

Monitorización de rentas de alquiler en áreas municipales



José Manuel García Rodes

Máster Universitario en
Ciencia de Datos

Área 5: Data Analysis y Big Data.
Streaming, Visualization & Big Data

Tutor/a de TF

Rafael Luque Ocaña

**Profesor/a responsable de
la asignatura**

Albert Solé Ribalta

01/2023

Universitat Oberta
de Catalunya



Esta obra está sujeta a una licencia de Reconocimiento-
NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

Ficha del Trabajo Final

Título del trabajo:	Monitorización de rentas de alquiler en áreas municipales
Nombre del autor/a:	José Manuel García Rodes
Nombre del Tutor/a de TF:	Rafael Luque Ocaña
Nombre del/de la PRA:	Nombre y dos apellidos
Fecha de entrega:	01/2023
Titulación o programa:	Máster Universitario en Ciencia de Datos
Área del Trabajo Final:	Área de TF. Área 5: Data Analysis y Big Data. Streaming, Visualization & Big Data
Idioma del trabajo:	Castellano
Palabras clave	Geolocalización. Observatorio. Alquiler. Vivienda.
Resumen del Trabajo	
<p>El objetivo del presente trabajo fin de máster es la creación de una herramienta que permita a los servicios públicos de vivienda de las distintas corporaciones locales (o a cualquier otra entidad o persona interesada) disponer de información periódica de la evolución de la renta de alquiler en su municipio o áreas de interés. El proceso automatiza la extracción de direcciones, características y rentas de las viviendas anunciadas en alquiler en los principales portales inmobiliarios, para posteriormente mediante su geolocalización asignarlas a las áreas de estudio, siendo más precisos en la ubicación de las viviendas y aumentando el tamaño de la muestra, lo que nos permite atomizar o crear zonas diferentes a las que se pueden obtener en los portales inmobiliarios.</p> <p>Adicionalmente, realizar extracciones diarias de los anuncios permite tener una estimación de la renta más rigurosa dado que una parte importante de estos tienden a ser eliminados en pocos días. Con todo ello se presenta la información en una visualización interactiva (cuadro de mando) accesible desde internet.</p>	

Abstract

The goal of this Project is to create a tool that allows public housing services from local corporations (or any other interested party) to track information about the evolution of housing rental prices within a town or area of interest. The process automates the extraction of data such as addresses, features or renting prices of the houses advertised in the main real state portals, to later assign them to the study areas through geolocation, being more accurate in the location of the houses and increasing the sample size, what allows to atomize or create different areas from those that can be obtained in real state portals.

Additionally, making daily extractions of the advertisements allows for a more rigorous estimation of the income, since a significant part of these tend to be eliminated in a few days. With all this, the information is presented in an interactive visualization (dashboard) accessible from the Internet.

Índice

1. Introducción	7
1.1. Contexto y justificación del Trabajo	7
1.2. Explicación de la motivación personal	8
1.3. Definición de los objetivos	9
1.4. Enfoque y método seguido	9
1.5. Planificación del trabajo	11
1.6. Breve resumen de productos obtenidos	11
1.6.1. Cuadro de mando	11
1.6.2. Conjunto de datos	14
1.6.3. Anuncios de particulares	14
2. Estado del arte	16
2.1. Situación actual	16
2.2. Proyectos similares	22
2.3. Conclusión	24
3. Métodos y resultados	26
3.1. Estructura de carpetas y archivos	26
3.2. Delimitación de zonas	27
3.2.1. Estudio de la ciudad	27
3.2.2. Geolocalización de áreas	29
3.3. Extracción de datos	31
3.3.1. Búsqueda en portales inmobiliarios	31
3.3.2. Web scraping	31
3.4. Depuración	38
3.5. Visualización	43
4. Conclusiones y trabajos futuros	46
5. Glosario	48
6. Bibliografía	50
7. Anexos	53
7.1. Anexo I. Web scraping	53
7.2. Anexo II. Depuración del conjunto de datos	63
7.3. Anexo III. Código de inserción del cuadro de mando en página web	82

Índice de Figuras

Figura 1. Descripción general del proceso	10
Figura 2. Planificación del proyecto.....	11
Figura 3. Cuadro de mando	12
Figura 4. Cuadro de mando. Playa de San Juan.....	13
Figura 5. Anuncios de particulares	15
Figura 6. Visor Ministerio de Transportes, Movilidad y Agenda Urbana	17
Figura 7. Visor Observatorio vasco de vivienda	18
Figura 8. Índice inmobiliario Fotocasa	20
Figura 9. Índice inmobiliario Idealista	21
Figura 10. Cuadro de mando Universitat Politècnica de València	22
Figura 11. Cuadro de mando. Vitoria-Gasteiz	23
Figura 12. Mapa cromático. Vitoria - Gasteiz	24
Figura 13. Estructura de carpetas y archivos	26
Figura 14. Límites administrativos nivel barrio. Fuente: Guía Urbana de Alicante	28
Figura 15. Ampliación figura 14. Fuente: Guía Urbana de Alicante	28
Figura 16. Dibujando los perímetros de los barrios	29
Figura 17. Áreas de estudio	30
Figura 18. Exportación a KML.....	30
Figura 19. Tráfico y compromiso. Septiembre 2022. Fuente: similarweb.com	31
Figura 20. Url inicial de fotocasa para las búsquedas de Alquiler en el municipio de Alicante.....	35
Figura 21. Inspección de la web inicial.....	36
Figura 22. Inspección web secundaria	37
Figura 23. Archivo resultado del raspado. anuncios_fotocasa 2022.09.22_15.30.12.xlsx	38
Figura 24. "Semi-line algorithm for determining whether a point is inside a polygonon", del.....	40
Figura 25. data (data.xlsx)	41
Figura 26. dormitorios (data.xlsx)	42
Figura 27. Intervalos renta (data.xlsx)	42
Figura 28. Estructura de tablas de Tableau.....	43
Figura 29. Cuadro de mando	44
Figura 30. Cuadro de mando. Playa de San Juan.....	45

1. Introducción

1.1. Contexto y justificación del Trabajo

La problemática del acceso a la vivienda es un tema que está en primera línea de la actualidad, es sabida la dificultad para el acceso a la vivienda sobre todo en los colectivos de edades más tempranas que buscan su primer acceso a ésta. Desde el punto de vista político es fundamental para una corporación local, autonómica, etc., conocer periódicamente cual es el estado del mercado inmobiliario y si existen zonas tensionadas o no. Son múltiples los indicadores que se llevan a cabo actualmente para tratar de dar respuesta a esta pregunta, los podemos encontrar a nivel estatal [1] y autonómico [2], indicadores en su mayoría sesgados o que abarcan áreas muy extensas. Se considera que, no dejando de tener sesgo, los indicadores que más se pueden aproximar al precio real son los basados en los anuncios de alquiler de vivienda realizados en los portales inmobiliarios de internet [3], [4], ya que como es sabido cualquier persona tiene acceso a esta información desde su smartphone, Tablet o PC y suele ser uno de los primeros lugares donde se acude para buscar vivienda y donde se anuncian inmobiliarias y particulares. Centrándonos en la información que ofrecen en estos portales, esta es bastante amplia y detallada, pero suele ser casi siempre para grandes áreas. Se va a tratar de aumentar la granularidad creando áreas geográficas de menor superficie o que se correspondan a zonas de especial interés para la corporación local, autonómica, etc.

La relevancia de la propuesta se basa en tener un mecanismo propio de observación del comportamiento del mercado inmobiliario del alquiler de vivienda para un área geográfica concreta con la periodicidad que se desee (diaria, mensual, trimestral, etc.). Esto permite, por una parte, realizar estudios longitudinales de la evolución de la renta para el municipio o de áreas concretas y por otro lado como un subproducto, aprovechando la información extraída, poder discriminar entre anuncios de inmobiliarias y particulares conociendo los anuncios nuevos publicados por estos últimos antes de que sean captados por inmobiliarias y poder de esta forma, captar esas viviendas para los programas de alquiler asequible de las corporaciones locales o autonómicas.

Lo innovador respecto a lo que hay actualmente es que se puede configurar el área a estudiar no teniendo que limitarse a la segmentación tradicional de municipio, barrio, distrito o sección censal, se pueden definir áreas de interés como unidades inferiores, superiores o combinaciones de las tradicionales.

La propuesta consiste en la realización de un procedimiento para la monitorización de la evolución de la renta de alquiler de vivienda en un municipio, barrio o área geográfica de interés. Se trata de automatizar todo el proceso de extracción de

direcciones, características y rentas de las viviendas anunciadas en alquiler en los principales portales inmobiliarios para posteriormente, mediante un proceso de geolocalización, asignarlas a las áreas de interés y poder realizar estudios estadísticos sobre la evolución de la renta, tipología de la vivienda, etc. Creando nuestras propias áreas se puede ser más precisos en la ubicación de las viviendas y se pueden atomizar o crear nuevas zonas diferentes a las que se obtienen en un portal inmobiliario, estatal o autonómico. Es importante destacar que la extracción puede tener una periodicidad diaria, de manera que diariamente tenemos todos los anuncios nuevos publicados en el portal no afectando la desaparición de estos en días posteriores.

1.2. Explicación de la motivación personal

Durante más de once años he trabajado como mediador entre propietarios/as e inquilinos/as en el alquiler de vivienda de particulares dentro del programa de *Alquiler Asequible de Vivienda del Patronato Municipal de la Vivienda de Alicante* [5]. A lo largo de estos años han sido multitud de viviendas las alquiladas a través del programa de todos los barrios de la ciudad de Alicante. Uno de los principales cometidos era tratar qué los propietarios/as accedieran a alquilar sus viviendas a través del programa por debajo del precio de mercado en su zona, para lo cual había que saber cuál era ese precio de mercado según las características de la vivienda ofrecida y la ubicación de ésta. Por este motivo se recurría a portales inmobiliarios para ver cuál era el precio de otras viviendas similares anunciadas en el entorno. De aquí partía el principal problema, al consultar los portales en un día concreto es posible que hubiese muy pocos o ningún anuncio para esa zona o que los anuncios fueran muy antiguos con precios fuera de mercado. Para solventar este problema se decidió crear alertas diarias en los principales portales inmobiliarios e ir almacenando los datos de cada vivienda recibida en una base de datos, con el consiguiente coste en tiempo y recursos para tener actualizada ésta. Por otro lado, nos encontrábamos con que las zonas en que los portales inmobiliarios dividían la ciudad eran demasiado extensas y heterogéneas como para poder estimar un precio concreto de una vivienda y nos vimos en la necesidad de subdividir estas zonas en otras de áreas más pequeñas y homogéneas para poder aproximar la renta de alquiler de un nuevo inmueble, lo que nos llevaba a asignar las direcciones postales a las nuevas zonas de forma manual, con el nuevamente coste en tiempo y recursos.

Dada la nueva información obtenida de ubicación de las viviendas encontradas, se pensó en crear una visualización de acceso público [6] para que la ciudadanía pudiera consultar el mercado inmobiliario del alquiler tanto en el municipio de Alicante como en alguna zona en concreto entre las que se había subdividido.

Por todo ello la motivación principal es automatizar todos estos procesos con el consiguiente ahorro en tiempo y recursos y hacerlos extensibles a cualquier otro municipio o zona de interés.

1.3. Definición de los objetivos

OBJETIVO PRINCIPAL

Creación de una herramienta para realizar una visualización interactiva (cuadro de mando) accesible desde internet, donde se muestre la evolución trimestral de la renta de alquiler de vivienda para un municipio concreto, los barrios de éste o zonas de interés en el que se haya subdividido, así como la tasa de variación con respecto al año anterior.

OBJETIVOS SECUNDARIOS

Conocer el precio medio y las características de una vivienda tipo que sirva para aproximar la renta de mercado de un nuevo inmueble similar en la zona.

Crear y mantener actualizado un banco de datos abierto a partir de los resultados obtenidos.

Tener acceso a los anuncios de alquiler de vivienda de particulares de forma temprana, antes de que desaparezcan del portal, para la posible captación de la vivienda.

1.4. Enfoque y método seguido

En cuanto a las posibles estrategias para alcanzar el objetivo principal se podía haber recurrido a los datos ya existentes en los portales inmobiliarios, estatales o autonómicos y haber realizado una visualización ad hoc para el municipio en concreto y para las zonas en las que se subdivida éste en el portal, pero la idea es tener una mayor granularidad y una muestra de anuncios más grande para poder llevar a cabo esta granularidad. Por ello se descarta esta posibilidad y se decide crear una metodología para la realización de la visualización con una muestra recogida por nosotros mismos.

En la *figura 1* se realiza la descripción general del proceso de recogida y monitorización de rentas de alquiler para su posterior almacenamiento y visualización.

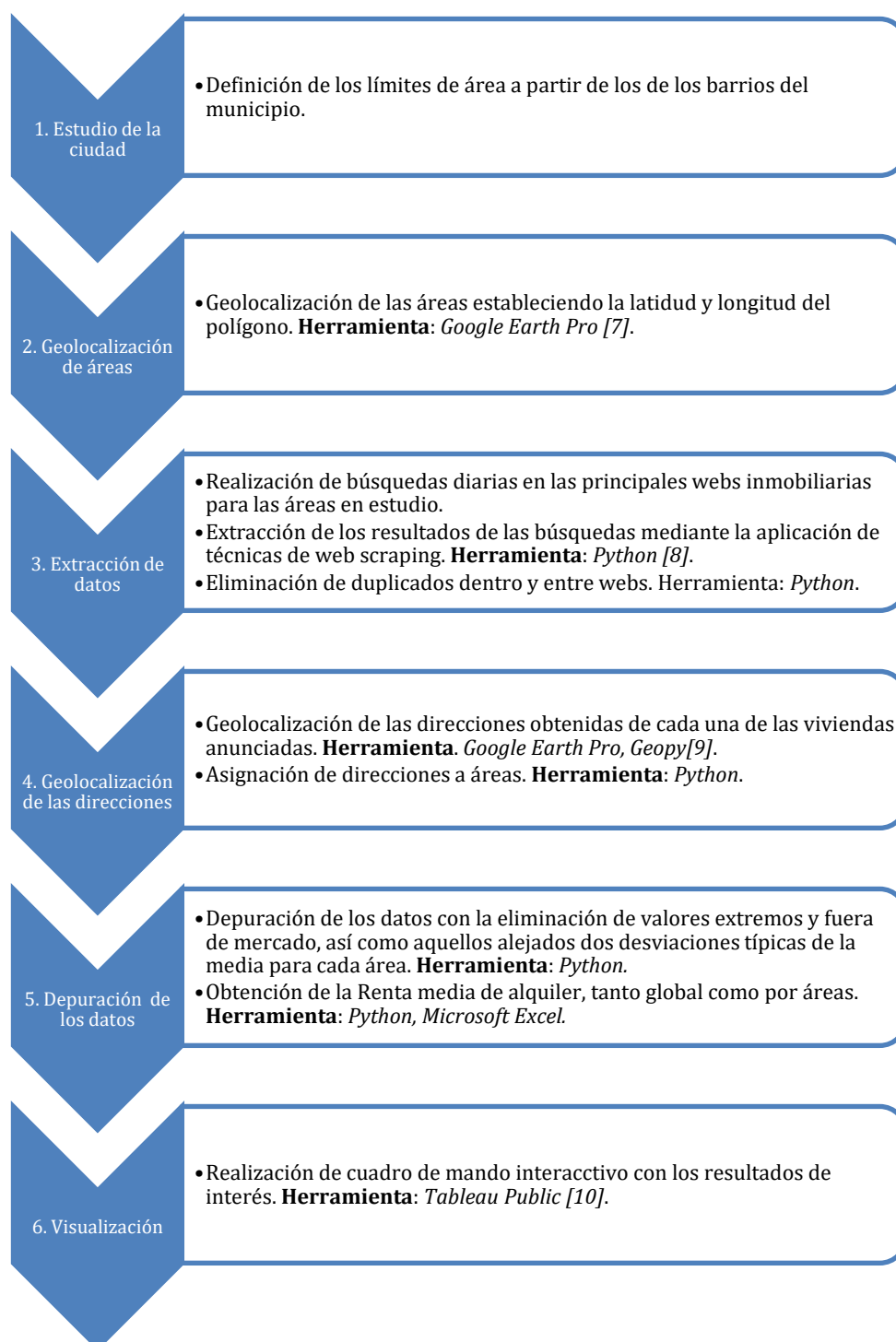


Figura 1. Descripción general del proceso

1.5. Planificación del trabajo

En la *figura 2*, se presenta la planificación del proyecto.

	Semana 1 10/Oct - 16/Oct	Semana 2 17/Oct - 23/Oct	Semana 3 - 4 24/Oct - 6/Nov	Semana 5 7/Nov - 13/Nov	Semana 6 - 7 - 8 14/Nov - 4/Dic	Semana 9 5/Dic - 11/Dic	Semana 10 - 11 12/Dic - 18/Dic	Semana 12 - 13 26/Dic - 8/Ene
Estudio del estado del arte								
Estudio del municipio y definición de las áreas de interés								
Geolocalización de áreas de interés								
Creación de la herramienta para la extracción de datos								
Puesta en marcha de la extracción diaria de los datos								
Extracción de la muestra piloto								
Creación de la herramienta para la geolocalización de las direcciones								
Creación de la visualización								
Herramienta de depuración de los datos								
Geolocalización de los datos								
Carga de datos en el cuadro de mando								
Depuración de errores								
Redacción de la memoria								

Figura 2. Planificación del proyecto

1.6. Breve resumen de productos obtenidos

1.6.1. Cuadro de mando

El objetivo final del proyecto es la realización de un cuadro de mando (*figura 3*), en el cual se muestren entre otros estadísticos; la renta media de alquiler y la tasa de variación de esta con respecto al año anterior para cada zona en estudio, junto con su evolución a lo largo de los trimestres.

Estos resultados se pueden obtener de forma individual para cada una de las zonas (o barrios) en las que hemos subdividido el municipio seleccionándola en el área del mapa correspondiente (*figura 4*).

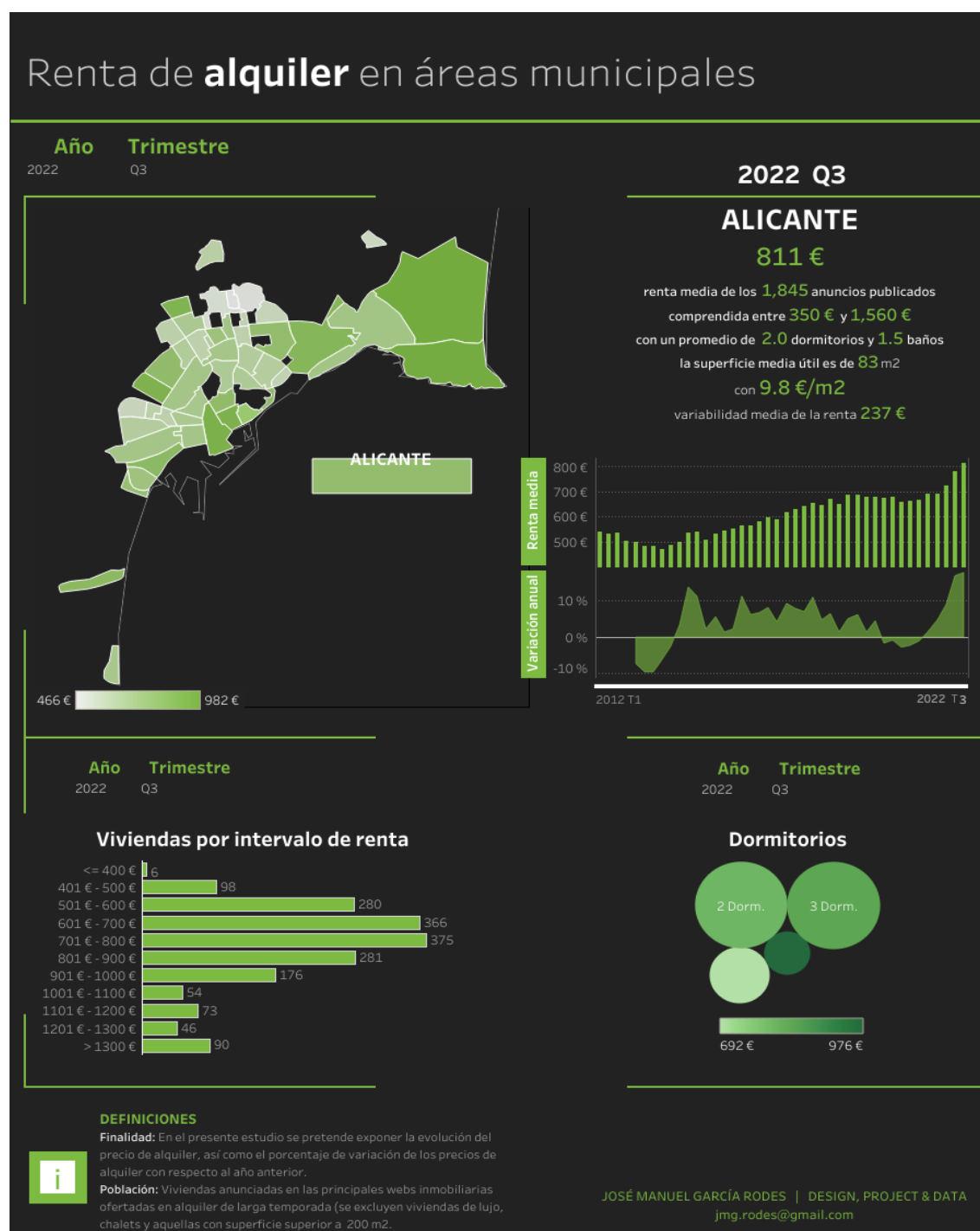
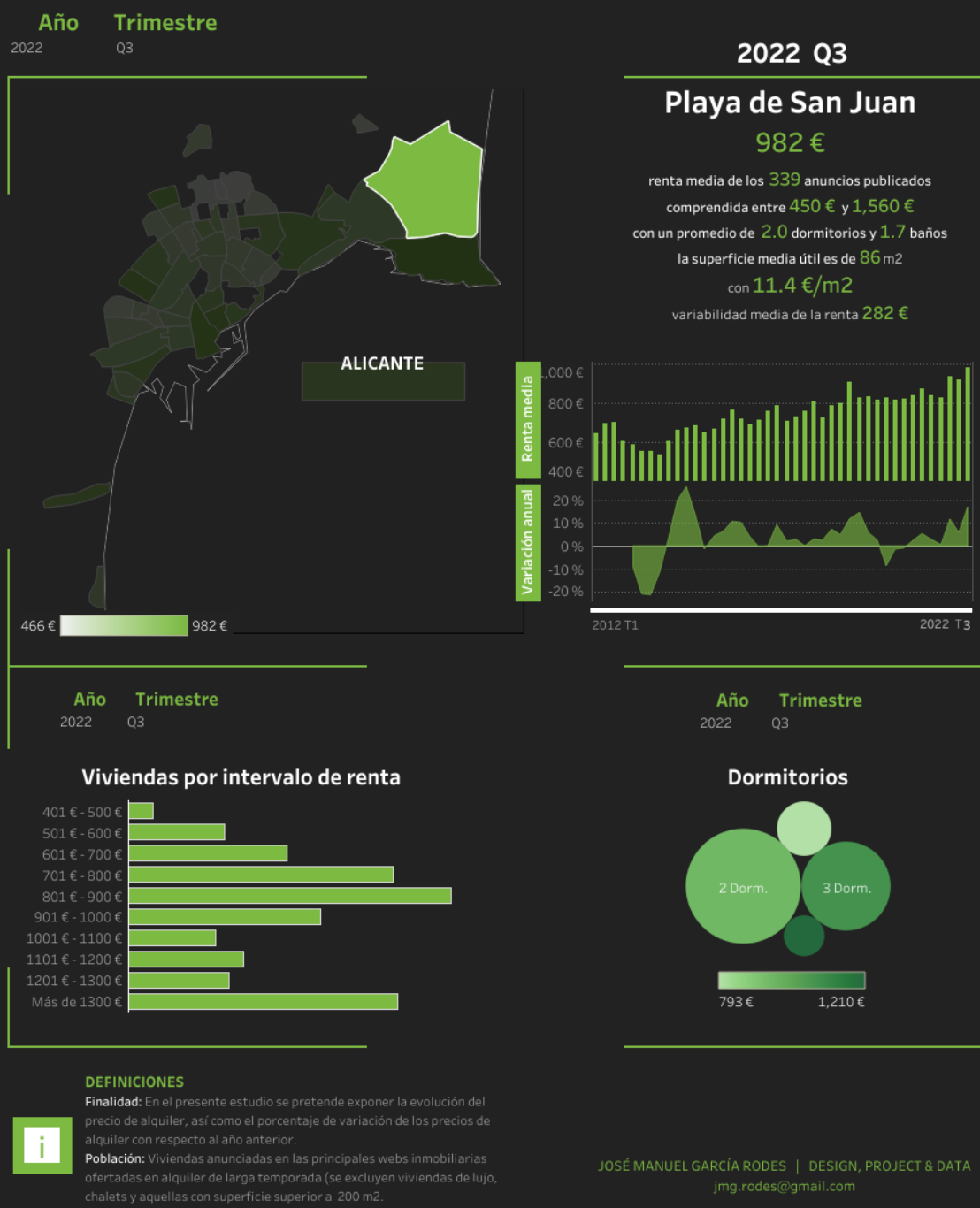


Figura 3. Cuadro de mando

Renta de **alquiler** en áreas municipales



Dormitorios

2 Dorm.

3 Dorm.

793 €

1,210 €

DEFINICIONES

Finalidad:

En el presente estudio se pretende exponer la evolución del precio de alquiler, así como el porcentaje de variación de los precios de alquiler con respecto al año anterior.

Población:

Viviendas anunciadas en las principales webs inmobiliarias ofertadas en alquiler de larga temporada (se excluyen viviendas de lujo, chalets y aquellas con superficie superior a 200 m2).

JOSÉ MANUEL GARCÍA RODES | DESIGN, PROJECT & DATA

jmg.rodes@gmail.com

Figura 4. Cuadro de mando. Playa de San Juan

1.6.2. Conjunto de datos

Para la realización del cuadro de mando anterior se utiliza el conjunto de datos obtenido del proceso de *web scraping* de los portales inmobiliarios y de la depuración de este que se describirá más adelante. Este conjunto de datos se puede subir al repositorio de datos abiertos del organismo correspondiente.

El conjunto de datos está formado por 11 variables y 1947 registros con los datos agregados por zonas, trimestres y años, correspondientes a los anuncios de alquiler de vivienda en la ciudad de Alicante publicados en internet por los principales portales inmobiliarios. En la depuración se excluyen viviendas de lujo, unifamiliares y chalés, eliminándose outliers y duplicados, así como, los valores que exceden por exceso o defecto dos desviaciones típicas de la media. La recogida de datos se ha venido realizando desde el cuarto trimestre del año 2021 al cuarto trimestre del año 2022 ambos inclusive. Para completar la serie se han añadido los datos existentes en el *Portal integral de información analítica y datos abiertos del Ayuntamiento de Alicante* [11].

Descripción de las variables:

- **Fecha anuncio:** Fecha de descarga de los datos del anuncio.
- **Id_Zona:** Número asignado a la zona correspondiente.
- **Zona:** Nombre de la zona
- **n:** número de viviendas anunciadas por zona.
- **Precio:** Renta media de alquiler anunciada.
- **m2_U:** Metros cuadrados útiles por término medio.
- **Dormitorios:** Número de dormitorios por término medio.
- **Baños:** Número de baños por término medio.
- **P_min:** Renta mínima de alquiler anunciada.
- **P_max:** Renta máxima de alquiler anunciada.
- **P_desv:** Desviación típica de la renta de alquiler.

En el conjunto de datos anterior se han seleccionado las variables de interés para la realización del cuadro de mando, pero el archivo inicial contiene muchas más características de la vivienda como son: el amueblamiento, el estado de conservación, año de construcción, reformas, etc. Este archivo se almacena para poder ser utilizado en futuros estudios.

1.6.3. Anuncios de particulares

Un tercer producto que extraemos es un listado diario de los anuncios insertados en las webs por propietarios/as particulares que gestionan el alquiler de la vivienda ellos mismos. Este listado puede resultar interesante desde el punto de vista de la captación de vivienda en alquiler, bien para incluirla en programas de alquiler

municipales o para profesionales del sector inmobiliario, evitando el acceso y recorrido de las webs inmobiliarias con el consiguiente ahorro en tiempo.

En este listado se almacena entre otras, la renta de alquiler, la dirección o barrio donde se encuentra la vivienda y el enlace directo a la web donde podemos obtener los datos del anunciante (figura 5).

	A	B	C	D	E	F	G	H	I	J	K	L
1	Anuncio	Fecha	Portal	Renta	Titular	Web	Dirección					
2	4	2022.12.21	Fotocasa	600 €	Anuncio Particular	https://www.fotocasa.es/es/alquiler/estefania	Piso de alquiler en Paseo Explanada de España, Barrio del Centro					
3	5	2022.12.21	Fotocasa	600 €	estefania	https://www.fotocasa.es/es/alquiler/particular	Piso de alquiler en Avenida de la Estación, Ensanche - Diputación					
4	7	2022.12.21	Fotocasa	800 €	Particular	https://www.fotocasa.es/es/alquiler/particular	Piso de alquiler en Plaza Barcelona, Pla de Bon Repós					
5	10	2022.12.21	Fotocasa	1.500 €	Particular	https://www.fotocasa.es/es/alquiler/particular	Ático de alquiler en Plaza Avenida Naciones 30, PAU 5					
6	24	2022.12.21	pisoscom	550 €	François	https://www.pisos.com/alquiler/	Piso en alquiler en Calle del Poeta Blas de Loma					
7												
8												
9												

Figura 5. Anuncios de particulares

2. Estado del arte

2.1. Situación actual

Desde hace unos pocos años a esta parte el interés por monitorizar el mercado del alquiler ha ido en aumento por parte de las administraciones públicas. A modo de ejemplo, el Ministerio de transportes, movilidad y agenda urbana, con el objetivo de garantizar la transparencia en el mercado de alquiler de vivienda, ha creado, en el marco del desarrollo del Sistema estatal de referencia del precio del alquiler de vivienda [12], el observatorio de vivienda y suelo. También encontramos el Índice de referencia de precios de alquiler de Catalunya creado en el marco de la Ley 11/2020 por la regulación el precio del alquiler [13]. Estos son tan solo dos ejemplos de cómo la problemática del alquiler y el acceso a la vivienda está en primera línea política.

A continuación, se pasa a mostrar distintos observatorios de vivienda que ofrecen diversas administraciones públicas y algún ente privado, junto con las características principales de estos en lo referente al alquiler.

OBSERVATORIO DE VIVIENDA Y SUELO. MINISTERIO DE TRANSPORTES, MOVILIDAD Y AGENDA URBANA [14]

Este observatorio, a través de la publicación de boletines estadísticos electrónicos, compara la vivienda residencial en alquiler frente a la de venta, con evolución de precios y rentabilidad del alquiler, incluyendo cuestiones referentes a la accesibilidad y la financiación, así como datos socioeconómicos referentes a la renta de los hogares, datos de lanzamientos hipotecarios y sobre rehabilitación. Para ello utiliza los datos disponibles en las principales fuentes estadísticas oficiales como son; el Banco de España, el Registro de la propiedad, el Instituto Nacional de Estadística, Eurostat, el Ministerio de Fomento y los depósitos de fianza de las CCAA, junto con el portal inmobiliario Fotocasa.

Dentro de este observatorio se encuentra el Índice de alquiler de vivienda, un visor interactivo en forma de mapa cromático (*figura 6*) que nos da información para todo el territorio nacional, con una granularidad a nivel censal de la renta de alquiler por m² para municipios de más de 25.000 habitantes.

La periodicidad de los datos es trimestral y podemos encontrar boletines desde el primer trimestre de 2012 hasta el más reciente que corresponde al 1T de 2022.

Sección censal

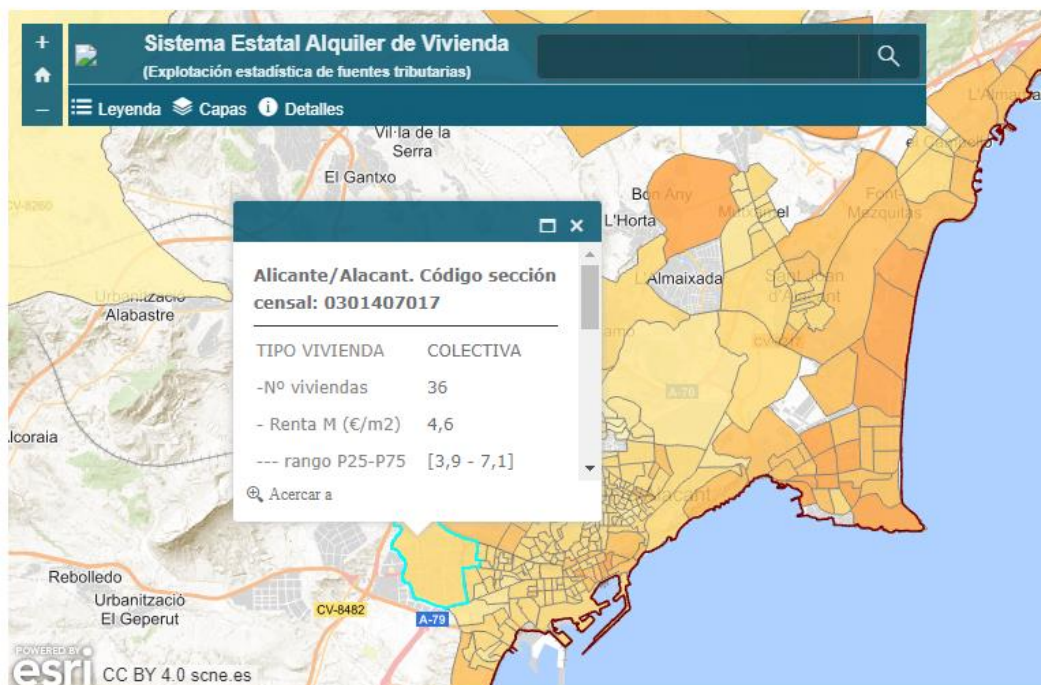


Figura 6. Visor Ministerio de Transportes, Movilidad y Agenda Urbana

AGÈNCIA DE L'HABITATGE DE CATALUNYA. GENERALITAT DE CATALUNYA [15]

En esta agencia podemos encontrar información relacionada con el alquiler de vivienda y demás temas de relacionados con esta. En lo que respecta a la información de rentas de alquiler, la página permite la consulta y obtención de un certificado con el índice de alquiler para los municipios de Catalunya a modo individual para un inmueble determinado, la renta de alquiler no puede sobrepasar lo establecido en este índice. Este dato se aporta únicamente para la vivienda consultada no ofreciendo más información de la zona o la ciudad.

La fuente de datos es propia y la fecha de la última actualización es del 09/03/2022.

OBSERVATORIO DE VIVIENDA ASEQUIBLE. PROVIVIENDA [16]

En este observatorio encontramos tan solo datos globales a nivel de España, no existe un índice de precios por municipio o zona más pequeña. Se trata de un observatorio que analiza de forma global la problemática del acceso a la vivienda en España.

OBSERVATORIO VALENCIANO DE VIVIENDA (OVV) [17]

Este es uno de los observatorios más antiguos que se ha encontrado. Dispone de un visor cartográfico que engloba muchas materias, no solo de vivienda. En lo referente a alquiler se puede obtener datos de la Comunitat Valenciana para zonas de un área bastante extensa, engloban a la práctica totalidad del municipio. Los datos están desactualizados, el último informe trimestral data del cuarto trimestre de 2016.

OBSERVATORIO VASCO DE LA VIVIENDA. DEPARTAMENTO DE PLANIFICACIÓN TERRITORIAL, VIVIENDA Y TRANSPORTES [18]

El observatorio vasco de la vivienda recoge la estadística oficial de la viceconsejería de vivienda que incluye estudios e informes del sector inmobiliario como son; informes del mercado de alquiler, del de compraventa de inmuebles, edificación, vivienda vacía, situación del sector inmobiliario, suelo y urbanización, y oferta inmobiliaria. Realizan sus propias encuestas sobre las personas arrendadoras, hogares en régimen de alquiler, etc. La periodicidad de los informes acerca del mercado del alquiler es trimestral y va desde 2016 hasta el más reciente que es del cuarto trimestre de 2021. La información de las rentas de alquiler se obtiene de los depósitos de fianzas realizados en el País Vasco.

Disponen de un visor [19] de precios de alquiler donde se puede consultar la renta media de alquiler en un mapa cromático con una granularidad de barrio. Los datos están actualizados a 2021.

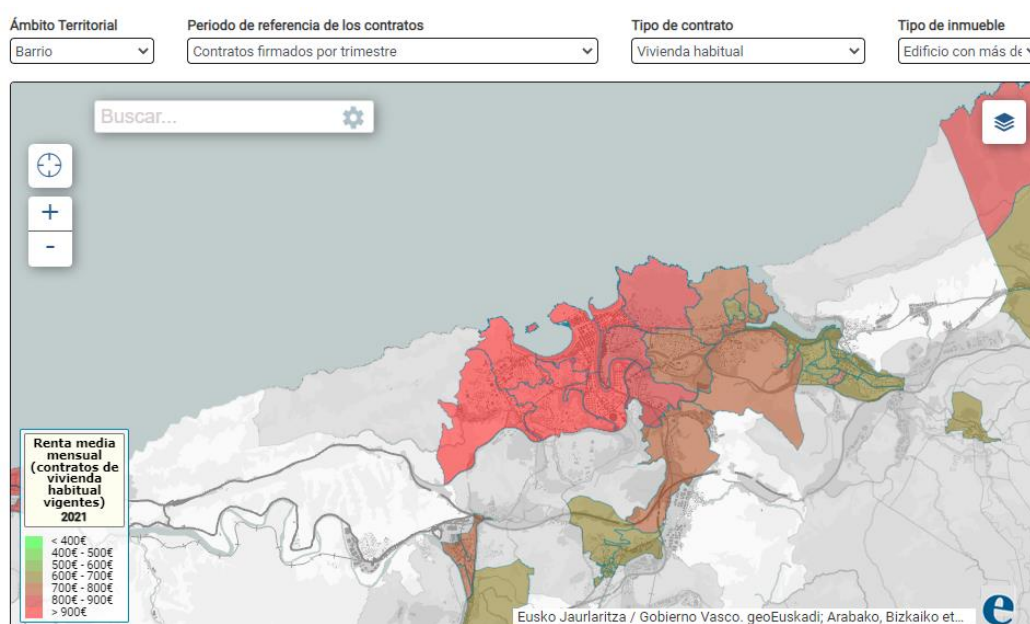


Figura 7. Visor Observatorio vasco de vivienda

OBSERVATORIO DE VIVIENDA Y SUELO DE CANTABRIA. GOBIERNO DE CANTABRIA. CONSEJERÍA DE EMPLEO Y POLÍTICAS SOCIALES. DIRECCIÓN GENERAL DE VIVIENDA [20]

En este observatorio en lo referente a alquiler realizan un informe en pdf de precios de venta y de alquiler. En este informe se incluyen datos relativos a la venta de vivienda y en lo referente al alquiler aporta datos extraídos de Fotocasa e Idealista por separado siendo el nivel de desagregación el que ofrecen estos portales inmobiliarios. También utilizan datos del Ministerio de transportes, movilidad y agenda urbana. La serie tiene una periodicidad mensual y comienza en 2014, siendo el último informe publicado de enero de 2022.

OBSERVATORIO DE VIVIENDA DE GALICIA. INSTITUTO GALEGO DA VIVIENDA E SOLO. XUNTA DE GALICIA [21]

Observatorio arroja datos de; alquiler, censo viviendas, características de las viviendas, construcción de edificios, precios vivienda, transacciones inmobiliarias, fianzas de los alquileres, hipotecas de vivienda, coyuntura. Ofrece mucha información respecto a de otros temas relacionados con la vivienda a parte del alquiler.

Las fuentes de información provienen del Instituto nacional de estadística (INE) y del Instituto gallego de estadística (IGE) y de los registros de contratos de alquiler de la Xunta de Galicia. La última publicación corresponde a junio de 2021 y comenzando la serie de datos de alquiler en 2014. Entre otros da información del número de contratos de alquiler firmados, renta media en euros e intervalos de renta. Tiene una periodicidad anual y la desagregación es por municipio.

OBSERVATORIO DE LA VIVIENDA DE ANDALUCÍA. JUNTA DE ANDALUCÍA. CONSEJERÍA DE FOMENTO, ARTICULACIÓN DEL TERRITORIO Y VIVIENDA [22]

Al parecer este observatorio está en fase de creación, únicamente se encuentra información acerca del órgano colegiado, no se ha encontrado nada más.

OBSERVATORIO DE VIVIENDA. DIARIO EL ESPAÑOL [23]

Observatorio de vivienda de noticias relacionadas con la vivienda en España. No aporta estudios longitudinales ni índices de precios.

ÍNDICE INMOBILIARIO DE PRECIOS FOTOCASA [24]

En esta sección del portal inmobiliario Fotocasa se puede consultar el precio por m² de las viviendas en una zona en concreto, tanto para la venta como el alquiler, así como una serie de datos estadísticos que se muestran en la figura 8. Estos datos son los más amplios a nivel de detalle y la fuente proviene de los propios anuncios publicados por Fotocasa. Con los datos de sus anunciantes Fotocasa elabora en su sala de prensa una gran cantidad de informes y artículos relativos al mercado inmobiliario y sus distintas problemáticas.

Precio del alquiler en Playa de San Juan - El Cabo de las Huertas

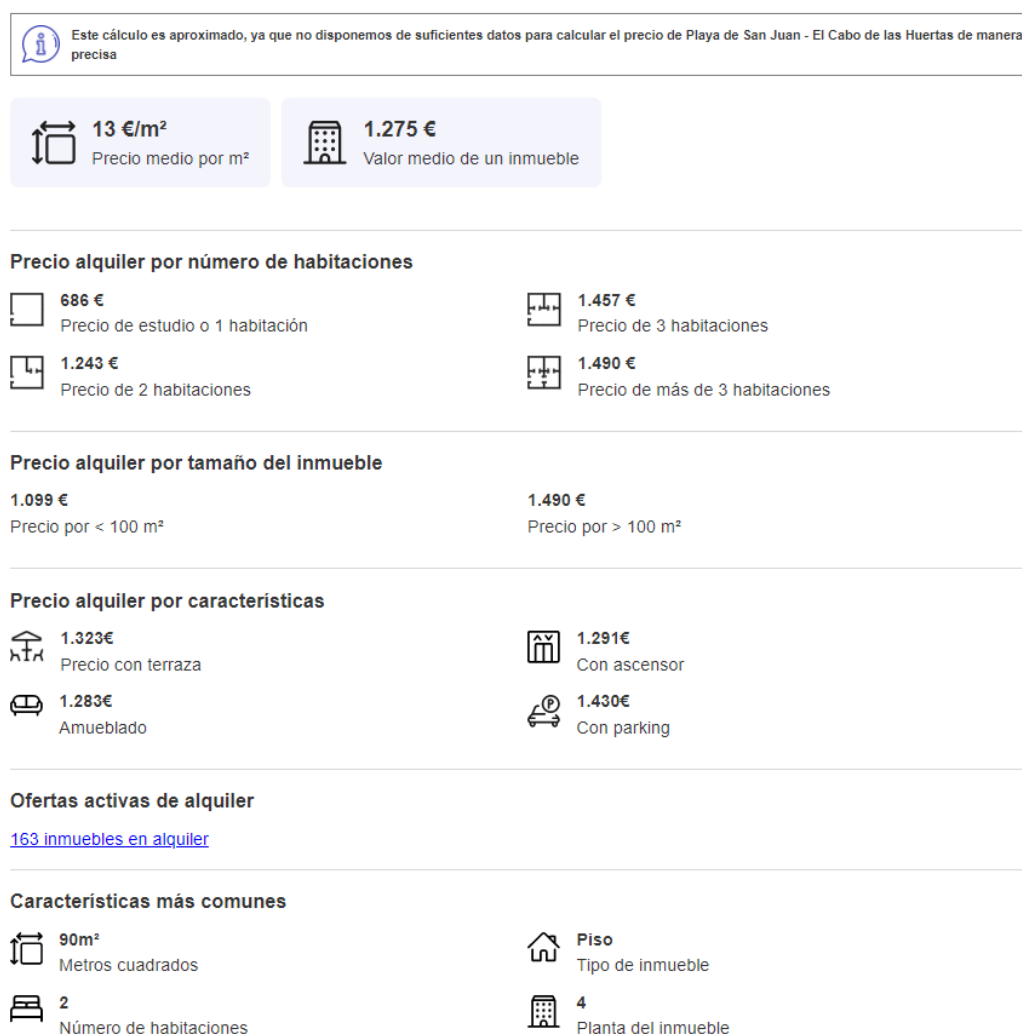


Figura 8. Índice inmobiliario Fotocasa

IDEALISTA. SALA DE PRENSA [25]

En el portal inmobiliario Idealista se encuentra la sección de prensa, donde a parte de una gran cantidad de artículos referidos al mercado inmobiliario se pueden encontrar Informes de precios tanto de venta como de alquiler para las distintas zonas en las que Idealista a dividido el territorio nacional, estas zonas suelen coincidir con agrupaciones de barrios. Al igual que ocurría con Fotocasa la fuente de datos es propia, provienen de los miles de anuncios que se insertan diariamente en el portal. En la figura 9 se muestra un ejemplo de la información ofrecida.

Evolución del precio de la vivienda en alquiler en Playa de San Juan-El Cabo

Septiembre 2022

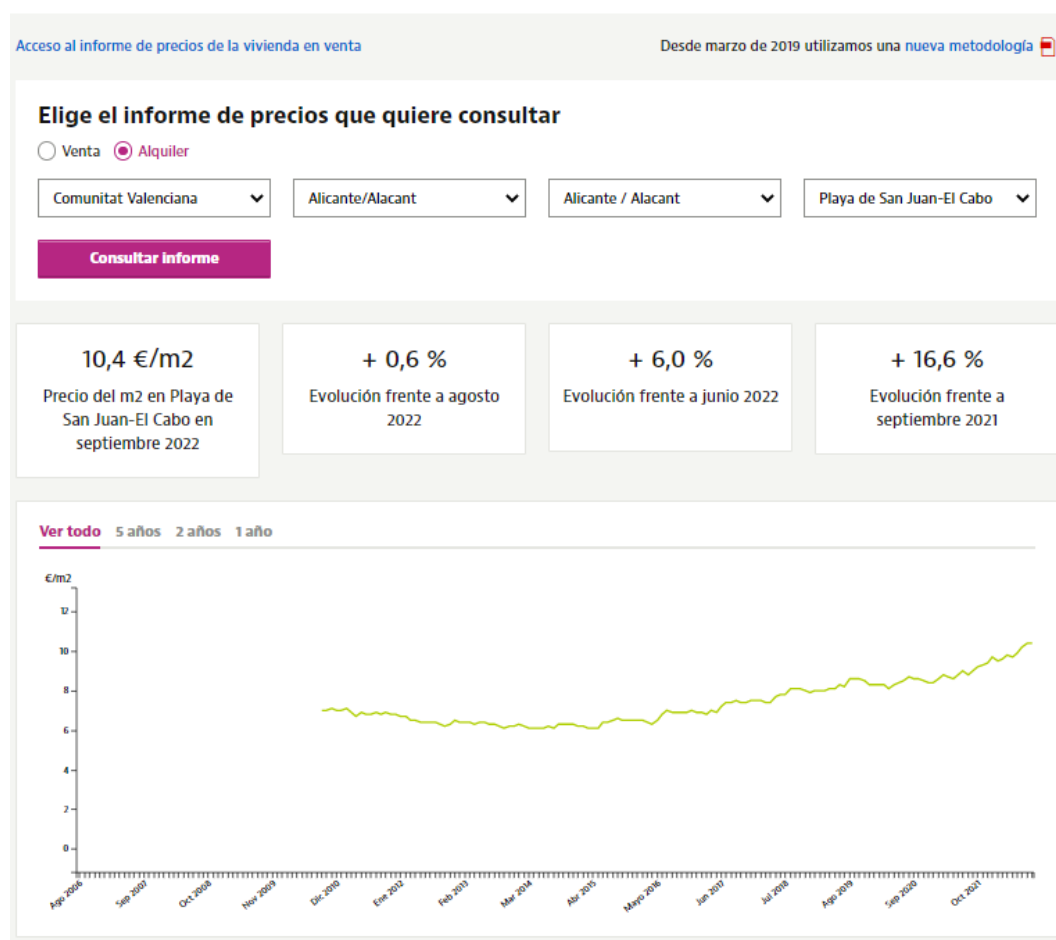


Figura 9. Índice inmobiliario Idealista

2.2. Proyectos similares

Se pasa a comentar las características principales de los dos observatorios que presentan una información y una presentación de resultados similar a la que se pretende realizar.

OBSERVATORIO DE VIVIENDA DE LA UNIVERSITAT POLITÈCNICA DE VALÈNCIA [26]

En este observatorio se presenta un informe extenso con datos referentes a la venta y alquiler de vivienda residencial. La desagregación es por distritos. La parte del alquiler no habla de dormitorios, baños. Las series temporales empiezan en 4T 2019 y el último informe publicado es del segundo trimestre de 2022.

Por otro lado, en la figura 10 se muestra el cuadro de mando [27] del observatorio que incluye un visor con un mapa cromático. Tiene una desagregación que llega hasta el barrio, dando información, entre otras cosas, del precio medio por número de habitaciones y distingue entre vivienda unifamiliar y plurifamiliar.



Figura 10. Cuadro de mando Universitat Politècnica de València

OBSERVATORIO DE LA VIVIENDA. VITORIA-GASTEIZ. SOCIEDAD URBANÍSTICA MUNICIPAL [28]

Presenta un cuadro de mando interactivo (figura 11) con información de vivienda en alquiler y venta que se puede filtrar por fecha, barrios e indicador. La última actualización es del segundo trimestre de 2022 y la serie comienza en el primer trimestre de 2021. El nivel máximo de desagregación es por barrio y los datos están extraídos de los depósitos de renta de alquiler del Gobierno vasco.

El observatorio también presenta mapas cromáticos de diversos temas, el mapa del alquiler (figura 12) es muy similar al que se quiere realizar.

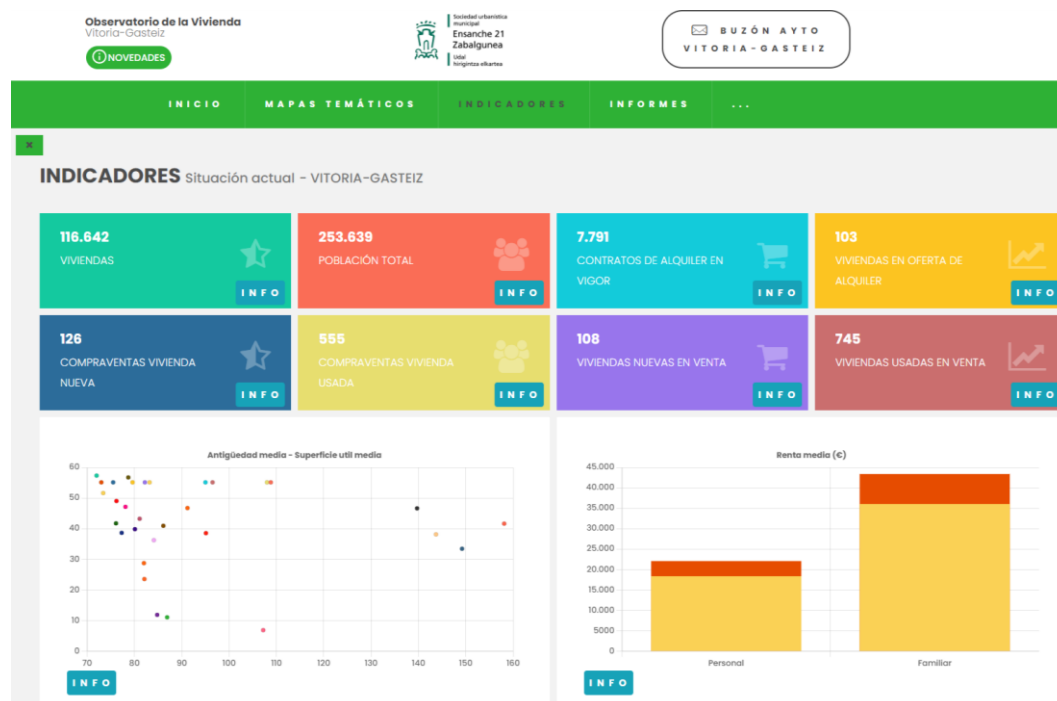


Figura 11. Cuadro de mando. Vitoria-Gasteiz

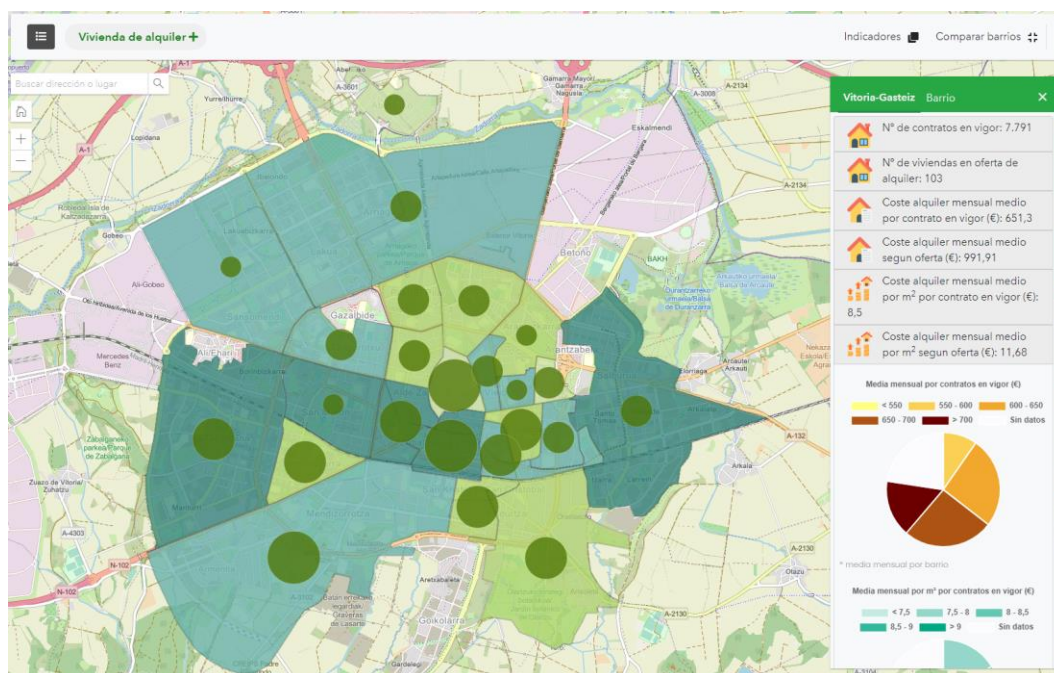


Figura 12. Mapa cromático. Vitoria - Gasteiz

2.3. Conclusión

Una vez analizados los distintos observatorios que se han encontrado en los diferentes organismos públicos y privados, se ha podido comprobar que en su mayoría se nutren de las mismas fuentes (depósitos de fianza de las CCAA, Hacienda, Idealista y Fotocasa). La mayoría de los informes y boletines, en lo referente al alquiler del mercado libre, se utilizan datos publicados por Idealista y Fotocasa, también se elaboran con los depósitos de fianza de las CCAA o con los datos del Ministerio de Hacienda, en estos dos últimos casos las muestras podrían estar sesgadas si los alquileres se declaran por un precio inferior al real.

La granularidad de la mayoría de ellos se ciñe a la ofrecida en sus informes por Idealista o Fotocasa, o bien por la segmentación de hacienda o la de las CCAA en sus depósitos de fianza.

Aunque los datos de los portales inmobiliarios también pueden estar sesgados o fuera de mercado, se piensa que pueden ser los que mejor reflejen la realidad del mercado libre por el volumen de viviendas anunciadas y por ser sitios recurrentes para encontrar vivienda en alquiler.

La herramienta que se plantea tiene las siguientes ventajas sobre las revisadas anteriormente:

- Se pueden crear nuestras propias zonas de interés al margen de las que establecen los portales inmobiliarios o los organismos públicos y con la granularidad que deseemos.
- Se tiene la posibilidad de obtener datos diarios de viviendas en alquiler, evitando la rápida desaparición de los anuncios y la pérdida de información valiosa.
- Se pueden extraer un mayor número de características de las viviendas como; orientación, amueblamiento, ascensor, garaje, urbanización, antigüedad, fotos, etc.
- Se discrimina entre anuncios de inmobiliarias y particulares, pudiendo tener acceso a la captación de las viviendas de estos últimos si fuera necesario.
- La información extraída es gratuita y no se depende de terceros para obtenerla.
- El dato que se ofrece es reciente. La mayoría de los indicadores disponibles tienen más de 9 meses o un año de antigüedad, a excepción de los dos últimos observatorios que tienen una antigüedad de un trimestre y los de Fotocasa o Idealista que tienen una periodicidad mensual.
- Al extraer anuncios de varios portales inmobiliarios, tras la eliminación de duplicados, el número de testigos es mayor que si solo contamos con un portal.

3. Métodos y resultados

3.1. Estructura de carpetas y archivos

En este primer apartado se muestra la estructura de carpetas y los archivos contenidos en ellas que va a tener el proyecto, a la cual se irá haciendo referencia a lo largo de la memoria. Las carpetas que cuelgan de la carpeta *Ouput* se irán creando de forma automática.

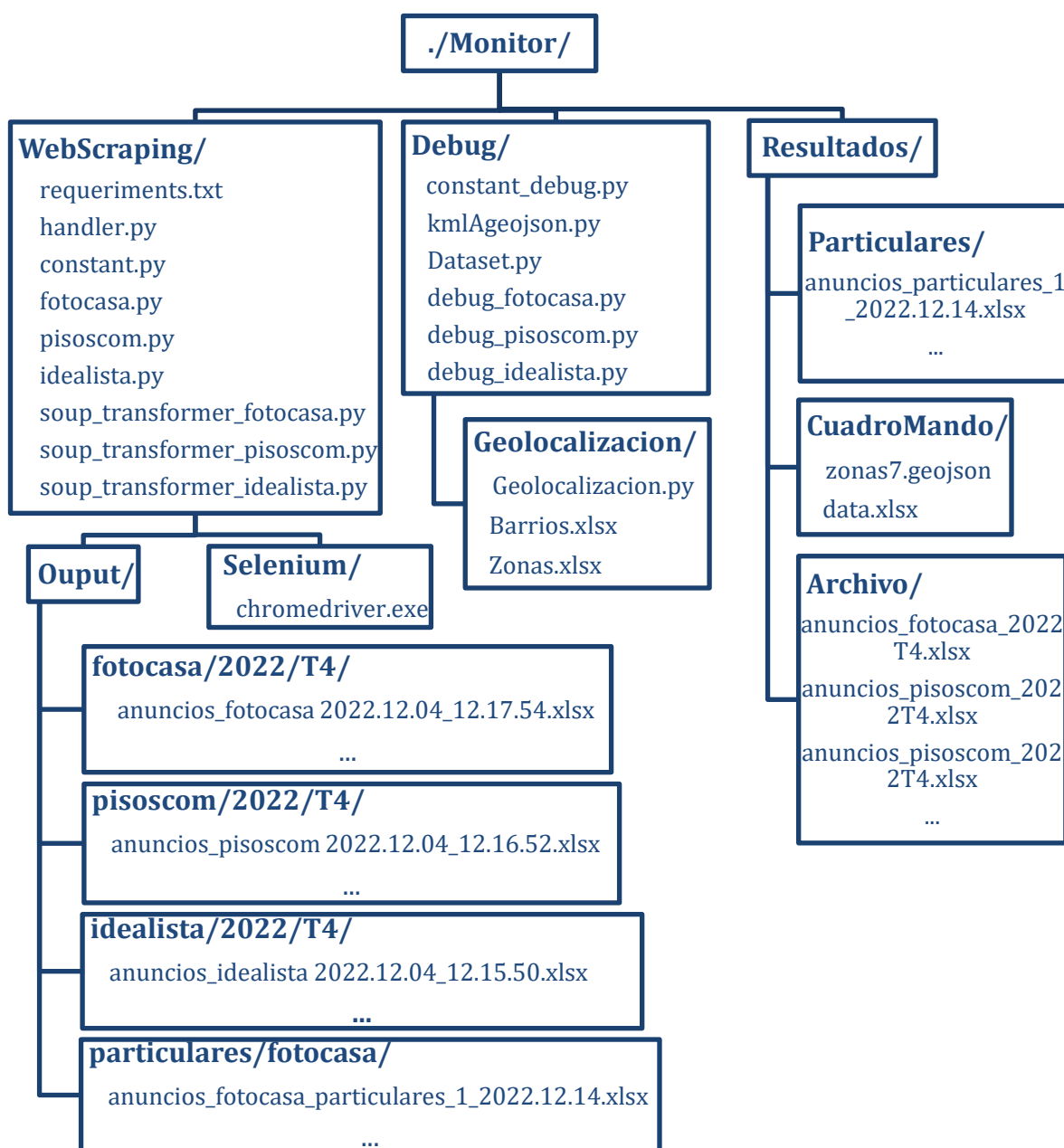


Figura 13. Estructura de carpetas y archivos

3.2. Delimitación de zonas

Para llevar a cabo el proyecto se va a tomar como municipio de estudio la ciudad de **Alicante**.

3.2.1. Estudio de la ciudad

Inicialmente se realiza el estudio partiendo de las áreas administrativas determinadas por los ayuntamientos, como pueden ser los barrios o los distritos, y a partir de estos límites se crean otros de inferior tamaño. Por lo general dentro de los barrios tradicionales suelen haber unas zonas más homogéneas que otras, pudiendo ser interesante tenerlas separadas. Estas zonas se suelen distinguir por el nivel socioeconómico de la población, el tipo de construcción, la antigüedad de los edificios, etc. Por ello es aconsejable consultar con expertos en la zona o realizar algún tipo de estudio antes de subdividir estas zonas administrativas. También podría interesar una zona en concreto basándose en algún criterio arbitrario.

La figura 14 muestra la división administrativa que aparece en la *Guía Urbana de Alicante* [29] de los barrios de la ciudad. Para el objetivo que nos ocupa se parte de esta división inicial y se añadirá alguna subdivisión de barrio según los criterios comentados anteriormente. En la figura 15 vemos una ampliación de la figura 14 para conocer en detalle las calles por las que discurren los límites de los barrios (líneas color morado). Dentro de cada área de barrio viene el nombre de este.

A partir de aquí y según las necesidades que se tengan, se pueden subdividir, eliminar o crear nuevas zonas siempre y cuando las nuevas divisiones sean excluyentes, es decir, una dirección solo puede pertenecer a una única zona.

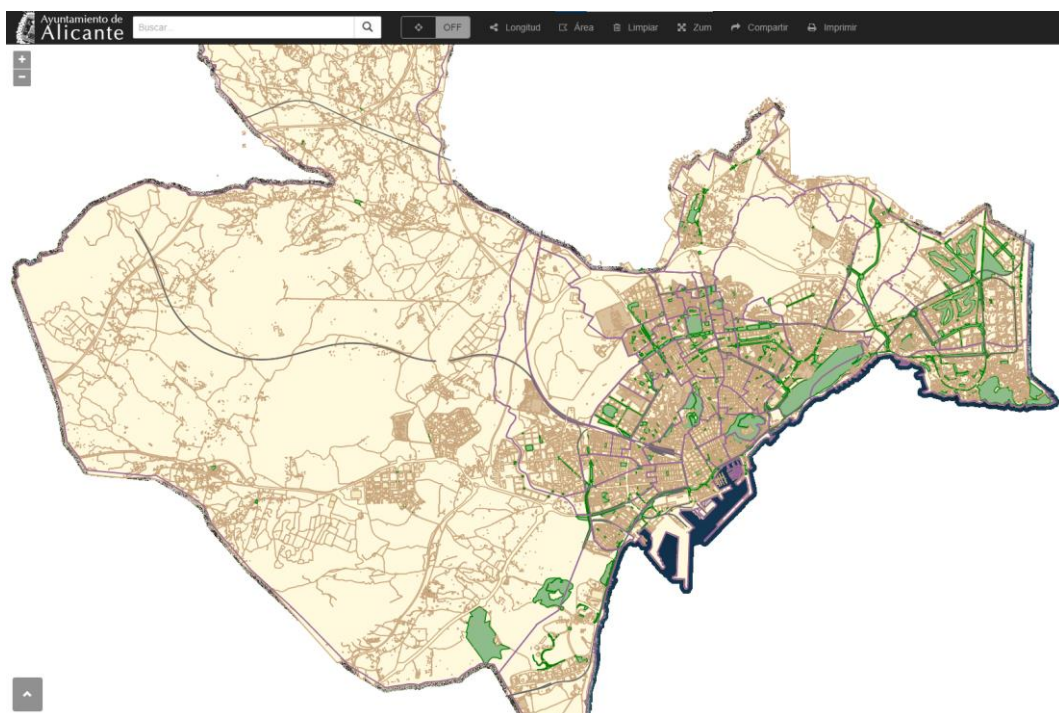


Figura 14. Límites administrativos nivel barrio. Fuente: Guía Urbana de Alicante

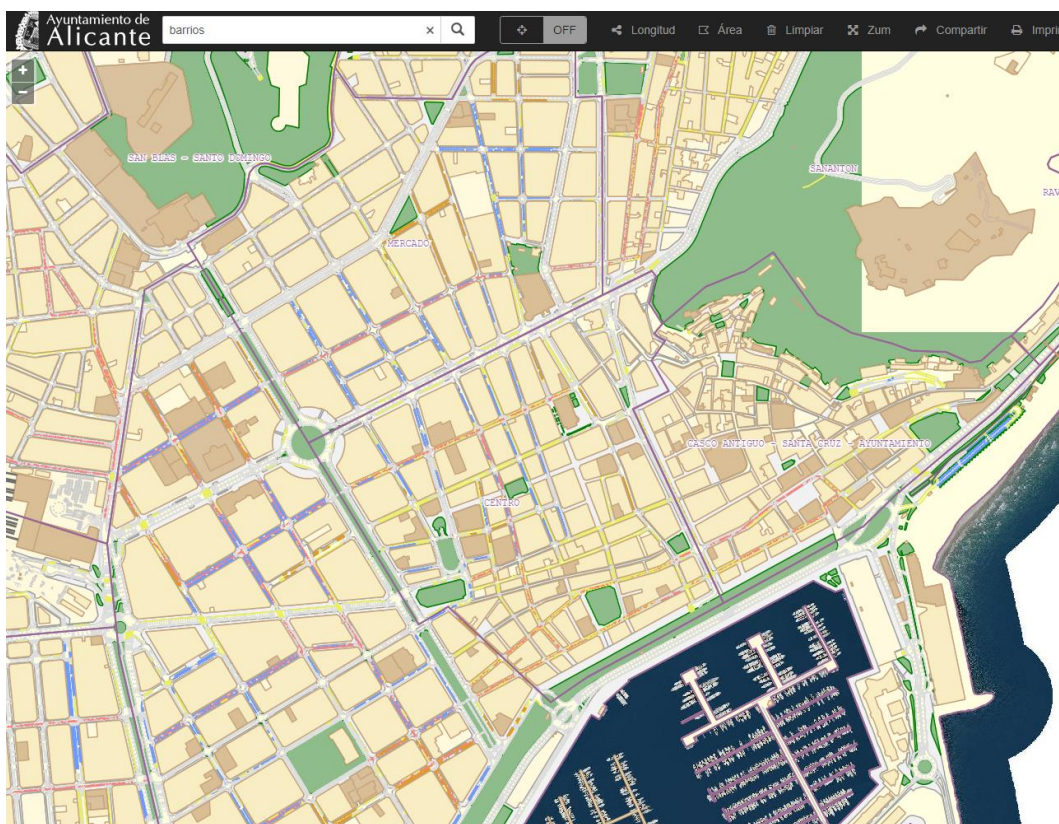


Figura 15. Ampliación figura 14. Fuente: Guía Urbana de Alicante

3.2.2. Geolocalización de áreas

Los distintos polígonos que forman los barrios de la ciudad se geolocalizan para obtener las coordenadas de sus vértices, para ello utilizamos la herramienta *Google Earth Web* [7] que nos permite dibujar los barrios (o zonas de interés) sobre el mapa y descargar sus coordenadas. En la figura 16 se muestra la geolocalización de los barrios *Ensanche – Diputación*, *Centro* y el comienzo de la geolocalización del barrio *Mercado*. El proceso consiste en ir dibujando el polígono que recorre el perímetro de cada uno de los barrios que aparecen en la figura 14. En la figura 17 se muestra el resultado final con las áreas de los barrios geolocalizadas.

Una vez tenemos definidas las áreas de nuestras zonas las exportamos a un archivo con formato *KML* (figura 18), en el cual están las coordenadas de los vértices de los polígonos que forman las zonas en estudio y mediante el *script kmlAgeojson.py* se pasan estas coordenadas a un archivo con formato *geojson* que será leído por el cuadro de mando realizado en *Tableau* [10] y nos permitirá dibujar el mapa con la delimitación de áreas.

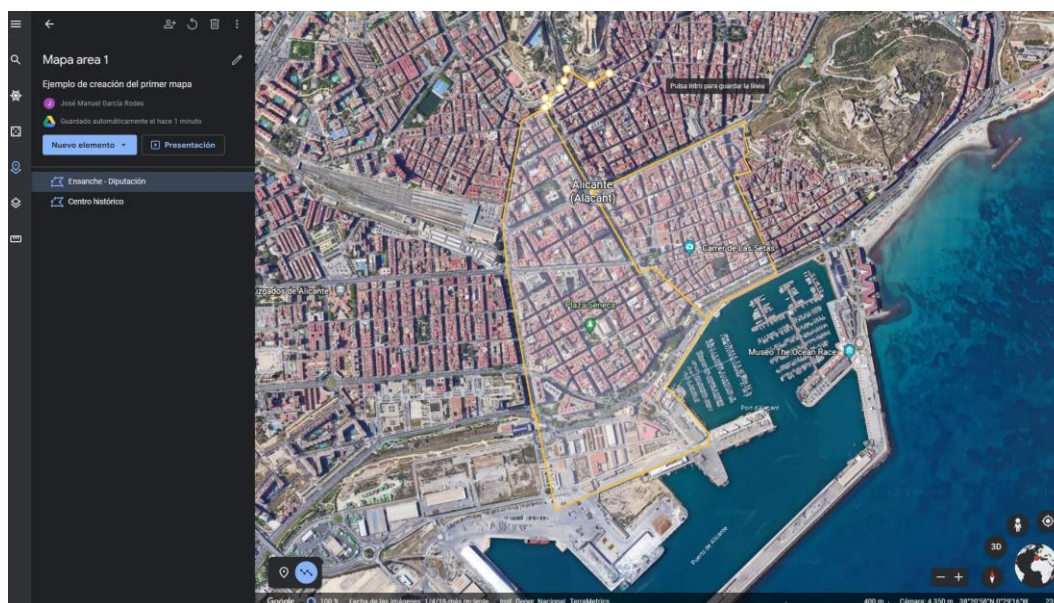


Figura 16. Dibujando los perímetros de los barrios

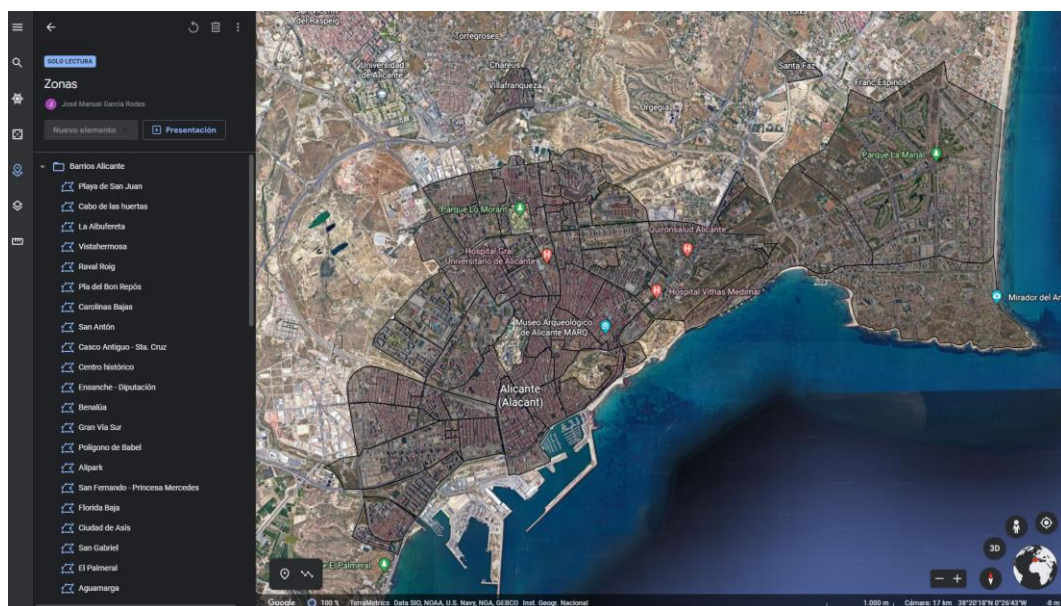


Figura 17. Áreas de estudio

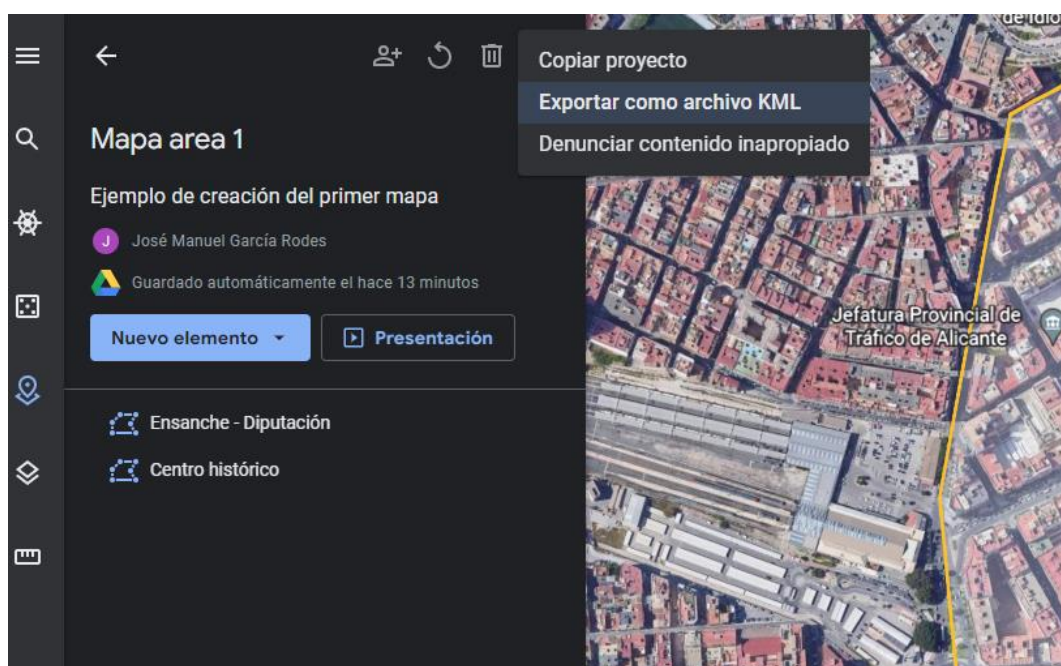


Figura 18. Exportación a KML

3.3. Extracción de datos

La tarea principal en cualquier estudio estadístico es la obtención de datos de calidad y de un volumen suficiente para poder extraer conclusiones válidas. Nuestra fuente de datos se obtiene de internet, de los principales portales inmobiliarios. En los siguientes apartados se describe el proceso de obtención de estos.

3.3.1. Búsqueda en portales inmobiliarios

En internet se pueden encontrar diferentes portales inmobiliarios con anuncios de alquiler de vivienda residencial. La página *similarweb* [30] da información del tráfico recibido por los portales inmobiliarios. Tras comprobar distintos portales como *idealista* [4], *fotocasa* [3], *habitaclia* [31] o *pisos.com* [32] (figura 19), encontramos que los dos principales portales inmobiliarios en España son *idealista* y *fotocasa*. Por ello el presente trabajo lo realizaremos sobre estos dos portales, pudiendo incorporar cuantos queramos en función de los recursos disponibles y la accesibilidad de estos. Añadimos también *pisos.com* para ampliar la muestra.

	id idealista.com	VS.	fc fotocasa.es		h habitaclia.com	VS.	p pisos.com
	id		fc		h		p
Visitas totales	54.2M		16.2M		11.4M		7.2M
Cambio desde el mes pasado	7.62% ▼		5.21% ▲		10.98% ▲		1.20% ▲
Promedio de duración de las visitas	00:08:33		00:05:54		00:05:39		00:04:29
Páginas por visita	20.14		4.95		6.08		3.67
Tasa de rebote	33.37%		40.68%		32.60%		46.13%

Figura 19. Tráfico y compromiso. Septiembre 2022. Fuente: *similarweb.com*

3.3.2. Web scraping

El *web scraping* o raspado web es una técnica utilizada, mediante programas de software, para extraer información de sitios web pudiendo simular cómo navegaría un ser humano en determinadas páginas. El objetivo puede ser transformar contenidos, almacenar datos de la web, reconocer estructuras de código HTML único, recopilar información, hacer minería y análisis de datos, automatizar la creación de enlaces, caza de tendencias, monitorización de la competencia, optimización de precios, etc. Son muchos sus usos, y en general son beneficiosos para cualquier proyecto digital.

Cuando se trata de raspado de páginas (*web scraping*) se habla de descargar información accesible al público en general, que no está protegida por usuario y

contraseña y a la que se puede acceder directamente a través de internet. No obstante, no todo vale, por ejemplo, si se realizan muchas peticiones en un corto periodo de tiempo a una web se podría saturar y provocar que esta dejase de funcionar. Para llenar el posible vacío legal existente acerca del raspado de páginas la Comisión Europea ha creado unas pautas de implementación en las directrices de la política de raspado web de Eurostat [33] que presentamos de forma resumida:

- Respete el protocolo de exclusión de robots.txt y solo siga los enlaces en la medida necesaria para mantener la calidad de las estadísticas;
- Respetar los deseos de los propietarios/as de sitios web según lo establecido en los términos y condiciones en la medida de lo posible para verificar esos términos y el raspado no es esencial para mantener la calidad de las estadísticas como lo implica la ley estadística;
- Identificarse en la cadena de usuario-agente y proporcionar canales de contacto. Esto podría incluir un enlace a una página web que explique el propósito del raspador y qué datos recopila, los detalles de contacto del equipo responsable e información sobre cómo optar por no participar y solicitar que se eliminen los datos extraídos;
- Siga las convenciones estándar de Internet para el scraping, como los estándares establecidos por el consorcio W3C;
- Sea transparente acerca de sus actividades de raspado web, posiblemente proporcionando información en el sitio web asociado;
- Informe a los propietarios/as de sitios web si se extrae una cantidad considerable de datos de forma regular. Este no sería el caso si un sitio web se raspa con una frecuencia baja y no se raspa en profundidad;
- Trate de minimizar la carga en los servidores web mediante:
 - o agregar tiempo de inactividad entre solicitudes,
 - o raspado en un momento del día durante el cual no se espera que el servidor web esté bajo una gran carga,
 - o optimizar la estrategia de raspado para minimizar el número de solicitudes a un dominio;
- Solo extraiga los datos dentro del alcance del mandato legal de la oficina de estadística y no reutilice ni distribuya los datos sin procesar;
- Maneje los datos extraídos de la web de forma segura; Evite el *web scraping* al usar API (Interfaz de programación de aplicaciones) públicas u otras opciones de provisión de datos cuando estén disponibles.

Siguiendo las directrices anteriores que afectan al proyecto se va a realizar *web scraping* con *Python* usando la librería *Requests* junto con *Beautifulsoup*, entre otras. La primera extrae el código de la página web y la segunda crea un árbol con todos los elementos del documento y puede ser utilizada para recopilar información. Cuando el acceso a la información con las librerías anteriores no es posible se puede combinar el uso del software *Selenium* [34] que se encarga de

imitar el comportamiento humano cuando se navega por una web y va accediendo a las distintas partes de las que interesa extraer la información.

La finalidad de este proceso es obtener el número suficiente de información de los anuncios de alquiler para poder nutrir al cuadro de mando y que la información resultante sea representativa del área en estudio. Se puede establecer cualquier periodicidad en la actualización del cuadro de mando siempre y cuando se hayan raspado anuncios suficientes para poder elaborar un informe de calidad. En nuestro caso por el tamaño de la ciudad de Alicante hemos establecido una periodicidad de 3 meses, por lo cual se realizada durante 92 días consecutivos el proceso de raspado anterior.

A continuación, se pasa a describir el proceso de extracción de datos para el portal **fotocasa**, siendo similar para los otros portales en estudio, salvando las particularidades de cada uno.

ESTUDIO DEL PORTAL

En primer lugar, se accede a su web principal, en este caso la de *fotocasa*, se realiza la búsqueda de viviendas en alquiler situadas en el municipio de Alicante y se ordenan por fecha del anuncio decreciente (*Más recientes*). En la figura 20 se muestra el resultado de esta búsqueda. Esta consulta muestra si nos desplazamos hasta el final de la página un total de 29 anuncios de viviendas residenciales en alquiler, siendo la *url* de la consulta la siguiente:

<https://www.fotocasa.es/es/alquiler/viviendas/alicante-alacant/todas-las-zonas/?sortType=publicationDate>

En función de la periodicidad con la que descarguemos los anuncios se tendrá que descargar la siguiente página o no. Por lo general en *fotocasa* hay en torno a 20 anuncios nuevos diarios para el municipio de Alicante, por lo que si se realiza una descarga diaria basta con visitar la primera página, descargándose 29 anuncios de los que posteriormente se quitarán los duplicados.

La *url* anterior es el enlace inicial para la realización del *web scraping*, aporta la información de las 29 *url*'s de los anuncios listados en cada página de la consulta, con las que acceder y extraer los datos que necesitamos de cada anuncio para formar nuestro conjunto de datos. La web de *fotocasa* tiene la particularidad de que cuando se realiza la consulta, solo carga los tres primeros anuncios por lo que, si únicamente se utiliza la librería **requests** para descargar el código *html* de la página como un objeto y posteriormente la librería **beautifulshop** para crear el árbol, solo se accede a la información de los tres primeros anuncios debido a que la página no se carga por completo y no da acceso a los 26 restantes. Para evitar esto se utiliza software libre **Selenium** que nos permite recorrer la página imitando el comportamiento humano, de modo que se programa para que se desplace

hasta el final y así se carguen todos los datos de la página y se puedan extraer las *url's* de cada uno de los anuncios.

EXTRACCIÓN NOMBRES ETIQUETAS

Llegados a este punto, se analiza la estructura de las webs de las que queremos extraer los datos. Se tiene por una parte la web inicial (figura 20) de donde solo interesa la *url* de cada anuncio y por otra cada una de estas *url's* que son similares y de la que se extraerán las características de los anuncios. Haciendo clic con el botón derecho del ratón en la web inicial, se accede al menú *inspeccionar* del cual extraemos los nombres de las etiquetas (*tags*) del código *html* (sopa) donde se encuentran los enlaces a los anuncios que aparecen en la búsqueda inicial (figura 21). Accedemos al primer enlace del anuncio y repetimos el proceso anterior para quedarnos con cada uno de los nombres de las etiquetas que nos interesan para construir el conjunto de datos (figura 22). Estos nombres de etiquetas serán utilizados por el código en *Python* para acceder a los datos de interés dentro de cada *url*.

DESCRIPCIÓN DEL CÓDIGO UTILIZADO

El código *Python* utilizado para la realización del *web scraping* al que se va a hacer referencia a continuación se presenta en el **anexo I**. Por no extender en exceso la memoria, solo se ha puesto la parte general y la referente a *fotocasa*, el código completo para el resto de las webs se puede consultar en el enlace a *GitHub* [35].

En el *script constant.py* se establecen los parámetros para la realización del *web scraping*, en este caso se han puesto las *url's* de las búsquedas ordenadas por más fecha reciente, el número de días consecutivos que se quiere realizar el raspado web, la frecuencia diaria y el número de páginas consultadas por visita a la web. Una vez establecidos los parámetros se ejecuta el *script handler.py* que hace las llamadas a las distintas funciones que realizan el raspado web, obteniéndose un archivo en formato *xlsx* (figura 23) con los resultados del raspado diario para cada uno de los portales en estudio y otro con los anuncios de particulares (figura 5) que son almacenados en *./Monitor/WebScraping/output/* (Ver **anexo II**).

El modo de ejecución es el siguiente:

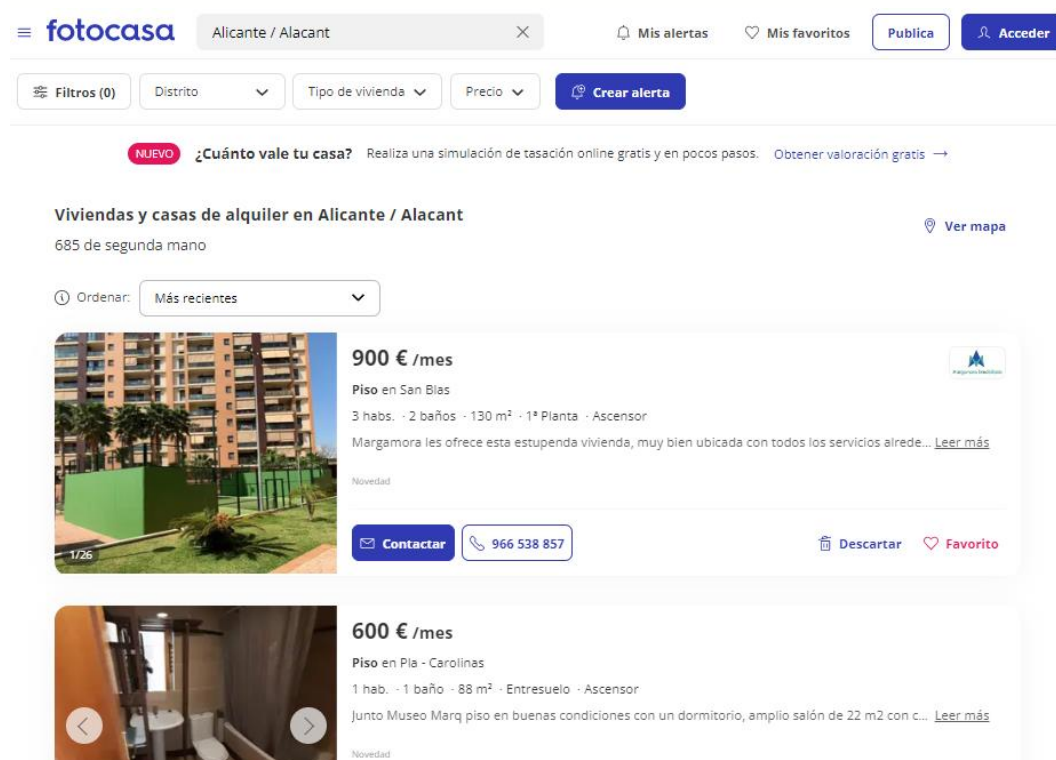
- En la carpeta *./Monitor/WebScraping* abrir el *script constant.py* y ajustar los parámetros. (NOTA: Está ya configurado por defecto para la ciudad de Alicante)
- Instalar las dependencias de *Python* contenidas en el fichero *requirements.txt*, ya sea a mano o a través de:

```
~/Monitorizacion_Alquileres/Monitor$ pip3 install -r requirements.txt
```
- Ejecutar el fichero *handler.py* desde el terminal de *Python* de la siguiente forma:

```
~/Monitorizacion_Alquileres/Monitor/WebScraping$ python handler.py
```

Cuando se ejecuta el *script* anterior, se hace una llamada a la función *human.get* que a su vez llama a la función *scraping_fotocasa* contenida ambas en el *script fotocasa.py*, esta última recibe la *url* de partida y el número de páginas a visitar, seguidamente inicia el navegador (*browser*) ejecutando *Chrome* [36] mediante el software *chromedriver* de *Selenium* tomando la *url* y llama a la función *action_get_page* que hace *scrolling* hasta el final de la página para que se cargue toda devolviendo el código html.

Una vez la acción anterior ha devuelto el código html, se llama a la función *transform_html_to_data* situada en el *script soup_transformer_fotocasa.py* que es la encargada de, utilizando la librería *BeautifulSoup*, crear la “sopa” inicial objeto del cual podemos extraer las *url*’s de los anuncios. Ahora para cada una de estas *url*’s mediante la librería *requests.get* se extrae el código que es transformado en formato *xml* utilizando nuevamente la librería *BeautifulSoup* y se almacena en la variable *soup* para cada uno de los anuncios. Se crea un bucle para extraer las características que se necesitan de la “sopa” (*soup*) de cada uno de los anuncios y se almacenan en una lista que se convertirá en un *dataframe* posteriormente.



fotocasa Alicante / Alacant X Mis alertas Mis favoritos Publica Acceder

Filtros (0) Distrito Tipo de vivienda Precio Crear alerta

NUEVO ¿Cuánto vale tu casa? Realiza una simulación de tasación online gratis y en pocos pasos. Obtener valoración gratis →

Viviendas y casas de alquiler en Alicante / Alacant Ver mapa

685 de segunda mano

Ordenar: Más recientes

900 € /mes
Piso en San Blas
3 hab. · 2 baños · 130 m² · 1ª Planta · Ascensor
Margamora les ofrece esta estupenda vivienda, muy bien ubicada con todos los servicios alrede... [Leer más](#)
Novedad
Contactar 966 538 857 Descartar Favorito

600 € /mes
Piso en Pla - Carolinas
1 hab. · 1 baño · 88 m² · Entresuelo · Ascensor
Junto Museo Marq piso en buenas condiciones con un dormitorio, amplio salón de 22 m2 con c... [Leer más](#)
Novedad

Figura 20. Url inicial de fotocasa para las búsquedas de Alquiler en el municipio de Alicante

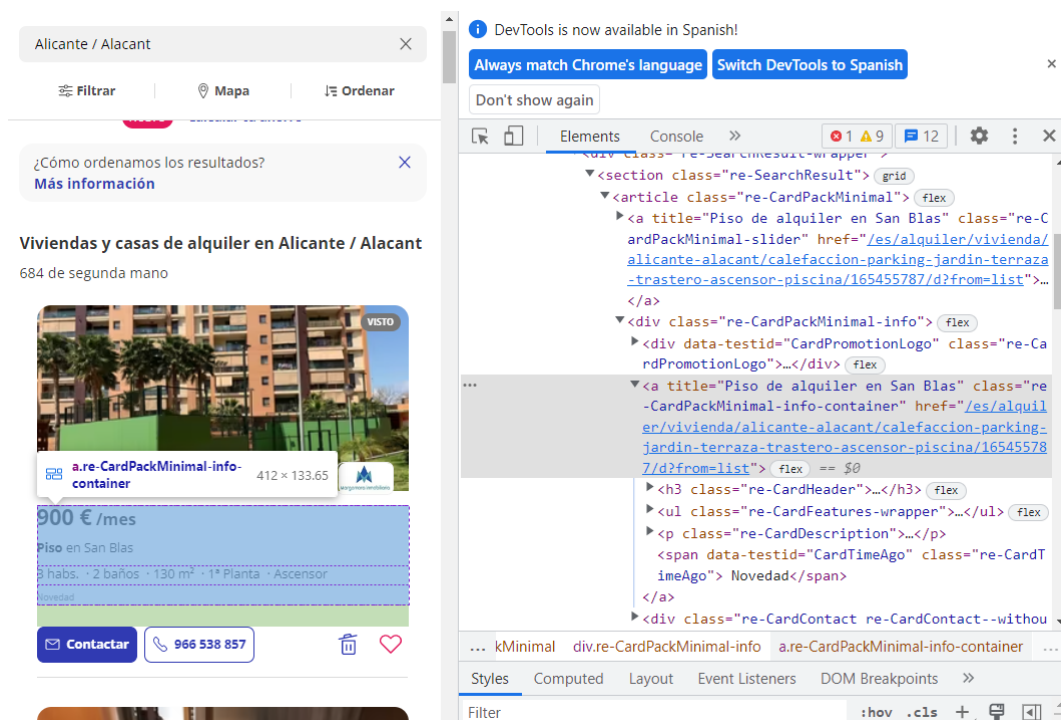
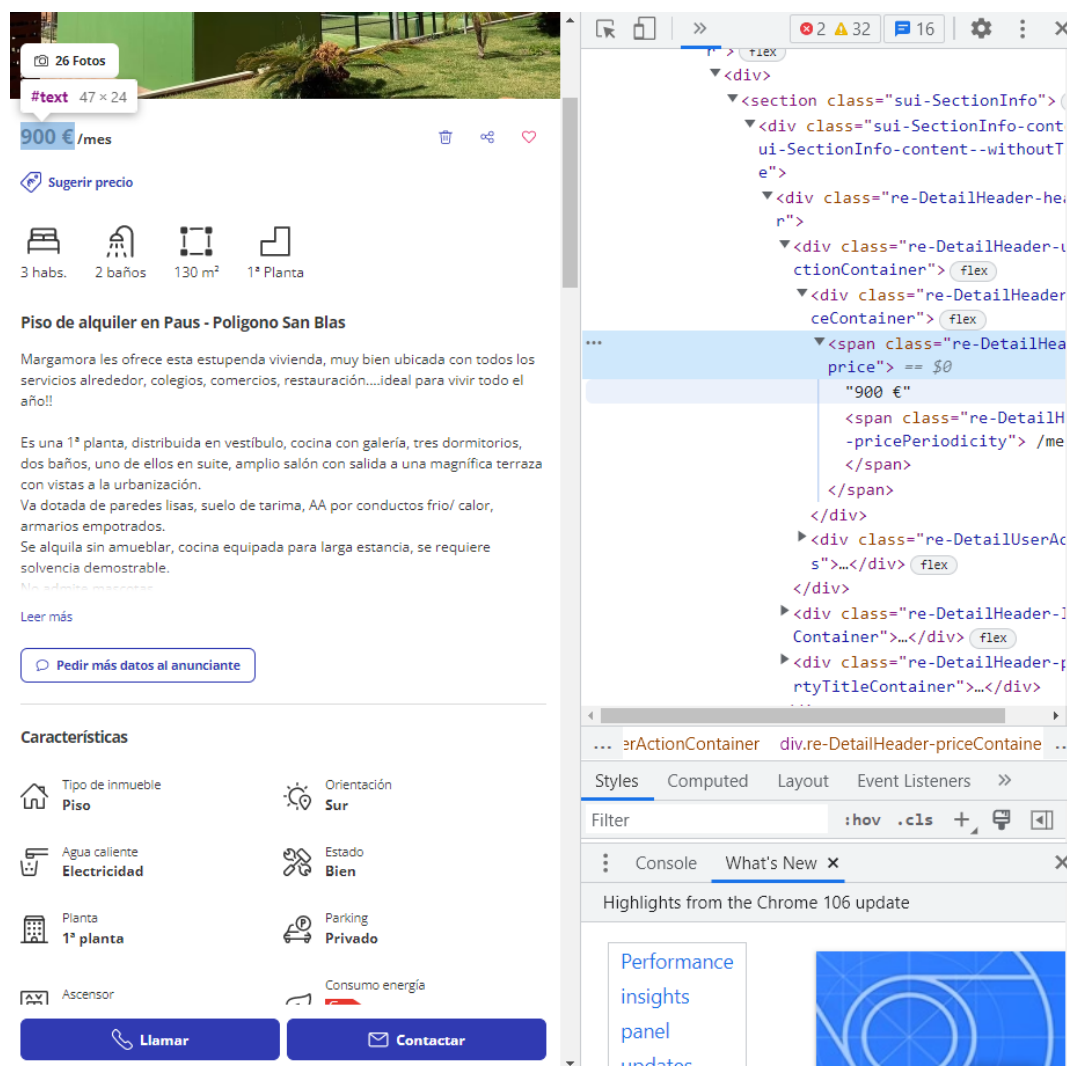


Figura 21. Inspección de la web inicial



The screenshot shows a web browser displaying a rental listing for 'Piso de alquiler en Paus - Poligono San Blas'. The listing includes a photo of a garden, a price of 900 €/mes, and details such as 3 bedrooms, 2 bathrooms, 130 m², and 1st floor. The browser's developer tools are open, showing the HTML structure of the page. The selected element is a `` tag with the class `re-DetailHeader-price`, which contains the text `== $0` and `"900 €"`. The developer tools also show the 'Styles' panel with the `flex` class applied to the element.

Figura 22. Inspección web secundaria

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
2	0	2022.09.22	Fotocasa	400 €	/mes	965 096 10	Inmobiliar	Referenci	https://w	Estudio de 1	baño	25	m²	Terraza	Electrodo	Cocina Eq	Estupendo	estudio para larga	temporada, sin habita			
3	1	2022.09.22	Fotocasa	550 €	/mes	966 534 85	Inmobiliar	Referenci	https://w	Piso de al	3	hab.	1	baño	90	m²	4ª	Planta	Aire acond	Armarios	Electrodo	Balcón
4	2	2022.09.22	Fotocasa	850 €	/mes	865 564 21	Inmobiliar	Referenci	https://w	Piso de al	1	hab.	1	baño	55	m²	Bajos	Balcón	Este herm	Situado e	Situado e	**El prese
5	3	2022.09.22	Fotocasa	1.000 €	/mes	865 564 11	Inmobiliar	Referenci	https://w	Piso de al	3	hab.	1	baño	100	m²	1ª	Planta	Aire acond	Armarios	Calefacci	Terraza
6	4	2022.09.22	Fotocasa	800 €	/mes	965 096 10	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	2	baños	120	m²	Aire acond	Armarios	Calefacci	Terraza	Electrodo	Lavadora
7	5	2022.09.22	Fotocasa	700 €	/mes	965 096 11	Inmobiliar	Referenci	https://w	Piso de al	1	hab.	2	baños	77	m²	2ª	Planta	Aire acond	Armarios	Gres Cerá	Cocina Of
8	6	2022.09.22	Fotocasa	750 €	/mes	966 531 34	RINCONCI	Referenci	https://w	Piso de al	2	hab.	1	baño	75	m²	8ª	Planta	Terraza	Electrodo	Zona Depi	Piscina co
9	7	2022.09.22	Fotocasa	700 €	/mes	Ver teléfc	particular: Laura	https://w	Piso de al	4	hab.	2	baños	154	m²	Aire acond	Terraza	Patio	Electrodo	Horno	Lavadora	
10	8	2022.09.22	Fotocasa	550 €	/mes	966 539 89	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	70	m²	Bajos	Aire acond	Armarios	Gres Cerá	Patio	Electrodo
11	9	2022.09.22	Fotocasa	480 €	/mes	966 539 89	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	65	m²	4ª	Planta	Armarios	Gres Cerá	Electrodo	Horno
12	10	2022.09.22	Fotocasa	750 €	/mes	Ver teléfc	particular: Valeria	https://w	Piso de al	2	hab.	2	baños	110	m²	Aire acond	Armarios	Terraza	Suite - cor	Visitas a partir del 10		
13	11	2022.09.22	Fotocasa	750 €	/mes	865 561 54	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	75	m²	8ª	Planta	Armarios	Jardín Priv	Terraza	Electrodo
14	12	2022.09.22	Fotocasa	600 €	/mes	966 535 88	SALAMAN	Referenci	https://w	Piso de al	2	hab.	2	baños	80	m²	1ª	Planta	Aire acond	Armarios	Gres Cerá	Cocina Of
15	13	2022.09.22	Fotocasa	900 €	/mes	865 561 54	Inmobiliar	Referenci	https://w	Piso de al	4	hab.	2	baños	145	m²	7ª	Planta	Aire acond	Armarios	Calefacci	Jardín Priv
16	14	2022.09.22	Fotocasa	800 €	/mes	865 561 54	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	2	baños	120	m²	7ª	Planta	Armarios	Terraza	Patio	Electrodo
17	15	2022.09.22	Fotocasa	575 €	/mes	865 561 54	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	70	m²	Armarios	Calefacci	Electrodo	Cocina Eq	ARQS GES	Agradable
18	16	2022.09.22	Fotocasa	1.700 €	/mes	Ver teléfc	particular: RUBEN GC	https://w	Piso de al	2	hab.	2	baños	98	m²	Aire acond	Armarios	Calefacci	Terraza	Trastero	Z. Comuni	
19	17	2022.09.22	Fotocasa	1.700 €	/mes	Ver teléfc	particular: Ruben Go	https://w	Piso de al	2	hab.	2	baños	98	m²	Aire acond	Armarios	Calefacci	Terraza	Trastero	Z. Comuni	
20	18	2022.09.22	Fotocasa	650 €	/mes	865 561 54	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	90	m²	1ª	Planta	Aire acond	Armarios	Calefacci	Terraza
21	19	2022.09.22	Fotocasa	750 €	/mes	911 792 25	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	2	baños	73	m²	4ª	Planta	Aire acond	Terraza	Cocina Eq	Nuevo cur
22	20	2022.09.22	Fotocasa	1.600 €	/mes	966 531 36	RESIDE INI	Referenci	https://w	Ático de a	4	hab.	2	baños	286	m²	9ª	Planta	Aire acond	Armarios	Calefacci	Gres Cerá
23	21	2022.09.22	Fotocasa	750 €	/mes	865 560 43	Inmobiliar	Referenci	https://w	Piso de al	3	hab.	1	baño	75	m²	8ª	Planta	Armarios	Terraza	Z. Comuni	Electrodo
24	22	2022.09.22	Fotocasa	1.200 €	/mes	965 095 53	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	60	m²	5ª	Planta	Aire acond	Armarios	Balcón	ALQUILER
25	23	2022.09.22	Fotocasa	950 €	/mes	865 560 51	Inmobiliar	Referenci	https://w	Ático de a	3	hab.	2	baños	70	m²	Calefacci	Terraza	Electrodo	Cocina Eq	DISPONIB	.
26	24	2022.09.22	Fotocasa	650 €	/mes	865 560 39	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	1	baño	49	m²	Armarios	Cocina Of	Redipso	ofrece vivienda	ubicada en una 6	
27	25	2022.09.22	Fotocasa	650 €	/mes	865 564 11	Inmobiliar	Referenci	https://w	Estudio de 1	baño	40	m²	1ª	Planta	Cocina Of	Electrodo	Balcón	Cocina Eq	*TORO SEI	ALQUILER	
28	26	2022.09.22	Fotocasa	700 €	/mes	865 561 62	Inmobiliar	Referenci	https://w	Piso de al	1	hab.	1	baño	50	m²	Aire acond	Armarios	Calefacci	Electrodo	Alarma	Puerta Bli
29	27	2022.09.22	Fotocasa	900 €	/mes	965 094 85	Inmobiliar	Referenci	https://w	Piso de al	3	hab.	2	baños	116	m²	4ª	Planta	Armarios	Balcón	Cocina Eq	Vivienda e
30	28	2022.09.22	Fotocasa	730 €	/mes	865 566 12	Inmobiliar	Referenci	https://w	Piso de al	2	hab.	30	m²	1ª	Planta	Aire acond	Armarios	Gres Cerá	Electrodo	Lavadora	Nevera
31																						

Figura 23. Archivo resultado del raspado. *anuncios_fotocasa 2022.09.22_15.30.12.xlsx*

3.4. Depuración

Una vez terminado el trimestre se pasa al tratamiento de los datos obtenidos en el *web scraping*, para ello ejecutamos el script *Dataset.py* (ver anexo II) situado en la ruta *./Monitor/Debug/* cuyo proceso de ejecución se describe a continuación:

- **Archivo general.** Diariamente se guarda el resultado del raspado anterior para cada portal en su carpeta y trimestre correspondiente, al final del trimestre se unen estos archivos dentro de cada portal utilizando la función *unir_excel()*, creando un único archivo con todos los datos del raspado para el periodo de estudio, en este caso el trimestre. Para el cuadro de mando que se pretende realizar no se van a utilizar todas las variables extraídas, por ello se guarda una copia en la ruta *./Monitor/Resultados/Archivo/* para su posterior utilización en futuros trabajos. Este proceso lo realiza la función *store_data()*.
- **Estandarización de los archivos.** No todos los anuncios publicados en una misma web tienen la misma cantidad de características (variables), teniendo distintas columnas cada registro del archivo de datos, por ello hay que insertar y eliminar columnas y registros para que quede perfectamente tabulado y se tenga cada tipo de dato en su columna correspondiente. Este proceso hay que realizarlo de forma independiente para cada uno de los portales raspados anteriormente, ya que tienen estructuras diferentes. El script *Dataset.py* hace una llamada a la función *debug_fotocasa()*,

`debug_pisoscom()` y `debug_idealista()` situadas en los *scripts* del mismo nombre que depuran los archivos. En el **anexo II** podemos ver el *script* para fotocasa (el resto de los portales se pueden consultar sus *scripts* en el enlace a GitHub [33]).

- **Eliminación de duplicados dentro de cada web.** Una vez hemos unido y estandarizado todos los registros de los anuncios del trimestre para un portal, se procede a eliminar los duplicados dentro del portal, para cada uno de los portales analizados, este proceso lo realiza también las funciones anteriores.
- **Unión de los conjuntos de datos.** En este punto se tienen todos los conjuntos de datos de cada portal con las mismas columnas y es ahí cuando se unen en un único conjunto de datos.
- **Eliminación de duplicados entre webs.** Manteniendo intacto el conjunto de datos de la web con mayor número de registros se eliminan los demás existentes en las otras *web's* que coincidan con estos.
- **Extracción de las direcciones.** En este punto ya se tienen todos los registros de interés, ahora se extraen las direcciones para su geolocalización, guardando la columna *direcciones* en un vector.
- **Geolocalización de las direcciones.** Para llevar a cabo este proceso existen distintas herramientas y *API's*, en este caso se utiliza *Nomiation de la librería Geopy* por ser gratuita (función `coord()`), con *Google Earth* o la *API* de *Google V8* se obtienen mejores resultados, pero para la primera el proceso no se puede automatizar y la segunda es de pago.
- **Asignación de coordenadas a las zonas.** Con las viviendas geolocalizadas se aplica el algoritmo *punto-polígono* [37] para la asignación de estas a sus zonas correspondientes. Este algoritmo consiste en dibujar líneas rectas que pasen por el punto formado por las coordenadas de la vivienda y contar el número de veces que la línea intercepta el polígono del área de una zona en concreto en las que hemos dividido la ciudad. Si el número de veces que es interceptado el polígono es impar, el punto estaría dentro, por el contrario, si es par estaría fuera (figura 24). Este proceso se introduce en un bucle y se repite para cada dirección con cada una de las áreas de las zonas.

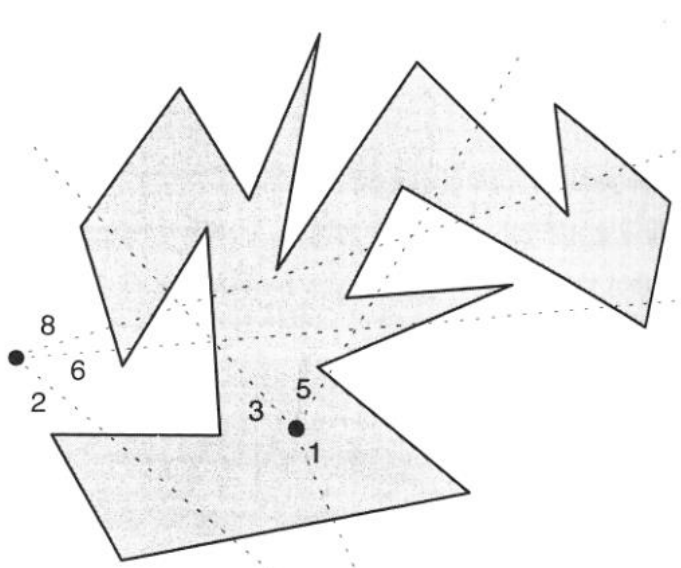


Figura 24. "Semi-line algorithm for determining whether a point is inside a polygon", del libro de Workboys [37]

Este algoritmo se ha implementado en el script `./Geolocalizacion/Geolocalización.py`, que se ejecuta automáticamente haciendo una llamada a la función `geolocalización()`.

- **Eliminación de duplicados dentro de la zona.** Con las viviendas asignadas a la zona se vuelve a realizar el proceso de eliminación de duplicados esta vez dentro de cada zona (función `elimina_dup()`).
- **Valores perdidos (NA).** El conjunto de datos va a ser agrupado en el en sucesivos procesos obteniendo medidas de tendencia central y de dispersión, por lo que los valores faltantes se sustituyen por las medias de la variable utilizando la función `medias()`.
- **Filtrado por los criterios de interés.** Una vez se tiene el conjunto de datos ordenado y depurado se filtra por los criterios de interés, estos se establecen en el script `constant_debug()` que para nuestro caso son: viviendas comprendidas entre 35 y 200 m2 útiles, de no más de 5 dormitorios y 3 baños. Se utiliza la función `val_extremos()` para el filtrado
- **Eliminación de outliers.** Para la eliminación de los valores extremos en primer lugar se aplica el algoritmo *K-nearest neighbors* (KNN) [38], sobre todo el conjunto, y después se eliminan los valores que disten por exceso o defecto más de 1,96 desviaciones típicas de la media dentro de cada zona. Este paso se realiza con la llamada a la función `val_extremos()`.

- **Agrupación de los datos.** Para un menor consumo de recursos se agrupa, mediante la función `estadisticos()`, el conjunto de datos por zonas obteniendo los estadísticos de interés de cada zona, así como para intervalos de rentas y dormitorios con las funciones `intervalos()` y `dormitorios()` respectivamente.
- **Acumulación de datos trimestrales.** Una vez se tienen los datos del periodo, para este caso el trimestre, los unimos con el conjunto de datos que contiene los trimestres anteriores desde que se realiza el estudio y se guardan en la carpeta `./Monitor/Resultados/CuadroMando` para que sean leídos por el cuadro de mando realizado en *Tableau*. Los valores agrupados del periodo comprendido entre el primer trimestre de 2012 y el cuarto trimestre de 2020 se han extraído del repositorio de datos abiertos del Ayuntamiento de Alicante [11].

El modo de ejecución es el siguiente:

- En la carpeta `./Monitor/Debug` abrir el script `constant_debug.py` y ajustar los parámetros. (NOTA: Está ya configurado por defecto con los parámetros de interés para el estudio).
- Estando en la carpeta `Debug`, ejecutar el fichero `Dataset.py`.
`~/Monitorizacion_Alquileres/Monitor/Debug$ python Dataset.py`

Esto realizará automáticamente todas las operaciones enumeradas anteriormente.

Fichero de salida:

Después de una ejecución correcta se generará un archivo en formato `xlsx` (figuras, 25,26,27) con tres pestañas, los datos se encuentran en:

`./Monitor/Resultados/CuadroMando/data.xlsx`

que es leído por el cuadro de mando realizado en *Tableau*.

	A	B	C	D	E	F	G	H	I	J	K
1	Fecha anuncio	Id_Zona	Zona	n	Precio	m2_U	Dormitorios	Baños	P_min	P_max	P_desv
2	2017-12-01 00:00:00	0	ALICANTE	1335	628,132584	94,0165663	2,59267913	1,55093555	190	1400	182,174406
3	2020-03-01 00:00:00	0	ALICANTE	1295	677,878764	89,9606178	2,43892829	1,4941452	345	1300	174,108354
4	2014-06-01 00:00:00	0	ALICANTE	1243	499,389068	93,1559486	2,63344051	1,54180064	150	1300	173,69177
5	2013-12-01 00:00:00	0	ALICANTE	1243	473,582462	87,7047466	2,54867257	1,54947707	220	1300	140,015637
6	2017-06-01 00:00:00	0	ALICANTE	1221	589,621622	93,8551842	2,66442953	1,54642857	230	1500	197,517532
7	2014-03-01 00:00:00	0	ALICANTE	1221	486,843571	91,4373464	2,6027846	1,51395731	200	1250	156,753083
8	2017-09-01 00:00:00	0	ALICANTE	1193	618,215423	94,8966695	2,64563526	1,51300236	200	1400	179,045517
9	2018-06-01 00:00:00	0	ALICANTE	1171	654,690009	93,3800171	2,6190061	1,53105862	275	1600	214,737531
10	2021-03-01 00:00:00	0	ALICANTE	1123	663,030276	91,50935	2,50626118	1,52092609	300	1300	185,656653
11	2019-06-01 00:00:00	0	ALICANTE	1070	687,36729	93,6542056	2,61516588	1,51526718	250	1600	225,243741
12	2021-06-01 00:00:00	0	ALICANTE	1050	666,94381	91,2342857	2,52743022	1,50383142	285	1350	190,185317
13	2013-06-01 00:00:00	0	ALICANTE	1043	483,337748	90,5110259	2,56855225	1,54458293	220	1300	142,895179
14	2013-09-01 00:00:00	0	ALICANTE	1040	486,215969	90,038961	2,57720058	1,54545455	220	1400	140,377766

Figura 25. data (data.xlsx)

	A	B	C	D	E	F	G	H	I	J	K
1	Fecha anuncio	Id_Zona	dormitorio	n	Precio						
2	2020-12-01 00:00:00	0	2 Dorm.	321	650,648						
3	2020-12-01 00:00:00	0	3 Dorm.	429	685,1865						
4	2020-12-01 00:00:00	0	4 o más De	81	754,6296						
5	2020-12-01 00:00:00	0	Estudio o	151	564,5033						
6	2021-03-01 00:00:00	0	2 Dorm.	327	659,3731						
7	2021-03-01 00:00:00	0	3 Dorm.	492	693,5833						
8	2021-03-01 00:00:00	0	4 o más De	119	730,9244						
9	2021-03-01 00:00:00	0	Estudio o	180	539,1389						
10	2021-06-01 00:00:00	0	2 Dorm.	329	672,8571						
11	2021-06-01 00:00:00	0	3 Dorm.	473	675,0127						
12	2021-06-01 00:00:00	0	4 o más De	103	780,7767						
13	2021-06-01 00:00:00	0	Estudio o	134	543,2836						
14	2021-09-01 00:00:00	0	2 Dorm.	324	687,608						

Figura 26. dormitorios (data.xlsx)

	A	B	C	D	E	F	G	H	I	J	K
1	Fecha anuncio	Id_Zona	intervalo	n							
2	2015-09-01 00:00:00	0	< 301 €	29							
3	2015-09-01 00:00:00	0	301 € - 400	245							
4	2015-09-01 00:00:00	0	401 € - 500	449							
5	2015-09-01 00:00:00	0	501 € - 600	325							
6	2015-09-01 00:00:00	0	601 € - 700	278							
7	2015-09-01 00:00:00	0	701 € - 800	91							
8	2015-09-01 00:00:00	0	801 € - 900	44							
9	2015-09-01 00:00:00	0	901 € - 1000	34							
10	2015-09-01 00:00:00	0	> 1000 €	51							
11	2015-12-01 00:00:00	0	< 301 €	76							
12	2015-12-01 00:00:00	0	301 € - 400	412							
13	2015-12-01 00:00:00	0	401 € - 500	492							
14	2015-12-01 00:00:00	0	501 € - 600	340							

Figura 27. Intervalos renta (data.xlsx)

3.5. Visualización

Por último, una vez depurado y agregado el conjunto de datos, este se utiliza para crear una visualización mediante un cuadro de mando [39] construido en *Tableau* (Figuras 29 y 30). En la figura 28 se muestra la estructura de tablas.

En el archivo *data.xlsx* obtenido en el proceso anterior se encuentran tres de las tablas anteriores (*data*, *dormitorios* e *intervalos rentas*) y en la ruta *./Monitor/Resultados/CuadroMando* se encuentra el archivo *zonas7.geojson* donde se han establecido las coordenadas de las zonas en las que se ha dividido la ciudad.

data+ (Varias conexiones)



Figura 28. Estructura de tablas de Tableau

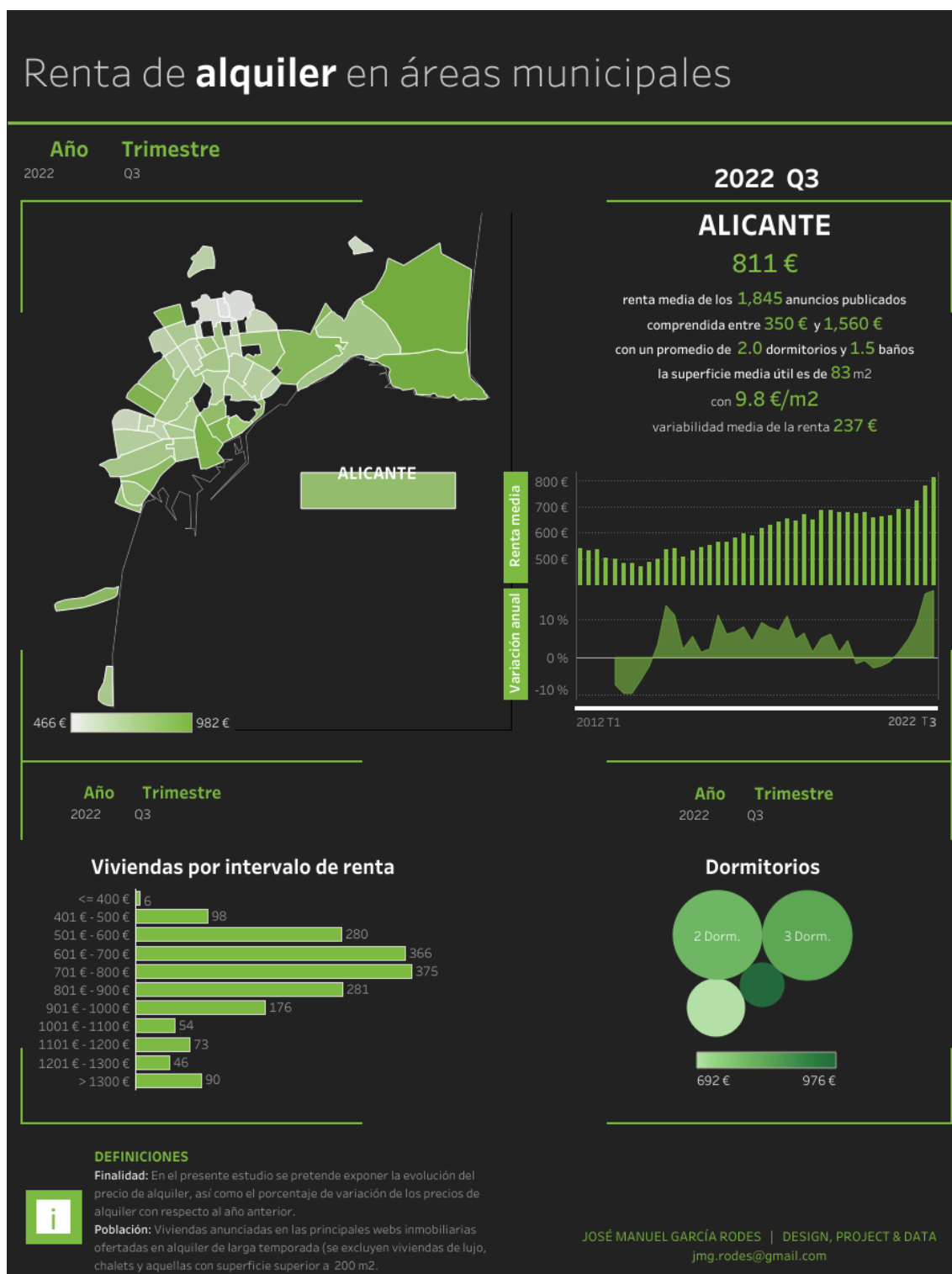
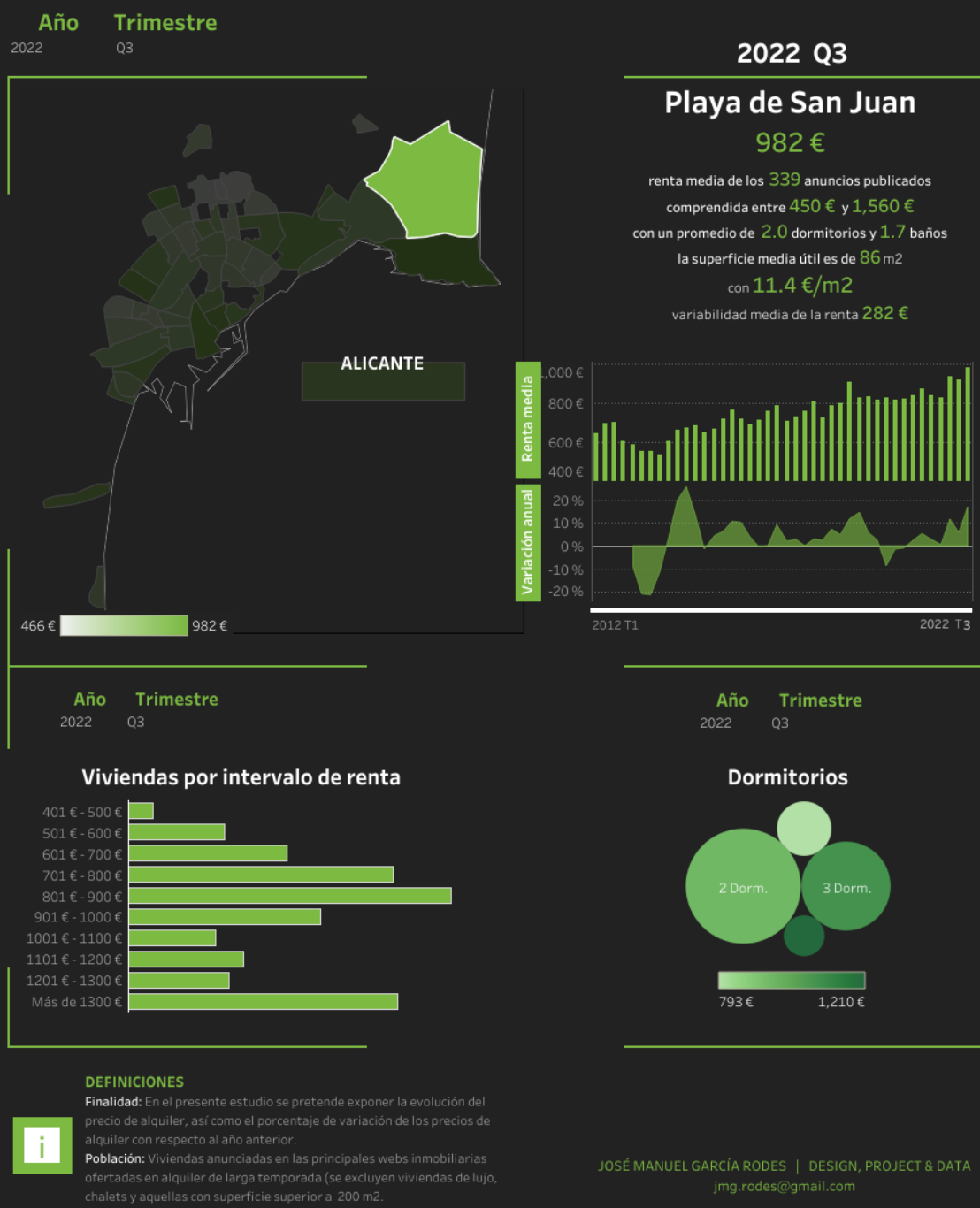


Figura 29. Cuadro de mando

Renta de **alquiler** en áreas municipales



Dormitorios

2 Dorm.

3 Dorm.

793 €

1,210 €

DEFINICIONES

Finalidad:

En el presente estudio se pretende exponer la evolución del precio de alquiler, así como el porcentaje de variación de los precios de alquiler con respecto al año anterior.

Población:

Viviendas anunciadas en las principales webs inmobiliarias ofertadas en alquiler de larga temporada (se excluyen viviendas de lujo, chalets y aquellas con superficie superior a 200 m2).

JOSÉ MANUEL GARCÍA RODES | DESIGN, PROJECT & DATA

jmg.rodes@gmail.com

Figura 30. Cuadro de mando. Playa de San Juan

4. Conclusiones y trabajos futuros

CONCLUSIONES

Con el proceso de *web scraping* se consigue extraer en tiempo real y de forma automática la información relativa a la oferta de vivienda en alquiler anunciada en las principales webs inmobiliarias para una determinada zona que se haya definido previamente, pudiendo discriminar entre la oferta procedente de profesionales y de propietarios/as particulares. Este proceso se podría llevar a cabo de manera manual, visitando diariamente los principales portales inmobiliarios y posteriormente accediendo a cada uno de los anuncios, con el consiguiente coste de tiempo y recursos, con lo que de esta forma se obtiene un primer beneficio.

Por otro lado, tanto si la información se extrae mediante *web scraping* o manual esta suele estar sin estructurar y hay que darle formato para poder extraer resultados valiosos de ella. Esta parte se soluciona con los *scripts*, que depuran y estructuran la información; quitando duplicados, valores extremos o fuera de mercado y completando los valores faltantes.

Por último, mediante el proceso de geolocalización de las direcciones, se obtienen las coordenadas geodésicas de las viviendas anunciadas en alquiler, pudiendo asignarlas a nuestras zonas de interés, para así realizar un estudio estadístico de la evolución de la renta media de alquiler para las zonas de interés, su tasa de variación con respecto al año anterior y demás estadísticos como pueden ser la desviación típica o el rango de alquiler.

Toda esta información recogida se presenta de forma interactiva en un cuadro de mando que puede ser insertado en una web (Anexo III) o consultado directamente en la página web [39] de *Tableau public*. La actualización del cuadro de mando se realiza automáticamente con la periodicidad que se haya establecido, evitando de esta forma la realización de informes estáticos que quedan desactualizados con el tiempo.

TRABAJOS FUTUROS

Durante el proceso de depuración de los datos resultantes del *web scraping*, se guarda un archivo con todas las variables extraídas y que no son utilizadas para la realización del cuadro de mando, como son: amueblamiento, estado de conservación, antigüedad del edificio, reformas, orientación, etc. Toda esta información puede ser utilizada para la realización de un **modelo de regresión lineal** para la estimación de la renta de alquiler, es decir, dadas una serie de características conocidas de la vivienda poder predecir su posible renta de alquiler en el mercado libre.

Otro posible trabajo futuro, sería añadir al *script* que realiza el *web scraping* la descarga y almacenamiento de las fotos de las viviendas anunciadas, para así

crear una base de datos con estas fotos, clasificando las viviendas según estén amuebladas, con reforma, bien conservadas, etc. previo visionado de las fotos. Con esta información se podría realizar un modelo supervisado de *machine Learning* o una red neuronal para clasificar nuevas viviendas anunciadas a partir de sus fotos. Esta información no suele aparecer en todos los anuncios o no es cierta, por lo que se tendría una información más fidedigna que podría ser utilizada entre otras cosas para predecir su valor de mercado.

Por último, utilizando la serie temporal resultante del precio de alquiler de vivienda o la de su tasa de variación, se pueden realizar modelos ARIMA para la predicción de la tendencia de la renta (o tasa de variación) de alquiler o para la estimación de esta.

5. Glosario

Python: Es un lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código, se utiliza para desarrollar aplicaciones de todo tipo. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.

Librería de Python: En programación una librería responde al conjunto de funcionalidades que permiten al usuario llevar a cabo nuevas tareas que antes no se podían realizar. Es decir, las librerías de Python responden al conjunto de implementaciones que permiten codificar este lenguaje, con el objeto de crear una interfaz independiente.

Script: En informática, un script, es una secuencia de comandos o guion (traduciendo desde inglés) es un término informal que se usa para designar un lenguaje de programación que se utiliza para manipular, personalizar y automatizar las instalaciones de un sistema existente.

Web scraping o raspado web: es una técnica utilizada mediante programas de software para extraer información de sitios web. Usualmente, estos programas simulan la navegación de un humano en la World Wide Web ya sea utilizando el protocolo HTTP manualmente, o incrustando un navegador en una aplicación.

Machine learning o aprendizaje automatizado: es el subcampo de las ciencias de la computación y una rama de la inteligencia artificial, cuyo objetivo es desarrollar técnicas que permitan que las computadoras aprendan.

Geojson: Es un formato estándar abierto diseñado para representar elementos geográficos sencillos, junto con sus atributos no espaciales, basado en *JavaScript Object Notation*. El formato es ampliamente utilizado en aplicaciones de cartografía en entornos web al permitir el intercambio de datos de manera rápida, ligera y sencilla.

Scrolling: Desplazamiento dentro de una página web.

Bucle: Un bucle o ciclo, en programación, es una secuencia de instrucciones de código que se ejecuta repetidas veces, hasta que la condición asignada a dicho bucle deja de cumplirse.

Dataframe: Conjunto de datos.

API: La interfaz de programación de aplicaciones, es un conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción.

ARIMA: En estadística y econometría, en particular en series temporales, un modelo autorregresivo integrado de promedio móvil o ARIMA (acrónimo del inglés

autoregressive integrated moving average) es un modelo estadístico que utiliza variaciones y regresiones de datos estadísticos con el fin de encontrar patrones para una predicción hacia el futuro. Se trata de un modelo dinámico de series temporales, es decir, las estimaciones futuras vienen explicadas por los datos del pasado y no por variables independientes.

6. Bibliografía

- [1] *MINISTERIO DE TRANSPORTES, MOVILIDAD Y AGENDA URBANA. Índice alquiler de vivienda.* [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.mitma.gob.es/vivienda/alquiler/indice-alquiler>
- [2] *Observatorio Valenciano de Vivienda (OVV). Comunitat Valenciana.* [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://habitatge.gva.es/es/web/vivienda-y-calidad-en-la-edificacion/observatorio-valenciano-de-vivienda>
- [3] *Fotocasa.* [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://prensa.fotocasa.es/informes/>
- [4] *Idealista.* [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.idealista.com/sala-de-prensa/informes-precio-vivienda/>
- [5] *Programa de alquiler asequible de vivienda. Patronato Municipal de la Vivienda de Alicante.* [en línea] [fecha de consulta: 30/12/2022]. Disponible en: <https://www.alicante.es/es/contenidos/programa-alquiler-asequible-vivienda>
- [6] *Observatorio Municipal de Vivienda en Alquiler de Alicante.* [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.alicante.es/es/contenidos/observatorio-municipal-vivienda-alquiler-alicante>
- [7] *Google earth.* [en línea] [fecha de consulta: 26 de octubre de 2022]. Disponible en: <https://earth.google.com/>
- [8] *Python Software Foundation.* Python [software]. Versión 3.10.7.
- [9] *Geopy* [software]. Versión 2.3.0.
- [10] *Tableau.* Tableau Public [software]. Versión 2022 T2.
- [11] *Portal integral de información analítica y datos abiertos.* Ayuntamiento de Alicante. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: https://datosabiertos.alicante.es/search/field_topic/urbanismo-y-vivienda-6
- [12] *Disposición adicional segunda. Sistema de índices de referencia del precio del alquiler de vivienda. Real Decreto-ley 7/2019, de 1 de marzo, de medidas urgentes en materia de vivienda y alquiler. «BOE» núm. 55, de 5 de marzo de 2019.*
- [13] *Ley 11/2020, de 18 de septiembre, de medidas urgentes en materia de contención de rentas en los contratos de arrendamiento de vivienda y de modificación de la Ley 18/2007, de la Ley 24/2015 y de la Ley 4/2016, relativas a la protección del derecho a la vivienda. «BOE» núm. 258, de 29 de septiembre de 2020*
- [14] *Observatorio de vivienda y suelo.* [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://www.mitma.gob.es/arquitectura-vivienda-y-suelo/urbanismo-y-politica-de-suelo/estudios-y-publicaciones/observatorio-de-vivienda-y-suelo>

- [15] *Índex de referència de preus de lloguer*. [en línea] [fecha de consulta 15 de octubre de 2022]. Disponible en: <http://agenciahabitatge.gencat.cat/indexdelloguer/>
- [16] *Observatorio de vivienda asequible*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://provivienda.org/observatorio/>
- [17] *Observatori Valencià d'Habitatge (OVH)*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://habitatge.gva.es/va/web/vivienda-y-calidad-en-la-edificacion/observatorio-valenciano-de-vivienda>
- [18] *Observatorio vasco de la vivienda*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://www.etxebide.euskadi.eus/x39-ovhome/es/>
- [19] *Consulta de la estadística del mercado del alquiler*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://apps.euskadi.eus/web01-apetxebi/es/ad23aConsultaWar/ema?locale=es>
- [20] *Observatorio de vivienda y suelo de Cantabria*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://www.observatoriovivienda.cantabria.es/>
- [21] *Observatorio de vivienda de Galicia*. [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.observatoriodavivenda.gal/es>
- [22] *Observatorio de la Vivienda de Andalucía*. [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.juntadeandalucia.es/organismos/fomentoarticulaciondelterritorioyvienda/consejeria/organos-colegiados/60411.html>
- [23] *Observatorio de vivienda. El Español*. [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.elespanol.com/invertia/observatorios/vivienda/>
- [24] *Índice Inmobiliario de precios Fotocasa*. [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.fotocasa.es/indice-precio-vivienda>
- [25] *Sala de prensa de Idealista*. [en línea] [fecha de consulta: 8 de octubre de 2022]. Disponible en: <https://www.idealista.com/sala-de-prensa/>
- [26] *Observatorio de vivienda de la UPV*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://observa.webs.upv.es/>
- [27] *Observatorio de vivienda de la UPV*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://observa.webs.upv.es/vivo/>
- [28] *Observatorio de la Vivienda. Vitoria-Gasteiz*. [en línea] [fecha de consulta: 15 de octubre de 2022]. Disponible en: <https://www.e21z.es/observatorio/es/>
- [29] *Guía Urbana de Alicante*. [en línea] [fecha de consulta: 26 de octubre de 2022]. Disponible en: <https://guiaurbana.alicante.es/>
- [30] *similarweb*. [en línea] [fecha de consulta: 02 de noviembre de 2022]. Disponible en: <https://www.similarweb.com/es/>
- [31] *habitaclia*. [en línea] [fecha de consulta: 02 de noviembre de 2022]. Disponible en: <https://www.habitaclia.com/>
- [32] *Pisos.com*. [en línea] [fecha de consulta: 02 de noviembre de 2022]. Disponible en: <https://www.pisos.com/>

- [33] *DIME - Directors of Methodology. Item 04 – Web scraping policy*. [en línea] [fecha de consulta: 27 de octubre de 2022].
https://ec.europa.eu/eurostat/cros/content/item-04-web-scraping-policy_en
- [34] *Selenium*. [en línea] [fecha de consulta: 26 de septiembre de 2022]. Disponible en: <https://www.selenium.dev/>
- [35] *GitHub*. [en línea] [fecha de consulta: 25 de diciembre de 2022]. Disponible en: https://github.com/jmg-rodes/Monitorizacion_Alquileres
- [36] *Google Chrome*. [en línea] [fecha de consulta: 26 de septiembre de 2022]. Disponible en: https://www.google.com/intl/es_es/chrome/
- [37] *GIS, A computing Perspective*, Michel F. Workboys, Editorial Taylor & Francis, London, 1995.
- [38] *Real Python*. [en línea] [fecha de consulta: 28 de septiembre de 2022]. Disponible en: <https://realpython.com/knn-python/>
- [39] *Monitorización renta de alquiler en áreas municipales*. [en línea] [fecha de consulta: 24 de diciembre de 2022]. Disponible en: <https://public.tableau.com/app/profile/jos.manuel6318/viz/Monitorizacinrentadealquiler/Alicante?publish=yes>

7. Anexos

7.1. Anexo I. Web scraping

cosntant.py

```
##### constant.py #####

##### CONSTANTES WEBSCRAPING (NO MODIFICAR) #####

ENCODING_UTF8 = 'utf-8'

HTTP_STATUS_OK = 200

REQUEST_TIMEOUT = 5

SELENIUM_SLEEP_TIME = 5

POS_FIRST = '1'
POS_SECOND = '2'

DATETIME_FORMAT = '%Y-%m-%d %H:%M:%S'

##### CRITERIOS DE BUSQUEDA #####

portales = ['pisoscom', 'fotocasa', 'idealista']

# BUSQUEDAS PARA LA CIUDAD DE ALICANTE
# Se realiza la búsqueda deseada en el portal y se copia la url
BASE_URL_pisoscom = 'https://www.pisos.com/alquiler/pisos-alicante_alacant/
feharecientedesde-desc/'
BASE_URL_fotocasa = 'https://www.fotocasa.es/es/alquiler/viviendas/alicante-
alacant/todas-las-zonas/l?sortType=publicationDate'
BASE_URL_idealista = 'https://www.idealista.com/alquiler-viviendas/alicante-
alacant-alicante/?ordenado-por=fecha-publicacion-desc'

DEFAULT_CITY = 'Alicante / Alacant'
DIAS = 1 # Días que queremos que se realice el webscraping
FRECUENCIA = 1 # Frecuencia diaria
DEFAULT_PAGES = 1 # Páginas diarias por webscraping

#####
```

requeriments.txt

```
requests==2.24.0
beautifulsoup4==4.9.3
selenium==3.141.0
geopy==2.3.0
kml2geojson==5.1.0
pyod==1.0.7
glob2==0.7
```

handler.py

```
##### handler.py #####

import particulares as provider_particulares
import fotocasa as provider_fotocasa
import idealista as provider_idealista
import pisoscom as provider_pisoscom

import constant as constant
import sys
from constant import *
import time

city = constant.DEFAULT_CITY
pages = constant.DEFAULT_PAGES

print('- City: ' + city)
print('- Pages: ' + str(pages))
print()
print('Launching...')

requests = 0
anuncios_pisoscom = []
anuncios_fotocasa = []
anuncios_idealista = []

# Días consecutivos para los que se realiza el webscraping
for i in range(DIAS) :
    # Número de veces que se visita la página x día
    for run in range(FRECUENCIA):
        try:
            provider_pisoscom.human_get(constant.BASE_URL_pisoscom, pages,
                                         run, anuncios_pisoscom)
        except:
            print('Ha fallado pisoscom')
        try:
            provider_fotocasa.human_get(constant.BASE_URL_fotocasa, pages,
```

```

run, anuncios_fotocasa)

except:
    print('Ha fallado fotocasa')
try:
    provider_idealista.human_get(constant.BASE_URL_idealista,
                                pages, run, anuncios_idealista)
except:
    print('Ha fallado idealista')
try:
    provider_particulares.anuncios_get(run)
except:
    print('Ha fallado particulares')

# Monitor the requests
requests += 1
print('Request: {}; Día: {}; Run: {}'.format(requests, i+1, run+1 ))

# número de segundos por webscraping (86400 para que lo haga cada
# 24 horas)
#segundos = 86400/FRECUENCIA
segundos = 5
time.sleep(segundos)

#provider_fotocasa.debug_dataset(df_fotocasa)
print ('***** SE HAN REALIZADO {} VISITAS A LAS WEBS *****'.format(i+1))

#####

```

particulares.py

```

##### particulares.py #####
from constant import *
from datetime import datetime
import pandas as pd

def anuncios_get(run):
    '''
    Se unen los anuncios de particulares en un dataframe
    '''

    requests = 0

    df_particulares = pd.DataFrame()

```

```
# Portales para los que se realiza el webscraping
for portal in portales:
    # Establecemos la fecha
    today = datetime.today()
    fecha = today.strftime("%Y.%m.%d")
    try:
        # Lectura xls del portal
        df_part = pd.read_excel(
            './ouput/particulares/' + portal + '/anuncios_' +
            portal + '_particulares' + '_' + str(run+1) + '_' + fecha + '.xlsx')
        #df_part = df_part.drop(['Unnamed: 0'], axis=1)
        df_particulares = pd.concat([df_particulares, df_part])

    except:
        print('Hay {} anuncio de {}'.format(run, portal))

# Monitor the requests
requests += 1
print('Request: {}; Portal: {}; Run: {}'.format(
    requests, portal, run+1 ))

# Almacenaje de los anuncios de particulares
df_particulares.columns = ['Anuncio', 'Fecha', 'Portal', 'Renta',
                           'Titular', 'Web', 'Dirección']
pd.DataFrame(df_particulares).to_excel(
    './Resultados/Particulares/anuncios_particulares_' +
    fecha + '_' + str(run+1) + '.xlsx', encoding='utf-8-sig')
print ('*** SE HAN REALIZADO {} VISITAS A LAS WEBS ***'.
    format(run+1))

#####
```

fotocasa.py

```
##### fotocasa.py #####

import logging
import time
import os
from selenium.webdriver.common.keys import Keys
import soup_transformer_fotocasa as transformer
from selenium.webdriver.common.by import By
from constant import *
```

```

from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.action_chains import ActionChains
from selenium import webdriver
from datetime import datetime
import pandas as pd

# Uso de selenium para cargar una web
# IMPORTANTE: El driver incluido es para Chrome versión 108
# Si se usa otra versión hay que descargar de nuevo el driver
# de https://chromedriver.chromium.org/downloads

def human_get(url:str, number_of_pages:int, run:int, anuncios_fotocasa):

    '''
    Función principal que extrae las características del anuncio y las
    almacena en un archivo.xlsx
    '''

    # Establecemos la fecha
    today = datetime.today()
    fecha_hora = today.strftime("%Y.%m.%d_%H.%M.%S")
    fecha = today.strftime("%Y.%m.%d")
    trim = str((today.month-1)//3)+1
    anyo = str(today.year)

    # Se acumulan todos los anuncios de cada día en una lista
    anuncios_fotocasa_dia,anuncios_fotocasa_dia2,anuncios_fotocasa_dia3 =
        scraping_fotocasa(url,number_of_pages)

    anuncios_fotocasa1 = []
    anuncios_fotocasa2 = []
    anuncios_fotocasa3 = []
    for anuncio in anuncios_fotocasa_dia:
        anuncios_fotocasa1.append(anuncio)
    for anuncio in anuncios_fotocasa_dia2:
        anuncios_fotocasa2.append(anuncio)
    for anuncio in anuncios_fotocasa_dia3:
        anuncios_fotocasa3.append(anuncio)

    # Se crea un dataframe con los anuncios acumulados
    df_fotocasa1 = pd.DataFrame(anuncios_fotocasa1)
    df_fotocasa2 = pd.DataFrame(anuncios_fotocasa2)
    df_fotocasa3 = pd.DataFrame(anuncios_fotocasa3)

    # Se unen los dataframes
    df_fotocasa = pd.concat([df_fotocasa1,df_fotocasa2,df_fotocasa3],axis=1,)

```

```

columns_names = df_fotocasa.columns.values
columns_names = list(columns_names)
columns_names = list(range(0, len(columns_names)))
df_fotocasa.columns = columns_names

# Se quitan los duplicados
df_fotocasa = df_fotocasa.drop_duplicates(
    df_fotocasa.columns[df_fotocasa.columns.isin([7])])
df_fotocasa = df_fotocasa.drop_duplicates(
    df_fotocasa.columns[df_fotocasa.columns.isin([8,9,11,13,15])])

# Se guardan los datos en un archivo en la carpeta del trimestre
os.makedirs("./ouput/fotocasa/"+anyo+"/"+T"+trim, exist_ok=True)
df_fotocasa.to_excel("./ouput/fotocasa/"+anyo+"/"+T"+trim+"/"+
                    "anuncios_fotocasa "+ fecha_hora +".xlsx",
                    encoding="utf-8-sig")

# Anuncios particulares del día
df_fotocasa_particulares = pd.DataFrame(
    df_fotocasa)[pd.DataFrame(
        df_fotocasa)[5].str.contains('particular:')]
df_fotocasa_particulares = pd.DataFrame(
    df_fotocasa_particulares)[pd.DataFrame(
        df_fotocasa_particulares)[0].str.contains(fecha)]
df_fotocasa_particulares = df_fotocasa_particulares.iloc[:, [0,1,2,6,7,8]]
df_fotocasa_particulares.columns = ['Fecha', 'Portal', 'Renta', 'Titular',
                                    'Web', 'Dirección' ]

# Se crea un excel con los anuncios de particulares
pd.DataFrame(df_fotocasa_particulares).to_excel(
    './ouput/particulares/fotocasa/anuncios_fotocasa_particulares_'+
    str(run+1) + '_' + fecha +'.xlsx', encoding='utf-8-sig')

def scraping_fotocasa (url_link, number_of_pages: int):

    """
    Función que recibe un link de fotocasa y extrae las características
    del anuncio.
    """

    # Preparando la monitorización del bucle
    requests = 0

    # Init

```



```

data_list = []
data_list2 = []
data_list3 = []
datetime_now = datetime.now()

browser = init_browser()
browser.maximize_window()

# Ir a la URL
browser.get(url_link)
time.sleep(SELENIUM_SLEEP_TIME)

# Obtención del html y transformación de los datos
html = action_get_page(browser)
transformer.transform_html_to_data(html, data_list, data_list2,
                                   data_list3, datetime_now)

# Ir a la página siguiente y desplazarse y repetir el proceso
j = 1
while j < number_of_pages:
    print('página: {}'.format(j))
    action_next_page(browser)
    print('action_get_page')
    time.sleep(20)
    html = action_get_page(browser)
    print('transform_html_to_data')
    transformer.transform_html_to_data(html, data_list, datetime_now)
    print('#####')
    #print(data_list[0])
    j += 1

# Stop selenium
browser.quit()

return data_list, data_list2, data_list3

def init_browser():
    # Apertura del navegador
    browser = webdriver.Chrome(executable_path=r'./selenium/chromedriver.exe')
    return browser

# Acción para desplazarse hacia abajo y obtener el código html
def action_get_page(browser):
    elem = browser.find_element(By.TAG_NAME, "body")

```

```
# Desplácese hasta el final de la página
no_of_pagedowns = 30

while no_of_pagedowns:
    elem.send_keys(Keys.PAGE_DOWN)

    time.sleep(1)
    no_of_pagedowns -= 1

# Ir al paginador
ActionChains(browser).move_to_element(
    browser.find_element(By.XPATH,
build_path_for_paginator())) .perform()
time.sleep(SELENIUM_SLEEP_TIME)

html = browser.page_source

return html

# Acción para ir a la página siguiente
def action_next_page(browser):
    next_page = browser.find_elements(By.XPATH, build_path_for_next_page())
    next_page = next_page[len(next_page)-1]

    next_page.click()

    time.sleep(SELENIUM_SLEEP_TIME)

# Obtener el xpath para la página siguiente
def build_path_for_next_page():
    return '//*[@id="App"]/div[2]/div[1]/main/div/div[4]/ul/li[7]/a'

# Obtener el xpath para el paginador
def build_path_for_paginator():
    return '//div[@class="re-Pagination"]'

#####
```

soup_transformer_fotocasa.py

```
##### soup_transformer_fotocasa.py #####

import logging
import requests
from datetime import datetime
from bs4 import BeautifulSoup

def transform_html_to_data(html, data_list, data_list2, data_list3, datetime_now):

    '''
    Función que recibe el html inicial, extrae las url's de los anuncios y
    devuelve tres listas con las variables de interés extraídas de los
    anuncios
    '''

    soup = BeautifulSoup(html, 'html.parser')

    # Extraemos los links 'a' de la sopa
    a_tag = []
    a_tag = soup.find_all('a',
                           attrs={'class': 're-CardPackMinimal-info-
container'})

    # Guardo las url's en una lista
    url_list = []
    for link in a_tag:
        url_list.append(link.get('href'))

    # Para cada url de la lista extraemos los datos del anuncio
    for i in range(len(url_list)):

        try:
            url_link_sec='https://www.fotocasa.es' + url_list[i]

            print ('Anuncio: {}; \n URL: https://www.fotocasa.es{}'.format(i, url_list[i]))

            req = requests.get(url = url_link_sec)
            soup = BeautifulSoup(req.text, "lxml")

            # Asignación de variables
            euro=[]
            caract=[]
            zona=[]
            desc =[]
            otras_caract =[]
```

```

extras = []
contacto = []
fecha_hora = []
today = datetime.today()
fecha_hora.append(today.strftime("%Y.%m.%d"))
portal = []
url_sec_list = []

portal.append('Fotocasa')

url_sec_list.append('https://www.fotocasa.es' + url_list[i])

euro_tag=soup.find('span','re-DetailHeader-price')
if euro_tag is not None:
    for st in euro_tag.stripped_strings:
        euro.append(st)

caract_tag=soup.find('ul','re-DetailHeader-features')
if caract_tag is not None:
    for st in caract_tag.stripped_strings:
        caract.append(st)

zona_tag=soup.find('h1','re-DetailHeader-propertyTitle')
if zona_tag is not None:
    for st in zona_tag.stripped_strings:
        zona.append(st)

desc_tag=soup.find('div','re-DetailFeaturesList')
if desc_tag is not None:
    for st in desc_tag.stripped_strings:
        desc.append(st)

otras_caract_tag=soup.find('section','re-DetailFeaturesList')
if otras_caract_tag is not None:
    for st in otras_caract_tag.stripped_strings:
        otras_caract.append(st)

extras_tag=soup.find('ul','re-DetailExtras-list')
if extras_tag is not None:
    for st in extras_tag.stripped_strings:
        extras.append(st)

contacto_tag=soup.find('div','re-ContactDetailPhoneContainer')
if contacto_tag is not None:
    for st in contacto_tag.stripped_strings:

```

```

        contacto.append(st)
        if len(contacto) <= 2:
            contacto.insert(1, 'Inmobiliaria')

        data_list.append(fecha_hora + portal + euro + contacto +
                        url_sec_list + zona + caract + otras_caract)
        data_list2.append(desc)
        data_list3.append(extras)

    except:
        logging.error('The url [' + url_list[i] + '] is advertasing')

    return data_list, data_list2, data_list3

#####

```

7.2. Anexo II. Depuración del conjunto de datos

constant_debug.py

```

##### constant_debug.py #####

##### CONSTANTES DEPURACIÓN DATASET #####

anyo = '2022'
trim = 'T4'

ciudad_Nominatim = "%s, Alacant / Alicante, l'Alacantí, Alacant / Alicante"

ciudad_google_earth = 'Alicante / Alacant'

provincia = 'Alicante'
pais = 'España'

# Criterios de filtrado del dataset para incluir el anuncio
m2_sup = 200 # m2 útiles máximo
m2_inf = 30 # m2 útiles mínimo
Dorm = 5 # máximo de dormitorios
banyos = 3 # máximo de baños

#####

```

Dataset.py

```
##### Dataset.py #####

import debug_fotocasa as fotocasa_dep
import debug_pisoscom as pisoscom_dep
from constant_debug import *
import Geolocalizacion.Geolocalizacion as geo
import pandas as pd
import numpy as np
import os
import glob
from datetime import datetime
from pyod.models.knn import KNN

import kml2geojson
# Llegados a este punto explicar las distintas opciones de geolocalización
from geopy.geocoders import Nominatim
#from geopy.geocoders import GoogleV3 # ---> De pago 0,005 cts la dirección

# Se muestra el año y el trimestre del informe

print(f"Año: {anyo}. Trimestre: {trim}")

##### FUNCIONES #####

def unir_excel(anyo, trim, portal, drop):

    '''
    Escribimos un loop que irá a través de cada uno de los nombres de archivo
    a través de globbing y el resultado final será la lista dataframes
    '''

    xlsx_files = glob.glob("../WebScraping/ouput/"+portal+"/"+anyo+"/"+
                           trim+"/"+ "*.xlsx")

    list_data = []

    for filename in xlsx_files:
        data = pd.read_excel(filename, thousands='.')
        list_data.append(data)

    anuncios = pd.concat(list_data, ignore_index=True).sort_values(
        1, ascending=True)

    # Primer filtrado para la eliminación de duplicados entre webs
    anuncios = anuncios.drop_duplicates()
```



```

        anuncios.columns[anuncios.columns.isin(drop)]

    return anuncios

def store_data(df, anyo, trim, portal):
    df.to_excel('../Resultados/Archivo/anuncios_'+portal+'_' + anyo +
                trim+'.xlsx', header=True, index=False, encoding='utf-8-sig')

def coord(direc, city):
    '''
    Función que recibe el municipio y las direcciones y devuelve las
    coordenadas utilizando "geopy.geocoders.Nominatim"
    '''

    coordenadas = []
    geolocator = Nominatim(user_agent="josemanuel@gmail.com")
    geocode = lambda query: geolocator.geocode(city % query)

    #for i in range(5):
    for i in range(len(direc)):
        print(i)
        try:
            location = geocode(direc[i])
            coordenadas.append((location.latitude, location.longitude))
            print(location.address)
        except:
            coordenadas.append((0.000, 0.000))
            print('Dirección no encontrada')

    # Guardamos el archivo depurado
    Dir = pd.DataFrame(coordenadas)
    Dir.columns = ['y', 'x']
    Dir = Dir[['x', 'y']]
    Dir['x'] = Dir['x'].replace(".", ",")
    Dir['id'] = range(1, len(Dir)+1)
    #Dir['id'] = range(len(Dir))
    Dir = Dir.iloc[:, [2, 0, 1]]

    return Dir

def direc_google(direc, ciudad, provincia, pais, anyo, trim):
    '''

```

```

Función que crea el archivo los parametros para crear el archivo para
geolocalizar utilizando "google earth"
'''

Direc_google_earth = pd.DataFrame(direc)
Direc_google_earth['id'] = range(1,len(Direc_google_earth)+1)
Direc_google_earth['Ciudad'] = ciudad
Direc_google_earth['Provincia'] = provincia
Direc_google_earth['Pais'] = pais
Direc_google_earth.rename(columns={0:'Calle'}, inplace=True)
Direc_google_earth = Direc_google_earth.loc[:,['id','Calle','Ciudad',
                                                'Provincia','Pais']]

Direc_google_earth.to_excel("./Geolocalizacion/Direcciones "+
                             anyo+trim+".xlsx", header=True, index=False,
                             encoding='utf-8-sig')

```

```

def coord_google(D_g, anyo, trim):

    '''
    Función que recibe las direcciones y crea el archivo xlsx para asignar las zonas
    '''

    Id = []
    x = []
    y = []

    coordenadas = pd.DataFrame()

    D = D_g[0]

    l = len(D['features'])

    for i in range(l):

        Id.append(int(D['features'][i]['properties']['id_2']))
        x.append(D['features'][i]['geometry']['coordinates'][0])
        y.append(D['features'][i]['geometry']['coordinates'][1])

    coordenadas['id'] = Id
    coordenadas['x'] = x
    coordenadas['y'] = y

    coordenadas.to_excel('./Geolocalizacion/coordenadas '+anyo+trim+'.xlsx',

```

```
header=True, index=False, encoding='utf-8-sig')
```

```
def zona(df, Direcciones_Zona):

    '''
    Función que añade las zonas al dataframe
    '''

    df['Id_Zona'] = Direcciones_Zona['Zona']
    df['Fecha anuncio'] = pd.to_datetime(df['Fecha anuncio'], format="%Y%m%d")

    # Guardamos los que no hemos localizado la ubicación
    # NOTA: Aquí cabría investigar por si rescatamos alguna dirección
    df_N = df[df['Id_Zona'] == 0]

    # Se seleccionan las filas con zona asignada
    df = df[df['Id_Zona'] != 0]
    df = df.drop(['Calle'], axis=1)

    # Renumeramos el índice
    df.index = range(0, len(df))

    # Añadimos las zonas
    df_Zonas = pd.read_excel('./Geolocalizacion/Zonas.xlsx')
    zona = []
    for i in range(len(df)):
        num = int(df.loc[i, ['Id_Zona']])
        zona.append(df_Zonas['Zonas'][num-1])

    df['Zona']=zona
    df = df.iloc[:, [0,1,8,3,2,4,5,6,7]]

    return df, df_N
```

```
def elimina_dup(df):

    '''
    Función que elimina los duplicados existentes en las webs
    secundarias en base a criterios más laxos y mantiene los
    registros de la web principal.
    '''

    df_fotocasa_Y = df[df['Fuente'].str.contains('Fotocasa')]
```

```
df_fotocasa_N = df[df['Fuente'].str.contains('Fotocasa')==False]
df = df.drop_duplicates(df.columns[df.columns.isin(
    ['Precio', 'Id_Zona', 'm2_U', 'Dormitorios', 'Baños'])))
df_fotocasa_N = df[df['Fuente'].str.contains('Fotocasa')==False]
df = pd.concat([df_fotocasa_Y, df_fotocasa_N])

return df
```

```
def medias(df):

    '''
    Función que sustituye los valores faltantes de las columnas "m2_U",
    "Dormitorios" y "Baños" por sus medias
    '''

    df['m2_U'] = pd.to_numeric(df['m2_U'])
    df['m2_U'] = df['m2_U'].fillna(df['m2_U'].mean())
    df['Dormitorios'] = pd.to_numeric(df['Dormitorios'])
    df['Dormitorios'] = df['Dormitorios'].fillna(df['Dormitorios'].mean())
    df['Baños'] = pd.to_numeric(df['Baños'])
    df['Baños'] = df['Baños'].fillna(df['Baños'].mean())

    df = df.drop(['m2_C'], axis=1)

    return df
```

```
def trimestre(df):

    '''
    Construcción de la columna trimestre y ordenación del dataset
    '''

    df['year'] = df['Fecha anuncio'].dt.year.astype(str)
    df['quarter'] = df['Fecha anuncio'].dt.quarter.astype(str)
    df['Trimestre'] = df['year'] + 'T' + df['quarter']

    df = df.iloc[:, [1, 10, 0, 2, 3, 4, 5, 6, 7]]

    return df
```

```
def val_extremos(df):
    '''
    Función que depura los valores extremos del dataset
    '''

    df = df[df['m2_U'] <= m2_sup] #m2_sup = 200
    df = df[df['m2_U'] >= m2_inf] #m2_inf = 30
    df = df[df['Dormitorios'] <= Dorm] # Dorm = 6
    df = df[df['Baños'] <= banyos] # banyos = 4

    # Detección y eliminación de outliers
    X = pd.DataFrame(data={'Precio':df['Precio']})
    #clf = KNN(contamination=0.18)
    clf = KNN()
    clf.fit(X)
    y_pred = clf.predict(X)
    outliers = list(X[y_pred == 1].index)
    df = df.drop(outliers, axis=0)

    # Se extraen las zonas con 5 o más observaciones
    zona = df.groupby(['Id_Zona'])['Zona'].count()
    zona = zona[zona > 4]
    num_zona = zona.index
    df = df[df.Id_Zona.isin(num_zona)]

    # Depuramos outlier dentro de cada zona
    df['Precio'] = df['Precio'].astype(str).astype(int)
    media = df.groupby(['Id_Zona'])['Precio'].mean()
    desv = df.groupby(['Id_Zona'])['Precio'].std()
    lim_inf = media - 1.96 * desv
    lim_sup = media + 1.96 * desv

    # Se eliminan los valores que se alejan de la media por exceso o
    # defecto 1,96 desviaciones típicas
    for i in num_zona:
        a = int(lim_inf[lim_inf.index == i])
        b = int(lim_sup[lim_inf.index == i])
        df_aux = df[df['Id_Zona'] == i]
        outliers = df_aux[(df_aux['Precio'] < a) | (df_aux['Precio'] > b)].index
        df = df.drop(outliers, axis=0)

    return df
```

```
def estadisticos(df):
    '''
    Función agrupa el dataframe con los estadísticos de interés
    '''
    #df = df_informe
    Id_Zona = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].count().index)

    Zona = pd.DataFrame(df.groupby(['Zona'])['Precio'].count().index)

    n = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].count())
    n.index = range(n.shape[0])

    precio = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].mean())
    precio.index = range(n.shape[0])

    desv = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].std())
    desv.index = range(n.shape[0])

    minimo = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].min())
    minimo.index = range(n.shape[0])

    maximo = pd.DataFrame(df.groupby(['Id_Zona'])['Precio'].max())
    maximo.index = range(n.shape[0])

    dormitorios = pd.DataFrame(df.groupby(['Id_Zona'])['Dormitorios'].mean())
    dormitorios.index = range(n.shape[0])

    baños = pd.DataFrame(df.groupby(['Id_Zona'])['Baños'].mean())
    baños.index = range(n.shape[0])

    m2_U = pd.DataFrame(df.groupby(['Id_Zona'])['m2_U'].mean())
    m2_U.index = range(n.shape[0])

    # Creamos la fecha tipo para el trimestre.
    if trim == 1:
        mes = '03'
    elif trim == 2:
        mes = '06'
    elif trim == 3:
        mes = '09'
    else:
        mes = '12'

    fecha_dt = '01'+'/'+'mes+'/'+'anyo
    #fecha_dt = datetime.strptime(fecha_dt, '%d/%m/%Y')
```

```

Fecha_anuncio = pd.DataFrame()
Fecha_anuncio.index = range(n.shape[0])
Fecha_anuncio['Fecha anuncio'] = fecha_dt

# Unimos el dataframe con los datos agrupados
df1 = pd.concat([Fecha_anuncio, Id_Zona, Zona, n, precio, m2_U,
                 dormitorios, baños, minimo, maximo, desv], axis=1,)
df1.columns = ['Fecha anuncio', 'Id_Zona', 'Zona', 'n', 'Precio',
               'm2_U', 'Dormitorios', 'Baños', 'P_min', 'P_max', 'P_desv']

# Añadir el valores para zona 0
n = df['Precio'].count()
precio = df['Precio'].mean()
m2_U = df['m2_U'].mean()
dormitorios = df['Dormitorios'].mean()
baños = df['Baños'].mean()
minimo = df['Precio'].min()
maximo = df['Precio'].max()
desv = df['Precio'].std()

total = {'Fecha anuncio': fecha_dt, 'Id_Zona': 0, 'Zona': ciudad_google_earth,
         'n': n, 'Precio': precio, 'm2_U': m2_U, 'Dormitorios': dormitorios,
         'Baños': baños, 'P_min': minimo, 'P_max': maximo, 'P_desv': desv}

total = pd.DataFrame(total, index=[0])

df = pd.concat([total, df1])
df.index = range(df.shape[0])

return df

```

```

def intervalos(df):
    '''
    Se crea la columna intervalo renta y se agrupa por zona e intervalo
    '''

    df['intervalo'] = False
    df['intervalo'] = df['Precio'].apply(
        lambda x: 'Menor o igual 400 €' if x<=400
        else ('401 € - 500 €' if (x>400) & (x<=500)
        else ('501 € - 600 €' if (x>500) & (x<=600)
        else ('601 € - 700 €' if (x>600) & (x<=700)

```



```

        else ('701 € - 800 €' if (x>700) & (x<=800)
              else ('801 € - 900 €' if (x>800) & (x<=900)
                    else ('901 € - 1000 €' if (x>900) & (x<=1000)
                          else ('1001 € - 1100 €' if (x>1000) & (x<=1100)
                                else ('1101 € - 1200 €' if (x>1100) & (x<=1200)
                                      else ('1201 € - 1300 €' if (x>1200) & (x<=1300)
                                            else ('Más de 1300 €'))))))))

n = pd.DataFrame(df.groupby(['Id_Zona', 'intervalo'])['Precio'].count())
n = n.reset_index()

# Creamos la fecha tipo para el trimestre.
if trim == 1:
    mes = '03'
elif trim == 2:
    mes = '06'
elif trim == 3:
    mes = '09'
else:
    mes = '12'

fecha_dt = '01'+'/'+mes+'/'+anyo
Fecha_anuncio = pd.DataFrame()
Fecha_anuncio.index = range(n.shape[0])
Fecha_anuncio['Fecha anuncio'] = fecha_dt

# Unimos el dataframe con los datos agrupados
df1 = pd.concat([Fecha_anuncio, n], axis=1)
df1.columns = ['Fecha anuncio', 'Id_Zona', 'intervalo', 'n']

# Añadir el valores para zona 0
n = pd.DataFrame(df.groupby(['intervalo'])['Precio'].count())
n = n.reset_index()
Fecha_anuncio = pd.DataFrame()
Fecha_anuncio.index = range(n.shape[0])
Fecha_anuncio['Fecha anuncio'] = fecha_dt
Id_Zona = pd.DataFrame()
Id_Zona.index = range(n.shape[0])
Id_Zona['Id_Zona'] = 0

df = pd.concat([Fecha_anuncio, Id_Zona, n], axis=1)
df.columns = ['Fecha anuncio', 'Id_Zona', 'intervalo', 'n']
df = pd.concat([df, df1], axis=0)
df.index = range(df.shape[0])

```

```
return df
```

```
def dormitorios(df):
```

```
'''
```

```
Se crea la columna dormitorios y se agrupa por zona e intervalo
```

```
'''
```

```
#df = df_informe
```

```
df['Interv_dorm'] = False
```

```
df['Interv_dorm'] = df['Dormitorios'].apply(
```

```
    lambda x: 'Estudio o 1 Dorm.' if x<2
```

```
    else ('2 Dorm.' if (x>=2) & (x<3)
```

```
    else ('3 Dorm.' if (x>=3) & (x<4)
```

```
    else ('4 o más Dorm.') ))
```

```
n = pd.DataFrame(df.groupby(
```

```
    ['Id_Zona', 'Interv_dorm'])['Precio'].count())
```

```
Precio = pd.DataFrame(
```

```
    df.groupby(['Id_Zona', 'Interv_dorm'])['Precio'].mean())
```

```
df1 = pd.concat(
```

```
    [n, Precio], axis=1)
```

```
df1 = df1.reset_index()
```

```
# Creamos la fecha tipo para el trimestre.
```

```
if trim == 1:
```

```
    mes = '03'
```

```
elif trim == 2:
```

```
    mes = '06'
```

```
elif trim == 3:
```

```
    mes = '09'
```

```
else:
```

```
    mes = '12'
```

```
fecha_dt = '01'+'/' +mes+'/' +anyo
```

```
Fecha_anuncio = pd.DataFrame()
```

```
Fecha_anuncio.index = range(n.shape[0])
```

```
Fecha_anuncio['Fecha anuncio'] = fecha_dt
```

```
# Unimos el dataframe con los datos agrupados
```

```
df1 = pd.concat([Fecha_anuncio, df1], axis=1)
```

```
df1.columns = ['Fecha anuncio', 'Id_Zona', 'Dormitorios', 'n', 'Precio']
```

```
# Añadir el valores para zona 0
n = pd.DataFrame(df.groupby(['Interv_dorm'])['Precio'].count())
Precio = pd.DataFrame(df.groupby(['Interv_dorm'])['Precio'].mean())
df = pd.concat([n, Precio], axis=1)
df = df.reset_index()

Fecha_anuncio = pd.DataFrame()
Fecha_anuncio.index = range(n.shape[0])
Fecha_anuncio['Fecha anuncio'] = fecha_dt
Id_Zona = pd.DataFrame()
Id_Zona.index = range(n.shape[0])
Id_Zona['Id_Zona'] = 0

df = pd.concat([Fecha_anuncio, Id_Zona, df], axis=1)
df.columns = ['Fecha anuncio', 'Id_Zona', 'Dormitorios', 'n', 'Precio']
df = pd.concat([df, df1], axis=0)
df.index = range(df.shape[0])

return df
```

```
##### CUERPO PRINCIPAL #####

# Leemos y unimos los dataset diarios de cada portal
df_fotocasa = unir_excel(anyo, trim, portal='fotocasa', drop=[7])
df_pisoscom = unir_excel(anyo, trim, portal='pisoscom', drop=[6])

# Guardamos los archivos por si queremos extraer otras variables posteriormente
store_data(df_fotocasa, anyo, trim, portal='fotocasa')
store_data(df_pisoscom, anyo, trim, portal='pisoscom')

# Depuramos y unimos los dataset
df_fotocasa_dep = fotocasa_dep.debug_dataset(df_fotocasa)
df_pisoscom_dep = pisoscom_dep.debug_dataset(df_pisoscom)
df_informe = pd.concat([df_fotocasa_dep, df_pisoscom_dep])
df_informe.columns = ['Fecha anuncio', 'Fuente', 'Precio', 'Calle', 'Zona',
                      'm2_C', 'm2_U', 'Dormitorios', 'Baños', 'Planta', 'Tipo']

# Quito los duplicados entre webs
df_informe = df_informe.drop_duplicates(
    df_informe.columns[df_informe.columns.isin(
        ['Calle', 'Zona', 'm2_U', 'Dormitorios', 'Baños'])])

df_informe.index = range(0, len(df_informe))
```

```
df_informe['Id Zona'] = False

df_informe = df_informe.drop(['Planta'], axis=1)
df_informe = df_informe.drop(['Tipo'], axis=1)

# obtenemos las coordenadas de las direcciones
direcciones = list(df_informe['Calle'])

##### NOMINATIM #####
# Se utiliza "geopy.geocoders.Nominatim" para obtener las coordenadas
Direcciones_coord = coord(direcciones, ciudad_Nominatim)
#####

# Se asignan las zonas a las coordenadas de las direcciones
Direcciones_Zona = geo.geolocalizacion(Direcciones_coord)

# Se añade la columna zona al dataset
df_informe, df_informe_N = zona(df_informe, Direcciones_Zona)

# Se eliminan las viviendas posibles duplicados de las sucesivas webs
# y se añaden a la principal
df_informe = elimina_dup(df_informe)

# Sustitución de los na por las medias
df_informe = medias(df_informe)

# Depuración del dataframe
# En este punto se depura el dataframe en función de los criterios establecidos
df_informe = val_extremos(df_informe)

# Agrupación del dataframe para obtener principales estadísticos
df_informe_agrup = estadisticos(df_informe)

# Agrupación del dataframe por zona e intervalo de renta para obtener
# el total de anuncios en cada categoría
df_intervalo_agrup = intervalos(df_informe)

# Agrupación del dataframe por zona y dormitorios para obtener el total
# de anuncios en cada categoría
df_dormitorio_agrup = dormitorios(df_informe)

# Se une con el dataset del trimestre anterior y se guarda en el imput
# del cuadro de mando
df_data = pd.read_excel(
    "../Resultados/CuadroMando/data.xlsx", sheet_name='data')
```

```
df_dormitorios = pd.read_excel(
    "../Resultados/CuadroMando/data.xlsx", sheet_name='dormitorios')
df_intervalos = pd.read_excel(
    "../Resultados/CuadroMando/data.xlsx", sheet_name='Intervalos renta')

df_data = pd.concat([df_data, df_informe_agrup])
df_dormitorios = pd.concat([df_dormitorios, df_dormitorio_agrup])
df_intervalos = pd.concat([df_intervalos, df_intervalo_agrup])

# Guarda datos los datos actualizados en un archivo xlsx en la carpeta
# imput del cuadro de mando
writer = pd.ExcelWriter("../Resultados/CuadroMando/data.xlsx")
df_data.to_excel(writer, sheet_name="data",
                  header=True, index=False, encoding='utf-8-sig')
df_dormitorios.to_excel(writer, sheet_name="dormitorios",
                        header=True, index=False, encoding='utf-8-sig')
df_intervalos.to_excel(writer, sheet_name="Intervalos renta",
                       header=True, index=False, encoding='utf-8-sig')

writer.save()
writer.close()

#####
```

debug_fotocasa.py

```
##### debug_fotocasa.py #####

def debug_dataset(df_fotocasa):

    '''
    Función que recibe como entrada el dataset en bruto y
    lo devuelve depurado
    '''

    # Se quitan los duplicados
    df_fotocasa = df_fotocasa.drop_duplicates(
        df_fotocasa.columns[df_fotocasa.columns.isin([7])])
    df_fotocasa = df_fotocasa.drop_duplicates(
        df_fotocasa.columns[df_fotocasa.columns.isin([8,9,11,13])])

    # Selección registros para el informe
    df_fotocasa = registros_informe(df_fotocasa)

    columns_names = df_fotocasa.columns.values
    columns_names = list(columns_names)
```

```
# Se insertan celdas en WWW
df_fotocasa = insert_www(df_fotocasa, columns_names)

# Se corrigen los NA
df_fotocasa = df_fotocasa.replace({None: "BLANCO33"})

# Se ordenan las columnas
df_fotocasa = orden_col(df_fotocasa, columns_names)

# Se elimina el caracter "." y "€"
df_fotocasa[2]= df_fotocasa[2].str.replace('.', '', regex=True)
df_fotocasa = extrae_valor(
    df_fotocasa, sep = ' €', col = 2, drop = 1, n_col = [2])

# Se seleccionan las variables que se necesitan para el informe
df_fotocasa = df_fotocasa.iloc[:, [0,1,16,7,12,8,10,14]]

# Se separa la columna dirección
df_fotocasa = extrae_direccion(df_fotocasa, columns_names)
columns_names = list(range(0,df_fotocasa.shape[1]))
df_fotocasa.columns = columns_names

# Corregimos los NA
df_fotocasa = correc_na(df_fotocasa, columns_names)

return df_fotocasa

def registros_informe(df):

    '''
    Selección registros para el informe
    '''

    df = df.drop(['Unnamed: 0'], axis=1)

    # Se quitan los anuncios sin renta
    df =df[df[2] != 'A consultar']
    df =df[df[8].str.contains('Casa')==False]

    # Se queda con las variables necesarias para el informe
    df = df.iloc[:, [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]]

    return df
```

```
def insert_www(df, columns_names):

    '''
    Inserción celdas en columna WWW
    '''

    df_Y = df[df[7].str.contains('https')]
    df_N = df[df[7].str.contains('https')==False]

    try:
        df_N.insert(6, 6, '', True)
        df_N = df_N.drop(16, axis=1)
        df_N.columns = columns_names

        df = pd.concat([df_Y, df_N])
    except:
        print("No hay ningún registro sin https")

    return df

def orden_col(df, columns_names):

    '''
    Ordenación de las columnas
    '''

    var = ['hab', 'baño', 'm²', 'Planta']
    n=10
    i=10
    for i in range(10,17,2):
        df_Y = df[df[i].str.contains(var[i-n])]
        df_N = df[df[i].str.contains(var[i-n])==False]
        df_N.insert(i-1, 'c', '', True)
        df_N.insert(i-1, 'c', '', True)
        df_N = df_N.drop([15,16], axis=1)
        df_N.columns = columns_names
        df = pd.concat([df_Y, df_N])
        n+=1

    return df

def extrae_valor(df, sep, col, drop, n_col):

    '''
```



```

    Extrae el valor deseado de la columna separando por caracter
    '''

    df_aux= df[col].str.split(sep, expand=True)
    df_aux= df_aux.drop([drop], axis=1)
    df = df.drop([col], axis=1)
    df_aux.columns = np.array(n_col)
    df = pd.concat([df, df_aux], axis=1)

    return df

def extrae_direccion(df, columns_names):

    '''
    Extrae la dirección de la columna
    '''

    # Se separa la columna dirección
    df_8= df[8].str.split(' en ', expand=True)
    df_9 = df_8.iloc[:, [0]]
    df_9= df_9[0].str.split(' de ', expand=True)
    df_9 = df_9.iloc[:, [0]]
    df_9.columns = [10]
    df_8= df_8[1].str.split(', ', expand=True)

    # Direcciones con número de portal
    df_8[2] = df_8[2].replace({None: "BLANCO33"})
    df_Y = df_8[df_8[2].str.contains("BLANCO33")]
    df_N = df_8[df_8[2].str.contains("BLANCO33")==False]
    df_N.iloc[:,0] = df_N.iloc[:,0] + ', ' + df_N.iloc[:,1]
    df_N = df_N.iloc[:, [0,2]]
    df_N.columns = [0,1]
    df_Y = df_Y.iloc[:, [0,1]]

    df_8 = pd.concat([df_Y, df_N])
    df_8.columns = [4,5]

    df = df.drop([8], axis=1)
    df = pd.concat([df, df_8, df_9], axis=1)
    df = df[[0,1,2,4,5,13,9,11,15,10]]
    df.insert(5, 5, '', True)

    return df

def correc_na(df, columns_names):

```

```
'''
Se renombran todos los valores vacios
'''

df[4] = df[4].replace({None: "BLANCO33"})
df_Y = df[df[4]=="BLANCO33"]
df_N = df[df[4]!="BLANCO33"]
df_Y = df_Y[[0,1,2,4,3,5,6,7,8,9,10]]

columns_names = list(range(0,df.shape[1]))
df_Y.columns = columns_names
df_Y[3]=None

df = pd.concat([df_Y, df_N])

df[4] = df[4].replace({None: "BLANCO33"})
df = df[df[4]!='BLANCO33']
df = df.sort_index()

return df

#####
```

Geolocalizacion.py

```
##### Geolocalizacion.py #####

import pandas as pd
import numpy as np

# GEOLOCALIZACION DIRECCIONES

def geolocalizacion(Direcciones dataframe):

    '''
    Función que asigna un número de zona a cada coordenada según al polígono
    que pertenece
    '''

    # Se lee el archivo con las coordenadas y el número de las áreas de cada zona
    Barrios_dataframe = pd.read_excel("./Geolocalizacion/Barrios.xlsx")
    pd.options.display.max_rows = 10
```

```
Direcciones_dataframe['Zona'] = 0

D = Barrios_dataframe.iat[(len(Barrios_dataframe)-1), 2]

result = np.zeros((len(Direcciones_dataframe), D)) # matriz de ceros

for s in range(len(Direcciones_dataframe)): # desde 0 hasta 540
    print('Dirección =', s+1)
    x = Direcciones_dataframe[Direcciones_dataframe['id'] == s+1]
    for t in range(D): # desde 0 hasta n° de zonas
        p = Barrios_dataframe[Barrios_dataframe['Zona'] == t+1]
        if len(p) > 0:
            i = 0
            j = len(p)-1
            dentro = 0
            for i in range(len(p)): # desde 0 hasta el tamaño de la Zona i

                if ((p.iat[i,4] < x.iat[0,2]) and
                    (p.iat[j,4] >= x.iat[0,2] )) or
                    ((p.iat[j,4] < x.iat[0,2]) and
                     (p.iat[i,4] >= x.iat[0,2]))):
                    A1 = float(p.iat[i,3])
                    A2 = float(x.iat[0,2] - p.iat[i,4])
                    A3 = float(p.iat[j,3] - p.iat[i,3])
                    A4 = float(p.iat[j,4] - p.iat[i,4])
                    AA = A1 + (A2 * A3) / A4

                    if (AA < x.iat[0,1]):
                        dentro = 1 - dentro

            j = i

        result[s,t] = dentro
        if result[s,t]==1:
            Direcciones_dataframe.iat[s,3] = t+1

    return Direcciones_dataframe

#####
```

7.3. Anexo III. Código de inserción del cuadro de mando en página web

Embed Code:

```
<div class='tableauPlaceholder' id='viz1672398963454' style='position: relative'><noscript><a
href='#'><img alt='Renta de alquiler en áreas municipales '
src='https://public.tableau.com/static/images/Mo/Monitorizacinrent
adealquiler/Alicante/1_rss.png' style='border: none' /></a></noscript><object
class='tableauViz' style='display:none;'><param name='host_url'
value='https%3A%2F%2Fpublic.tableau.com%2F' /> <param name='embed_code_version' value='3'
/> <param name='site_root' value='' /><param name='name'
value='Monitorizacinrentadealquiler/Alicante' /><param name='tabs' value='no' /><param
name='toolbar' value='yes' /><param name='static_image'
value='https://public.tableau.com/static/images/Mo/Monitorizacinre
ntadealquiler/Alicante/1.png' /> <param name='animate_transition' value='yes' /><param
name='display_static_image' value='yes' /><param name='display_spinner' value='yes' /><param
name='display_overlay' value='yes' /><param name='display_count' value='yes' /><param
name='language' value='en-US' /><param name='filter' value='publish=yes' /></object></div>
<script type='text/javascript'>
    var divElement =
document.getElementById('viz1672398963454');
    var vizElement =
divElement.getElementsByTagName('object')[0];
    if ( divElement.offsetWidth > 800 ) {
vizElement.style.width='900px';vizElement.style.height='1227px';} else if ( divElement.offsetWidth >
500 ) { vizElement.style.width='900px';vizElement.style.height='1227px';} else {
vizElement.style.width='100%';vizElement.style.height='3577px';}
    var scriptElement =
document.createElement('script');
    scriptElement.src =
'https://public.tableau.com/javascripts/api/viz_v1.js';
vizElement.parentNode.insertBefore(scriptElement, vizElement);
</script>
```