In [48]:

```
#Nuea columna con las 3 categorias de precios
precio_cal = listings[["id","price","review_scores_rating","number_of_reviews"]].dropna
()
precio_cal.head()
```

Out[48]:

	id	price	review_scores_rating	number_of_reviews
0	18628	54.0	89.0	37
1	19864	65.0	91.0	56
2	21512	40.0	79.0	36
3	21853	17.0	90.0	26
4	23021	90.0	80.0	55

In [49]:

```
precio_cal.describe()
```

Out[49]:

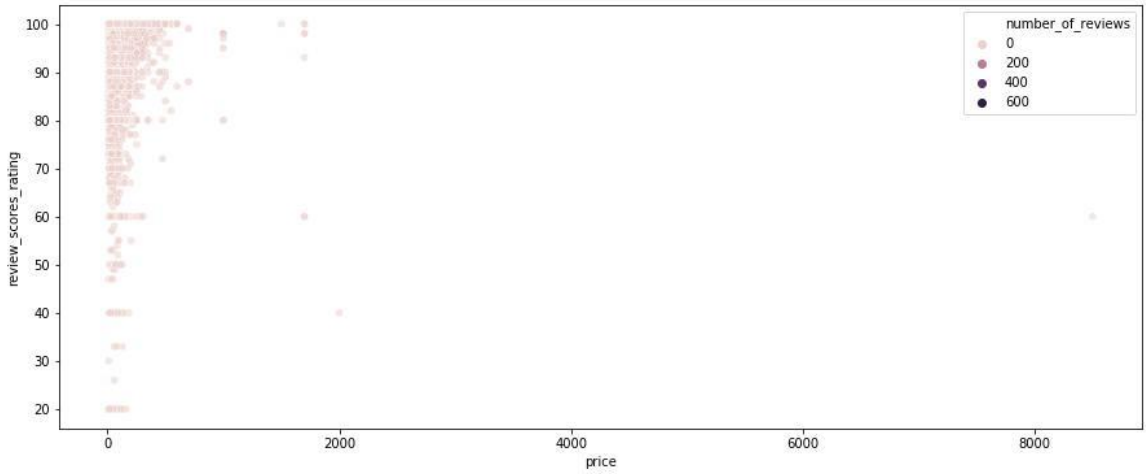
	id	price	review_scores_rating	number_of_reviews
count	1.311800e+04	13118.000000	13118.000000	13118.000000
mean	1.336239e+07	69.445114	92.426666	34.292804
std	6.686888e+06	100.457766	8.857210	48.666034
min	1.862800e+04	8.000000	20.000000	1.000000
25%	7.819681e+06	35.000000	90.000000	4.000000
50%	1.500547e+07	58.000000	95.000000	15.000000
75%	1.911964e+07	85.000000	98.000000	43.000000
max	2.268655e+07	8500.000000	100.000000	488.000000

In [50]:

```
#Relación precio y calificación
f, ax = plt.subplots(figsize=(15, 6))
seaborn.scatterplot(x='price',y='review_scores_rating',hue='number_of_reviews',alpha=0.
5,data=precio_cal)
```

Out[50]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29e23f5efd0>
```



In [51]:

```
#Quitamos outliers
```

In [52]:

```
precio_cal_wo_outl = precio_cal
```

In [53]:

```
def remove_outlier(df_in, col_name):
q1 = df_in[col_name].quantile(0.25)
q3 = df_in[col_name].quantile(0.75)
iqr = q3-q1 #Interquartile range
fence_low      = q1-1.5*iqr
fence_high = q3+1.5*iqr
df_out = df_in.loc[(df_in[col_name] > fence_low) & (df_in[col_name] < fence_high)]
return df_out
```

In [54]:

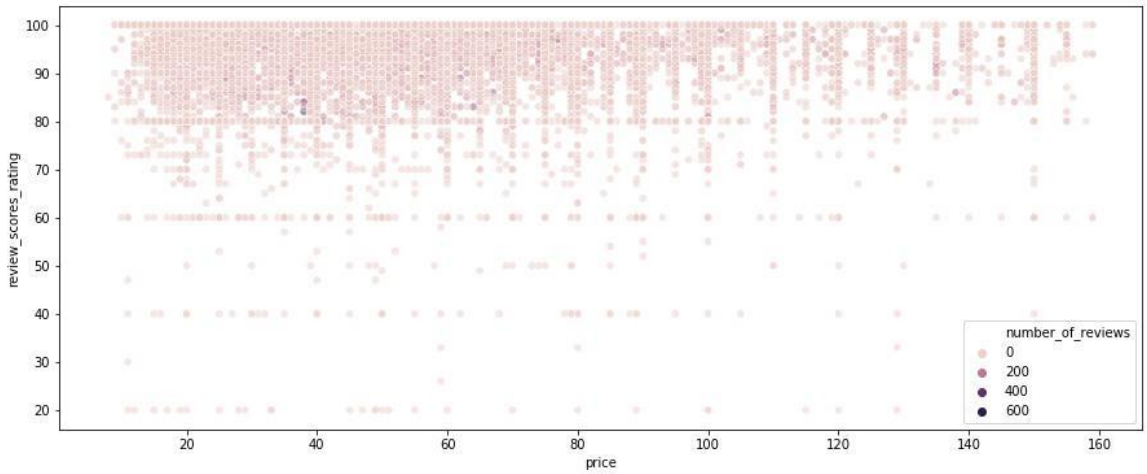
```
precio_cal_wo_outl =remove_outlier(precio_cal_wo_outl, "price")
```

In [55]:

```
f, ax = plt.subplots(figsize=(15, 6))
seaborn.scatterplot(x='price',y='review_scores_rating',hue='number_of_reviews',alpha=0.
5,data=precio_cal_wo_outl)
```

Out[55]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29e23ae9278>
```



In [ ]:

```

```

In [56]:

```
#Algunas de las variables que tienen un peso significativo en el precio
```

In [57]:

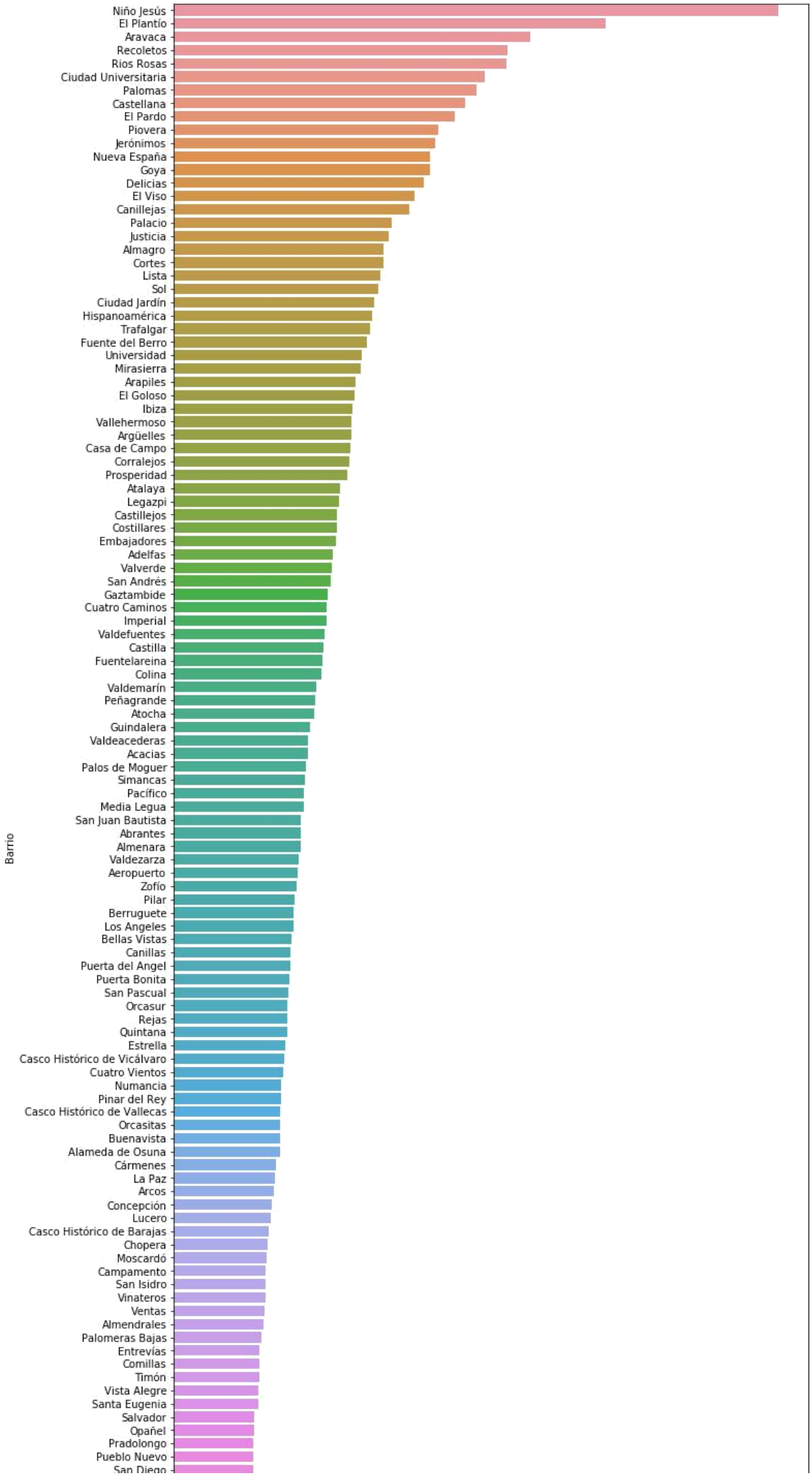
```
# Quitar aquellos datos que no presenten ningún valor en el precio
listings_precios = listings[listings.price.notnull()]
```

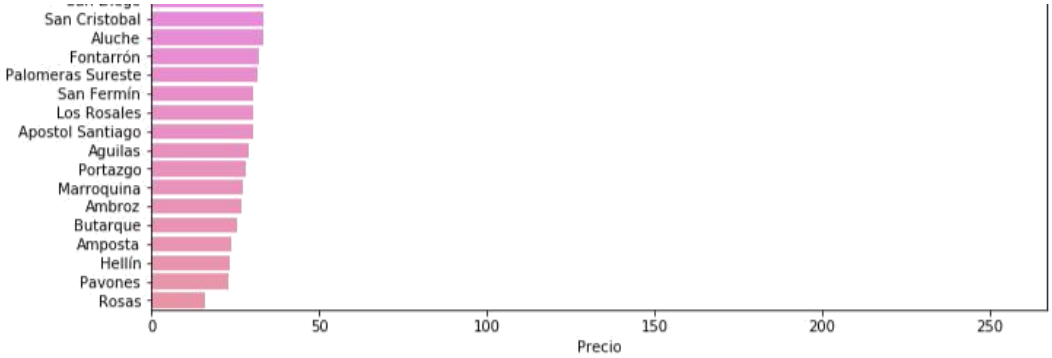
In [58]:

```
#Precios según barrio de Madrid
def plot_precio_barrio (colname, listings = listings, fig_row_size=11, fig_col_size=30
):
price_col=listings_precios.groupby(colname).mean()[["price"]]
price_col.reset_index(inplace=True)
f, ax = plt.subplots(figsize=(fig_row_size, fig_col_size))
seaborn.barplot(y = colname, x="price", data= price_col.sort_values(by="price", asc
ending=False))
ax.set_xlabel(xlabel="Precio")
ax.set_ylabel(ylabel="Barrio")
```

In [59]:

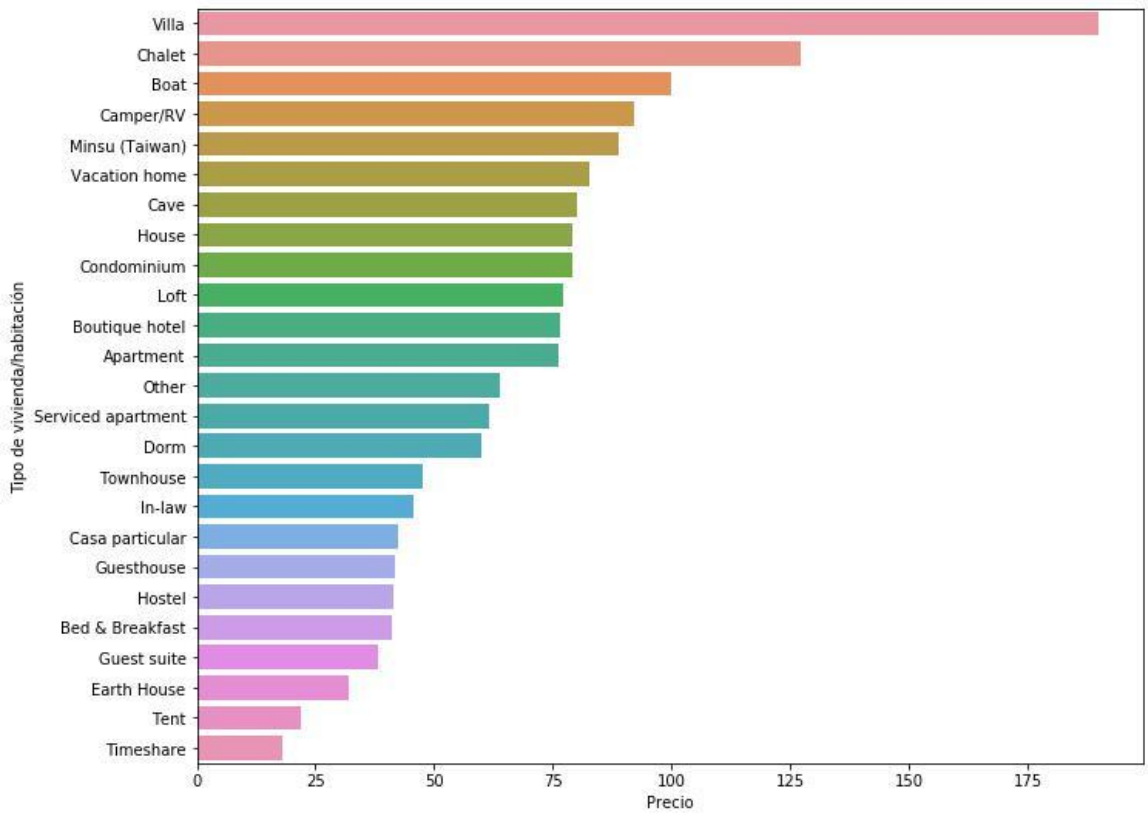
```
plot_precio_barrio("neighbourhood_cleansed", listings = listings_precios)
```


Barrio
Niño Jesús
El Plantío
Aravaca
Recoletos
Rios Rosas
Ciudad Universitaria
Palomas
Castellana
El Pardo
Piovera
Jerónimos
Nueva España
Goya
Delicias
El Viso
Canillejas
Palacio
Justicia
Almagro
Cortes
Lista
Sol
Ciudad Jardín
Hispanoamérica
Trafalgar
Fuente del Berro
Universidad
Mirasierra
Arapiles
El Goloso
Ibiza
Vallehermoso
Argüelles
Casa de Campo
Corralejos
Prosperidad
Atalaya
Legazpi
Castillejos
Costillares
Embajadores
Adelfas
Valverde
San Andrés
Gaztambide
Cuatro Caminos
Imperial
Valdefuentes
Castilla
Fuentelareina
Colina
Valdemarín
Peñagrande
Atocha
Guindalera
Valdeacederas
Acacias
Palos de Moguer
Simancas
Pacífico
Media Legua
San Juan Bautista
Abrantes
Almenara
Valdezarza
Aeropuerto
Zofío
Pilar
Berruguete
Los Angeles
Bellas Vistas
Canillas
Puerta del Angel
Puerta Bonita
San Pascual
Orcasur
Rejas
Quintana
Estrella
Casco Histórico de Vicálvaro
Cuatro Vientos
Numancia
Pinar del Rey
Casco Histórico de Vallecas
Orcasitas
Buenavista
Alameda de Osuna
Cármenes
La Paz
Arcos
Concepción
Lucero
Casco Histórico de Barajas
Chopera
Moscardó
Campamento
San Isidro
Vinateros
Ventas
Almendrales
Palomeras Bajas
Entrevías
Comillas
Timón
Vista Alegre
Santa Eugenia
Salvador
Opañel
Pradolongo
Pueblo Nuevo
San Diego



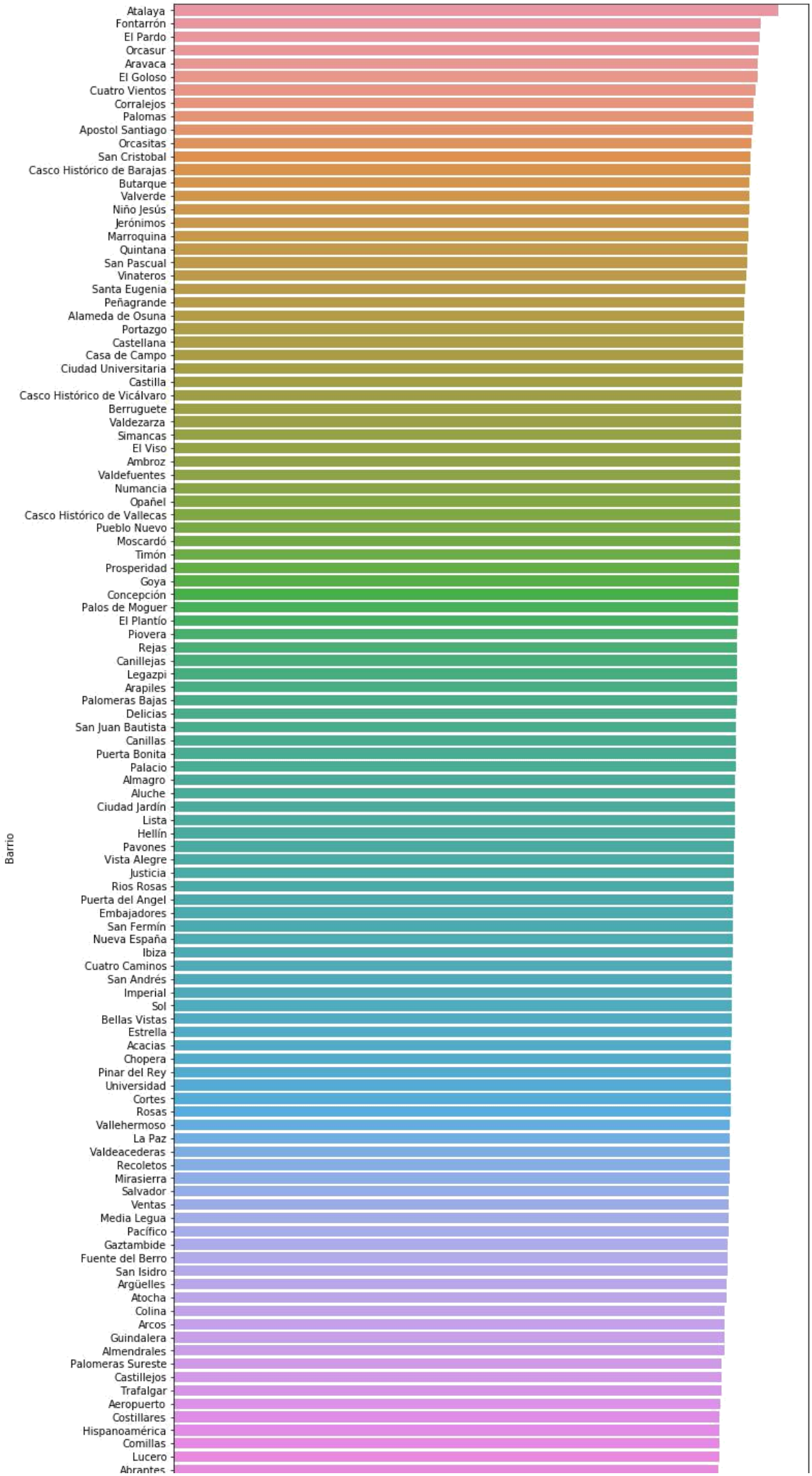
In [60]:

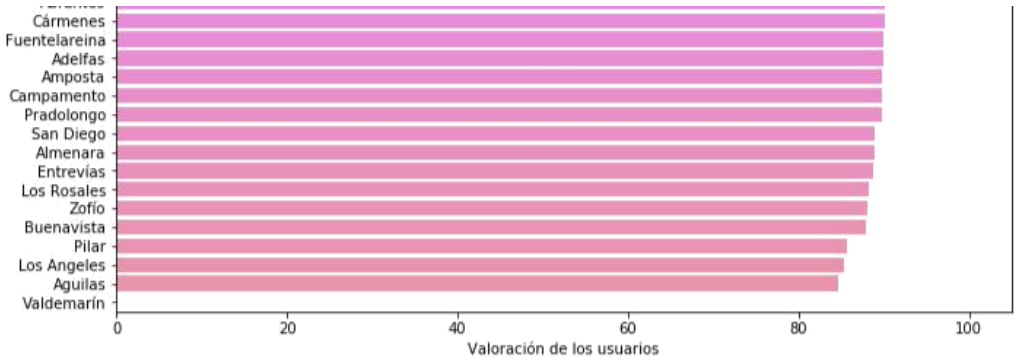
```
#Precio según vivenda/habitación alquilada
def plot_precio_tipov (colname, listings = listings, fig_row_size=11, fig_col_size=9):
price_col=listings_precios.groupby(colname).mean()[["price"]]
price_col.reset_index(inplace=True)
f, ax = plt.subplots(figsize=(fig_row_size, fig_col_size))
seaborn.barplot(y = colname, x="price", data= price_col.sort_values(by="price", asc
ending=False))
ax.set_xlabel(xlabel="Precio")
ax.set_ylabel(ylabel="Tipo de vivienda/habitación")

plot_precio_tipov("property_type", listings = listings_precios)
```



In [61]:

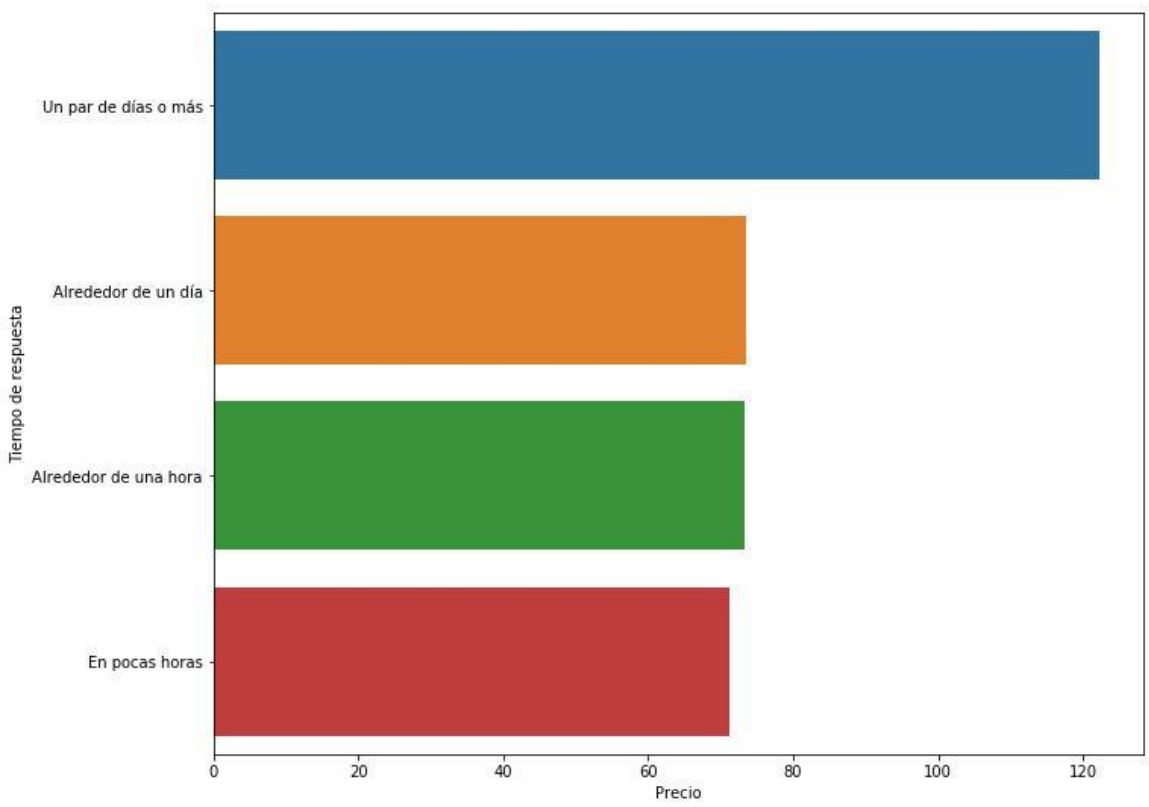
```
#Precio según vivenda/habitación alquilada
def plot_valor_tipov (colname, listings = listings, fig_row_size=11, fig_col_size=30):
price_valor=listings_precios.groupby(colname).mean()[["review_scores_rating"]]
price_valor.reset_index(inplace=True)
f, ax = plt.subplots(figsize=(fig_row_size, fig_col_size))
seaborn.barplot(y = colname, x="review_scores_rating", data= price_valor.sort_value
s(by="review_scores_rating", ascending=False))
ax.set_xlabel(xlabel="Valoración de los usuarios")
ax.set_ylabel(ylabel="Barrio")

plot_valor_tipov("neighbourhood_cleansed", listings = listings_precios)
```


Barrio
Atalaya
Fontarrón
El Pardo
Orcasur
Aravaca
El Goloso
Cuatro Vientos
Corralejos
Palomas
Apostol Santiago
Orcasitas
San Cristobal
Casco Histórico de Barajas
Butarque
Valverde
Niño Jesús
Jerónimos
Marroquina
Quintana
San Pascual
Vinateros
Santa Eugenia
Peñagrande
Alameda de Osuna
Portazgo
Castellana
Casa de Campo
Ciudad Universitaria
Castilla
Casco Histórico de Vicálvaro
Berruguete
Valdezarza
Simancas
El Viso
Ambroz
Valdefuentes
Numancia
Opañel
Casco Histórico de Vallecas
Pueblo Nuevo
Moscardó
Timón
Prosperidad
Goya
Concepción
Palos de Moguer
El Plantío
Piovera
Rejas
Canillejas
Legazpi
Arapiles
Palomeras Bajas
Delicias
San Juan Bautista
Canillas
Puerta Bonita
Palacio
Almagro
Aluche
Ciudad Jardín
Lista
Hellín
Pavones
Vista Alegre
Justicia
Rios Rosas
Puerta del Angel
Embajadores
San Fermín
Nueva España
Ibiza
Cuatro Caminos
San Andrés
Imperial
Sol
Bellas Vistas
Estrella
Acacias
Chopera
Pinar del Rey
Universidad
Cortes
Rosas
Vallehermoso
La Paz
Valdeacederas
Recoletos
Mirasierra
Salvador
Ventas
Media Legua
Pacífico
Gaztambide
Fuente del Berro
San Isidro
Argüelles
Atocha
Colina
Arcos
Guindalera
Almendrales
Palomeras Sureste
Castillejos
Trafalgar
Aeropuerto
Costillares
Hispanoamérica
Comillas
Lucero
Abrantes


Cármenes
Fuentelareina
Adelfas
Amposta
Campamento
Pradolongo
San Diego
Almenara
Entrevías
Los Rosales
Zofío
Buenavista
Pilar
Los Angeles
Aguilas
Valdemarín
0
20
40
60
80
100
Valoración de los usuarios

In [62]:

```
#Relación entre el precio del alquiler y la velocidad de respuesta del host cuando reci
be un mensaje

mod_listings_precios= listings_precios
mod_listings_precios["host_response_time"] = mod_listings_precios["host_response_time" ].replace({"a
few days or more":"Un par de días o más",

"within a day":"Alrededor de un día",

"within an hour":"Alrededor de una hora",

"within a few hours":"En pocas horas"})
def plot_precio_respuesta (colname, listings = listings, fig_row_size=11, fig_col_size=
9):
price_col=listings_precios.groupby(colname).mean()[["price"]]
price_col.reset_index(inplace=True)
f, ax = plt.subplots(figsize=(fig_row_size, fig_col_size))
seaborn.barplot(y = colname, x="price", data= price_col.sort_values(by="price", asc
ending=False))
ax.set_xlabel(xlabel="Precio")
ax.set_ylabel(ylabel="Tiempo de respuesta")

plot_precio_respuesta("host_response_time", listings = listings_precios)
```



In [ ]:

In [63]:

```
#Buscar correlaciones numericas entre los valores de listings

listings_precios_num = listings_precios.select_dtypes(include=["float64","int"])
listings_precios.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 16313 entries, 0 to 16312
Data columns (total 97 columns):
id                                16313  non-null int64
listing_url                       16313  non-null object
scrape_id                         16313  non-null int64
last_scraped                      16313  non-null object
name                              16302  non-null object
summary                           15860  non-null object
space                             11798  non-null object
description                       16305  non-null object
experiences_offered               16313  non-null object
neighborhood_overview             10343  non-null object
notes                             6734 non-null object
transit                           10186  non-null object
access                            9057 non-null object
interaction                       9196 non-null object
house_rules                       10189  non-null object
thumbnail_url                     0 non-null float64
medium_url                        0 non-null float64
picture_url                       16313  non-null object
xl_picture_url                    0 non-null float64
host_id                           16313  non-null int64
host_url                          16313  non-null object
host_name                         16284  non-null object
host_since                        16284  non-null datetime64[ns]
host_location                     16223  non-null object
host_about                        9965 non-null object
host_response_time                13805  non-null object
host_response_rate                13805  non-null float64
host_acceptance_rate              0 non-null float64
host_is_superhost                 16284  non-null object
host_thumbnail_url                16284  non-null object
host_picture_url                  16284  non-null object
host_neighbourhood                11888  non-null object
host_listings_count               16284  non-null float64
host_total_listings_count         16284  non-null float64
host_verifications                16313  non-null object
host_has_profile_pic              16284  non-null object
host_identity_verified            16284  non-null object
street                            16313  non-null object
neighbourhood                     16309  non-null object
neighbourhood_cleansed            16313  non-null object
neighbourhood_group_cleansed      16313  non-null object
city                              16311  non-null object
state                             16230  non-null object
zipcode                           15693  non-null object
market                            16262  non-null object
smart_location                    16313  non-null object
country_code                      16313  non-null object
```

```
country                           16312  non-null object
latitude                          16313  non-null float64
longitude                         16313  non-null float64
is_location_exact                 16313  non-null object
property_type                     16313  non-null object
room_type                         16313  non-null object
accommodates                      16313  non-null int64
bathrooms                         16275  non-null float64
bedrooms                          16308  non-null float64
beds                              16281  non-null float64
bed_type                          16313  non-null object

amenities                         16313  non-null object
square_feet                       446 non-null float64
price                             16313  non-null float64
weekly_price                      2879 non-null object
monthly_price                     2703 non-null object
security_deposit                  10095  non-null object
cleaning_fee                      11400  non-null object
guests_included                   16313  non-null int64
extra_people                      16313  non-null object
minimum_nights                    16313  non-null int64
maximum_nights                    16313  non-null int64
calendar_updated                  16313  non-null object
has_availability                  16313  non-null object
availability_30                   16313  non-null int64
availability_60                   16313  non-null int64
availability_90                   16313  non-null int64
availability_365                  16313  non-null int64
calendar_last_scraped             16313  non-null object
number_of_reviews                 16313  non-null int64
first_review                      13261  non-null object
last_review                       13290  non-null object
review_scores_rating              13118  non-null float64
review_scores_accuracy            13105  non-null float64
review_scores_cleanliness         13111  non-null float64
review_scores_checkin             13089  non-null float64
review_scores_communication       13104  non-null float64
review_scores_location            13083  non-null float64
review_scores_value               13082  non-null float64
requires_license                  16313  non-null object
license                           1511 non-null object
jurisdiction_names                0 non-null float64
instant_bookable                  16313  non-null object
is_business_travel_ready          16313  non-null object
cancellation_policy               16313  non-null object
require_guest_profile_picture     16313  non-null object
require_guest_phone_verification  16313  non-null object
calculated_host_listings_count    16313  non-null int64
reviews_per_month                 13261  non-null float64
cat_precio                        16313  non-null object
dtypes: datetime64[ns](1), float64(23), int64(13), object(60)
memory usage: 12.2+ MB
```

In [64]:

```
#quitar columnas irrelevantes
listings_precios_num = listings_precios.drop(["id","scrape_id","host_id","latitude","lo
ngitude",
                                              "jurisdiction_names","neighbourhood_g
roup_cleansed",
"license","has_availability","neighbou
rhood_group_cleansed",
"thumbnail_url","medium_url","xl_picture_u
rl",
"host_acceptance_rate"],
axis=1)
```

In [65]:

```
#Correlación
corr = listings_precios_num.corr()
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
cmap = seaborn.diverging_palette(220, 10, as_cmap=True)
f, ax = plt.subplots(figsize=(11, 9))
seaborn.heatmap(corr,mask=mask, cmap=cmap, vmax=.3, center=0,
                square=True, linewidths=.5, cbar_kws={"shrink": .5})
```

Out[65]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x29e22a99748>
```