Racism in the housing market

Technical Writeup

Michael Brauweiler Vadim Goryainov

Technical University of Munich
March 15, 2018

The goal of our project is to research the latent racism a person might encounter while searching for accommodation. We wanted to analyse whether persons with a foreign name are less likely to get a positive response to an application for accommodation than the ones with a German name and whether this response rate is correlated with the local support of populist parties. To acquire the data, we wrote a tool that sent automated requests to hundreds of offers on the website `www.wg-gesucht.de` in seven German cities. Thus we could show, that there is indeed a correlation between your name and your success in finding accommodation. Also, a correlation with local support for the political party *AfD* is shown.

1 The Automated Tool

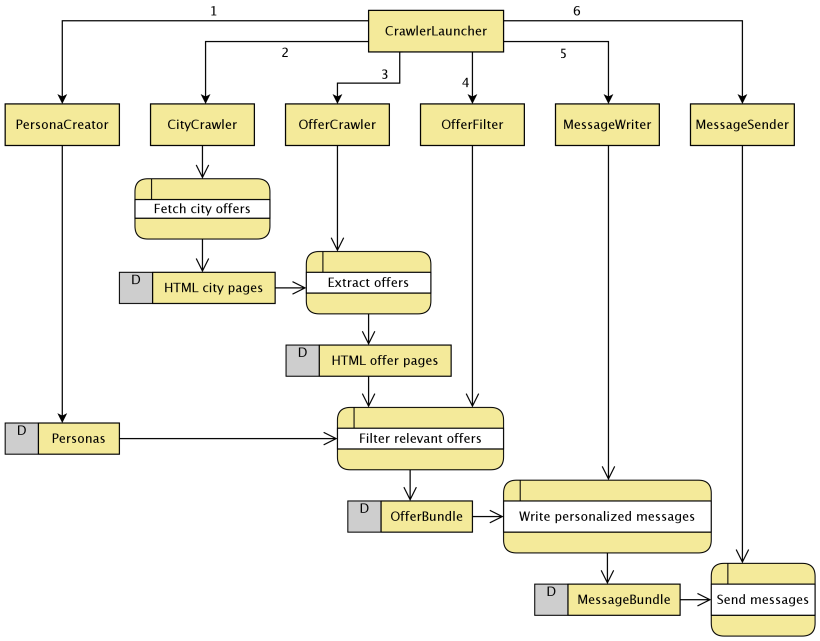


Figure 1: UML Data Flow Diagram

1.1 Setup and Workflow

Our tool consists of several components that are executed in sequential order (see *Figure* 1). The results of each component are saved on the hard drive and not only on the main memory. This allowed us to shutdown the program after running several components, check the interim results and resume the procedure. It also made sure that the results were not lost in case of an unexpected crash of the program - this was especially important as the tool runs for several hours per city.
One big issue while crawling `www.wg-gesucht.de` and sending automated requests to the offers found was to avoid detection as a web bot by the website. For that reason we made a component StealthManager that handles the connection process to the website and is called by every other component. The only library we used apart from the standard JavaSE-1.8 library was the open source library Jsoup[1]. We used latter as a convenient way to examine the components in the HTML files.

1.2 Classes

1.2.1 PersonaCreator

This simple class was used to create and store the information for each of the two personas a Java *.properties* file. This included:

- the given name
- the surname
- the age
- the gender
- the text which is sent in formal applications
- the text which is sent in informal applications
- the email provider (`gmx.de`[2] in our case)

[1] `www.jsoup.org`
[2] *GMX Mail* is a free e-mail service which supports an easy sign-up process without the need to submit a phone number.

1.2.2 StealthManager

StealthManager
- cookiesDatabase : Map<String, Map<String, String>
+ hide(Connection connection) - getRandomUserAgent() - getUserCookies(String userAgent)

Our StealthManager class is a layer between every connecting class and `wg-gesucht.de` (see *Figure* 2). It anonymizes every connection by randomly picking an entry from a list with 50 common user agents (*getRandomUserAgent()* method). An user agent string is used by web browsers to provide information about the connecting device. This way, websites can provide special representations for e.g. smartphones but it enables also to track suspicious users.
The StealthManager class ensures consistency by providing an own cookie database for every user agent, simulating distinct users. Also, it internally adds certain cookies to the databases used by the website to mark users which already passed the security warning and captcha test. Other classes connect to the website through the *hide(Connection connection)* method, which delays incoming queries for a random amount of seconds. A *Connection* is an object type used by the *JSoup* package consisting an URL.

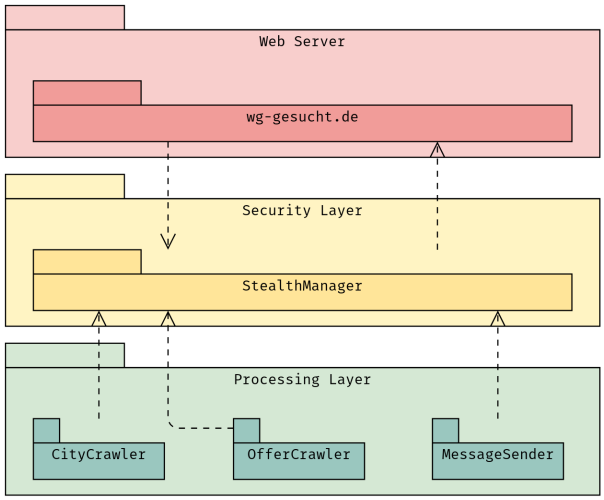


Figure 2: UML Package Diagram showing the security layering

1.2.3 CityCrawler

Our class CityCrawler is used to fetch the HTML pages with offer listings from the website `wg-gesucht.de`. It is callable either through the *updateAll()* method to fetch and update all known cities (and thus *maxPages* pages in every city) or through the *updateCityList()* method, which takes an ArrayList with cityIDs as an argument.
Every city page has an unique ID (e.g. Munich has the ID 90), which can be extracted by URL manipulation. The *maxCities* integer provides the maximal cityID (used by *updateAll()* method) and the *cities* HashMap provides a mapping of known cityID - city name pairs. This is needed to avoid repeatedly using a wrong URL when connecting to city pages and thus increasing the hazard of being detected as a bot. Connections to the website are processed by the StealthManager class for the same reason.

CityCrawler
- cities : Map<Integer, String> - maxCities : int - maxPages: int
+ getCityList() + updateAll() + updateCityList(List<Integer> list)

OfferCrawler
- filePathCities : String - filePathContacts : String - baseContactFilepath : String - lastContactFilepath : String - lastContact : int - contactCounter: int
+ readURLsFromFileSystem() + searchOffersInFile(Document doc, String city, int cityId) + saveContactInFile(String url, String contactname)

1.2.4 OfferCrawler

Our class OfferCrawler is used to extract all the offers from the HTML pages that the CityCrawler has downloaded previously. For that it simply iterates through all the saved HTML files in the directory specified by *filePathCities* and searches for a specific headline that every offer in `www.wg-gesucht.de` has. However, the headline for advertisements from partner websites differs from mentioned headline, so we did not have to take care of filtering those. The OfferCrawler connects with the URL in every offer and downloads the page to the folder specified in *filePathContacts*. An example for this procedure is given in *Figure* 4. To cover and resume the offer extraction after an unexpected crash of the program, the OfferCrawler saves the count of the last offer found in a file on the hard drive.

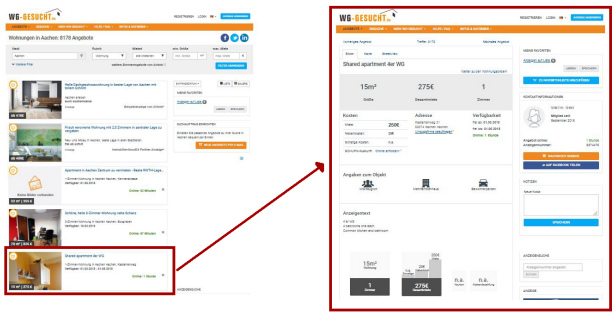


Figure 4: Example of a website downloaded by the CityCrawler and an offer extracted by the OfferCrawler.

1.2.5 OfferFilter

OfferFilter
- filteredDocs : HTMLDocument[] - group1 : Document[] - group2 : Document[]
- checkDocAndAdd(Document doc, int cityId) - randomSplitHalf()

We couldn't use every offer that the OfferCrawler extracted, as some of the offers preemptively excluded persons of certain genders and age groups. Our two personas (as defined in the *.properties* file generated by PersonaCreator) are male and 22 years old, so we had to filter any offer that wouldn't accept them. The OfferFilter takes the output of the OfferCrawler and searches the offer description for the genders

and age boundaries the lessors are willing to accept. This information is always found under specific headlines in the offers of `www.wg-gesucht.de`, so we hadn't to search the whole text for mentions of age and gender. Also, the OfferFilter discards offers if the contact form is inaccessible. After filtering the offers, the OfferFilter splits those randomly into two preferably equally sized groups.

1.2.6 MessageWriter

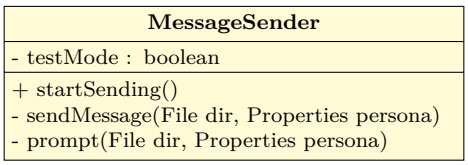
MessageWriter
- persona : Properties
+ writeMsgs(Document[] group1, Document[] group2) - writeMsg(Document offerDoc, Document contactForm, int groupNr, int cityId)

The MessageWriter takes the two sets (realised as an enumeration class) that the OfferFilter has created and writes the respective applications. This is done by using the application text (see 1.1) which previously has been saved on the hard drive by the PersonaCreator. The name of the lessor can be extracted from the contact form on the offer page. The resulting text is saved on the hard drive and could be checked manually before executing the MessageSender.
We decided to send two different kinds of messages - formal ones and and informal ones - as we assumed that different persons would expect different salutations. Our idea was to decide what kind of message to write based on whether they gave their last name or not. Persons that only gave their first name received an informal message. In most cases, the gender of the person can be found out by examining the label of the profile picture of the lessor (there is either a male or female profile picture). In all other cases we chose a gender neutral salutation. This resulted in the following possible salutations:

- Hallo ___, (gender neutral)
- Sehr geehrte Frau ___, (female, formal)
- Sehr geehrter Herr ___, (male, formal)
- Liebe ___, (female, informal)
- Lieber ___, (male, informal)

Unfortunately, the division into formal and informal message groups did not seem to be very successful, as some of the lessors reacted with confusion to the formal salutations.

1.2.7 MessageSender



The MessageSender is responsible for the final step of the process: it accesses the contact form by searching for certain tags in every offer selected by the OfferFilter and fills the 'send a message'-form with data generated by the PersonaCreator and MessageWriter. Also it makes sure to request a copy of the application via e-mail, so that we could double-check that the right amount of applications was sent.
In addition, it consists of a special mode for testing purposes and a prompt option, which asks to try again in case of a failure (e.g. broken internet connection).

2 The Experiment

2.1 Setup

Our initial hypothesis stated that there is a positive correlation between the amount of right wing party voters in a certain city and it's *racism score*, calculated by comparing the success of the two almost identical profiles on `www.wg-gesucht.de`. Our goal was to show that a foreign name makes you less successful in finding accommodation. To exclude any other possible reason for rejection, we decided to vary only the foreign tone of the name in both personas. Finally, we came up with *Sebastian Winkler* and *Achmet Bukhari* and the following flat share offer request:

> Hallo ___,
> mein Name ist ___ und ich studiere im 3. Semester Maschinenbau. Zurzeit bin ich 22 Jahre alt (werde aber bald 23) und suche im Moment nach einer netten Wohngemeinschaft. Ich bin Nichtraucher und habe keinerlei (mir bekannten) Allergien. In meiner Freizeit mache ich gerne Sport (im Winter Skilaufen, sonst Volleyball, 'Fitnessstudio') und spiele Gitarre (keine Sorge, nicht daheim, nur mit meiner Band im Bandproberaum). Außerdem koche ich ab und zu gerne (meine Freunde empfehlen mich weiter (:). Ich würde mir gerne die Wohnung mal anschauen, aber ich kann dir auch erst mehr Informationen zu mir senden wenn gewollt. (Facebook, Telefonnummer).
> Ich würde mich freuen von dir zur hören!
> Beste Grüße, ___.

The text differs depending on whether the message was classified as formal or informal (see 1.2.6).

Furthermore, we chose seven German cities to conduct our experiment by searching the results of the last Bundestag elections [1] for cities meeting two criteria:

1. Relatively extreme percentage (low or high) of *AfD*[3] voters
2. High usage of `www.wg-gesucht.de` for housing market purposes (at least 40 flat share offers within a week)

We selected the following cities:

Name	AfD percentage
Münster	4.9%
Osnabrück	7.3%
München	8.4%
Erfurt	18.2%
Leipzig	18.3%
Dresden	23.2%
Chemnitz	24.3%

Therefore, we had two independent variables, namely, whether we used the foreign or the German profile name, and the percentage of *AfD* voters. The dependent variable was the degree of positive responses we got. We used an own email address at *gmx.de* for every city/persona combination in accordance to the following pattern: *forename.surname.cityID@gmx.de*

[3] *Alternative für Deutschland* is a populist party, which achieved 12,6% of all votes at the Bundestag election in 2017

To increase reliability, we classified the emails into distinct folders with the following codebook:

Positive answer	Negative answer	Personal check	Excluded answer
Mentioning time and place of house viewing	Apologising for already giving the room away	Asking for a link to Facebook profile	Checking correct understanding of offer
Asking for a time frame for the house viewing	No answer at all	Asking for current employment	Asking for subletting secretly
Providing or asking for a phone number for meeting purposes		Asking for a residence permit / photos / certificate of bond	

Our automatic tool crawls recent offers from a chosen city and randomly splits them into two groups. Thus, every offer has the same chance to be selected into either group. This two offer bundles get transformed to message bundles, which then are sent using the form on each offer site.
Exactly one week after the process of sending the messages, we logged into the related mailboxes and classified every email using our codebook. E-mails arriving after classification got discarded and did not influence our statistics.
Until 8th of January 2018 we made preparations and wrote the tool. Within a time frame of 5 days we started sending the messages. On 20th January we classified the last email and started analysing the results.

2.2 Results

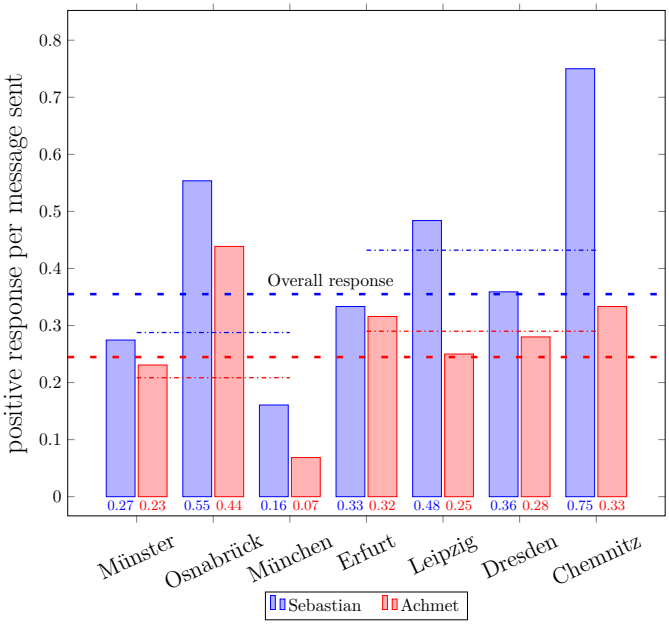


Figure 6: Results in all cities in ascending *AfD* percentage order.

By looking at the responses we could see that our persona with the foreign name (*Achmet Bukhari*, 24.5% positive responses) received 10.5% less positive answers than the one with the German name (*Sebastian Winkler*, 35.0% overall positive responses). Furthermore it is evident, that it becomes less likely to receive an answer (be it positive or negative) to a request with an increasing number of inhabitants in the city where the request is sent. The gap between positive answers for *Sebastian Winkler* is higher in cities with a high percentage of *AfD*-supporters ($> 18.5\%$) than in cities with a low percentage ($< 8.5\%$).

2.3 Problems

As already mentioned in section 1.1, one big obstacle we had to overcome was the website's bot checker. Our StealthManager component made sure that the program paused for a random amount of time – between 8 and 18 seconds – before connecting to the website and randomly chose different user agents. Still the bot checker blocked us after about one hundred connections to the website which required us to manually change the VPN server which we used for connecting.
The long waiting time between every connection raised another issue: our program had to run for several hours to send a few hundred applications. To avoid losing all progress in case of a crash of the program we decided to save the results of every component on the hard drive. This also allowed us to interrupt the program and to check the results manually before proceeding.

2.4 Ethical Considerations

Reflecting the answers we got, we filled seats on house viewing events which could have been given to real people. Also, we got answers from stakeholders, who invited us exclusively on a specific date. We could have saved them a lot of time, if we had arranged an automatic reply on all mail boxes beforehand, informing the stakeholders about the study or politely declining the invitation.
In addition, it should be noted, that our experiment may have violated the terms and conditions of `wg-gesucht.de`. Before and during the experiment, we took actions to conceal our identity, but in our estimation, our program barely had an effect on the performance of the website or it's user experience. This aspect should be considered in greater detail when executing bigger studies.

References

[1] Der Bundeswahlleiter: Statistisches Bundesamt, *Zweitstimmenanteile in den Wahlkreisen nach Parteien*. `https://www.bundeswahlleiter.de/bundestagswahlen/2017/ergebnisse/weitere-ergebnisse.html`, Wiesbaden, Deutschland, 2017.