Research and Innovation

Martí Municoy, Alba Gordó, Jan-Hendrik Niemann
and Andreas Radke

November 17, 2017

Abstract

TODO: modify the abstract s.t. it fits to the content The aim of this project is to find and visualize the events posted on Meetup.com in different cities in order to analyze the density of planned activities and to make a social study regarding typical meeting times and most usual activities and interests. One aspect may concern different meeting times in Spain and Germany or group size. We will focus our research on cultural activities like languages tandems and sports. The data shall be obtained via the Meetup-API and OpenStreetMap. The most important data source for this project is the Meetup-API where we have to request the events with different Python packages like "requests" or "meetup-api". Our second task is to visualize the locations of the events on a given city map obtained by data resources like OpenStreetMap. To plot the locations we could use "matplotlib" or similar Python plotting libraries. One further aspect could be to look for interest of twitter users to make recommends to activities nearby. The geotag of the tweets provides the central information for the search.

Contents

1 Summary and Goals

The human being is social. Every day there are millions of events. Thousands of them are posted at the webpage `Meetup.com`. It is a website where people can seek and find other people with the same interest. It helps to organize events and makes them public.

This is the reason why we chose `Meetup.com` to analyze the behavior and the preferences of cities and its inhabitants all over the world. Almost all `Meetup.com` activities contain information about their location, expressed in terms of latitude and longitude. Thus, by using the `Meetup.com` API, we can request and extract a dataset "latitude-longitude" for all the available `Meetup.com` events. Then, we can use this dataset to plot all these located activities on a `GoogleMaps.com` map. Our aim, here, is to create and visualize a density map, a so-called heatmap, where one can see all the activities for a selected city.

Furthermore, `Meetup.com` API allows the user to get activities arranged by category. We want to use this tool to consider separately different types of events like "Food & Drink" or "Sports" and study their relative predominance in different cities. By combining this information with the geographical and/or political structure of a city, one may can also identify trending districts and neighborhoods in a city. To do so we make use of GeoJSON-files which can contain the polygonal shape of a district and its name. There are several websites where one can get GeoJSON data or create his own data. By using the information from these websites, a GeoJSON dataset is created with all the coordinates of the districts for all the studied cities. This constitutes the third data source that is used in this work.

A fourth data source is `Wikipedia.org` where a web scraping script is applied on to extract some useful population data such as the population number, the population density or the land area of each district. With all this information, one can use the following factor to evaluate the social activity per district. This factor is given as

$$\text{Social activity} = \frac{\#\text{ events}}{\#\text{ person}}.$$

Once downloaded, we can access to our data off-line. This is due to the fact that we store it locally by using csv (comma-separated files).

This report is structured as follows in 6 sections. Firstly, in section 2 we discuss the data life-cycle of our data. In section 3, we present both, the third-party tools that we used and the custom methods that we developed in our own, to achieve this project. They mainly, are developed by using Python. Afterwards, we present our results in section 4 and, in section 5, we make a legal and ethical disquisition. Finally, in section 6 we discuss the limitations of our work and make proposals for future work.

2 Data Life-cycle

In the following section we have a closer look at the terms "data life-cycle" and "data life-cycle management".

"Data life-cycle" is a term which tries to describe the life of data and information. Since there is no unique definition of a data life-cycle, we define the folloing six stages: creation, storage, use, sharing, archiving and destruction [7], [9].

1. **Creation**: data is created. It can be created as a structured or unstructured set of data, can be obtained by collection, measurements, generated or gathered. We create our data by using the `Meetup.com` API. By performing a web scraping at `Wikipedia.org` we gather the number of population, the population density and the land area for different cities and their districts. In this way, we request and create the data which is going to be needed for this project.

2. **Storage**: once the data is created, it has to be stored. Depending on where we want to apply this data for, there are different and optimal kinds of storage solutions. However, this is not subject of this report. In our case, both, the data gathered from `Meetup.com` and `Wikipedia.org`, are saved as csv-files on the hard drive.

3. **Use**: data can be viewed, modified analyzed, corrected, interpreted, visualized, joined,... We use our data to visualize the social activity of cities. We join our data from different sources and visualize it on a given map which can be seen as data as well. This data visualization

and its further treatment can be used to obtain new data such as the measurement of the social activity per district introduced in section 1.

4. **Sharing**: data can be shared. There are several ways of sharing by using different file formats, applications or operating environments. In this work, we used csv-files and GeoJSON-files to share data from one method to another. For instance, the district coordinates are read from a GeoJSON-file, the Meetup events and the population data from a csv-file. Then, these are the files created by the scripts to share data along different methods.

5. **Archiving**: when the data leaves the active use, one should archive it in a suitable way. Since archiving is very similar to storaging, one can use the same methods. One difference may be that saved data needs to be compressed to save storage or protected to prevent unwanted changed. Therefore, the access is slightly more difficult than for data in active use. The file formats used in this project does not satisfy this statement. This is due to the fact that the file formats are chosen to ease the access to their data rather than to save storage space. If we want to keep this data for a long time, we should think on developing a method to compress this data.

6. **Destruction**: there is a last stage in the data life-cycle. There are several reasons why one should destroy data. On one hand, the volume of archived data increases and one needs to save storage to store new data. On the other hand, data gets old and could be not needed anymore. Or, if it is not up to date, it can be replaced by newer versions. The second case is applicable for our work. During every single run of the program new csv-files with event data and new csv-files containing populations data are created. Especially the event data underlie a rapid change.

The term "data life-cycle management" describes the process that helps to organize the flow of data throughout its life-cycle.

3 Data Tools

3.1 Python Packages

The main work for our project is done in the Python programming language in version 3.6 [8]. This programming language is used to develop different packages. Each package is in charge of managing a specific dataset and it

does a certain job. In this section, we are going to introduce all the packages that we have developed and we use to retrieve the data of this project.

3.2 Meetup Package

This package has the main purpose of retrieve the information of all the available events in a city. Its main module is called *mu_ requests.py*. This module makes use of the *Python 3.6 Requests* package to submit queries to Meetup. It is able to get information about all the available categories and events. Then, it can also parse the data coming from the Meetup Client, whose format is expressed in JSON, and it can store it in a local drive by writing a custom csv file. For instance, all the events found in a city they are written in this way *./csv/name_ of_ the_ city.csv*. This custom csv file contains all the events of a city arranged by category.

3.3 Mapping Package

This is the package responsible of plotting all the located events on a map. To do this, first, it needs to read all the events data from the previous csv-file. Then, it uses the read coordinates to display events on a Google Maps map. This functionality uses the *gmaps* library along with a *Jupyter* notebook which allow us to display data on a map.

It has some tools that allows us manage which events we want to display. For example, it allows us to plot only those events that are scheduled in a certain time range or those events which belong to a specific category. We can also control the way the script colors the event marks or their opacity.

Another interesting tool that this package offers is to plot the districts of a city. In this case, GEOJSON files with the delimiting points for each district are required. If you want to plot them according to their population, population data for each district is also required. The script read these files and parses the data and it can plot and paint the districts according to the number of events per capita that each district has.

3.4 Scraping Package

This package is involved on retrieving data from *Wikipedia*. Particularly, it searches for the pages belonging to the districts of a city to retrieve information about their population from one of their tables. This process is usually known as web scraping. To check for the appropriate tables, it uses *BeautifulSoup* Package that is an html parser.

Then, the script is designed to look for these tables in an intelligent way. So, it is able to check different Wikipedia addresses to get the right information. Firstly, it looks for the English articles but, in some cases, best articles are written in other languages rather than English. So, it is also able to check pages written in Spanish and Deutsch.

Once it gets the right population table it comes the hardest part. It needs to move around the table to look for the proper cells. The functions that are used in the parsing process are able to work with tables, words and numbers expressed in different formats. For instance, it supports tables with joined cells or which arrange their information in different ways.

3.5 Plotting Package

The last developed Package is used to plot data from all these retrieved events. It is useful to view and compare the amount of events for each category or city. This script makes use of *pygal package* to create these plots.

3.6 Data

As already noted we used the data sources that are summarized below:

- Meetup API: To get the events in the examined cities.
- Wikipedia: For obtaining data about the population density of the cities and their districts.
- Various resources for GeoJSON files: To visualize districts onto Google Maps. TODO
- Google Maps: As a general back-end for our visualization.

4 Results

To draw meaningful conclusions we created several types of visualization of the data. First and foremost we plotted the open events onto a map - either as a point map or as a occurrences distribution map for each city district.

In Figure 1 one can see an example of Barcelona with open events plotted as points. The different colors represent different meetup categories.

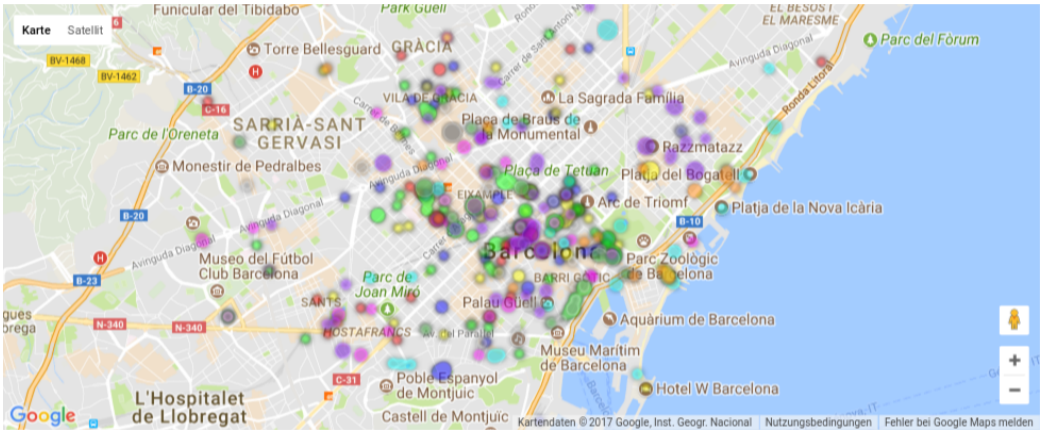
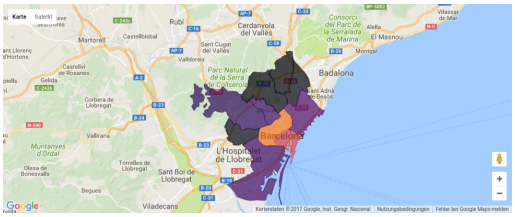
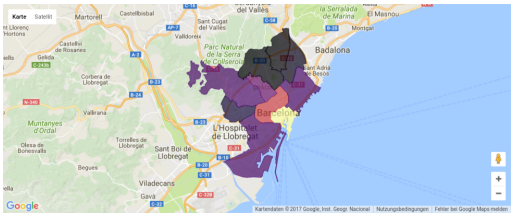


Figure 1: Snippet of Barcelona with Open Events

Figure 2: Snippet of Barcelona with Language & Ethnic Identity Events

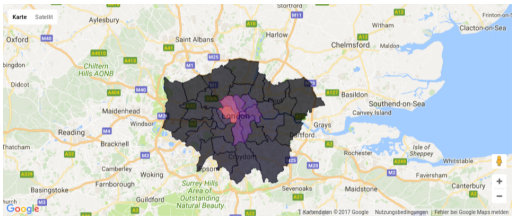


(a) Activity per District

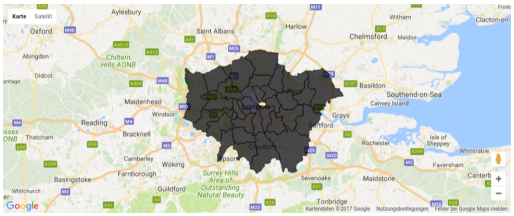


(b) Activity per Capita per District

Figure 3: Barcelona

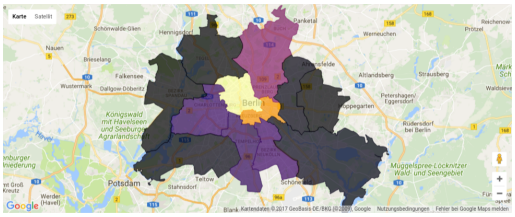


(a) Activity per District

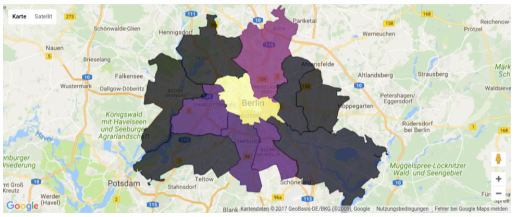


(b) Activity per Capita per District

Figure 4: London

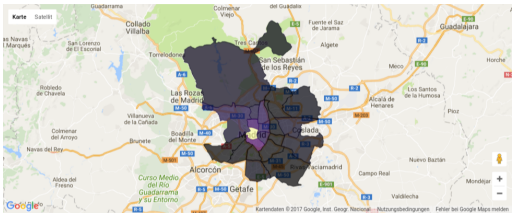


(a) Activity per District

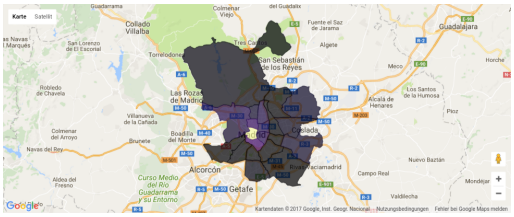


(b) Activity per Capita per District

Figure 5: Berlin

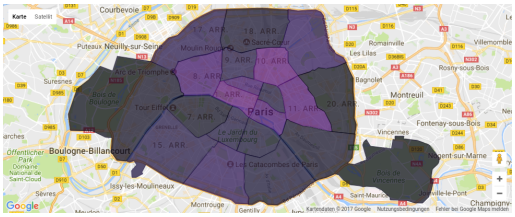


(a) Activity per District

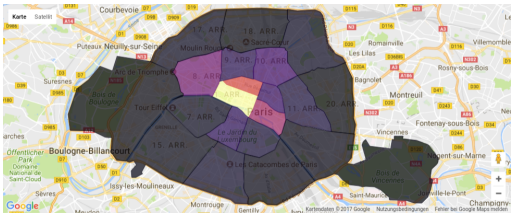


(b) Activity per Capita per District

Figure 6: Madrid

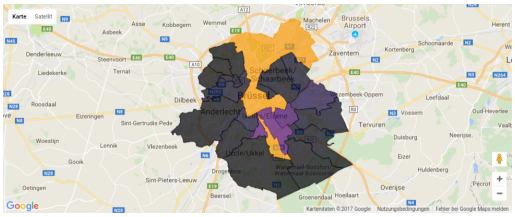


(a) Activity per District

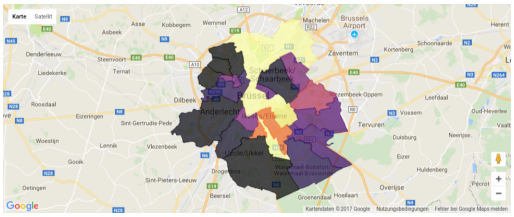


(b) Activity per Capita per District

Figure 7: Paris

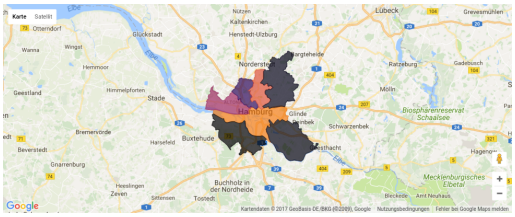


(a) Activity per District

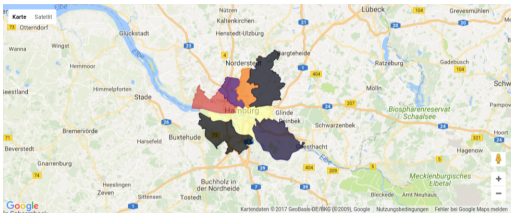


(b) Activity per Capita per District

Figure 8: Brussels

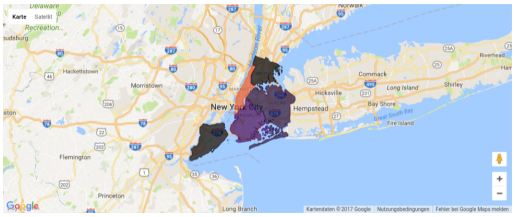


(a) Activity per District

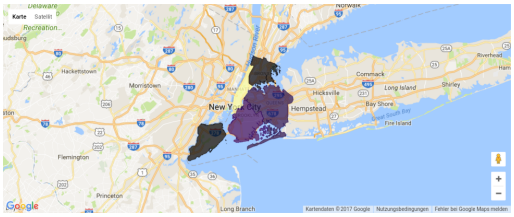


(b) Activity per Capita per District

Figure 9: Hamburg

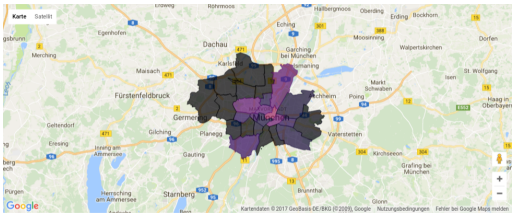


(a) Activity per District

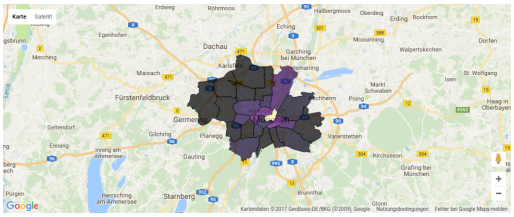


(b) Activity per Capita per District

Figure 10: New York City

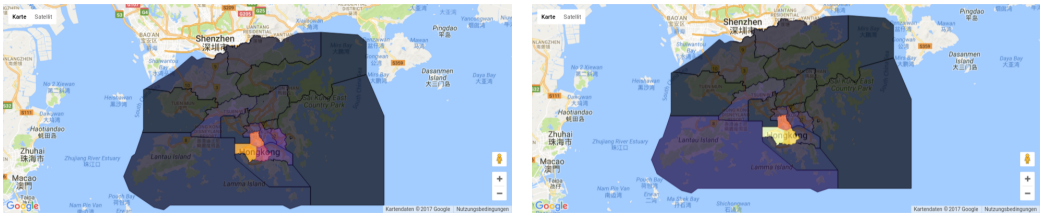


(a) Activity per District



(b) Activity per Capita per District

Figure 11: Munich



(a) Activity per District (b) Activity per Capita per District

Figure 12: Hong Kong

5 Legal and Ethical Issues

6 Limitations and Future Work

One major issue is that this project is depending strongly on the quality of the data available at `Meetup.com`. We have to rely on the users of `Meetup.com` so this issue can hardly be improved. One idea would be to integrate other social services like `Facebook.com` to extrend our data source and to get more independent. In this way one could also fix the problem that `Meetup.com` might be less known in non-English speaking countries. Having a wide base as data source one gets more independent of local and regional preferences regarding the social networks.
Another problem is that this project has the potential to be extended in the following way: So far our program is restricted to predefined cities. It is not possible to select new cities or regions. This is due to the fact that we work with GeoJSON-files which are saved locally on the hard drive. The second problem is that some Wikipedia pages for the cities are not coherent so that one needs to be careful. One improvement could be to create a user interface and the possibility to select new cities and regions. By selecting single categories of events one could get special information, e.g. where to go running. Additional, in an application, one could automatically create suitable graphics, charts and tables.
Another limitation is as well related to `Meetup.com`. Since we only have one data source for events, we only can give sufficient criterions for districts. A district having lots of events seems to be trening. However, a district having less events is not automatically less interesting and popular. It just means that its inhabitants are not using `Meetup.com`. Again, one should take more data sources into account.

Our tool could help social scientists to provide theire hypothesis with proving data. Beside literature, studies and surveys our tool seems to be a

good method for analyzing the behaviour and preferences of populations.

Appendix

Sources of GeoJSON Files

Table 1: GeoJSON Sources

City	Source
Berlin	https://github.com/m-hoerz/berlin-shapes
Barcelona	https://github.com/martgnz/bcn-geodata/find/master
Hamburg	https://matthiassuessen1975.carto.com/tables/bezirke_in_hamburg/public/map
München	https://mucx.carto.com/tables/osm_muc_bezirke/public
Paris	https://github.com/blackmad/neighborhoods/blob/master/gn-paris.geojson
New York	https://github.com/blackmad/neighborhoods/blob/master/new-york-city-boroughs.geojson
London	https://joshuaboyd1.carto.com/tables/london_boroughs_proper/public
Madrid	https://github.com/codeforamerica/click_that_hood/blob/master/public/data/madrid-districts.geojson
Sydney	http://insideairbnb.com/get-the-data.html
Brussels	http://insideairbnb.com/get-the-data.html
Hong Kong	http://insideairbnb.com/get-the-data.html
Vancouver	http://insideairbnb.com/get-the-data.html
San Francisco	http://insideairbnb.com/get-the-data.html
Palo Alto	http://insideairbnb.com/get-the-data.html
Chicago	http://insideairbnb.com/get-the-data.html
Los Angeles	http://insideairbnb.com/get-the-data.html

References

[1] Google Inc. Google Maps. `https://maps.google.com`, 2017. Interactive online Map Service.

[2] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[3] Kenneth Reitz. requests - Python HTTP for Humans, version 2.14.2. `https://pypi.python.org/pypi/requests`, 2017. Python Library for sending HTTP requests.

[4] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, Carol Willing, and Jupyter Development Team. Jupyter notebooks – a publishing format for reproducible computational workflows. pages 87 – 90, 2016.

[5] Meetup Inc. Meetup API. https://www.meetup.com/meetup_api/, 2017. [Online; accessed 11-November-2017].

[6] Pascal Buignon. gmaps, version 0.70. https://github.com/pbugnion/gmaps, 2017. Plugin for including interactive Google maps in the Jupyter Notebook.

[7] Philipps Universität Marburg. Was versteht man unter dem Data Life Cycle / Daten-Lebenszyklus? https://www.uni-marburg.de/projekte/forschungsdaten/faq/datalifecycle, 2014. [Online; accessed 15-November-2017].

[8] Python Software Foundation. Python Language reference, version 3.6. https://python.org, 2017.

[9] Spirion LCC. What is Data Lifecycle Management? https://www.spirion.com/data-lifecycle-management/, 2017. [Online; accessed 15-November-2017].