Paper Title*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address

***Abstract*—Last thing that will be written. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Mauris cursus mattis molestie a iaculis at erat pellentesque adipiscing. Vitae auctor eu augue ut lectus arcu bibendum. Dui id ornare arcu odio ut sem. Tellus at urna condimentum mattis. Sed euismod nisi porta lorem mollis aliquam. Orci eu lobortis elementum nibh tellus molestie. Posuere ac ut consequat semper viverra nam libero.**

***Index Terms*—swiss real estate, webscraping, user interface, transparency, economic research**

I. Introduction

Transparency in the real estate market has long been associated with favorable business environments. While Switzerland is ranked 11th in the JLL "Global Real Estate Transparency Index", there are only a few platforms that may allow end-market investors to compare prices and assess the worth of a property. The Swiss market is dominated by private valuers and brokers ("Agents"), who refrain from citing exact addresses and informing buyers the right price of properties on the market. Hence, leading to vast information asymmetry between Agents and potential buyers. However, the rise of web platforms is slowly erasing information asymmetry, where potential buyers may compare different properties without the use of Agents. Nevertheless, even with the use of these revolutionary real estate web platforms, it is still difficult to assess for a potential buyer what drives the price of their target property. Is it undervalued? Overvalued? At the right price? Therefore, despite advancements in enhancing transparency of the housing market, there are still improvements to be made to minimize information asymmetry, so that potential buyers may make rational and informed decisions before investing in real property.

II. Research Question

A. Problem

Real Estate has long been a very opaque asset class, with subjective pricing methods; as opposed to asset classes that trade in the open market (stocks, bonds, derivatives, etc.). Historical transaction data is difficult to obtain, and the fact that each property is radically different in its characteristics makes it difficult for inexperienced buyers to assess property value.

Therefore, buyers must rely on Agents, who have better knowledge of the housing market. Rutherford et al. (2005) show that Agents sell their own houses for a premium of approximately 4.5%, whereas Levitt and Syverson (2008) find that houses owned by Agents sell for approximately 3.7% more than other houses. The evidence suggests the presence of Agents' information advantage in the housing market. Why do Agents pay lower prices when buying their own houses? One explanation is related to information asymmetries in the housing market. Real estate agents have information advantages over less informed "nonagent" buyers. These studies also show that Agents will use this information to their own personal advantage, which may enter in conflict with the interest of their clients.

With the apparition of e-commerce real estate platforms, potential buyers have access to current listings with prices, characteristics, and pictures. This may help in reducing information asymmetry to a certain extent. However, these listing are actual listings, where historical listings are not available. Moreover, the data is not formatted and structured into databases to make informed comparisons. Finally, the price that is reflected in listings may not be the final prices, as there is generally a negotiation process involved before the transaction closes.

Therefore, we find the lack of a real open-source platform, showing past listings in a structured fashion to be necessary in reducing information asymmetry and enhancing transparency of the housing market in Switzerland.

B. Objective

Our goal is to provide Swiss potential home buyers with an open-source platform, that lists all previous past listings, which are ordered by NPA (Swiss postal codes), price, square meters and number of rooms. This platform would pull data automatically in certain periodic intervals, which would provide with a wide range of historical data, useful to understand trends. This platform would give an edge both to professional Agents and potential buyers, effectively reducing information asymmetry between the two parties.

Such platforms would also be relevant for statistics by authorities, insurances, mortgage lenders, as well as advertisers.

C. Scope

Considering the wide array of properties in Switzerland, we decided to focus our attention to Lausanne and its suburbs. This will permit us to test out the first version of our platform, with limited property data and historical data. However, the platform is scalable to become a nationwide platform.

III. Methodology

A big difference between real estate and other publicly traded assets is that there is no centralized platform where offers can be tracked. In the case of asset classes such as bonds and equity this service is provided by multiple sources such as Bloomberg, Refinitiv and Morningstar.

Something similar is missing for the Swiss real estate market. Usually real estate on the various websites is presented as a list of single "tiles", along with a brief overview of the key data of the corresponding property (for reference see figure 1). Normally there is no possibility to see the average cost per square meter in the ZIP code and no real way to compare similar estates.

Given that the data is neither available via an API nor through an already established database, the decision was made to write an algorithm that scrapes the required data from a website.

A. Choice of Website

Whilst there exist numerous websites that act as middlemen between buyer and seller such as `www.immoscout24.ch` and `www.homegate.ch`, all of them suffer from the same underlying problem mentioned in the introduction. Selecting and pricing real estate by clicking through listing after listing is cumbersome and inefficient. Furthermore, with the data being split up over many websites, a buyer might not find his ideal new property, simply by looking on the wrong platform.

An attempt at creating a more transparent and efficient real estate market was made by Comparis. On their website, listings from many sources are aggregated and displayed according to their key characteristics. However, also `www.comparis.ch` does not offer tools for further analysis and does not readily display historical data, except for increases or decreases in the price in percentage.

For the purpose of creating a database which provides a user with information on as many properties in a given area as possible, `www.comparis.ch` is the ideal target for scraping data.

B. Web Scraping

When creating a web scraping application in Python, there are a number of packages and tools to choose from. These modules include, but are not limited to `requests`, `BeautifulSoup`, `Scrapy` and last but not least, `Selenium`. Each come with their advantages and disadvantages. The former two modules, `requests` and `BeautifulSoup` both come with the bonus of ease of use. The latter two modules offer more functionality, but have a steeper learning curve.

After several attempts, the choice boiled down to a combination of `Selenium` and `Scrapy`. This decision was based on the need to not only collect data on the "top level" of the listing (see 1), but also the information on the detail page of every single URL.

Fig. 1. Typical Listing on Comparis

The need for more data was not the only factor that led to this choice. Today's websites are populated with JavaScript's that procedurally generate the content the user can see. Generating further content may be triggered by simple actions such as scrolling on a page, clicking a button, or by more complex actions such as hovering over a certain area or logging in somewhere. For a web scraping service that relies on parsing HTML (such as `requests` and `BeautifulSoup`), this poses a major issue. This is mainly because they cannot trigger a JavaScript by themselves and therefore cannot generate the source code that they could extract and parse.

C. Scrapy

Of the largest web scrapers available for python, `Scrapy` offers many options and usually serves the needs of larger scale web scraping projects. The main feature that lead to the use of `Scrapy` for this project was the possibility of using a so-called spider. The spider is set up by using the `scrapy genspider Comparis comparis.ch` command, along with several other files that allow granular customization of the spider. In using a spider, a web scraper has to define a class (based on the class `scrapy.Spider`, so it inherits all its attributes), which tells `Scrapy` how and in which sequence to scrape which parts of a given website.

Information on websites is not readily available for scraping per se. This entails that the output of the scraping process can be very disorganized and full of unwanted HTML tags. Here `Scrapy` offers a very attractive solution, namely the `ItemLoader`, which is defined as a class in the `items.py` file. Whilst `ItemLoader` certainly adds to the steepness of the learning curve, it is a worthwhile endeavor, since the scraped data will come out clean already. `ItemLoader` "cleans" the data when it comes in, stores it while the spider is running, and generates an output based on the user's preference.

Furthermore, `Scrapy` can send many requests to a website at once, thus speeding up the data gathering process tremendously. However, even when using all the request headers that a normal browser window sends to a website, the `Scrapy` spider's were constantly IP-blocked.

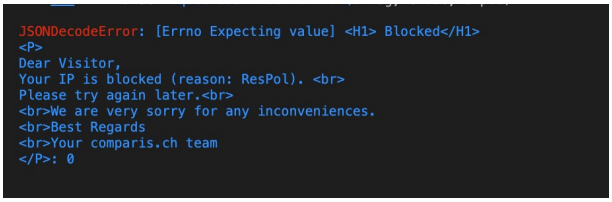

Fig. 2. IP block when running the spiders

D. Selenium

Whilst `Scrapy` works well with extracting data from the website's source code (expecially using XPaths), it is not possible to navigate JavaScript heavy websites. Furthermore, even with the implementation of rotating proxies, the IP-block could not be circumvented. This is where `Selenium` comes in: `Selenium` allows a user to control any browser environment, provided the corresponding driver has been installed. Controlling a browser offers several advantages, the primary advantage being that the request comes from an actual browser, which means it can hardly get blocked. The downside is that by sending the requests through a browser, the amount of requests per time is practically reduced to one. This increased the runtime of the program drastically, however it ensured that the data collection was successful.

With some additional effort, combining `Selenium` with `Scrapy` can offer the best two worlds: reliable data gathering on the one hand, and clean data extraction on the other hand.

E. User platform

To make the dataset scrapped from immoscout more user friendly we decided to build a graphical user interface(GUI). This way the user will be able to search for a property within a certain price range, zip code or with specific features, instead or having to go through the full scrapped dataset. We used the open source library tkinter to model the GUI. We chose this library as it is pre installed in python. Moreover it is stable and flexible and provides simple syntax. Tkinter is the perfect tool for this first prototype. However, if the project grows tkinter will not be powerful enough, therefore we recommend switching to PyQT another UI or even REACT for a more professional look.

F. Structure

IV. Implementation

- Scrapy - version 2.6.1
- Selenium - version 4.1.5
- Webdriver_manager - version 3.5.4
- Numpy - version 1.22.3
- Pandas - version 1.4.2
- Time
- Datetime

A. Subsection

V. Conclusion

A. Subsection

References