Búsqueda de inmuebles

Recuperación de Información y Recomendaciones en la Web

Grupo 17

Docente
Libertad Tansini

Integrantes
Federico Grunwald
Marcos Toscano
Soledad Rivas

Tabla de contenido

1. Introducción

El siguiente informe presenta una solución al problema cotidiano de búsqueda de inmuebles.
El objetivo es brindar una forma sencilla y eficaz para acceder a la mayor cantidad de inmuebles posibles y desplegar la información de los mismos de manera clara y completa.

2. Problema

En la actualidad, el conseguir un inmueble (ya sea para compra, alquiler u alquiler temporario) es un trabajo muy tedioso.

Esto se debe a que existen muchas fuentes posibles de donde extraer la información de los mismos, donde cada una de ellas presenta una gran cantidad de inmuebles, y a que cada sitio presenta los datos que considera más relevantes para los usuarios, mostrando los mismos en formatos diferentes.

Hoy en día, si un usuario desea buscar un inmueble, debe navegar por cada una de las páginas en la que se detalla la información del mismo, lo que a veces termina por frustrar al usuario en una búsqueda que debería poder realizarse de forma amena.

3. Motivación

La **motivación** para llevar a cabo este proyecto fue la observación de la **gran dificultad** que se plantea al momento de realizar la **búsqueda de inmuebles**.

La gran cantidad de fuentes y de información que se puede obtener es de tal magnitud que muchas veces el usuario termina agobiando por la misma y no existe un sitio que agrupe toda esa información y que la presente de forma normalizada.

4. Enfoque de solución

Para uniformizar el modo en que la información es presentada al usuario, construiremos una aplicación web la cual extrae la información de varias fuentes y luego las muestra en un mapa donde se utilizan filtros y se muestran los detalles del inmueble seleccionado.

La solución consta de dos grandes etapas:

Primera etapa: Extracción de la información

El primer paso del proyecto fue el obtener la información desde la web. Para esto se utilizó la herramienta **Scrapy**, que permite la construcción de “arañas” que van “trepando” la red y obteniendo información al mismo tiempo.

Debido a la gran cantidad de información disponible, se optó por extraer información solamente de **dos fuentes**.

De la **información** disponible que presentaban los inmuebles, se extrajo solamente aquella que era **más relevante para el usuario**, siendo estos los siguientes datos:

- Tipo de moneda del precio
- Precio
- Identificador de la propiedad (en el sitio)
- URL de la publicación
- Estado de la propiedad (Alquiler, Alquiler temporario o Venta)
- Latitud y longitud
- Origen del inmueble (MercadoLibre o Gallito)
- Dirección
- Barrio
- Descripción
- Cantidad de baños
- Cantidad de Cuartos
- Metros cuadrados construidos y totales
- Teléfono de contacto
- URLs de las imágenes de los inmuebles publicados

Segunda etapa: Presentación de la información

El segundo paso consistió en tomar la **información** obtenida y presentarla al usuario de forma **normalizada** y **completa**.

Se buscó como meta principal que la **interfaz** sea **intuitiva** y **sencilla**, por lo que se decidió incluir un mapa en el cuál se pudiera ir **filtrando** los inmuebles a medida que el usuario se desplaza por el mismo.

5. Herramientas y tecnologías utilizadas

A nivel de proyecto

Docker[11] es un proyecto de código abierto que **automatiza el despliegue de aplicaciones** dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de Virtualización a nivel de sistema operativo en Linux

Se utilizaron archivos de configuración que automatizan el **despliegue del proyecto en ambiente local**.

Backend

Python fue el lenguaje de programación utilizado para el desarrollo en el backend.

Django[5] es un framework (conjunto de componentes que ayudan a desarrollar sitios web más fácil y rápidamente) para aplicaciones web gratuito y open source, escrito en Python.

Django-rest-framework**[**6**]** es una herramienta para construir aplicaciones web que permite exponer, en una **API REST,** la estructura y los datos de un proyecto de Django.

Scrapy[4] es una librería open source de Python que sirve para extraer información de la web de forma rápida y fácil. Se utilizó en la creación de dos arañas cuyo resultado terminó en la extracción de 24000 inmuebles de Mercadolibre y 8000 en el gallito aproximadamente.

El sitio de **MercadoLibre[**2**]** cuenta con una **API**[1] que permitió extraer la información de forma sencilla, mientras tanto, en el sitio del El Gallito se tuvo que implementar la extracción de datos de forma manual, utilizando Xpath y expresiones regulares.

PostgreSQL[7] es un Sistema de gestión de bases de datos relacional orientado a objetos y libre, se utilizó como motor para la Base de Datos donde se guardó la información de los inmuebles.

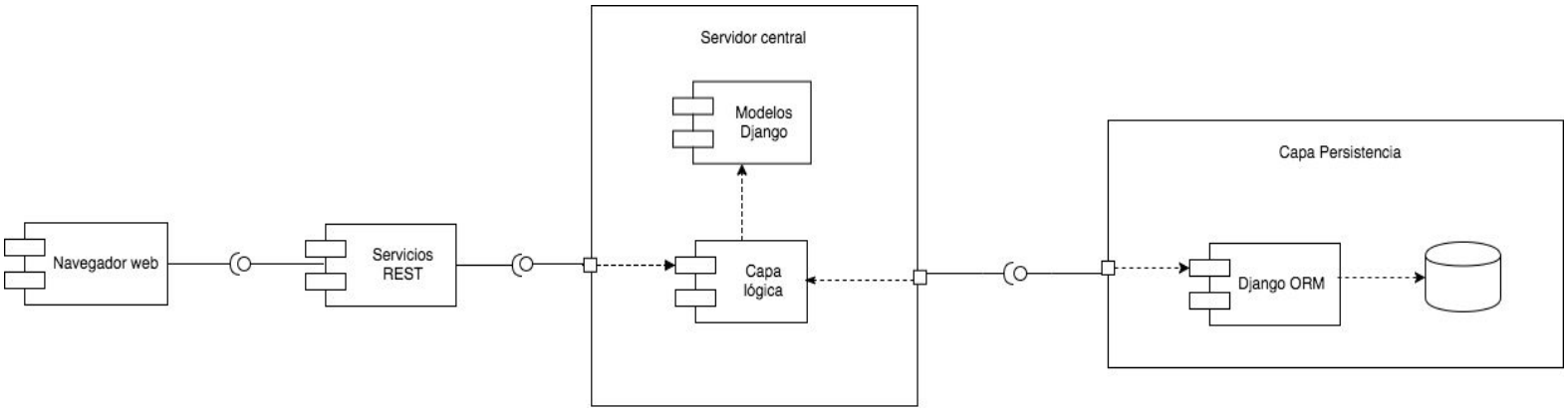
Frontend

React[8] es una biblioteca de JavaScript para crear interfaces de usuario. Lo mantienen **Facebook**, **Instagram** y una comunidad de desarrolladores y corporaciones individuales.

Google Maps[9] es un servidor de aplicaciones de mapas en la web, se utilizó para exponer los datos de los inmuebles al usuario. Más específicamente se utilizó un **componente de REACT[10]** que facilita el manejo de los pines en el mapa

6. Componentes del sistema

Para la **comunicación** de los diferentes componentes del sistema de optó por utilizar **servicios REST**.

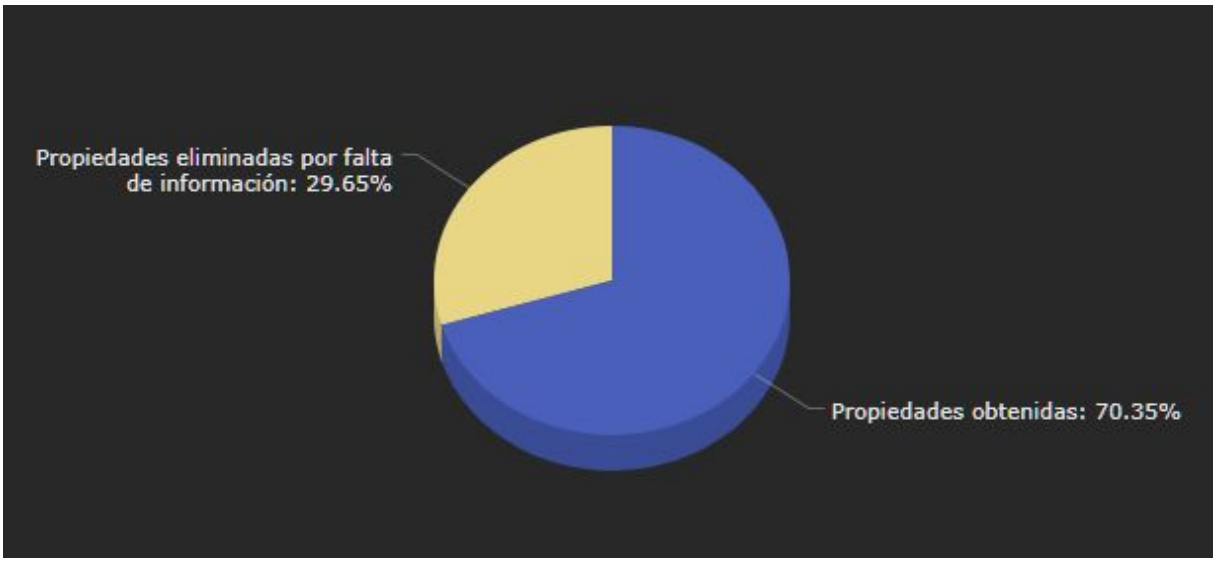


7. Evaluación y resultados

A modo de **comparación**, se analizaron primero por separado los datos obtenidos por ambas **fuentes de información**.

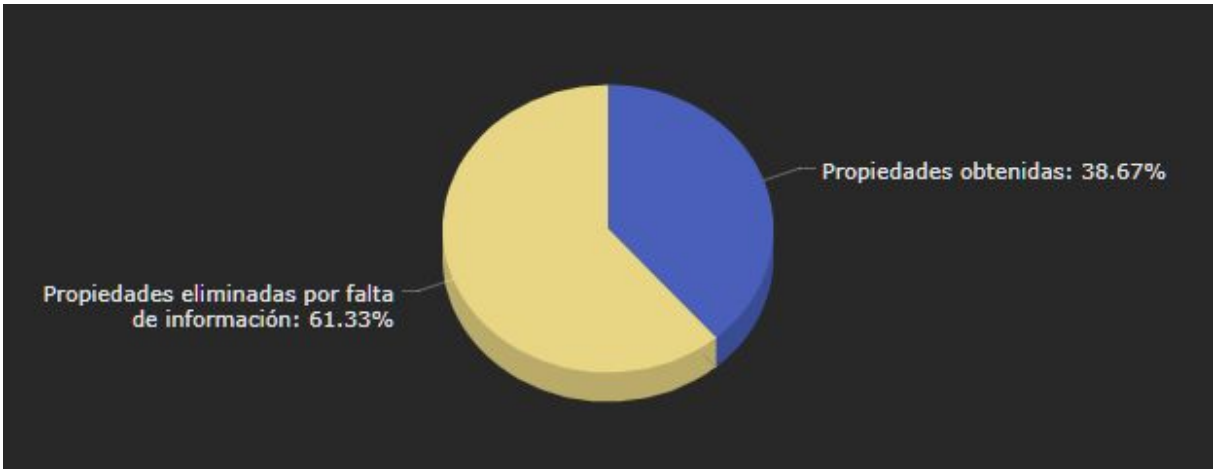
Para el caso de **MercadoLibre**, se obtuvo la siguiente proporción en cuanto a la calidad de los datos.

- Cantidad de propiedades **obtenidas**: 23.795
- Cantidad de propiedades **eliminadas**: 10.028



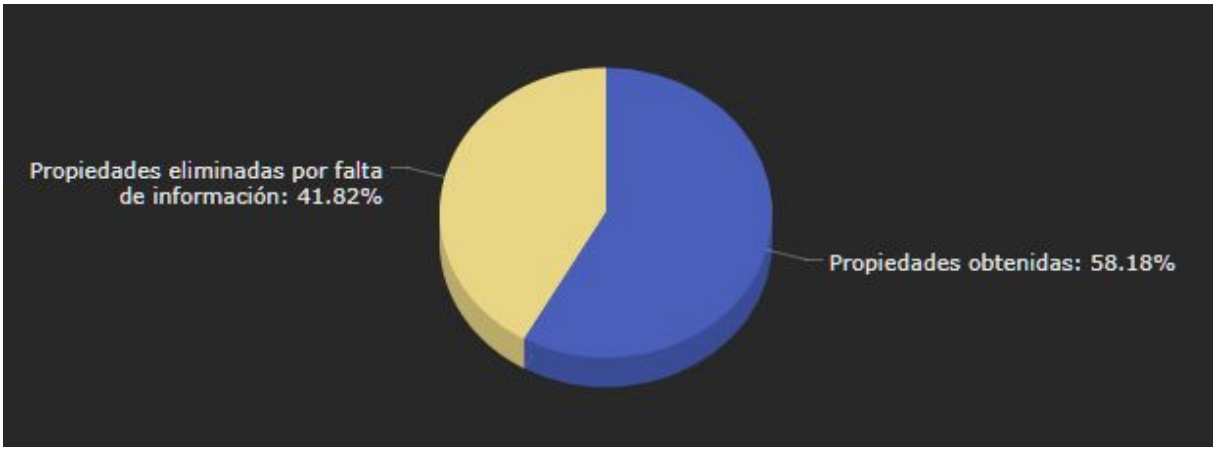
Para el caso de **El Gallito**, los resultados fueron los siguientes:

- Cantidad de propiedades **obtenidas**: 8.155
- Cantidad de propiedades **eliminadas**: 12.936



Se puede apreciar en los resultados presentados que la **calidad** de los datos presentados en **MercadoLibre supera** ampliamente los presentados por **El Gallito**.
Una de las posibles explicaciones de este fenómeno, puede ser que MercadoLibre cuenta con una API (lo que tiende a normalizar la información dentro del sitio y a presentar publicaciones más completas y homogéneas), mientras que para el Gallito se tuvo que utilizar otro método de obtención de la información (uso de XPath y expresiones regulares), lo que genera una solución menos robusta y con mayor probabilidad de fallas.

Si tomamos en cuenta los resultados anteriores, obtenemos un total de **31.950** propiedades **obtenidas** y **22.964** propiedades **eliminadas** por falta de información. Lo que genera la siguiente proporción:



Utilizando la definición de Precisión vista en clase:

$$P = \frac{Documentos\ relevantes\ recuperados}{Documentos\ recuperados}$$

Podemos calcular la Precisión de los resultados obtenidos mediante la operación

$$P = \frac{31950}{54914} = 0,5818$$

Al momento de calcular el Recall, definido como

$$R = \frac{Documentos\ relevantes\ recuperados}{Documentos\ relevantes}$$

Notamos que la cantidad total de documentos relevantes, dada la forma en la cual se realizó la recuperación de datos, es la misma que la cantidad de documentos relevantes recuperados, ya que se obtuvieron todos los datos, luego se filtraron los relevantes y los otros fueron eliminados (por ejemplo por falta de información en la publicación como las coordenadas geográficas).

8. Problemas encontrados

Problemas con los datos

- Surgieron varios problemas al **unificar** datos de distintas fuentes debido a que la calidad de los datos contenidos no es la mejor y el formato no se mantiene en todos los casos.
- Se encontraron **diferencias semánticas** en las fuentes elegidas, por ejemplo en el caso de MercadoLibre se maneja el concepto de ambientes mientras que en el Gallito el de dormitorios.
- Se encontraron muchas **publicaciones duplicadas** tanto en MercadoLibre como en Gallito.
- **Direcciones mal ingresadas** o abreviaciones de calles incorrectas
- **Precios irracionales**, como casas en venta a USD 1**.**

Problemas para extraer los datos

- La api de **Mercadolibre** utilizada tiene **documentación muy escasa** sobre sus métodos y cómo utilizar los filtros de búsqueda requeridos.
- **Gallito no tiene API**, por lo que se tuvo que realizar scraping utilizando xpath, provocando que su **velocidad** de scrapeo fuera mucho **más lenta** comparado con la de una API.

9. Caso de uso de la web

<u>El funcionamiento de la web es el siguiente:</u>

1. El **usuario selecciona** una serie de **filtros** como ser cantidad de ambientes, cantidad de baños, precio de los anteriores etc.
2. Se realiza la **consulta** a la base de datos y se **despliega** cada **inmueble** obtenido en el **mapa** según su longitud y latitud utilizando **Google Maps**.
3. Si se oprime el botón '**Más información**', debajo del mapa se muestran más datos como ser la dirección, fotos, link hacia la publicación original, etc.

10. Conclusiones

Luego de finalizar el proyecto concluimos que cumplimos con los objetivos propuestos, destacándose entre ellos el aprendizaje de técnicas de **Web Scraping** y **georeferenciación** que permitieron lograr un prototipo de sistema en el tiempo dado.

Este prototipo **permite al usuario** tener un sitio con **información unificada** de diferentes fuentes, ubicadas en el mapa. El mismo puede ser de gran utilidad si se avanza en el trabajo a futuro mencionado en el punto anterior.

Sobre recolección de datos en la web, se puede concluir que la **calidad difiere** mucho en cada publicación. Esto se debe a que ambas fuentes de datos elegidas, tienen como fuente de información de las publicaciones a usuarios.

11. Trabajo futuro

Gracias a los resultados obtenidos, se concluyó que sería posible construir una aplicación de **mayor escala** (o seguir desarrollando la presentada en el proyecto) que contenga **más fuentes de información**.

Se podría **incluir** también un **manejo de usuarios**, donde los mismos puedan agendar visitas a los inmuebles, guardar sus favoritas, tener una funcionalidad de comparación entre inmuebles que permita calcular un índice dependiendo de las características de los mismos y ver cuál es mejor.

Asimismo, dada la cantidad de datos manejados por la aplicación, se podrían implementar algoritmos de **aprendizaje automático** para los siguientes propósitos:

- La **predicción de precio**, por ejemplo, para saber si un inmueble está sobrevaluado o no dependiendo de sus características (barrio, cantidad de metros construidos, cantidad de cuartos y baños, etc.).

- **Recomendación** basada en experiencia de otros usuarios.

- Dados un conjunto de filtros, generar un **ranking de resultados** donde se muestran los mismos ponderados por algún índice calculado basado en las características del inmueble.

Como etapa final del proyecto se podría implementar la **reserva del inmueble** vía la aplicación, utilizando componentes del **gobierno electrónico** que se espera tener en un futuro, se podrían gestionar todos los documentos relacionados a la venta y alquiler de inmuebles, **brindando una mayor comodidad al usuario.**

12. Referencias

[1] - http://developers.mercadolibre.com/
[2] - https://inmuebles.mercadolibre.com.uy/casas/
[3] - http://www.gallito.com.uy/inmuebles/casas
[4] - https://scrapy.org/
[5] - https://www.djangoproject.com/
[6] - http://www.django-rest-framework.org/
[7] - https://www.postgresql.org/
[8] - https://reactjs.org/
[9] - https://www.google.com/maps
[10] - https://github.com/istarkov/google-map-react
[11] - https://www.docker.com/