Università Cattolica Sacro Cuore

Statistical and Actuarial Sciences

mj: Data Business Analytics

RESTful Scraping API for Real Estate data, a Spatial Bayesian modeling perspective with INLA

Author:
Niccolò Salvini

Supervisor:
Dr. Marco DellaVedova

Assistant Supervisor:
Dr. Vincenzo Nardelli

AY 2019 / 2020

RESTful Scraping API for Real Estate data, a Spatial Bayesian modeling perspective with INLA

Niccolò Salvini[1]

Lastest build: 03 gennaio, 2021

[1] https://niccolosalvini.netlify.app/

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Trento:

- Argomento
- Problema
- Obiettivi
- Metodo
- Struttura della tesi

Main themes:

- Research Question
- Milan Real Estate Controversies in relation to research question
- why the API (perchè mi mancano i dati e perchè è il futuro)
- Open Data discussion personal hope of data sharing and benefits from oper source
- Why a Bayesian approach
- Why INLA

According to Google research devoted to the analysis of USA real estate buyers' habits (National Association of Realtors, Google 2012), out of 10 % homebuyers use the Internet as one of the primary research tools, and 52 % of customers start their search using the Internet. According to the information presented by Google, the number of real estate searches grows up yearly by 22 %. The main reason for customers is to find information which can reduce purchasing related risks (Peterson & Merino, 2003). In Latvia, 79.7 % of the population now goes online regularly (Latvijas Interneta Asociācija, 2015). Latvian Internet users have the same habits as the Internet users of the USA or developed Western countries. 7 out of 10 Internet users (70 %) look for information on web pages (CSP, 2015). Real estate buyers read general information about a real estate object and compare prices of various real estate objects

As a general discussion technologies implied can be thought as the distance between a service running locally on a laptop and something that it can actually be put into production, shared among company stakeholders, solving business related problems. When such technologies are applied data scientist and interlocutors gradually close the gap. Insights are better communicated, data is up-to-date and automation can save time. Nonetheless when the infrastructure is structured with vision then integrating or substituting existing technologies is not trivial. Anyway technologies can not be always embedded because they might be exclusively designed to work only on certain back ends, therefore some choices are not into discussion. With foresight RStudio by setting future-oriented guidelines has spent a lot of effort giving its users an easy, integrated and interconnected environment. By that it is meant that the RStudio community has tried to either integrate or open the possibility to a number of technologies that fill the blanks in their weaker parts. On top of many, an entire package has been dedicated to democratize REST APIs (Plumber (Trestle Technology, LLC, 2018)). As a further example developers in RStudio have created an entire new paradigm i.e. Shiny (Chang et al., 2020), a popular web app development package, that enforces the developer to have front-end and back-end technologies tied up in the same IDE. They also added performance monitoring and optimization packages that are fitted into shiny such as shinytest [metti tag] and shinyloadtest [metti tag] to simulate sessions and verify network traffic congestion.

Chapter 2

Web Scraping

The following chapter covers a modern technique and up-to-date libraries for web scraping in R and related challenges with a focus on the immobiliare.it case. The easiest way of collecting information from websites involves inspecting and manipulating URLs. While browsing web pages urls under the hood compose themselves in the navigation bar according to a certain encoded semantic, specifying domain, url paths and url parameters. As a matter of fact websites can be regarded as a complex system of nested folders where urls are the relative file path to access to the content desired. That implies if a function can mimic the url semantic by providing decoded, human-readable arguments, then any content in the website can be filtered and later accessed. Once urls are reproduced and stored into a variable, scraping function are called on these set of address. The major time improvement with respect to classical crawlers comes from retrieving a set of urls instead of gathering the entire website sitemap and then searching through keywords. This is made exploiting a scraping workflow library, namely rvest Wickham (2019), which takes inspiration from the scraping gold standard Python library Beautiful Soup Contributors (2004). A search algorithm strategy, calibrated on the specific scraping context, is nested inside scraping functions whose skeleton is reproduced for all the information demanded. The search strategy exploits different CSS queries that point to different data location within the web page, with the aim to be sure to carry out a comprehensive exploration and to deliver data where available. Functions are then wrapped up around a “main” function that simplifies portability and enables parallelization. Both of the search logic flows are fronted presenting their pseudocode. HTTP protocol communication and related main concepts are introduced with the aim to familiarize with internet communication layers focusing on traffic from web server to web clients and viceversa and principal actors involved. By doing that some opinionated but highly diffused scraping best practices are incorporated into scraping taking inspiration by a further R library Polite Perepolkin (2019). Revised best practices try to balance scraping efficiency and *web etiquette.* As a result optimization happens both from the *web server* side; this is tackled by kindly asking for permission and sending delayed server requests rate based on Robotstxt file requirements. As well as from the *web client* by preventing scraping discontinuity caused by server blocks thanks to *User Agent* (HTTP request headers) pool *rotation* and *fail dealers. Parallel* computing is carried out due to data obsolescence, since the market on which scraping functions are called is vivid and responsive. At first concepts and are introduced and then main centered on the specific R parallel ecosystem. Then a run time scraping benchmark is presented for two different modern parallel back end options future Bengtsson (2020b) and doParallel Corporation and Weston (2020), along with two parallel looping paradigma, furrr Vaughan and Dancho (2018) and foreach Microsoft and Weston (2020). Both of the two combinations have showed similar results, nevertheless the former offers a more {Tidiverse} coherence and a more comfortable debugging experience. Furthermore an overview of the still unlocked challenges is provided on the open source project and possible improvements, with the sincere hope that the effort put might be extended or integrated. In the end a heuristic overview on the main legal aspect is offered bringing also an orientation inspired by a “fair balance” trade off between web scraping practices and the most common claims. Then a court case study about a well known scraping litigation is brought demonstrating how law is not solid in this area and how scrapign market should find its own equilibrium.

2.1 A Gentle Introduction on Web Scraping

Definition 2.1 (Scraping)**.** Web Scraping is a technique aimed at extracting unstructured data from static or dynamic internet web pages and collecting it in a structured way. Automated data collection, web extraction, web crawling, or web data mining are often used as synonyms to web scraping.

The World Wide Web (WWW or just the web") data accessible today is calculated in zettabytes (Cisco, 2020) (1 zettabyte = 10^{21} bytes). This huge volume of data provides a wealth of resources for researchers and practitioners to obtain new insights in real time regarding individuals, organizations, or even macro-level socio-technical phenomena (Krotov and Tennyson, 2018). Unsurprisingly, researchers of Information Systems are increasingly turning to the internet for data that can address their research questions (2018). Taking advantage of the immense web data also requires a programmatic approach and a strong foundation in different web technologies. Besides the large amount of web access and data to be analyzed, there are three rather common problems related to big web data: variety, velocity, and veracity (Krotov and Silva, 2018), each of which exposes a singular aspect of scraping and constitutes some of the challenges fronted in later chapters. *Variety* mainly accounts the most commonly used mark-up languages on the web used for content creation and organization such as such as HTML (htm, 2020), CSS (css, 2020), XML (contributors, 2020j) and JSON (contributors, 2020g). Sometimes Javascript (contributors, 2020f) components are also embedded into websites. In this cases dedicated parsers are required which would add a further stumbling block. Starting from these points scraping requires at least to know the ropes of these technologies which are also assumed in this analysis. Indeed later a section will be partially dedicated to familiarize with the inner internet mechanism taht manages the exhanges of information, such as HTTP protocols. *Velocity*: web data are in continuous flow status: it is created in real time, modified and changed continuously. This massively impacts the analysis that relies on those data which, as times passes, becomes obsolete. However it also suggests the speed and pace at which data should be collected. From one hand data should be gathered as quicker as possible so that analysis or softwares are up-to-date, section 2.7. From the other this should happen in any case by constraining speed to common shared scraping best practices, which require occasional rests in requesting information. The latter issue is faced later in section 2.4. *Veracity*: Internet data quality and usability are still surrounded by confusion. A researcher can never entirely be sure if the data he wants are available on the internet and if the data are sufficiently accurate to be used in analysis. While for the former point data can be, from a theoretical perspective, accurately selected it can not be by any means predicted to exist. This is a crucial aspects of scraping, it should take care of dealing with the possibility to fail, in other words to be lost. The latter point that points to data quality is also crucial and it is assessed by a thoughtful market analysis that mostly assures the importance of immobiliare.it as a top player in italian real estate. As a consequence data coming from reliable source is also assumed to be reliable. Furthermore web scraping can be mainly applied in two common forms as in (Dogucu and Cetinkaya, 2020): the first is browser scraping, where extractions takes place through HTML/XML parser with regular expression matching from the website's source code. The second uses programming interfaces for applications, usually referred to APIs. The main goal of this chapter is to combine the two by shaping a HTML parser and make the code portable into an RESTful API whose software structure is found at **??**. Regardless of how difficult it is the process of scraping, the circle introduced in aut (2014) and here revised with respect to the end goal, is almost always the same. Most scraping activities include the following sequence of tasks:

- Identification of data
- Algorithm search Strategy
- Data collection
- Data preprocess
- Data conversion

- Debugging and maintenance
- Portability

Scraping essentially is a clever combination and iteration of the previous tasks which should be heavily shaped on the target website or websites.

2.2 Anatomy of a url and reverse engineering

Uniform Resource Locators (URLs) (contributors, 2020h), also known as web addresses, determine the location of websites and other web contents and mechanism to retrieve it. Each URL begins with a scheme, purple in figure 2.2, specifying the protocol used to communicate client-application-server contact i.e. HTTPS. Other schemes, such as ftp (File transmission protocol) or mailto for email addresses correspond to SMTP (Simple Mail Transfer Protocol) standards.

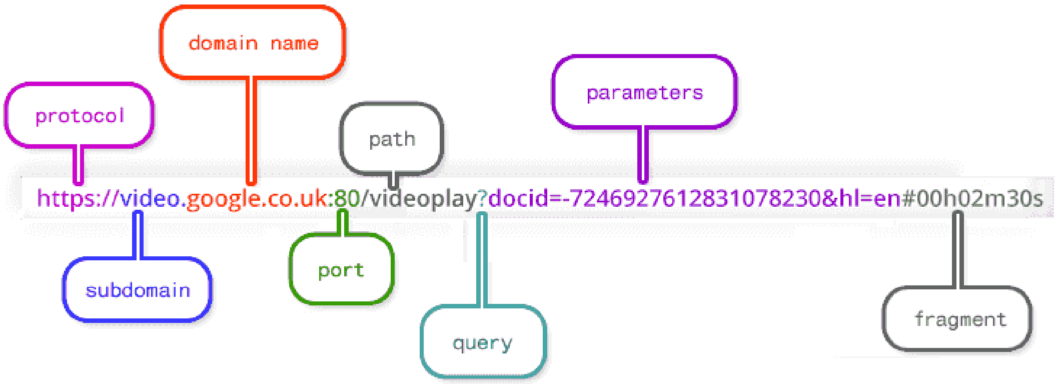

domain name
protocol
path
parameters
https://video.google.co.uk:80/videoplay?docid=-7246927612831078230&hl=en#00h02m30s
subdomain
port
query
fragment

Figure 2.1: General url Anatomy, Doepud (2010) source

The *domain name* in red in figure 2.2, with a specific ID, indicates the server name where the interest resource is stored. The domain name is sometimes accompanied with the *port* whose main role is to point to the open door where data transportation happens. Default port for Transmission Control Protocol (*TCP*) is 80. In the following chapter ports will play a crucial role since a compose file wiill instruct containers (3.2) to open and route communication through these channels and then converge the whole traffic into a secured one 3.3. Typically domain name are designed to be human friendly, indeed any domain name has its own proprietary IP and public IP (IPv4 and IPv6) address which is a complex number, e.g. local machine IP is 95.235.246.91. Here intervenes *DNS* whose function is to redirect domain name to IP and vice versa. The path specifies where the requested resource is located on the server and typically refers to a file or directory. Moving towards folders requires name path locations separated by slashes. In certain scenarios, URLs provide additional *path* information that allows the server to correctly process the request. The URL of a dynamic page (those of who are generated from a data base or user-generated web content) sometimes shows a *query* with one or more parameters followed by a question mark. *Parameters* follow a "field=value" scheme each of which is separated by a "&" character for every different parameter field, 6th character from the end in figure 2.2 parameter space. Each further parameter field added is appended at the end of the url string enabling complex and specific search queries. *Fragments* are an internal page reference often referred to as an anchor. They are typically at the end of a URL, starting with a hash (#) and pointing to specific part of a HTML document. Note that this is a full browser client-side operation and fragments are used to display only certain parts of already rendered website. The easiest way of collecting information from websites often involves inspecting and manipulating URLs which refer to the content requested. Therefore urls are the starting point for any scraper and they are also the only functional argument to be feeding to. Url parameters indeed help to route web clients to the requested information and this is done *under the hood* and unconsciously while the user is randomly browsing the web page. Let us assume to express the necessity to scrape only certain information from a generic website. Then if this data to be scraped can be circumscribed by properly setting url parameters, under those circumstances it is also demanded a way to consciously compose these urls. In other words a function (or a sevice) needs to mould urls by mimicking their mutations operated by browser while the user is navigating. As a matter of fact as filters (i.e. parameters) are applied to the initial search page for immobiliare.it then parameters populate the navigation bar. For each of the filters specified a new parameter field and its own selected value are appended to the url string according to the website specific *semantic*. Each website has its own sematic. As an example are applied directly to the domain name https://www .immobiliare.it/ the following filters (this is completely done in the dedicated panel):

- rental market (the alternative is selling)
- city is Milan
- bathrooms must be 2
- constrained zones search on "Cenisio" and "Fiera"

The resulting url is:

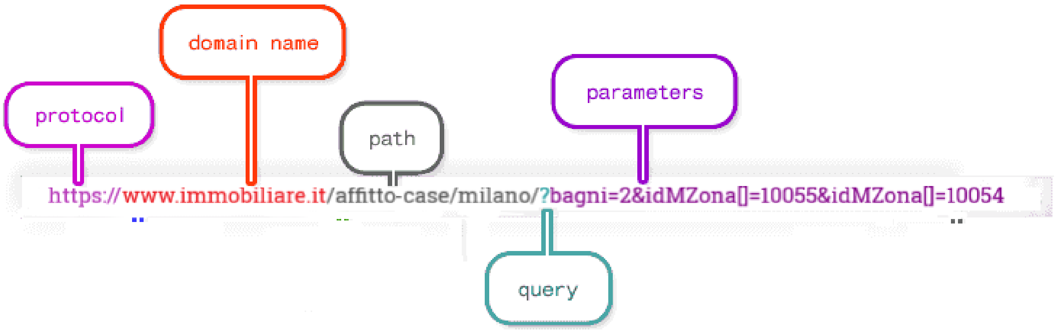

domain name
protocol
path
parameters
https://www.immobiliare.it/affitto-case/milano/?bagni=2&idMZona[]=10055&idMZona[]=10054
query

Figure 2.2: immobiliare url composed according to some filters, author's source

At first glance immobiliare.it does not enact a *clean url* (contributors, 2020a) structure, which ultimately would simplify a lot the reverse sematic process. In truth parameters follows a chaotic semantic. While the filter city="milano" is located in the url path the remaining are transferred to the url parameters. For some of them the logical trace back from the parameter field-value to their exact meaning is neat, i.e. bagni=2 and can be said clean. Unfortunately for the others the sematinc is not obvious and requires a further reverse layer to deal with. In addition to this fact the url in figure 2.2 readdress to the very first page of the search meeting the filter requirements, which groups the first 25 rental advertisements. The following set of 25 items can be reached by relocating the url address to: https://www.immobiliare.it/ affitto −case/milano/?bagni=2&idMZona[]=10055&idMZona[]=10054&pag=2. The alteration regards the last part of the url and constitutes a further url parameter field to look for. Note that the first url does not contain "&pag=1". For each page browsed by the user the resulting url happen to be the same plus the prefix "&pag=" glued with the correspondent n page number (from now on referred as *pagination*). This is carried on until pagination reaches either a stopping criteria argument or the last browsable web page (pagination sets as default option 10 pages, which returns a total of 250 rental advertisement). Therefore for each reversed semantic url are obtainable 25 different links, see figure 2.3 that is where actually are disclosed relevant data and the object of scraping. Links belonging to the 25 items set share the same anatomy: https ://www.immobiliare.it/annunci/84172630/ where the last *path* is associated to a unique ID, characteristic of the rental property. Unfortunately there is not any available solution to retrieve all the existing ID, therefore links needs to collected directly from "main" url in figure 2.2 as an obvious choice.

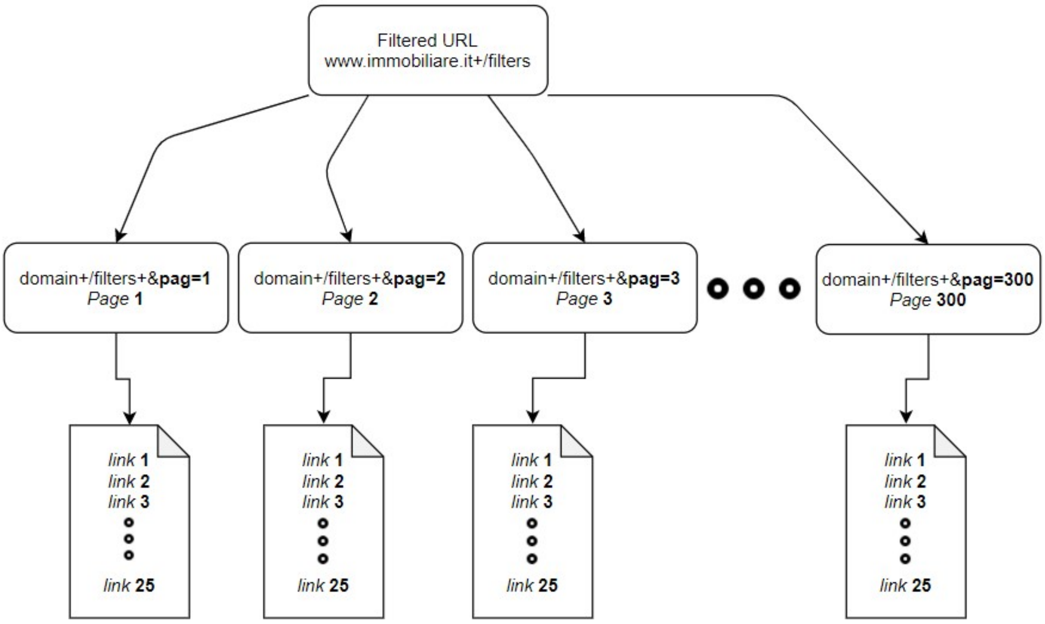


Figure 2.3: immobiliare.it website structure, author's source

Therefore 4 steps are required to *reverse engineer* the url (2014) and to ultimately make the links access available:

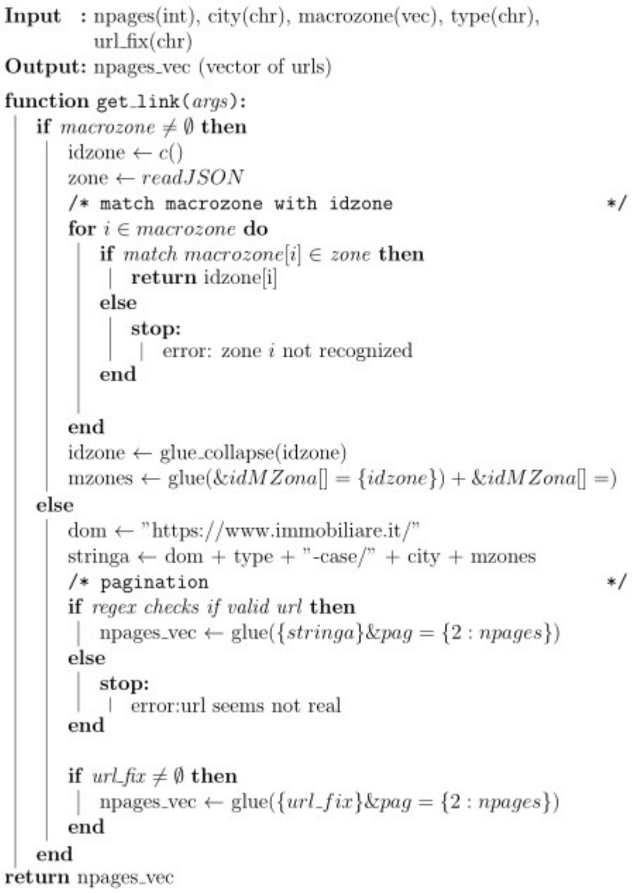
1. Determine how the URL syntactic works for each parameter field
2. Build the url based on the reverse engineered sematic:

    a. those ones who need an ID parameter value

    b. clean ones who need to only explicit the parameter value

3. Pagination is applied to the url until stopping criteria are met
4. Obtain the links for each urls previously generated

Once identified the inner composition mechanism by an insistent trial and error then the url restoration could starts. Parameter values for those that requires an ID in figure 2.2 are encoded is a way that each number is associated to its respective zone ID e.g. as in figure 2.2 “Fiera” to 10055 or “Cenisio” to 10054. Therefore a decoding function should map the parameter value number back to the correspondent human understandable name e.g from 10055 to “Fiera” or 10054 to “Cenisio”, the exactly opposite logical operation. The decoding function exploits a JSON database file that matches for each ID its respective zone name exploiting suitable JSON properties. The JSON file is previously compounded by generating a number of random url queries and subsequently assigning the query names to their respective ID. As soon as the JSON file is populated the function can take advantage of that database and compose freely urls at need. Pagination generates a set of urls based on the number of pages requested. In the end for each urls belonging to the set of urls, links are collected by a dedicated function and stored into a vector. Furthermore if the user necessitate to directly supply a precomposed url the algorithm overrides the previous object and applies pagination on the provided url. The pseudocode in figure 2.4 paints the logic behid the reverse egineering process.



```
Input  : npages(int), city(chr), macrozone(vec), type(chr),
         url_fix(chr)
Output: npages_vec (vector of urls)
function get_link(args):
    if $macrozone \neq \emptyset$ then
        idzone ← $c()$
        zone ← $readJSON$
        /* match macrozone with idzone                        */
        for $i \in macrozone$ do
            if match $macrozone[i] \in zone$ then
                return idzone[i]
            else
                stop:
                    error: zone $i$ not recognized
            end
        end
        idzone ← glue_collapse(idzone)
        mzones ← glue($\&idMZona[] = \{idzone\}) + \&idMZona[] =$)
    else
        dom ← "https://www.immobiliare.it/"
        stringa ← dom + type + "-case/" + city + mzones
        /* pagination                                          */
        if regex checks if valid url then
            npages_vec ← glue($\{stringa\}\&pag = \{2 : npages\}$)
        else
            stop:
                error:url seems not real
        end

        if $url\_fix \neq \emptyset$ then
            npages_vec ← glue($\{url\_fix\}\&pag = \{2 : npages\}$)
        end
    end
return npages_vec
```

Figure 2.4: pseudo code algorithm to reverse engineer url, author's source

2.2.1 Scraping with `rvest`

Reverse engineered urls are then feeded to the scraper which arranges the process of scraping according to the flow chart imposed by rvest in figure 2.5. The sequential path followed is highlighted by the light blue wavy line and offers one solution among all the alternative ways to get to the final content. The left part with respect to the central dashed line of figure 2.5 takes care of setting up the session and parsing the response. As a consequence at first scraping in consistent way requires to open a session class object with html_session. Session arguments demands both the target url, as built in 2.4 and the request headers that the user may need to send to the web server. Data attached to the web server request will be further explored later in section2.5.1, tough they are mainly 4: User Agents, emails references, additional info and proxy servers. The session class objects contains a number of useful data regarding either the user log info and the target website such as: the url, the response, cookies, session times etc. Once the connection is established (response 200), functions that come after the opening rely on the object and its response. In other words while the session is happening the user will be authorized with the already provided headers data. As a result jumps from a link to the following within the same session are registered by in the object. Most importantly sessions contain the xml/html content response of the webpage, that is where data needs to be accessed.

Indeed at the right of dashed line in 2.5 are represented the last two steps configured into two rvest(Wickham, 2019) functions that locate the data within the HTML nodes and convert it into a human readble text, i.e. from HTML/XML to text. The most of the times can be crafty to find the exact HTML node or nodes set that contains the data wanted, expecially when HTML trees are deep nested and dense. html_node function provides and argument that grants to specify a simple CSS/XPATH selector which may group a set of HTML/XML nodes or a single node. Help comes from an open source library and browser extension technology named SelectorGadget. Selector-

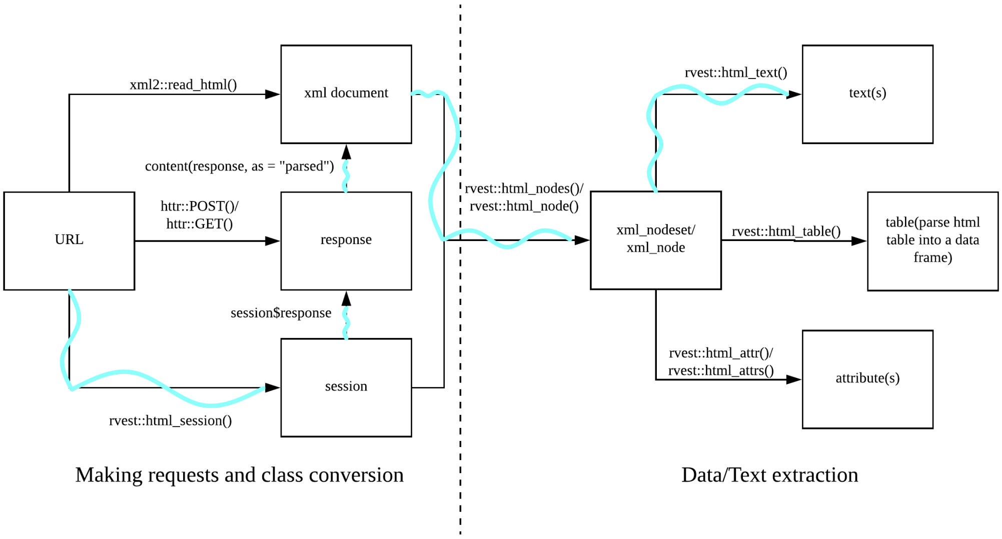


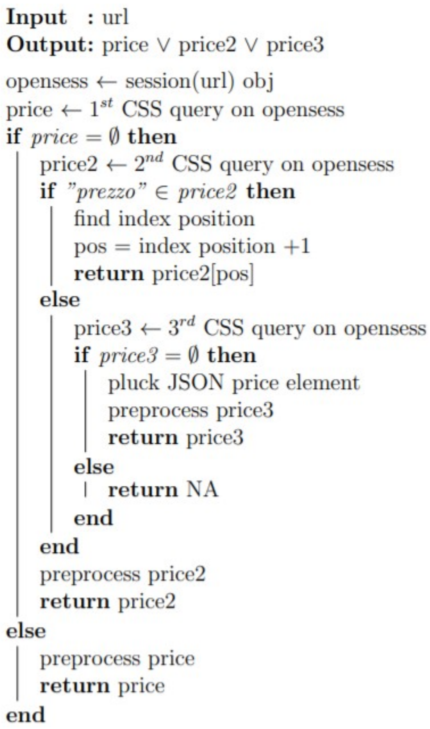
Figure 2.5: rvest general flow chart, author's source

Gadget (Cantino and Maxwell, 2013) which is a JavaScript bookmark that allows to interactively explore which CSS selector is needed to gather desired components from a webpage. Data can be repeated many times into webpages so the same information can be found at multiple CSS queries. This fact highlights one of the priciples embodied in the follwing section 2.3 accroding to which searching methods gravitates. Once the CSS query points address to the desired content then data finally needs to be converted into text. This is what explicitly achieve hmlt_text.

2.3 Searching Technique for Scarping

The present algorithm in figure 2.6 imposes a nested sequential search strategy and gravitates around the fact that data within the same webpage is repeated many times, so multiple CSS queries are avaialble for the same information. Furthermore since data is repeated has also less probability to be missed. CSS sequential searches are calibrated from the highest probability of appearance in that CSS selector location to the lowest so that most visited locations are also the the most likely to grab data. A session object opensess, the one seen in

2.5), is initialized sending urls built in 2.4. The object opensess constitutes a check point obj because it is reused more than once along the algorithm flow. The object contains session data as well as HTML/XML content. Immediately after another object price parses the sessions and points to a CSS query through a set of HTML nodes. The CSS location .im—mainFeatures___title addresses a precise group of data which are found right below the main title. Expectations are that monthly price amount in that location is a single character vector string, containing price along with unnecessary non-UTF characters. Then the algorithm bumps into the first if statement. The logical condition checks whether the object price first CSS search went lost. If it does not the algorithm directly jumps to the end of the algorithm and returns a preprocessed single quantity. Indeed if it does it considers again the check point opensess and hits with a second css query .im—features___value , .im—features___title, pointing to a second data location. Note that the whole search is done within the same session (i.e. reusing the same session object), so no more additional request headers 2.5.1 has to be sent). Since the second CSS query points to data sequentially stored into a list object, the newly initialized price2 is a type list object containing various information. Then the algorithm flows through a second if statement that checks whether ”prezzo” is matched in the list, if it does the algorithm returns the +1 position index element with respect to the “prezzo” position. This happens because data in the list are stored by couples sequentially (as a flattened list), e.g. list(title, “Appartamento Sempione”, energy class, “G”, “prezzo”, 1200/al mese). Then in the end a third CSS query is called and a further nested if statement checks the emptiness of the latest CSS query. price3 points to a hidden JSON object within the HTML content. If even the last search is lost then the algorithm escapes in the else statement by setting NA_Character_, ending with any CSS query is able to find price data. The search skeleton used for price scraping constitutes a standard reusable search method in the analysis for all the scraping functions. However for some of the information not all the CSS location points are available and the algorithm is forced to be rooted to only certain paths, e.g. “condizionatore” can not be found under main title and so on.

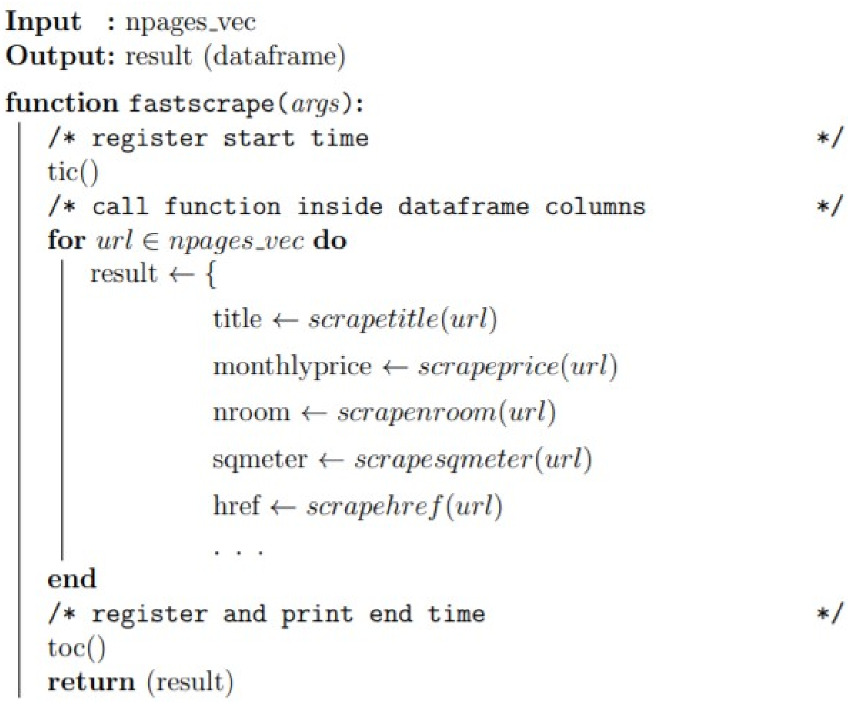


```
Input  : url
Output: price ∨ price2 ∨ price3
opensess ← session(url) obj
price ← $1^{st}$ CSS query on opensess
if $price = \emptyset$ then
    price2 ← $2^{nd}$ CSS query on opensess
    if "prezzo" ∈ price2 then
        find index position
        pos = index position +1
        return price2[pos]
    else
        price3 ← $3^{rd}$ CSS query on opensess
        if $price3 = \emptyset$ then
            pluck JSON price element
            preprocess price3
            return price3
        else
            return NA
        end
    end
    preprocess price2
    return price2
else
    preprocess price
    return price
end
```

Figure 2.6: pseudo code algorithm for price search, author’s source

Once all the functions have been designed and optimized with respect to their scraping target they need to be grouped into a single “main” *fastscrape* function, figure 2.7. This is accomplished into the API function endpoint addressed in section 3.1.1, which also applies some sanitization, next chapter section 3.1.2, on users inputs and administers also some security API measures, as in 3.1.3 outside the main function. Moreover as it will be clear in section 2.7 the parallel back end will be registered outside the scraping function for unexplained inner functioning reasons.

pseudo code



```
Input  : npages_vec
Output: result (dataframe)
function fastscrape(args):
    /* register start time                                  */
    tic()
    /* call function inside dataframe columns               */
    for url ∈ npages_vec do
        result ← {
                    title ← scrapetitle(url)
                    monthlyprice ← scrapeprice(url)
                    nroom ← scrapenroom(url)
                    sqmeter ← scrapesqmeter(url)
                    href ← scrapehref(url)
                    . . .
    end
    /* register and print end time                          */
    toc()
    return (result)
```

Figure 2.7: pseudo code algorithm structure for fatstscrape, author's source

2.4 Scraping Best Practices and Security provisions

Web scraping have to naturally interact multiple times with both the *client* and *server side* and as a result many precautions must be seriously taken into consideration. From the server side a scraper can forward as many requests as it could (in the form of sessions opened) which might cause a traffic bottleneck (DOS attack contributors (2020b)) impacting the overall server capacity. As a further side effect it can confuse the nature of traffic due to fake user agents 2.5.1 and proxy servers, consequently analytics reports might be driven off track. Those are a small portion of the reasons why most of the servers have their dedicated Robots.txt files. Robots.txt Meissner (2020) are a way to kindly ask webbots, spiders, crawlers to access or not access certain parts of a webpage. The de facto "standard" never made it beyond a *informal* "Network Working Group INTERNET DRAFT". Nonetheless, the use of robots.txt files is widespread due to the vast number of web crawlers (e.g. Wikipedia robot[1], Google robot[2]). Bots from the own Google, Yahoo adhere to the rules defined in robots.txt files, although their *interpretation* might differ.

Robots.txt files (Meissner and Ren, 2020) essentially are plain text and always found at the root of a website's domain. The syntax of the files follows a field-name value scheme with optional preceding user-agent. Blocks are separated by blank lines and the omission of a user-agent field (which directly corresponds to the HTTP user-agent field, cleared later in 2.5.1) is seen as referring to all bots. The whole set of possible field names are pinpointed in Google (2020), some important are: user-agent, disallow, allow, crawl-delay, sitemap and host. Some common standard interpretations are:

- Finding no robots.txt file at the server (e.g. HTTP status code 404) implies full permission.
- Sub-domains should have their own robots.txt file, if not it is assumed full permission.
- Redirects from subdomain www to the domain is considered no domain change - so whatever is found at the end of the redirect is considered to be the robots.txt file for the subdomain originally requested.

A comprehensive scraping library that observes a web etiquette is the polite (Perepolkin, 2019) package. Polite combines the effects of robotstxt, ratelimitr (2018) to limit sequential session requests together with the memoise (Wickham et al., 2017) for robotstxt response caching. Even though the solution meets the politeness requirements (from server and client side) ratelimitr is not designed to run in parallel as documented in the vignette (Shah, 2018). This is a strong limitation as a result the library is not applied. However the 3 simple but effective web etiquette principles around, which is the package wrapped up, describe what are the guidelines for a "polite" session:

[1] https://en.wikipedia.org/robots.txt

[2] https://www.google.com/robots.txt

> The three pillars of a polite session are seeking permission, taking slowly and never asking twice.
>
> — Dmytro Perepolkin, polite author

The three pillars constituting the *Ethical* web scraping manifesto (Densmore, 2019) are considered as common shared *best practices* that are aimed to self regularize scrapers. In any case these guidelines have to be intended as eventual practices and by no means as required by the law. However many scrapers themselves, as website administrators or analyst, have been fighting for many and almost certainly coming years with bots, spiders and derivatives. As a matter of fact intensive scraping might fake out real client navigation logs and confuse digital footprint, which results in induced distorted analytics. On the other hand if data is not given and APIs are not available, then scraping is sadly no more an option. Therefore throughout this analysis the goal will be trying to find a well balanced trade-off between interests on the main actors involved. Newly balanced (with respect to the author thought) guidelines would try to implement at first an obedient check on the validity of the path to be scraped through a cached function. Secondly it will try to limit request rates to a more feasible time delay, by keeping into the equation also the client time constraints. In addition it should also guarantee the continuity in time of scraping not only from the possibility to fail section **??**(possibly)), but also from servers "unfair" denial (in section 2.5.1). With that said a custom function caches the results of robotstxt into a variable i.e. polite_permission . Then paths allowed are evaluated prior any scraping function execution. Results should either negate or affirm the contingency to scrape the following url address.

```
## Memoised Function:
## [1] TRUE
```

polite_permission is then reused to check, if any, the server suggestion on crawl relay. In this particular context no delays are kindly advised. As a default polite selection delay request rates are set equal to 2.5 seconds. Delayed are managed through the purrr stack. At first a rate object is initialized based on polite_permission results, as a consequence a rate_sleep delay is defined and iterated together with any request sent, as in Lionel Henry RStudio (2020b).

```
get_delay = function(memoised_robot, domain) {

    message(glue(”Refreshing␣robots.txt␣data␣for␣%s...␣
      {domain}”))
    temp = memoised_robot$crawl_delay

    if (length(temp) > 0 && !is.na(temp[1, ]$value)) {
        star = dplyr::filter(temp, useragent == ”*”)
        if (nrow(star) == 0)
            star = temp[1, ]
        as.numeric(star$value[1])
    } else {
        2.5
    }

}
get_delay(rbtxt_memoised, domain = dom)
```

```
## [1] 2.5
```

2.5 HTTP overview

URLs in browsers as in 2.1 are a way to access to web content whether this accounts for getting from one location to the following, or checking mails, listening to musics or downloading files. However Internet communication is a stratification of layers conceived to make the whole complex system working. In this context URLs are the layer responsible for *user interaction.* The rest of the layers which involve techniques, standards and protocols, are called in ensemble Internet Protocols Suite (*IPS*) (2014). The TCP (Transmission Control Protocol) and IP (Internet Protocol) are two of the most important actors on the IP Suite and their role is to represent the *Internet layer* (IP) and the *transportation layer* (TCP). In addition they also guarantee trusted transmission of data. Inner properties and mechanism are beyond the scope of the analysis, luckily they are not required in this context. Resting on the transportation layer there are specialized message exchange protocols such as the HTTP (Hyper Text Transfer Protocol), FTP (File Transfer Protocol). In addition, there may be also e-mail exchange protocols for transmission, such as Post Office Protocol (POP) for Email Recovery and for storage and retrieval, IMAP (Internet Message Acccess Protocol). The aforementioned protocols describe default vocabulary and procedures to address particular tasks for both clients and servers and they can be ascribed to the transportation layer. HTTP is the most widespread, versatile enough to ask for almost any resource from a server and can also be used to transfer data to the server rather than to recover. In figure 2.8 is painted a schematized version of a User-Server general HTTP communication/session. If a website e.g. immobiliare.it[3] is browsed from a web browser e.g. Chrome (the HTTP client) then the client is interrogating a Domain Name System (DNS) which IP address is associated with the domain part of the URL. When the browser has obtained the IP address as response from the DNS server then connection/session is established with HTTP server via TCP/IP. From a content perspective data is gathered piece-by-piece, if the content is not exclusively HTML then the the browser client renders the content as soon as it receives data.

Furthermore HTTP should be borne in mind two essential facts.

- HTTP is not only a hypertext protocol, but it is used for all sorts of services (i.e. APIs)
- HTTP is a *stateless* protocol, which means that response and request

[3] https://www.immobiliare.it/

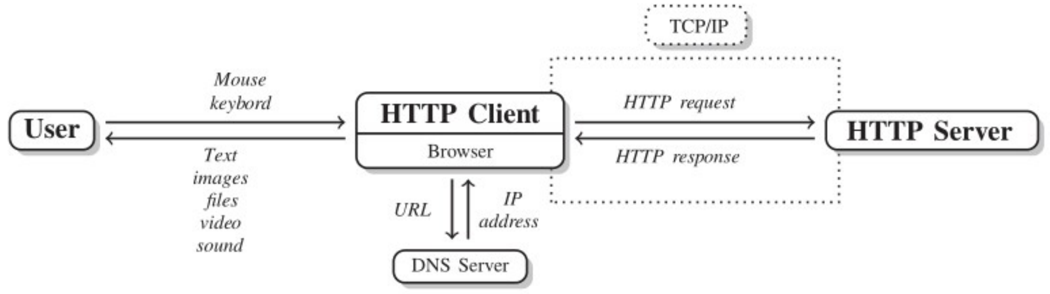


Figure 2.8: User-Server communication scheme via HTTP, source aut (2014)

and the the resulting interaction between client and server is managed by default as it were the first time they connected.

HTTP messages consist of three sections – start line, *headers* and body – whether client requests or server response messages. The start line of the server response begins with a declaration on the highest HTTP version. The header underneath the start line contains client and server meta data on the preferences for content displaying. Headers are arranged in a collection of name-value pairs. The body of an HTTP message contains the content of the response and can be either binary or plain text. Predominantly in the context of the analysis are headers which are the fields where lay the definition of the actions to take upon receiving a request or response. For what concerns the analysis the focus is on *identification headers* (both in request and response) whose role is to describe user preferences when sessions are opened i.e. default language, optimization of content display throughout devices. The exploited ones are:

(metti anche altra roba da altre fonti)

- *Authorization* (request): authentication field allows to insert personal credentials to access to dedicated content (log in to the immobiliare.it account). Credentials in requests are not really encrypted, rather they are encoded with Base64. Therefore they can be easily exposed to security breaches, which it does not happen when using HTTP (HTTP

Security) that applies encryption to basic authentication, extensively presented in 3.4.

- *Set-Cookie* (response): Cookies allow the server to keep user identity. They are a method to transform *stateless HTTP* communication (second point in previous) into a secure discussion in which potential answers rely on past talks.
- *User-Agent* (request), faced in next section 2.5.1.

2.5.1 User Agent and further Identification Headers Spoofing

Definition 2.2 (User Agents)**.** The user Agent (from now on refered as UA) “retrieves, renders and facilitates end-user interaction with Web content” (User:Jallan, 2011).

In the HTTP identification headers the UA string is often considered as *content negotiator* (contributors, 2020i). One the basis of the UA, the web server can negotiate different CSS loadings, custom JavaScript delivery or it can automatically translate content on UA language preferences (WhoIsHostingThis.com, 2020). UA is a dense content string that includes many User details: the user application or software, the operating system (and versions), the web client, the web client’s version, as well as the web engine responsible for content displaying (e.g. AppleWebKit). When the request is sent from an application software as R (no web browser), the default UA is set as:

```
## [1] ”libcurl/7.64.1 r−curl/4.3 httr/1.4.2”
```

Indeed when a request comes from a web browser a full UA example and further components breakdown is (current machine):

Mozilla/5.0 (Windows NT 6.3; WOW64)AppleWebKit/537.36 (KHTML,
like Gecko)Chrome/45.0.2454.85 Safari/537.36

- The browser application is Mozilla build version 5.0 (i.e. 'Mozilla-compatible').
- The Operating System is Windows NT 6.3; WOW64, running on Windows.
- The Web Kit provides a set of core classes to display web content in windows (UserAgentString.com, 1999), build version 537.36.
- The Web Browser is Chrome version 45.0.2454.85.
- The client is based on Safari, build 537.36.

The UA string is also one of the main responsible according to which Web crawlers and scrapers through a dedicated name field in robotstxt 2.4 may be ousted from accessing certain parts of a website. Since many requests are sent the web server may encounter insistently the same UA (since the device from which they are sent is the same) and as consequence it may block requests associated to the own UA. In order to avoid server block the scraping technique adopted in this analysis rotates a pool of UAs. That means each time requests are sent a different set of headers are drawn from the pool and then combined. The more the pool is populated the larger are the UA combinations. The solution proposed tries in addition to automatically and periodically resample the pool as soon as the website from which U agents ID are extracted updates newer UA strings.

```
set.seed(27)
url = "https://user-agents.net/"
agents = read_html(url) %>% html_nodes(css = ".agents_
    list␣li") %>% html_text()

agents[sample(1)]
```

```
## [1] "Mozilla/5.0 (Linux; Android 7.0; SM-G928F Build
    /NRD90M; wv) AppleWebKit/537.36 (KHTML, like Gecko)
    Version/4.0 Chrome/87.0.4280.101 Mobile Safari
    /537.36 [FB_IAB/FB4A;FBAV/300.2.0.58.129;]"
```

The same procedure has been applied to the From identification header attached to the request. E-mails, that are randomly generated from a website, are scraped and subsequently stored into a variable. A further way to imagine what this is considering low level API calls to dedicated servers nested into a more general higher level API. However UA field name technology has been recently sentenced as superseded in favor of a newer (2020) proactive content negotiator named *Hints* contributors (2020e).

An even more secure approach may be accomplished rotating proxy servers between the back and forth sending-receiving process.

Definition 2.3 (Proxy Server)**.** A proxy server is a gateway between the web user and the web server.

While the user is exploiting a proxy server, internet traffic in the form of HTTP request, figure 2.9 flows through the proxy server on its way to the HTTP server requested. The request then comes back through that same proxy server and then the proxy server forwards the data received from the website back to the client. Form a user perspective benefits regards:

- anonymity on the Web
- overcome restriction access to IPs imposed to requests coming from certain locations

Many proxy servers are offered as paid version. In this particular case security barrijers are not that high and this suggests to not apply them. As a further disclaimer many online services are providing free proxies server access, but this comes at a personal security cost due to a couple of reasons:

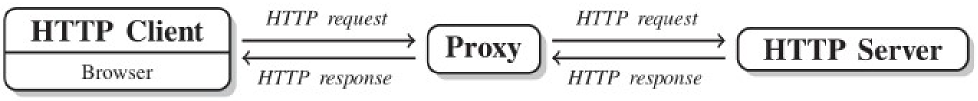
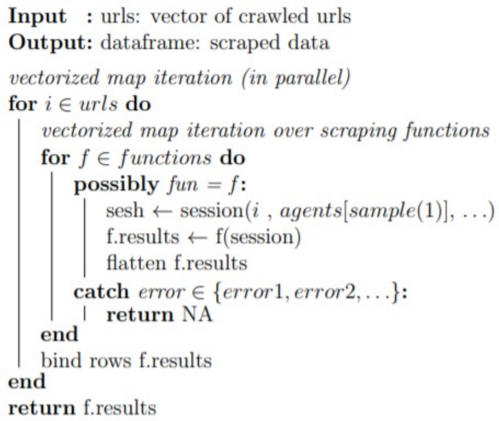


Figure 2.9: proxy middle man„ source aut (2014)

- Free plan Proxies are shared among a number of different clients, so as long as someone has used them in the past for illegal purposes the client is indirectly inheriting their legal infringements.
- Very cheap proxies, for sure all of the ones free, have the activity redirected on their servers monitored, profiling in some cases a user privacy violation issue.

2.6 Dealing with failure

During scraping many difficulties coming from different source of problems are met. Some of them may come from the website's layout changes (2.3), some of them may regard internet connection, some other may have been caused by security breaches (section 2.5.1). One of the most inefficient event it can happen is an unexpected error thrown while sending requests that causes all the data acquired and future coming going lost. In this particular context is even more worrying since scraping "main" functions is able to call 34 different functions each of which points to a different data location. Within a single function invocation, pagination contributes to initialize 10 pages. Each single page includes 25 different single links (**??**) leading to a number of 8500 single data pieces. Unfortunately the probability given 8500 associated to one piece being lost, unparsed is frankly high. For all the reasons said scraping functions needs to deal with the possibility to fail. This is carried out by the implementation of purrr vectorization function map (and its derivatives) and the adverb possibly Lionel Henry RStudio (2020a). *Possibly* takes as argument a function (map iteration over a list) and returns a modified version. In this case, the modified function returns an empty dataframe regardless of the error thrown. The approach is strongly encouraged when functions need to be mapped over large objects and time consuming processes as outlined in Hadley Wickham (2017) section 21.6. Moreover vecrotizaion is not only applied to a vector of urls, but also to a set of functions defined in the environemnt.



```
Input  : urls: vector of crawled urls
Output: dataframe: scraped data
vectorized map iteration (in parallel)
for $i \in urls$ do
    vectorized map iteration over scraping functions
    for $f \in functions$ do
        possibly $fun = f$:
            sesh ← session($i$ , $agents[sample(1)]$, ...)
            f.results ← f(session)
            flatten f.results
        catch $error \in \{error1, error2, \ldots\}$:
            return NA
    end
    bind rows f.results
end
return f.results
```

Figure 2.10: pseudo code for a generic set of functions applied with possibly fail dealers , author's source

2.7 Parallel Scraping

Scraping run time is crucial when dealing with dynamic web pages. This assumption is stronger in Real Estate rental markets where time to market is a massive competitive advantage. From a run time perspective the dimension of the problem requires as many html session opened as single links crawled (refer to previous section 2.6). As a result computation needs to be *parallelized* in order to be feasible. The extraordinary amount of time taken in a non-parallel environment is caused by R executing scraping on a single processor *sequentially* url-by-url in a queue, left part of figure 2.11 (i.e. single threaded computing).

Definition 2.4 (parallel)**.** *Parallel execution* is characterized as multiple operations taking place over overlapping time periods. (Eddelbuettel, 2020)

This requires multiple execution units and modern processors architecture provide multiple cores on a single processor and a way to redistribute computation (i.e. multi threaded computing). As a result tasks can be split into smaller chunks over processors and then multiple cores for each processor, right part of figure 2.11. Therefore Parallel scraping (sometimes improperly called asynchronous[4]) functions are proposed, so that computation do not employ vast

[4] https://medium.com/@cummingsi1993/the-difference-between-asynchronous-and-parallel-6400729fa897

cpu time (i.e. cpu-bound) and space.

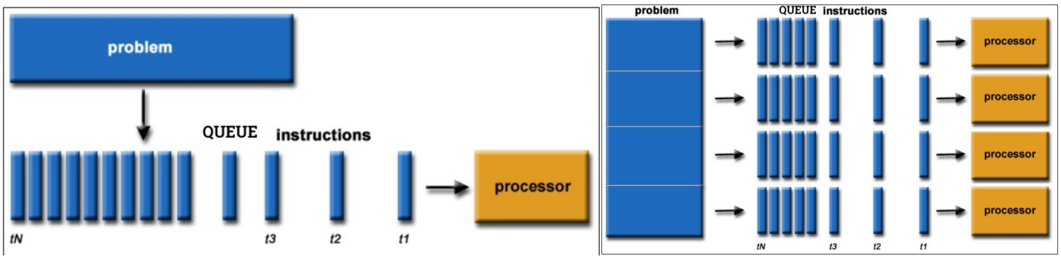


Figure 2.11: single threaded computing vs parallel computing, Barney (2020) source

Parallel execution heavily depends on hardware and software choice. Linux environments offers multi-core computation through *forking* (contributors, 2020c) (only on Linux) so that global variables are directly inherited by child processes. As a matter of fact when computation are split over cores they need to import whatever it takes to be carried out, such as libraries, variables, functions. From a certain angle they need to be treated as a containerized stand-alone environments. This can not happen in Windows (local machine) since it does not support multicore.

```
future::supportsMulticore()
```

```
## [1] FALSE
```

Cluster processing is an alternative to multi-core processing, where parallelization takes place through a collection of separate processes running in the background. The parent R session instructs the dependencies that needs to be sent to the children sessions. This is done by registering the parallel back end. Arguments to be supplied mainly regards the strategy (i.e. multi-core cluster, also said multisession) and the *working group*. The working group is a software concept (par, 2020), that points out the number of processes and their relative computing power/memory allocation according to which the task is going to be split. Moreover from a strictly theoretic perspective the *workers* (i.e. working group single units) can be greater than the number of physical cores detected. Although parallel libraries as a default choice (and choice for this analysis) initializes *as many workers as* physical HT (i.e. Hyper Threaded) *cores.* Parallel looping constructor libraries generally pops up as a direct cause of new parallel packages. The latest research activity by Bengtsson Bengtsson (2020c) indeed tries to unify all the previous back ends under the same umbrella of doFuture . The latter library allows to register many back ends for the most popular parallel looping options solving both the dependency inheritance problem and the OS agnostic challenge. The two alternatives proposed for going parallel are Future Bengtsson (2020b) with furrr Vaughan and Dancho (2018) and doFuture (2020c) along with the foreach Microsoft and Weston (2020) loop constructor. The former is a generic, low-level API for parallel processing as in Bengtsson (2020d). The latter takes inspiration by the previous work and it provides a back-end agnostic version of doParallel Corporation and Weston (2020). Further concepts on parallel computing are beyond the scope of the analysis. However they can be explored in Barney (2020), which may offers a comprehensive perspective on Parallel theory both on hardware and software. Indeed for a full reference on the R parallel ecosystem, run time simulations and advanced algorithm back end design strategies, the authorities are par (2020). If the interest is to cut short theory and directly put existing R code into parallel, a valuable resource is covered in blog[5], which also investigate the main debugging aspects.

2.7.1 Parallel furrr+future

cerca di centrare di più su scraping

Simulations are conducted on a not-rate-delayed (section 2.4) and restricted set of functions which may be considered as a "lightweight" version of the final API scraping endpoint. As a disclaimer run time simulations may not be really representative to the problem since they are performed on a windows 10, Intel(R) Core(TM) i7-8750H 12 cores RAM 16.0 GB local machine. Indeed

[5] https://nceas.github.io/oss-lessons/parallel-computing-in-r/parallel-computing-in-r.html

the API is served on a Linux Ubuntu distro t3.micro 2 cores RAM 1.0 GB server which may adopt forking. Simulations for the reasons said can only offer a run time performance approximation for both of the parallel + looping constructor combinations.

The first simulation considers furrr which enables mapping (i.e. vectorization with map) through a list of urls with purrr and parallelization with Future . Future gravitates around a programming concept called "future", initially introduced in late 70's by Baker (Baker and Hewitt, 1977). Futures are abstractions for values that may be available at a certain time point later (2020b). These values are result of an evaluated expression, this allows to actually divide the assignment operation from the proper result computation. Futures have two stages *unresolved* or *resolved*. If the value is queried while the future is still unresolved, the current process is blocked until the stage is resolved. The time and the way futures are resolved massively relies on which strategy is used to evaluate them. For instance, a future can be resolved using a *sequential* strategy, which means it is resolved in the current R session. Other strategies registered with plan(), such as *multi-core* (on Linux) and *multisession*, may resolve futures in parallel, as already pointed out, by evaluating expressions on the current machine in forked processes or concurrently on a cluster of R background sessions. With parallel futures the current/main R process does not get "bottlenecked", which means it is available for further processing while the futures are being resolved in separate processes running in the background. Therefore with a "multisession" plan are opened as many R background sessions as workers/cores on which chunks of futures (urls) are split and resolved in parallel. From an algorithmic point of view It can be compared to *a divide and conquer* strategy where the target urls are at first redistributed among workers/cores (unresolved) through background sessions and then are scraped in equally distributed chunks (resolved). Furthermore furrr has also a convenient tuning option which can interfere with the redistribution scheduling of urls' chunks over workers. The argument scheduling can adjust the average number of chunks per worker. Setting it equal to 2 brings *dinamicity* (2018) to the scheduling so that if at some point a worker is busier then chunks are sent to the more free ones.

(migliore rappresentazione)

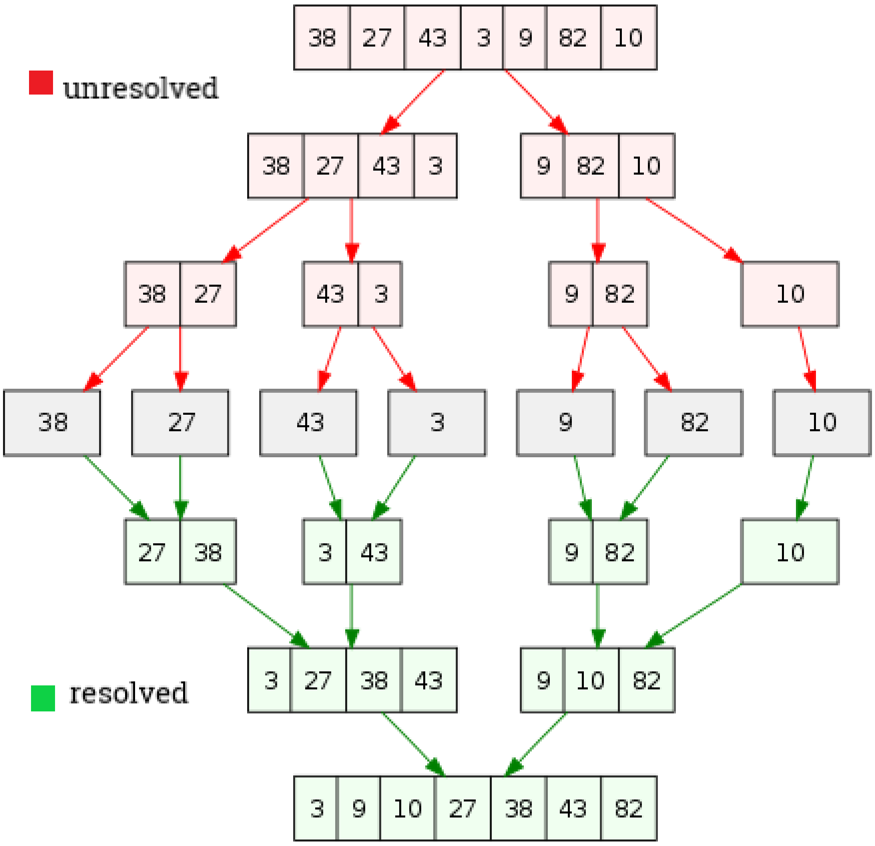


Figure 2.12: futures reimagined as divide and conquer, author's source

The upper plot in figure 2.13 are 20 simulations of 100 url (2500 data points) performed by the lightweight scraping. On the x-axis are plotted the 20 simulations and on the y-axis are represented the respective elapsed times. One major point to breakdown is the first simulation run time measurement which is considerably higher with respect to the others i.e. 15 sec vs mean 7.72 sec. Empirical demonstrations traces this behavior back to the opening time for all the background sessions. As a result the more are the back ground sessions/workers, the more it would be the time occupied to pop up all the sessions. As opposite whence many sessions are opened the mean execution time for each simulation is slightly less. The lower plot in in figure 2.13 tries to capture the run time slope behavior of the scraping function when urls (1 to 80) are cumulated one by one. The first iteration scrapes 1 single url, the second iteration 2, the third 3 etc. Three replications of the experiment has been made, evidenced by three colours. The former urls are more time consuming confirming the hypothesis casted before. Variability within the first 40 urls for the three repetitions does not show diversion. However It slightly increases when the 40 threshold is trespassed. Two outliers in the yellow line are visible in the nearby of 50 and 60. It can be assumed that workers in that urls neighbor may be overloaded but no evidences are witnessed on cores activity as in plot 2.14. The measured computational complexity of scraping when n is number of urls seems to be much more less than linear $\mathcal{O}(0.06n)$.

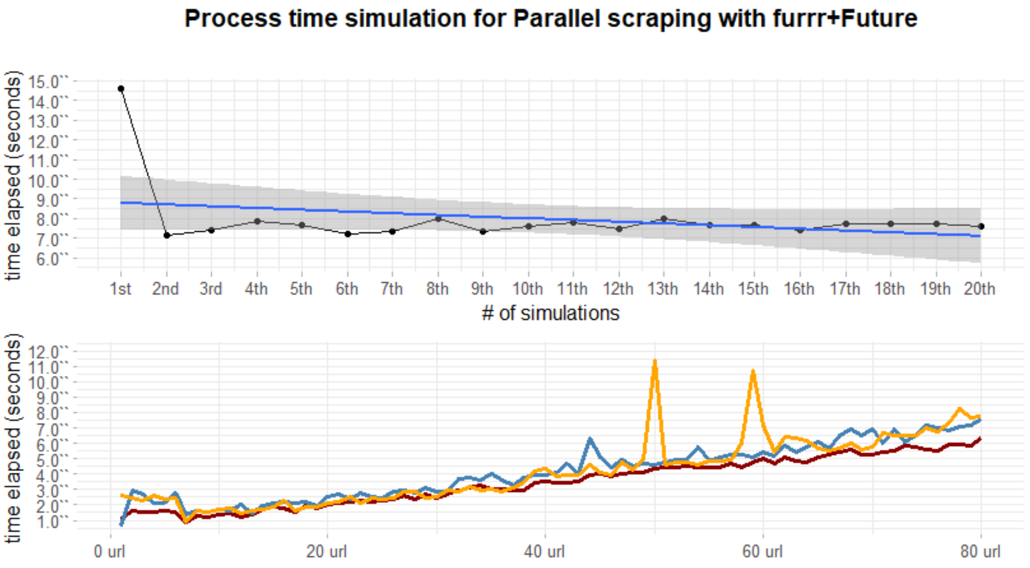
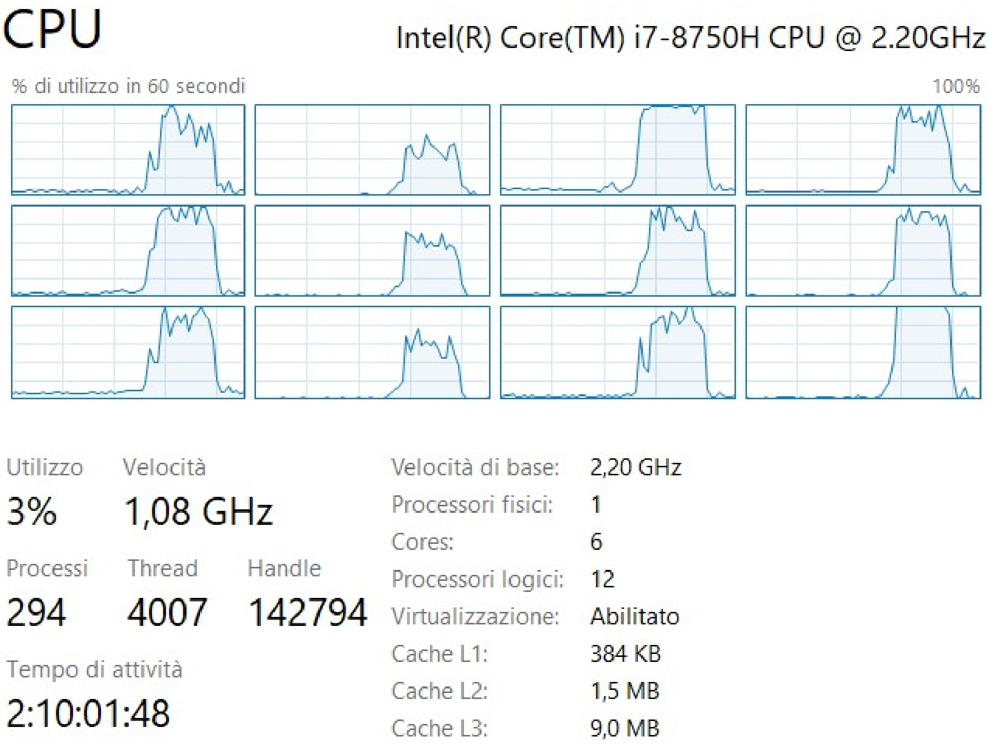


Figure 2.13: computational complexity analysis with Furrr

2.7.2 Parallel foreach+doFuture

A second attempt tries to encapsulate foreach (Microsoft and Weston, 2020) originally developed by Microsoft R, being a very fast loop alternative, parallelized with doFuture. The package registered with older back ends required rigorous effort to specify exact dependencies for child process inside foreach arguments .export. From a certain angle the approach could led to an indirect



CPU

Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz

Utilizzo
3%

Velocità
1,08 GHz

Processi
294

Thread
4007

Handle
142794

Tempo di attività
2:10:01:48

Velocità di base:	2,20 GHz
Processori fisici:	1
Cores:	6
Processori logici:	12
Virtualizzazione:	Abilitato
Cache L1:	384 KB
Cache L2:	1,5 MB
Cache L3:	9,0 MB

Figure 2.14: local machine monitoring of cores during parallel scraping

benefit from memory optimization. If global variables needs to be stated than the developer might be forced to focus on limiting packages exporting. Indeed since doFuture implements optimized auto dependency search this problem may be considered solved as in Bengtsson (2020c). Two major looping related speed improvements may come from .inorder and .multicombine arguments which both take advantage of parallel split disorder a subsequent recombination of results. In the context where data collection order matters this is extremely wrong, but since in this case order is defined through url composition based on criteria expressed inside nodes contents this can be totally applied. A drawback of enabling .multicombine is a worst debugging experience since errors are thrown at the end when results are reunited and no traceback of the error is given.

The upper part in 2.15 displays lower initialization lag from R sessions opening and parallel execution that also lead to a lower mean execution time of 6.42 seconds. No other interesting behavior are detected. THe lower plot displays high similarities with the curves in 2.13 highlighting an outlier in the same proximities of 45/50 urls. The blue simulation repetition shows an uncommon pattern that is not seen in the other plot. Segmented variability from 40 to 80 suggests a higher value which may be addressed do instability. As a result the approach is discarded in favor of furrr + future which also offers both a comfortable {Tidyverse} oriented framework and offers and easy debugging experience.

2.8 Open Challenges and Further Improvements

Although results are quite encouraging still one of the main challenges remains unsolved. In fact optimization has involved each scraping layer but scraping function must be continuously kept up with the immobiliare.it changes, particularly the crawling part. Indeed the proper scraping part, with some further

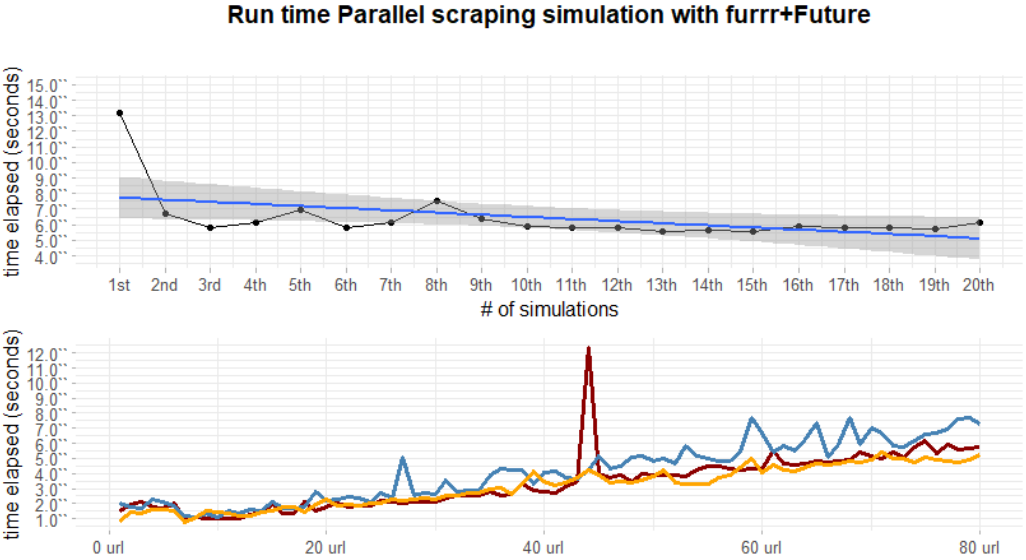


Figure 2.15: computational complexity analysis with Furrr

adjustments can take care of auto-search for exact information within the web page even if the design changes. This idea is massively applied in the package Rcrawler Khalil (2018), which doesn't apply segmentation in crawling, instead it downloads the entire website and then inspects targeted keywords. The major benefit relies in crawling HTML/XML that are agreed to be generally lightweight in this way the process does not weigh down the run time. Whence all files are saved locally the algorithm applies search methodologies within the HTML files. Run time performance with algorithm of this kind with respect to the amount of data gathered are very efficient, nevertheless results are not always effective due to keywords disambiguation. Afterall a way safer and time saving approach to general scraping may be including complex theme specific regular expressions on the HTML text which univocally identify CSS data location. With that said the idea would be an unsupervised algorithm that combines the traditional browser search + a selector gadget for CSS conversion. As a disclaimer Rcrawler is designed to be flexible to scrape a vast number of websites. As opposite the scraping functions here presented are exclusively built on top of immobiliare.it, even though they can be extended with a small effort to other category related website **??**. On the parallel computing side a further boosts might be added with parallel distributed processing through HPC (high-performance computing) clusters by future.batchtools Bengtsson (2020a). The package implements a generic future wrapper around batchtools with job scheduler like Torque, Slurm, Sge and many more. A higher level API built on top of the Future framework that exploits Google Cloud Engine Clusters[6] i.e.cloudyR allows to distribute computation on Google machines.

2.9 Legal Profiles

There is a legitimate *gray* web scrapers area, not only creating possible new uses for the data collected, but also potentially damaging the host website (Media, 2017). Online platforms and the hunger for new data insights are increasingly seeking to defend their information from web scrapers. Online platforms are unfortunately not always successful distinguishing users between polite, impolite and criminals, risking new ones Valid competition and creativity. A minimal but effective self limitation can really be The courts have fought hard to achieve clear judgments Cases for web scraping. The crucial obstacle is a coherent verdicts to ascertain “Kitchen Sink” (MERRIAM-WEBSTER, 2018), the standard web claim discontroversy on scraping. Kitchen Sink main arguments are:

- Legal lawsuits under the Computer Fraud and Abuse Act (CFAA) allegation that the defendant “overtook” allowed access *miss lit*
- Copyright infringement charges under the Digital Millennium Act or federal copyright law Copyright laws *miss lit*
- “state trespass to chattels claims” *miss lit*
- Contract agreements terms violation claims

A second challenge to clear verdicts is that there are several purposes for which a business model operates in a continuum of social acceptability hiring web

[6] https://cloudyr.github.io/googleComputeEngineR/articles/massive-parallel.html

scrapers (Maheedharan, 2016). As an example Googlebot ranks, sorts and indexes search results for Google users, without which the search would not optimized and business would not profit from this fact. A more coherent case is represented by the exploitation of scraping on online Real Estate advertiser whose importance is ascertained by the fact that they are the fist house purchase media 52% (USA survey) and searches are expected to be growing by 22% per yeay (Peterson and Merino, 2003). On the other hand it is estimated that the 20% of crawlers are actually DoSsing scrapers (LaFrance, 2017) causing an economic damage, as in 3.1.3. Unfortunately the majority of web scraping services fall between these two extremes (2017). The most discussed and observed case regards Linkedin Corp. "Linkedin" vs hiQ Labs Inc. ("hiQ") whose claim argues the exploitation of the former personal profile data to offer a series of HR services. The litigation started by accusing hiQ with "using processes to improperly, and without authorization, access and copy data from LinkedIn's website" (Corporation, 2017). As reinforcement to their arguments they presented also a citation directly form their terms raising the point on copying, web scarping prohibition without their explicit consent. Furthermore Linkedin noted that technical barriers were taken into existence to restrict the access of hiQ to the platform and warned of a breach of state and federal law by ignoring such barriers (2017). As a response Hiq submitted a temporary restricting order to prevent LinkedIn from denying the access to their platform, focusing on the aspect that the motivation was led by an anticompetitive intent. Linkedin's response brought into the litigation CFAA which actually pollutes argumentation since Court evidences that CFAA is ambiguous whether it is granting or restringing the access to public available website (Edward G. Black, 2017). In other words CFAA intent was to protect user data when they are authenticated with passwords, and in no case out of these borders. In the end the litigation moved to the Ninth Circuit where in 2019 it uphelds the preliminary injunction to prohibit LinkedIn from continuing to provide access to publicly accessible LinkedIn member profiles to the claimant hiQ Labs (contributors, 2020d). Immobiliare.it expressed similarly to linkedin in their terms (immobiliare.it, 2020) the prohibition to reproduce any form of intellectual property in their web pages. This once again does not imply the inaccessibility to scraping their content and should not in any case prevent the usage of an open sourced tool inspired by research grounds and efficient market choices.

Chapter 3

API Technology Stack

The previous chapter has encapsulated the main concepts behind the design of consistent, secure, and fast scraping functions with R. In truth challenges not only regard scraping per se, but also the way and how many times the service has to interact with different clients. Indeed the fact that functions are compressed into scripts does not imply that are shareable and portable. As a consequence when they are executed they also need to be at first understood and secondly loaded into a dedicated R environment (higher and lower dependecies). Moreover results are actually computed, whether in parallel or not, with local machines resources that are limited in many sense. In the end exposing the service traffic to the public or internally to a network of servers requires authentication and precautions from both malicious attack and privacy dangers. Actually from a restricted personal usage perspective what has been done since now is totally feasible. But in a large-scale orientation where different stakeholders should gather massive amount of data, then a unsuitable service may cause an enormous waste of time. The following chapter tries to capture the essence and its specific context usage of each technology involved that address each aforementioned issues raised. In parallel it highlights the *fil rouge* that ties together the chronological order according to which the stack has been developed.

The recipe proposed serves a REST Plumber API (an R framework) with 2 endpoints each of which calls Parallel scraping functions accomodated in section 2. Precautions regards sanitization of user inputs, anti-Dossing strategies and log monitoring. The software enviroment is containerized in a Docker container and *Composed* with a further container housing NGINX proxy server. Orchestration is managed with docker compose. NGINX with SSL certificates bring HTTPS communication and authentication. An AWS free tier EC2 server hosts the orchestation of containers and the IP is made Elastic. Furthermore the software development is made automatic with a straightforward composition of cloud services that ignites sequential building when local changes are pushed to cloud repository.

Technology stack:

- GitHub version control and CI
- Plumber REST API, section 3.1.1
- Docker containers and compose, section 3.2
- NGINX reverse proxy, section 3.3
- HTTPS and SSL certificates 3.4
- AWS EC2 3.5

immagine infrastruttura da rivedere

As a further note each single part of this thesis is comprised according to the service inspiring criteria of portability and self containerization. RMarkdown (Allaire et al., 2020) documents (book's chapters) are knitted and then rendered as .html files extension through Bookdown (Xie, 2016). The resulting .html documents are instructed by a .yml file and then are rendered as Gitbook format. Gitbooks are interactive online documentation that are employed to build books, technical documentation as well as research papers. That makes thesis' graphs and codes interactive. Moreover Gitbooks enables custom buttons to directly create a push request on documents from source, which may be convenient for continous review. It is able to directly share documents on social media, as well as to change the the documents' style, such as fonts and

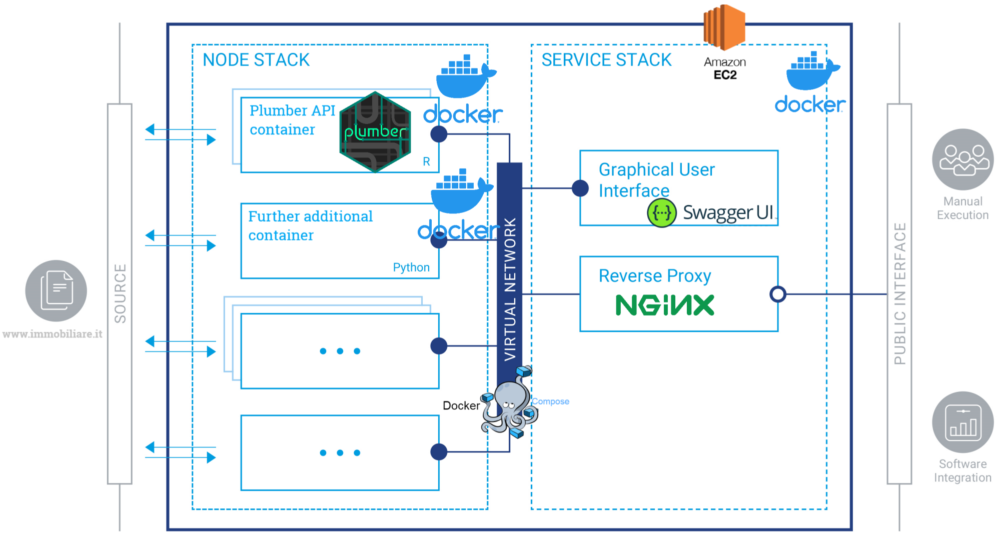


Figure 3.1: complete infrastructure, author's source

background color.
Files are ultimately pushed into a Github repository[1], that enables thesis version controlling. By a simple trick with GH pages a new branch is opened. .html files are displayed into a sub domain repository hosted at link[2], as a consequence thesis is also linked online. Moreover the Gitbook is able to produce also a .pdf version output through a Xelatex engine. Xelatex compiles .Rmd documents according to formatting rules contained in both a .tex template and a further .ylm and produces the final .pdf paper output.

3.1 RESTful API

Definition 3.1 (API)**.** API stands for application programming interface and it is a set of definitions and protocols for building and integrating application software. Most importantly APIs let a product or a service communicate with other products and services without having to know how they're implemented.

API may be considered as a mediator between users or clients sending a re-

[1] https://github.com/NiccoloSalvini/thesis
[2] https://niccolosalvini.github.io/thesis/

quest, left part figure 3.2 and the web resource or services they want (returning back a response) middle part figure 3.2. It also acts as means of exchanging resources and knowledge with an entity, as databases left part figure 3.2 preserving protection, control and authentication, such as who gets access to what. There are many types of APIs that exploit different media and architectures to communicate with apps or services.

Definition 3.2 (REST)**.** The specification REST stands for **RE**presentational **S**tate **T**ransfer and is a set of *architectural principles.*

If a client request is made using a REST API, a representation of the resource state is passed to a requestor or *endpoint* (wha, 2018). This information is returned in one of the formats of multiple formats using *HTTP*: JSON, HTML, XLT, Python, PHP or simple text. JSON is the most common programming language to use because it is language agnostic (2018) and easy to interpretable both for people and machines. REST architecture depends upon HTTP, as a matter of fact REST API inherits from HTTP the stateless property, second pillar in 2.5. Calling a REST API (producing a request) demands composing a well defined URL lower part in figure 3.2, whose semantic is able to uniquely identify the resource or a group of resources (sending back a response) along with the most common HTTP methods, as GET, PUT, POST, DELETE.

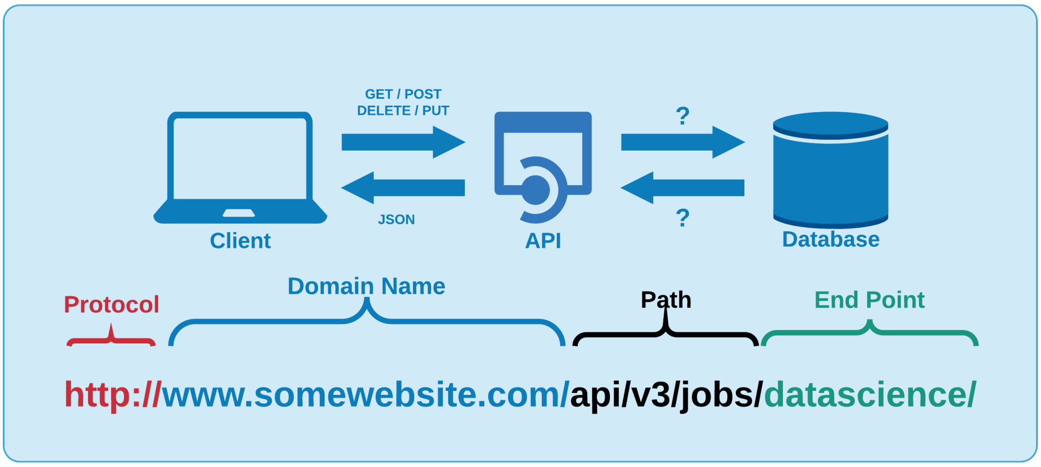


Figure 3.2: API general functioning, unknown source

When an API adheres to REST 3.2 principles is said RESTful. Principles are:

- A client-server architecture made up of clients, servers, and resources, with requests managed through HTTP method.
- Stateless client-server communication, meaning no client information is stored between requests and each request is separate and disconnected.
- Cacheable data that streamlines client-server interactions.
- A uniform interface between components so that information is transferred in a standard form. This requires that:
  - resources requested are identifiable and separate from the representations sent to the client.
  - resources can be manipulated by the client via the representation they receive because the representation contains enough information to do so.
  - self-descriptive messages returned to the client have enough information to describe how the client should process it.
- A layered system that organizes each type of server (those responsible for security, load-balancing, etc.) involved the retrieval of requested information into hierarchies, invisible to the client.

RESTful APIs receives HTTP request inputs and elaborates them through endpoints. Endpoints are the final step in the process of submitting an API request and they can interpreted as the responsibles for generating a response (2018). Further documentation and differences between HTTP and REST API can be found to this reference[3]. Open and popular RESTful API examples are:

- BigQuery API: A data platform for customers to create, manage, share and query data.
- YouTube Data API v3: The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists, and channels.

[3] https://docs.aws.amazon.com/it_it/apigateway/latest/developerguide/http-api-vs-rest.html

- Skyscanner Flight Search API: The Skyscanner API lets you search for flights & get flight prices from Skyscanner's database of prices, as well as get live quotes directly from ticketing agencies.
- Openweathermap API: current weather data for any location on Earth including over 200,000 cities.

3.1.1 Plumber HTTP API

Plumber is a R framework (**?**), allowing users to construct HTTP APIs simply by adding decoration comment to the existing R code, in this context to scraping code. Decorations are a special type of comments that suggests to Plumber where and when the API specifications start. Below is reported a toy API with 3 endpoints inspired by the original documentation. Endpoints in the following code chunk are identifiable by the three distinguishing comment blocks, separated by the aforementioned decorations. http API specifications require the user to set the endpoint description (first comment), to specify the parameters role and input type (second), and the http method i.e. GET, POST followed by the endpoints invokation verb e.g. echo, plot, sum (third).

```
# plumber.R


#* Echo back the input
#* @param msg The message to echo
#* @get /echo
function(msg=””) {
  list(msg = paste0(”The␣message␣is:␣’”, msg, ”’”))
}


#* Plot a histogram
#* @param n the number of samples
#* @serializer png
#* @get /plot
```

```
function(n = 10) {
  rand = rnorm(n = n)
  hist(rand)
}

#* Return the sum of two numbers
#* @param a The first number to add
#* @param b The second number to add
#* @post /sum
function(a, b) {
  as.numeric(a) + as.numeric(b)
}
```

Once HTTP api calls are sent to machines belonging to a single server or a network of servers, whether it is public or private they converge through endpoints. Endpoints execute functions involving the parameters specified through the call and by default response is JSON type. The first endpoint "echo" invocation simply *echoes* back the text that it was sent. The second endpoints generates a histogram *plot* i.e. .png file based on a Normally distributed sample whose observation number are "n". The third endpoint calculates the *sum* of a couple of number attached to the call. Scraping function are then implemented within the API framework and arguments becomes parameters for an incoming request. Many more options may induce plumber endpoints to respond in the preferred fashion, in any case are beyond the scope of the analysis. Exposing APIs on a private network or on a public server security is a major issue. Concerns and thought process therefore must adapt accordingly. There are several variables and consequent attacks that should be considered while creating Plumber APIs, but the focus will differ depending on the API audience. For example if APIs are offered without authentication on the Internet, potential vulnerabilities should seriously convince the api maintainer to properly account each of them. Three in the context of the analysis are critical:

- Sanitization
- Denial Of Service (DoS)
- Logging

3.1.2 Sanitization

Whenever APIs accept input from a random user this is directly injected into functions through endpoints, therefore the worst case scenario should be prepared. In the context of the analysis users are required to specify to the endpoints arguments such as cities, zones, number of pages and many others. Chances are that users might either misspell inputs or use different encoding (accents) or rather use capital letters when functions are capital sensitive. Endpoints should take account of the behavior by sanitizing whatever it comes into the function. The process at first requires an intense and creative investigation on what it can be misused and how. Then Secondly new functional inputs are defined so that they take the user generated input and give back a sanitized version inside the function. In the code chunk below are shown a couple of examples of sanitization of inputs:

```
tipo = tolower(type) %>% str_trim()
citta = tolower(city) %>% iconv(to = ”ASCII//TRANSLIT”)
    %>% str_trim()
macroozone = tolower(macrozone) %>% iconv(to = ”ASCII//
   TRANSLIT”) %>% str_trim()
```

Inputs make their entrance into functions through arguments “type”, “city” and “macrozone” and are immediately preprocessed. They are in sequence converted to lower cases, then extra spaces are trimmed, in the end accents are flattened.

3.1.3 Denial Of Service (DoS)

Denial-of-service attacks (DoS) are used to temporarily shut down a server or service through traffic congestion. A DoS scenario could be triggered accidentally by a malicious user requesting the server for an infinite looping task. Other scenarios might depict a malicious hacker who uses a large number of machines to repeatedly make time consuming requests to occupy the server, this is the case of DDoS (Distributed Denial of Service). DoS or DDoS attacks may also induce anomalies and deprives system resources, which in the context of hosting services may result in astronomical fees charged. Dos attack as a conseuquence may also induce distorted website/API logs analytics, leading to distorted reports. A simple but effective approach tries to limit and stop the number of request sent:

```
if (npages > 300 & npages > 0){
                              msg = ”Don’t␣DoS␣me!”
                              res$status = 500 # code num:
                                  Bad request
                              stop(list(error=jsonlite::unbox
                                  (msg)))
                  }
```

The code chunk above intercepts DoS attacks by limiting to 300 the number of pages to be server to the API. Furthermore it converts outputs error messages printed on console into JSON format and then pass them as output. This simplify distinguishing malicious attacks from a type errors. DDoS attacks are secured by SSL certificates and Authentication covered later in the chapter.

3.1.4 Logging

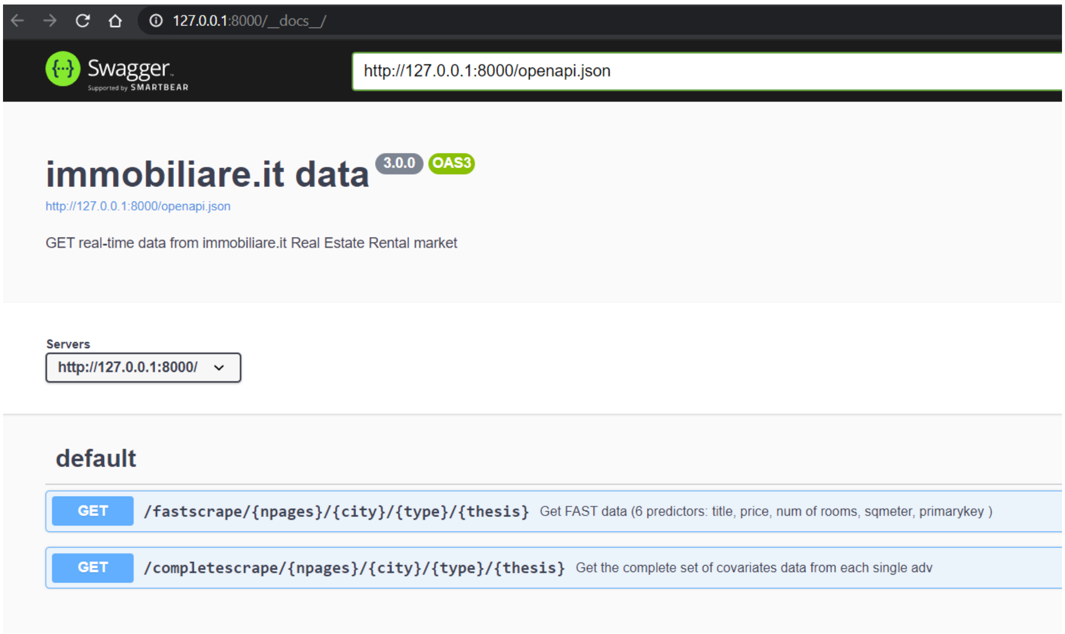
Plumber uses “filters” that can be resorted to describe a “pipeline” for processing incoming request. This enables API maintainers to separate complex logic into discrete, comprehensible steps. Usually, before trying to find an endpoint that satisfies a request, Plumber passes the request through the *filters*. When APIs are called, requests pass through filters one at a time and Plumber forwards i.e. forward() the request to the next filter untill the endpoints. Filters applications ranges from excluding client request based on request parameters or may offer also a thin layer of authentication. Filters might also be used as a logging for requests where logging, i.e. the act of keeping a log (**?**), is recording events in an operating system or running software from other users of communication software, or messages among different subjects. A request log filter might have this appearance:

```
#* Log information
#* @filter logging
function(req){
            cat(as.character(Sys.time()), "-",
                req$HTTP_USER_AGENT, "@",
                req$REMOTE_ADDR, "\n",
                req$QUERY_STRING, "\n")
            plumber::forward()
}
```

The above filter parses the request through the default request argument req, then it prints out messages about the incoming User Agent (i.e. HTTP_USER_AGENT) (section 2.5.1), the REMOTE_ADDR which is the IP address of the client making the request (2018) and the QUERY_STRING that records the parameters directly sent the endpoint. This helps to traceback clients activity on the API as well as detecting misuse.

3.1.5 RESTful API documentation

The service disposes of 2 endpoints */fastscrape* , */completescrape*. Parameters are the same for both of the endpoints since they rely on the same reverse engineering url algorithm 2.4, in section 2.1. Moreover Plumber APIs are natively wrapped up around Swagger UI helping development team or end users to imagine and communicate with the resources of the API without any discharge logic (SMARTBEAR, 2019). The OpenAPI (formerly referred to as Swagger), with the visual documentation facilitates backend implementation and client side consumption, as well as being automatically created by the APIs specification (parameters, endpoints…). Some of the major assets in Swagger UI are: The user interface works in any environment, whether locally or web and it is suited for all main browsers.



http://127.0.0.1:8000/openapi.json

immobiliare.it data

http://127.0.0.1:8000/openapi.json

GET real-time data from immobiliare.it Real Estate Rental market

http://127.0.0.1:8000/

default

GET /fastscrape/{npages}/{city}/{type}/{thesis} Get FAST data (6 predictors: title, price, num of rooms, sqmeter, primarykey)

GET /completescrape/{npages}/{city}/{type}/{thesis} Get the complete set of covariates data from each single adv

Figure 3.3: Swagger UI in localhost on port 8000, author's source

3.2 Docker

The API up to this point needs a dedicated lightweight software environment that minimizes dependencies both improving performances and enabling *cloud computing* coverage. A fast growing technology known as *Docker* might extend these capabilities.

Definition 3.3 (Docker and Containers)**.** *Docker* (Merkel, 2014) is a software technology to create and deploy applications using containers. *Docker containers* are a standard unit of software (i.e. software boxes) where everything needed for applications, such as libraries or dependencies can be run reliably and quickly. Containers are also portable, in the sense that they can be taken from one computing environment to the following without further adaptations.

Containers can be thought as a software abstraction that groups code and dependencies together. One critical advantage of containers is that multiple containers can run on the same machine with the same OS along with their specific dependencies (docker compose). Each container can run its own isolated process in the user space, so that each task/application is exhaustively and complementary to the other. The fact that containers are treated singularly enables a collaborative framework that it also simplifies bugs isolation.

Actually *Docker containers* are the build stage of *Docker Images.* Docker images therefore are the starting point to containerize a software environment. They are built up from a series of software layers each of which represents an instruction in the image's *Dockerfile* (2020a) . In addition images can be open sourced and reused through Docker Hub. *Docker Hub* is a web service provided by Docker for searching and sharing container images with other teams or developers in the community. Docker Hub can authorize third party applications as GitHub entailing an collaborative image version control, this would be critical for software development as disguised in section (3.6).

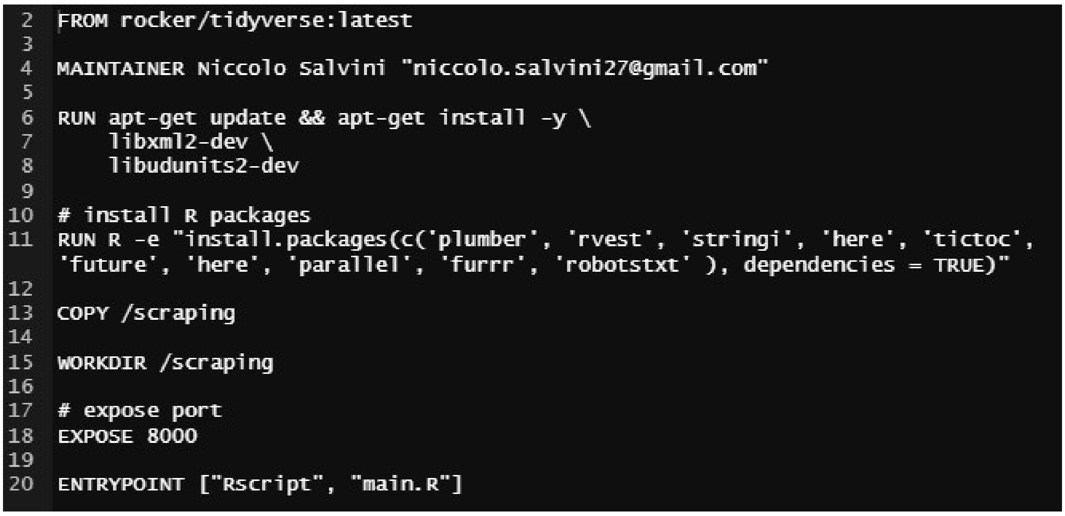
3.2.1 REST-API container

Docker can build containers from images by reading instructions from a Dockerfile. A Dockerfile is a text document that contains the commands/rules a generic user could call on the CLI to assemble an image. Executing the command docker build from working directory the user can trigger the build. Building consists of executing sequentially several command-line instructions that specifies the software environment. As a matter of fact the concept of containers takes inspiration by the fact that many single software layers are stacked up over at the following. An open source project named rocker[4] al-

[4] https://www.rocker-project.org/images/

ready disposes of a group of pre-set task-specific image configurations from which further custom dockerfile can be built on top of. Therefore images are overwritten with higher level dependencies i.e. package libraries, since lower levels Linux dependencies are already partially handled. Indeed as in 3.6 an automatic development workflow is proposed that triggers the building of the image when changes are pushed to github.

The custom Dockerfile in figure 3.4) is able to build the rest-api container:



```
FROM rocker/tidyverse:latest

MAINTAINER Niccolo Salvini "niccolo.salvini27@gmail.com"

RUN apt-get update && apt-get install -y \
    libxml2-dev \
    libudunits2-dev

# install R packages
RUN R -e "install.packages(c('plumber', 'rvest', 'stringi', 'here', 'tictoc',
'future', 'here', 'parallel', 'furrr', 'robotstxt' ), dependencies = TRUE)"

COPY /scraping

WORKDIR /scraping

# expose port
EXPOSE 8000

ENTRYPOINT ["Rscript", "main.R"]
```

Figure 3.4: Custom Dockerfile from salvini/api-immobiliare Docker Hub repository, author's source

Each line from the Dockerfile has its own specific role:

- FROM rocker/tidyverse:latest : The command imports from rocker project a pre set config guration containing the latest version of base-R along with the tidyverse (Wickham et al., 2019) R packages collection.
- MAINTAINER Niccolo Salvini "niccolo.salvini27@gmail.com" : The command tags the maintainer and its e-mail contact information.
- RUN apt−get update && apt−get install −y \ libxml2−dev \ libudunits2−dev : The command update and install remaining "lower" Linux dependencies required to run Plumber and rvest.

- RUN R –e ”install.packages(c(’tictoc ’,’ here ’,’...’, dependencies= TRUE) : The command install all the further lower level libraries required to execute the files. Since some packages have inner dependencies the option dependencies=TRUE is necessary.
- COPY /scraping : The command tells Docker to copy into the container files in /scraping folder (where API specs are)
- WORKDIR /scraping : The command tells Docker to set /scraping as working directory
- EXPOSE 8000 : The commands instructs Docker engine that the container listens on the specified network ports 8000 at runtime. Default *transportation* layer is TCP.
- ENTRYPOINT [”Rscript”, ”main.R”] : the command tells docker engine to execute the file main.R where are contained the Plumber router options (i.e. host and port specifications).

3.3 NGINX reverse Proxy Server and Authorization

Proxy server in this context offers the opposite angle for the exact same security problem. As a matter of fact the downside is that they can be exploited for the same reason for which they have been criticized at the end of section (2.5.1). Reverse Proxy server are a special type of gateway 2.3 that is usually located behind a private network firewall to route client requests to the corresponding backend server (NGINX, 2014). An reverse proxy offers an extra abstraction and control level to ensure that network traffic flows smoothly between clients and servers. NGINX as it can be inferred by its official documentation empowers traffic flows by:

- *Load balancing*: A reverse proxy server will stand guard inf front of back end servers and redirect requests across a group of servers in a way that maximizes speed and capacity usage as well as not overloading the server. When a server crashes, the load balancer redirects traffic to the other online servers (note required since traffic is not expected to be enormous).
- *Web acceleration*: A reverse proxies can condense input and output data by caching frequently requested information. This assists traffic between clients and servers avoiding to request data more than once. They can also apply SSL encryption, which improves their performance by eliminating loads form web servers.
- *Security and anonymity*: A reverse proxy server protects identities and serves as an additional protection against security threats seen in subsections (3.1.2, **??**) by intercepting requests before they reach end server.

When a user calls the API, NGINX acts as a gateway asking for credentials and registering log data (HTTP identification headers). If the request has already been asked then cached response is returned. If it does not then the request is routed to the endpoint, service A and B in figure 3.5. Endpoints elaborate the request into the response, which flows back at first to the gateway and then finally to the client. Logging data can be feeded to a dashboard monitoring traffic and API exposure.

Then without any further software installation, the developer can simply make use of the latest NGINX open sourced image to build the NGINX proxy server within the same image. A further configuration file should manage NGINX inner settings that links volumes, ports and IPs, but details are beyond the scope of the analysis.

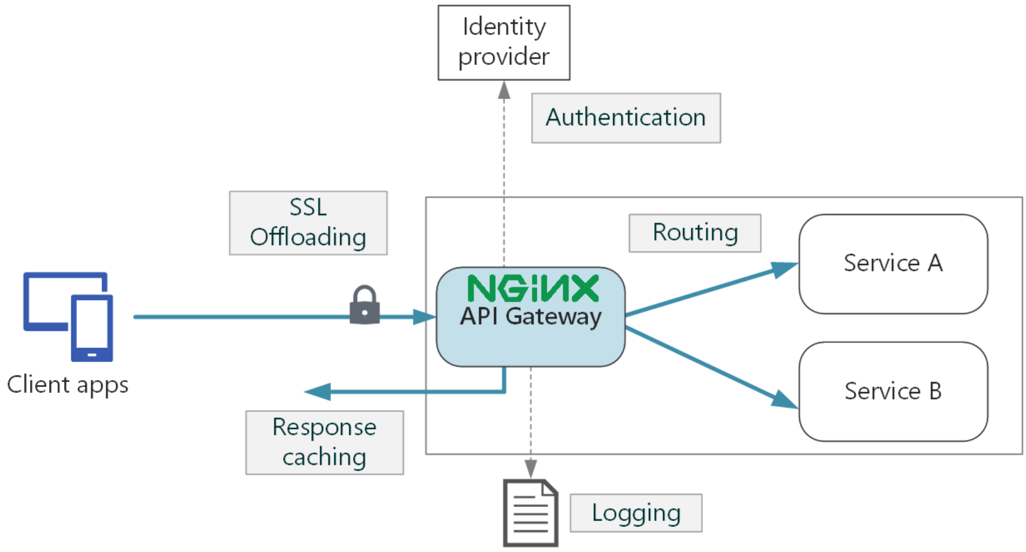


Figure 3.5: NGINX gateway redirecting an incoming request, source Microsoft (2018)

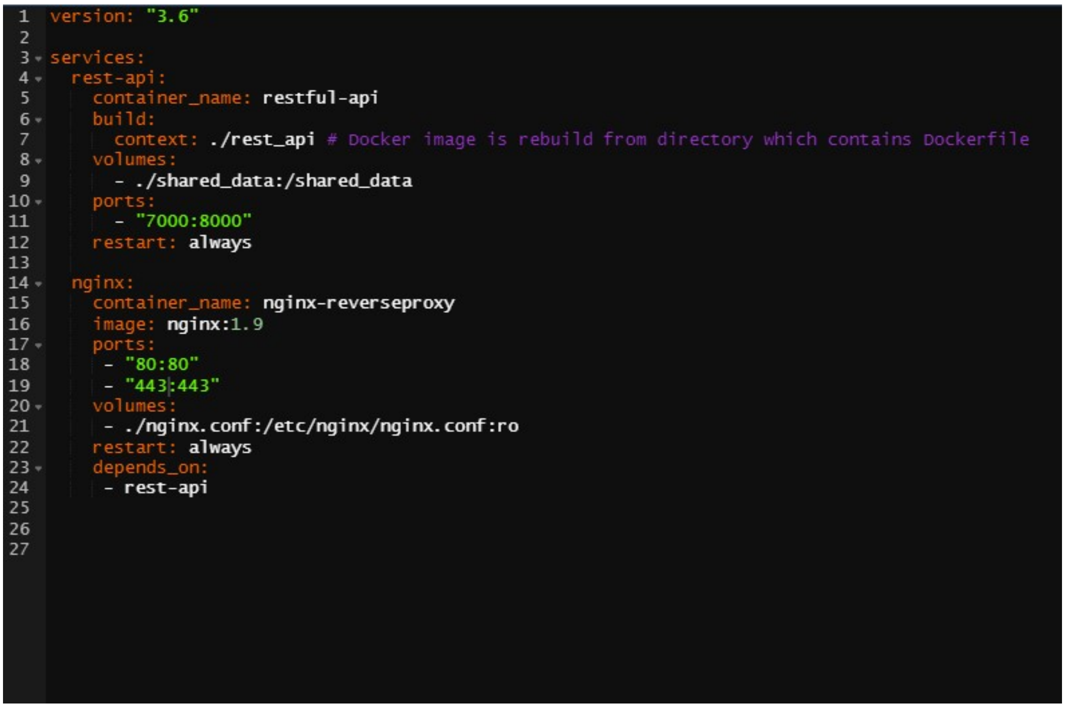
3.3.1 Docker-Compose

Compose is a multi-container Docker framework to define and run complex application and services, that means isolated stand alone containers can communicate one to each other. The orchestration of containers is managed through a .yaml file docker-compose.yml that configures the whole set of facilities designed for the services. Services are then build with a single docker command docker−compose up on the basis of the instruction yaml file. Compose also enables *Volumes*, which are used in Docker for **data persistence** (Inc., 2020b). VOlumes allows to keep data secured from docker stop or delete containers. Docker Volumes ina nutshell works as the linkage between a physical file system path (folders, directories) plugged/**mounted** into the virtual container file system path. The main properties for Composition as in documentation (Inc., 2020a) regards:

- *A central host in multiple isolated environments*: Compose provides a label for a project to distinguish environments. The default project name is the directory path.
- *Preserve data volumes (which are the preferred Docker mechanism to con-*

sume data generated and used by containers) when building containers: Compose maintains all of the services' Volumes. When Docker-compose is running, it transfers the Volumes from the old container into the new container when it detects containers from previous runs. This method guarantees that no Volume data produced is lost.

- *Only recreate containers that have changed*: Compose caches the configuration and re-use the same containers when it reboot a service which hasn't changed. Re-use containers means it provides really fast improvements to the environment when developing complex services.
- *Variables and moving a composition between environments*: Compose supports file variables which might be used to adjust any composition to various environments or users.



```yaml
version: "3.6"

services:
  rest-api:
    container_name: restful-api
    build:
      context: ./rest_api # Docker image is rebuild from directory which contains Dockerfile
    volumes:
      - ./shared_data:/shared_data
    ports:
      - "7000:8000"
    restart: always

  nginx:
    container_name: nginx-reverseproxy
    image: nginx:1.9
    ports:
     - "80:80"
     - "443:443"
    volumes:
     - ./nginx.conf:/etc/nginx/nginx.conf:ro
    restart: always
    depends_on:
     - rest-api
```

Figure 3.6: Docker Compose YAML file orchestration for NGINX container and RESTful API, author's source

The first line defines the versions according to which Compose orchestrates containers. Services are two, the rest-api and NGINX. The former whose name is restful-api builds the container starting from the Dockerfile contained into the rest_api path specified as in 3.2.1. The first service also mounts Volumes from shared_data to itself in the virtual container file path. Port 8000 opened (3.2.1) is linked to the 7000 one. Restarting set to “always” secures the latest container version. The latter service is NGINX 3.3 which proxies traffic from 80 and 443 based on .conf file whose specifications are beyond the scope of the analysis. Volumes makes sure that specifications are arranged in the default path choice. The option depends_on defines the chrono-logical condition accroding to which containers should be run and then composed.

3.4 HTTPS(ecure) and SSL certificates

Communication even though properly secured and distributed with NGINX is still not encrypted so sensitive data are still being exposed. HTTP Secured (HTTPS) is a product of HTTP combined with SSL/TLS (Secure Sockets Layer/Transportation Security Layer) protocol rather than a protocol itself. HTTPS is strictly appropriate when transmitting sensitive data such as banking or on-line buying. Its importance is highlighted by the fact that nowadays is rather more common to find HTTPS than HTTP. The Secured method encrypts all contacts between the client and the server, figure 3.7. The HTTPS scheme in actual is “HTTPS” and its default port is 443 (that should be exposed too in the dockerfile). SSL runs as a sublayer of the *application* layer (a further internet layer, refer to section 2.5), this ensures that HTTP messages are encrypted prior being transmitted to the server (SSL Offloading box in figure 3.5) whether it is a proxy server or directly a web server.

From a port communication point of view at first NGINX container listens to on port 80 and that it is where all requests should relay to IP_macchine_address:8000. As a result the RESTful API is published on Port 80 instead of 8000. Secondly it listens on the SSL port 443 and points to the key and certificate files that have been created in the Dockerfile.

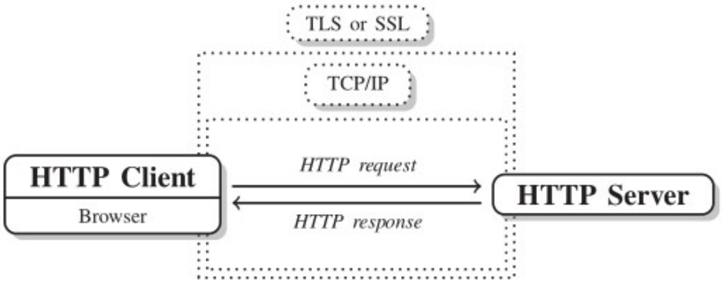


Figure 3.7: HTTPS encryption with SSL certificates, source aut (2014)

3.5 AWS EC2 instance

From henceforth the software container environment can be reproduced regardless of the mounted OS. Scalability and accessibility is worked out by exporting orchestrated containers on a web server. There are many hosting services options that varies primary on the budget and consequently on the API audience. A flexible cloud provider combined with NGINX load balancing *ceteris paribus* may offer a stable and reliable service for a reasonable price, even considering a bad-case scenario where requests are many and insistent.

Definition 3.4 (AWS EC2)**.** Amazon Elastic Compute Cloud (EC2) is a web service that contributes to a secure, flexible computing capacity in the AWS cloud. EC2 allows to rent as many virtual servers as needed with customized capacity, security and storage.

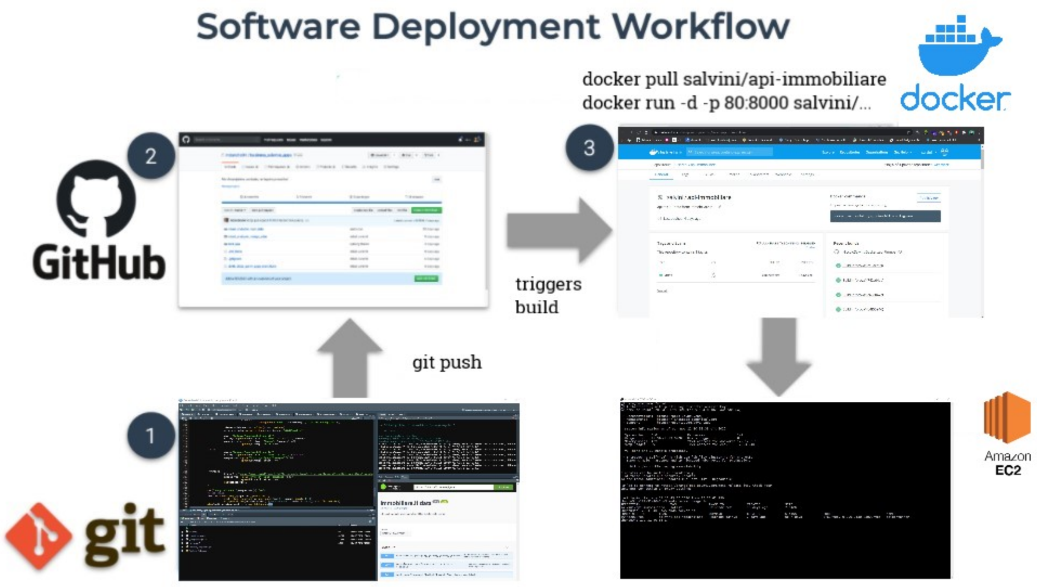
AWS EC2 represents a popular hosting option, most importantly this path is already narrowed by many open source contributors. The selected target is a AWS free tier t3.micro whose specifications are: 2 vCPU (Virtual CPU), 1 GB memory and a mounted Ubuntu distribution OS. Moreover T3 instances are known to provide device, memory, and network resource balance and have been developed for applications that experience transient spikes in use with modest CPU use. They are designed especially to serve low-latency interactive applications, small and medium databases, virtual desktops, development environments, code repositories, and business-critical applications, therefore they suit the needs. Prior any new instance initialization AWS offers to tune servers' set up options. Networking and VPC are chosen to be left as is since they can always be updated at need. Storage is increased to 30 GB which represents the free tier eligibility upper limit. Tags are not required. Indeed security at first needs to account for a SSH connection that allows communication with the server i.e. open port 22. Secondly it should account for port openings accordingly to NGINX configuration file and rest-api dockerfile (3.2.1), port 80 (default for TCP) and 443 (HTTPS default) are opened. Once the instance is running the server can be accessed through SSH (Putty or Bash depending on the OS) behind authentication.

3.6 Software development Workflow

Software changes can happen quite often due to the responsive nature of the RESTful API task. This requires a modern stack of cloud technologies to keep track, revert and quickly update software versions, even adapt software architecture. As a result minor changes are made locally to the project and then are directly pushed with GIT into a GitHub repository, upper left quadrant 3.8. The above mentioned repository communicates with a DockerHub repository that triggers the rebuilt of docker images whenever changes are pushed, upper right quadrant 3.8. Images are built along with tags so that versions are controlled and if that may be the the case of error, reverted. Debugging passes through logs auto generated by the docker engine. The build stage in R since 2019 required long time due to package compiling, but since the appearance of Rstudio package manager[5] which includes beta support for pre-compiled R packages they can be installed 3 times faster (Nolis, 2020). When newer images are available they can be pulled from the EC2 server and rerun in detached mode. Massive software changes are managed through GitHub branches, even though it must be kept in mind to swith lautomatic building brench. The Ec2 server is associated to an Elastic IPs address allowing to reuse the address

[5] https://packagemanager.rstudio.com/client/#/

for external databases connections and DNS services as Cloudflare (for SSL certificates). Moreover elastic IPs are effective when the EC2 server stands in need of upgrading or downgrading or when the server may fail, thus restoring and apply the IP address for a new server.



```
docker pull salvini/api-immobiliare
docker run -d -p 80:8000 salvini/...
```

Figure 3.8: Sofwtare development workflow, author's source

3.7 Further Integrations

From a software point of view a more robust code can be obtained embedding recent R software development frameworks as Golem Colin Fay (2020) into the existing code. The framework stimulates the usage of modules According to the latest literature APIs (as well as Shiny) should be treated as R packages (Santiago (2020) aligns with the idea) as argued in section 4.2 Colin Fay (2020) and in the comment[6] by Dean Attali. As a consequence of that *TDD* (i.e.Test-Driven Development TDD (2004)) through the tools of Wickham and Bryan (2020) and Wickham (2011) during package building can make software more robust, organized and production graded. Loadtest **?** can help figure out

[6] https://deanattali.com/2015/04/21/r-package-shiny-app/

CI/CT It can be missed to cite a popular API development integration service and automate testing tool Postman[7], which does its best when POST requests endpoints are served, since for the moment they are not required it is not used. Pins is an r packages this link[8] software development framework and tools for testing this work[9]

[7] https://www.postman.com/

[8] https://rstudio.com/resources/rstudioconf-2020/deploying-end-to-end-data-science-with-shiny-plumber-and-pins/?mkt_tok=eyJpIjoiTmprNU1USXhPVEprWXpNMSIsInQiOiJtTUhKVzlvSjVIV2hKc0NRNVU1NTRQYSsrRGd5MWMy

[9] https://github.com/isteves/plumbplumb

Chapter 4

INLA computation

INLA (Rue et al., 2009) stands for Integrated Nested Laplace approximation and constitutes a computational alternative to traditional MCMC methods. INLA does approximate Bayesian inference on special type of models called LGM (Latent Gaussian Models) due to the fact that they are *computationally* convenient. Benefits are many, some of them are capitalized by Rue in (de Rencontres Mathématiques, 2018):

- Low computational costs, even for large models.
- It provides high accuracy.
- Can define very complex models within that framework.
- Most important statistical models are LGM.
- Very good support for spatial models.
- Implementation of spatio-temporal model enabled.

INLA in few words uses a combination of analytics approximations and numerical integration to obtain an approximated posterior distribution of the parameters in a shorter time period. The chronological steps in the explanation follows the route sailed by Moraga in Moraga (2019), with the author choice to skip details. As a matter of fact the aim of the chapter is to provide a comprehensive intuition oriented to the immediate application of the methodology, without stepping too long on mathematical details. By the way details e.g model assessment and control options are handled under the hood by the package and can be tuned within the main function, most of them are covered by Gómez Rubio (2020). Notation is imported from Marta Blangiardo (2015), and quite differ from the one presented in the original paper by Rue, Chopin and Martino (2009). As notation remarks, bold symbols are considered as vectors, so each time they occur they have to be considered like the *ensamble* of their values. In addition $\tilde{\pi}$ in section 4.2 are the Laplace approximation of the underlying integrals. Moreover the inner functioning of Laplace approximation and its special usage within the INLA setting is far from the scope, but an easy shortcut oriented to INLA is in Marta Blangiardo (2015).

INLA can fit only Latent Gaussian type of models and the following work tries to encapsulate its properties. As a consequence a problem can be reshaped into the LGM framework with the explicit purpose to explore its benefits. When models are reduced to LGMs then joint posterior distribution can be rewritten and then approximated with INLA. A hierarchical bayesian structure on the model will help to integrate many parameter and hyperparameter levels and simplify interpretation. Generic Information on the project and the R-INLA package are contained in the initial part to last section 4. In the end a brief application on a toy spatial dataset is proposed with the aim to familiarize with the methodology and to come to grip with INLA results.

4.1 Latent Gaussian Models LGM

Given some observations $y_{i\ldots n}$ in order to define a Latent Gaussain Model within the bayesian framework it is convenient to specify at first an *exponential family* (Gaussian, Poisson, Exponential...) distribution function characterized by some parameters ϕ_i (usually expressed by the mean $E\left(y_i\right))$) and some other hyper-parameters $\psi_k, \forall k \in \ 1 \ldots K$. The parameter ϕ_i can be defined as an additive *latent linear predictor* η_i, as pointed out by Krainski and Rubio ((2019)) through a link function $g(\cdot)$, i.e. $g\left(\phi_i\right) = \eta_i$. A comprehensive expression of the linear predictor takes into account all the possible effects on covariates:

$$\eta_i = \beta_0 + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f_l \left(z_{li}\right)$$

where β_0 is the intercept, $\beta = \{\beta_1, \ldots, \beta_M\}$ are the coefficient that quantifies the linear effects on covariates $x = (x_1, \ldots, x_M)$ and $f_l(\cdot), \forall l \in 1 \ldots L$ are a set of random effects defined in terms of a z set of covariates $z = (z_1, \ldots, z_L)$ (e.g. rw, ar1). As a consequence of the last assumption the class of LGM can receive a wide range of models e.g. GLM, GAM, GLMM, linear models and spatio-temporal models. This constitutes one of the main advantages of INLA, which can fit many different models, starting from simpler and ending with more complex. Contributors recently are extending the methodology to many areas as well as models moreover they are trying to incorporate INLA with non gaussian latent models as Rubio (2020) pointed out. All the latent components can be conveniently grouped into a variable denoted with θ such that: $\theta = \{\beta_0, \beta, f\}$ and the same can de done for hyper parameters $\psi = \{\psi_1, \ldots, \psi_K\}$. Then the probability distribution conditioned to parameters and hyper parameters is then:

$$y_i \mid \theta, \psi \sim \pi \left(y_i \mid \theta, \psi\right)$$

Since data $(y_1, \ldots, y_n)$ is drawn by the same distribution family but it is conditioned to parameters which are conditional independent (i.e. $\pi\left(\theta_i, \theta_j \mid \theta_{-i,j}\right) = \pi\left(\theta_i \mid \theta_{-i,j}\right) \pi\left(\theta_j \mid \theta_{-i,j}\right)$) (Rue and Held, 2005) then the joint distribution is given by the product of all the independent parameters i.e. the likelihood. Moreover the Product operator index i ranges from 1 to n, i.e. $\mathbf{I} = \{1 \ldots n\}$. When an observation is missing so the corresponding $i \notin \mathbf{I}$ INLA automatically will not include it in the model avoiding errors (2020). As a consequence the likelihood expression is:

$$\pi(y \mid \theta, \psi)=\prod_{i \in \mathbb{I}} \pi\left(y_i \mid \theta_i, \psi\right) \tag{4.1}$$

Each data point is connected to one combination θ_i out of all the possible linear combinations of elements in θ *latent field*. The latent aspect of the field regards the undergoing existence of many parameter combination alternatives. Furthermore hyper parameters are by definition independent, in other words ψ will be the product of many univariate priors (Gómez Rubio, 2020). A Multivariate Normal distribution is imposed on the latent field θ such that it is centered in 0 with precision matrix $Q(\psi)$ (the inverse of the covariance matrix $Q^{-1}(\psi)$) depending only on ψ hyper parameter vector i.e., $\theta \sim \operatorname{Normal}\left(\mathbf{0}, Q^{-1}(\psi)\right)$. As a notation remark some authors choose to keep the covariance matrix expression as Q and its inverse precision matrix as Q^{-1}. This is strongly not encouraged fro two reasons: the first is that the default hyperpramater option in INLA R package uses the precision matrix, the second it over complicates notation when writing down conditional expectation as Rue pointed out *miss lit.* However notation for covariance function introduced in chapter 5.2.1 i.e. Matérn has to be expressed through covariance matrix, this passage will be cleared out in the dedicated section so that confusion is avoided. The exponential family density function is then expressed through:

$$\pi(\theta \mid \psi)=(2 \pi)^{-n / 2}|Q(\psi)|^{1 / 2} \exp \left(-\frac{1}{2} \theta Q(\psi) \theta\right) \tag{4.2}$$

The conditional independence assumption on the latent field θ leads $Q(\psi)$ to be a sparse precision matrix since for a general pair of combinations θ_i and θ_j the resulting element in the precision matrix is 0 i.e. $\theta_i \perp \theta_j \mid \theta_{-i, j} \Longleftrightarrow Q_{ij}(\psi)=0$ (2015). A probability distribution function with those characteristics is said *Gaussian Markov random field* (**GMRF**). GMRF as a matter of fact are Gaussian variables with Markov properties which are encoded in the precision matrix Q (Rue et al., 2009). (puoi dire di più) From here it comes the source of run time computation saving, inherited using GMRF for inference. As a consequence of GMRF representation of the latent field, matrices are sparse so numerical methods can be exploited (Marta Blangiardo, 2015). *Moreover when Gaussian Process (see chapter 5.1), which are used to integrate spatial components, are represented as GMRF through SPDE (Stochastic Partial Differential Equations) approach, then INLA can be used as a computing choice. This last assumption will be framed in chapter 4 and verified in chapter 6.* Once priors distributions are specified for ψ then the joint posterior distribution for θ and ψ is

$$\pi(\theta, \psi \mid y) \propto \underbrace{\pi(\psi)}_{\text{prior}} \times \underbrace{\pi(\theta \mid \psi)}_{\text{GMRF}} \times \underbrace{\prod_{i=1}^{n} \pi\left(y_i \mid \theta_i, \psi\right)}_{\text{likelihood}}$$

Last expression is said a Latent Gaussian Models, **LGM**, if the whole set of assumptions imposed since now are met. Therefore all models that can be reduced to a LGM representation are able to host INLA methodology. Then plugging in the *likelihood* (4.1) and *GMRF* (4.2) expression the posterior distribution can be rewritten as

$$\begin{aligned}
\pi(\theta, \psi \mid y) &\propto \pi(\psi) \times \pi(\theta \mid \psi) \times \pi(y \mid \theta, \psi) \\
&\propto \pi(\psi) \times \pi(\theta \mid \psi) \times \prod_{i=1}^{n} \pi\left(y_i \mid \theta_i, \psi\right) \\
&\propto \pi(\psi) \times |Q(\psi)|^{1/2} \exp\left(-\frac{1}{2}\theta' Q(\psi)\theta\right) \times \prod_{i}^{n} \exp\left(\log\left(\pi\left(y_i \mid \theta_i, \psi\right)\right)\right)
\end{aligned}$$

And by joining exponents by their multiplicative property it is obtained

$$\pi(\theta, \psi \mid y) \propto \pi(\psi) \times |Q(\psi)|^{1/2} \exp\left(-\frac{1}{2}\theta' Q(\psi)\theta + \sum^{n} \log\left(\pi\left(y_i \mid \theta_i, \psi\right)\right)\right) \tag{4.3}$$

4.2 Approximation in INLA setting

INLA is not going to try to estimate the whole posterior distribution from expression (4.3). Instead it will try to estimate the posterior marginal distribution effects for each θ_i combination in the latent parameter θ, given the hyper parameter priors specification ψ_k. Proper estimation methods however are beyond the scope of the analysis, further excellent references are suggested in their respective part by Rubio (2020) in section 2.2.2 and Blangiardo & Cameletti (2015) in section 4.7.2. The marginal posterior distribution function for each latent parameter element θ_i is

$$\pi(\theta_i \mid y) = \int \pi(\theta, \psi \mid \mathbf{y}) \pi(\psi \mid \mathbf{y}) d\psi \tag{4.4}$$

The posterior marginal integral for each hyper parameter \$ _{k}, k 1, ..., K \$ is

$$\pi\left(\psi_k \mid y\right) = \int \pi(\psi \mid y) d\psi_{-k}$$

where the notation ψ_{-k} is a vector of hyper parameters ψ without considering kth element ψ_k.

The goal is to have approximated solution for latent parameter posterior distributions. To this purpose A *hierarchical procedure* is now imposed since the "lower" hyper parameter integral, whose approximation for the moment does not exist, is nested inside the "upper" parameter integral that takes hyper param as integrand. Hierarchical structures are welcomed very warmly since they are convenient later in order to fit a hierarchical bayesian model approached in the next chapter 5.5. While many approximation strategies are provided and many others are emerging for both the hyper param and for the latent field, the common ground remains to unnest the structure in two steps such that:

- step 1: compute the Laplace approximation $\tilde{\pi}\left(\psi \mid y\right)$ for each hyper parameters marginal: $\tilde{\pi}\left(\psi_k \mid y\right)$

- step 2: compute Laplace approximation $\tilde{\pi}\left(\theta_i \mid \psi, y\right)$ marginals for the parameters given the hyper parameter approximation in step 1: $\tilde{\pi}\left(\theta_i \mid y\right) \approx \int \tilde{\pi}\left(\theta_i \mid \psi, y\right) \quad \underbrace{\tilde{\pi}(\psi \mid y)}_{\text{Estim. in step 1}} \quad \mathrm{d}\psi$

Then plugging approximation in the integral observed in (4.4) it is obtained:

$$\tilde{\pi}\left(\theta_i \mid y\right) \approx \int \tilde{\pi}\left(\theta_i \mid \psi, y\right) \tilde{\pi}(\psi \mid y) \mathrm{d}\psi$$

In the end INLA by its default approximation strategy thrpugh *simplified Laplace approximation* uses the following numerical approximation to compute marginals:

$$\tilde{\pi}\left(\theta_i \mid y\right) \approx \sum_j \tilde{\pi}\left(\theta_i \mid \psi^{(j)}, y\right) \tilde{\pi}\left(\psi^{(j)} \mid y\right) \Delta_j$$

where $\{\psi^{(j)}\}$ are a set of values of the hyper param ψ grid used for numerical integration, each of which associated to a specific weight Δ_j. The more the weight Δ_j is heavy the more the integration point is relevant. Details on how INLA finds those points is beyond the scope, but the strategy and grids seraches are offered in the appendix follwing both Rubio and Blangiardo.

4.2.1 further approximations (prolly do not note include)

INLA focus on this specific integration points by setting up a regular grid about the posterior mode of ψ with CCD (central composite design) centered in the mode (Gómez Rubio, 2020).

The approximation $\tilde{\pi}\left(\theta_i \mid y\right)$ can take different forms and be computed in different ways. Rue et al. (2009) also discuss how this approximation should be in order to reduce the numerical error (Krainski, 2019).

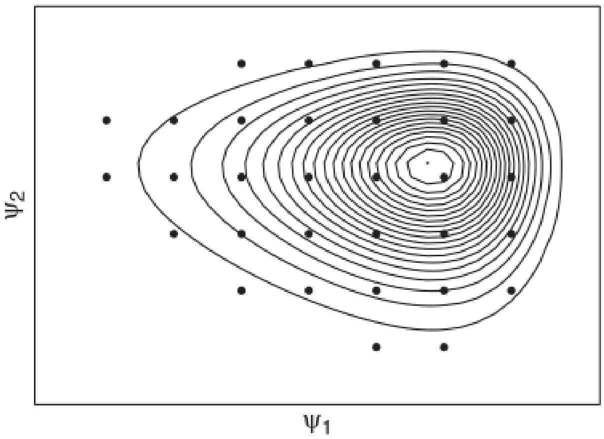


Figure 4.1: CCD to spdetoy dataset, source Marta Blangiardo (2015)

Following Gómez Rubio (2020), approximations of the joint posterior for the hyper paramer $\tilde{\pi}\left(\psi_k \mid y\right)$ is used to compute the marginals for the latent effects and hyper parameters in this way:

$$\tilde{\pi}(\psi \mid \mathbf{y}) \propto \left.\frac{\pi(\theta, \psi, y)}{\tilde{\pi}_G(\theta \mid \psi, y)}\right|_{\theta=\theta^*(\psi)}$$

In the previous equation $\tilde{\pi}_G(\theta \mid \psi, y)$ is a gaussian approximation to the full condition of the latent effect $\theta^*(\psi)$ is the mode for a given value of the hyper param vector ψ

At this point there exists three types of approximations for $\pi\left(\theta \mid \psi, y\right)$

- first with a gaussian approximation, estimating mean $\mu_i(\psi)$ and variance $\sigma_i^2(\psi)$.
- second using the *Laplace Approximation.*
- third using *simplified Laplace Approximation*

(rivedere meglio)

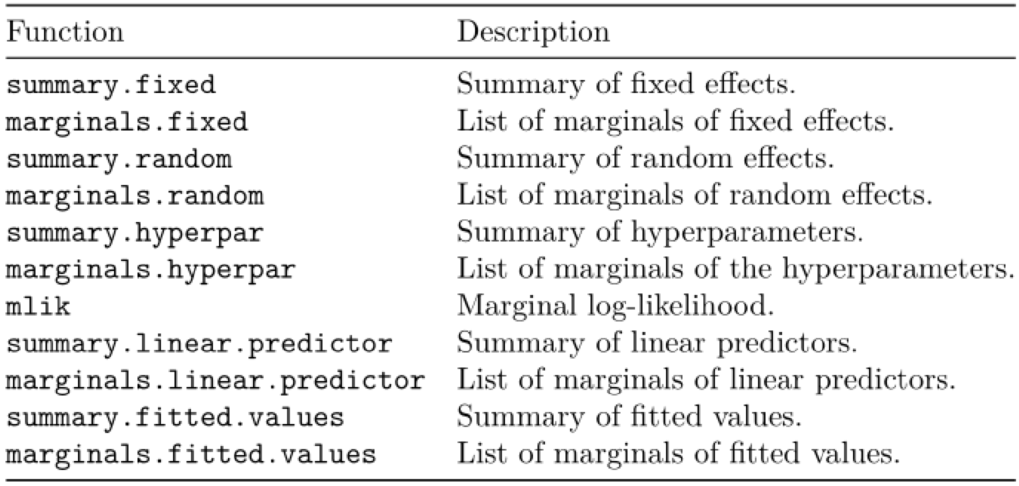
4.3 R-INLA package in a bayesian hierarchical regression perspective

4.3.1 Overview

INLA computations and methodology is developed by the R-INLA project whose package is available on their website[1]. Download is not on CRAN (the Comprehensive R Archive Network) so a special source repo link, which is maintained by authors and collaborators, has to be optioned. The website offers also a forum where a daily discussion group is opened and an active community is keen to answer. Moreover It also contains a number of reference books, among which some of them are fully open sourced as gitbook. Furthermore as Havaard Rue has pointed out in a web-lecture on the topic, the project is gaining importance due to its new applications and recent use cases, but by no means it is replacing the older MCMC methods, rather INLA can integrate pre existing procedures. The core function of the package is inla ()and it works as many other regression functions like glm(), lm() or gam(). Inla function takes as arguments the formula (where are response and linear predictor), the data (expects a data.frame obj) on which estimation is desired together with the distribution of the data. Many other methods inside the function can be added through lists, such as control .family and control . fixed which let the analyst specifying priors distribution both for θ parameters, ψ hyper parameters and the outcome precision τ, default values are non-informative. control . fixed as said regulates prior specification through a plain list when there only a single fixed effect, instead it does it with nested lists when fixed effects are greater than 2, a guided example might better display the behaviour: control . fixed = list (mean = list(a = 1, b = 2, default = 0)) In the chuck above it is assigned prior mean equal to 1 for fixed effect “a” and equal 2 for “b”; the rest of the prior means are set equal to 0. Inla objects are inla.dataframe summary-type lists containing the results from model fitting. Results con-

[1] http://www.r-inla.org

tained in the object are specified in the table below, even though some of them requires special method: (se riesco più elegante in kable) Following Krainski & Rubio (2019) observations $y(s_1), \dots, y(s_n)$ are taken from a toy generated dataset and a hierarchical linear regression is fitted.



Function	Description
`summary.fixed`	Summary of fixed effects.
`marginals.fixed`	List of marginals of fixed effects.
`summary.random`	Summary of random effects.
`marginals.random`	List of marginals of random effects.
`summary.hyperpar`	Summary of hyperparameters.
`marginals.hyperpar`	List of marginals of the hyperparameters.
`mlik`	Marginal log-likelihood.
`summary.linear.predictor`	Summary of linear predictors.
`marginals.linear.predictor`	List of marginals of linear predictors.
`summary.fitted.values`	Summary of fitted values.
`marginals.fitted.values`	List of marginals of fitted values.

Figure 4.2: summary table list object, source: Krainski (2019)

4.3.2 Linear Predictor

SPDEtoy dataset, that has a spatial component, is generated from a y_i Gaussian variable; its moments are μ_i and precision τ.

The formula that describe the linear predictor has to be called directly inside the inla() function or it can be stored in the environment into a variable. The mean moment in the gaussian distribution μ_i is expressed as the *linear predictor* η_i (i.e. $E\left(y_i \mid \beta_0, \dots, \beta_M, x_{i1}, \dots, x_{iM}\right) = \eta_i$). The function that maps the linear predictor into the parameter space is identity as in the initial part of section 4.1 i.e. $\eta_i = \beta_0 + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f_l\left(z_{li}\right)$. After including s_1 and s_2 spatial covariates the linear predictor takes the following form: $\beta_0 + \beta_1 s_{1i} + \beta_2 s_{2i}$, where once again β_0 is the fixed effect i.e. intercept and the β_j are the linear effect on covariates. INLA allows also to include non-linear effects with the f() method inside the formula. f are foundamental since they are

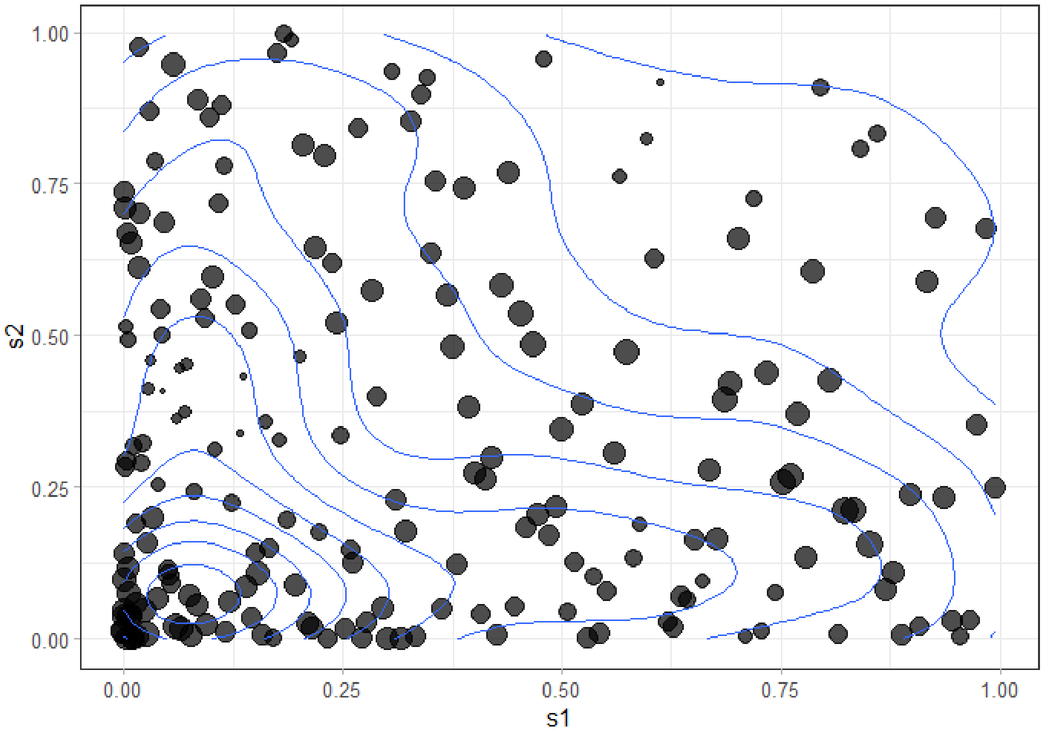


Figure 4.3: SPDEtoy plot, author's source

used to incorporate the spatial component in the model through the Matérn covariance function, this will be shown in section (boh). Once the formula is decided then priors has to be picked up; for the intercept a customary choice is uniform. Prior for Gaussian latent parameters are vague and they have 0 mean and 0.001 precision, then the prior for τ is a Gamma with parameters 1 and 0.00005. Prior initial choice can be later adapted.

The summary of the model parameters is:

$$
\begin{aligned}
&y_i \sim N\left(\mu_i, \tau^{-1}\right), i = 1, \dots, 200 \\
&\mu_i = \beta_0 + \beta_1 s_{1i} + \beta_2 s_{2i} \\
&\beta_0 \sim \text{ Uniform} \\
&\beta_j \sim N\left(0, 0.001^{-1}\right), j = 1, 2 \\
&\tau \sim Ga(1, 0.00005)
\end{aligned}
$$

```
data(”SPDEtoy”)
```

```
formula = y ~ s1 + s2
m0 = inla(formula, data = SPDEtoy)
```

	mean	sd	0.025quant	0.5quant	0.975quant	mode	
(Intercept)	10.1321487	0.2422118	9.6561033	10.1321422	10.6077866	10.1321497	7
s1	0.7624296	0.4293757	-0.0814701	0.7624179	1.6056053	0.7624315	7
s2	-1.5836768	0.4293757	-2.4275704	-1.5836906	-0.7404955	-1.5836811	7

The output offers among the others a summary of the posterior marginal values for intercept, coefficient and covariates, as well as precision. Below the plots for the parameters and hyperparameters. From the summary it can be seen that the mean for s2 is negative, so the more the value of the y-coordinates increases the more the output decreases, that is truer looking at the SPDEtoy cotour plot. Plots can be generated by calling the plot function on the inla object, however the one generated below are ggplot2 outputs coming from the $marginals.fixed list object.

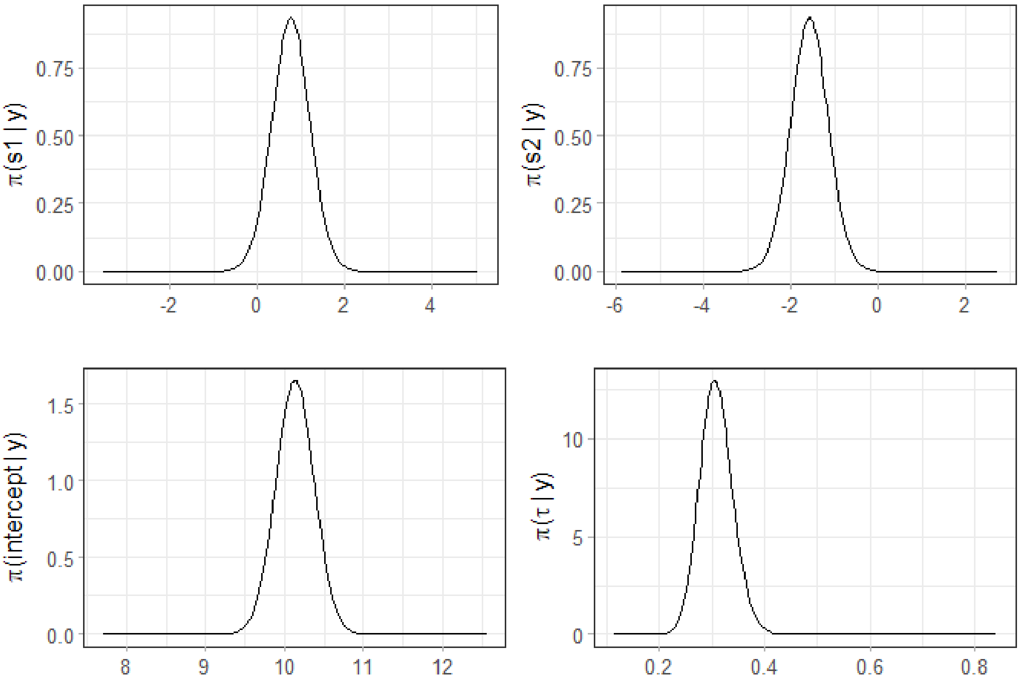


Figure 4.4: linear predictor marginals, author's creation

R-Inla also has r-base fashion function to compute statistics on marginal posterior distributions for the density, distribution as well as the quantile function respectively inla .dmarginal, inla .pmarginal and inla .qmarginal. One major option which is conveniently packed into a dedicated function computes the higher posterior density credibility interval inla .hpdmarginal for a given covariate's coefficient, such that $\int_{q_1}^{q_2} \tilde{\pi}\left(\beta_2 \mid y\right) \mathrm{d}\beta_2 = 0.90$ zwith .1 Confidence Level, in table @ref(tab:higer_posterior_density_interval).

	low	high
level:0.9	-2.291268	-0.879445

Recall that the interpretation is different from the frequentist: in Bayesian statistics β_j comes from probability distribution, while frequenstists considers β_j as fixed unknown quantity whose estimator (random variable conditioned to data) is used to infer the value (2015).

Chapter 5

Point Referenced Data Modeling

Geostatistical data are a collection of samples of geo type data indexed by coordinates (e.g. latlong, eastings and northings) that originate from a spatially continuous phenomenon (Moraga, 2019). Data as such can monitor a vast range of phenomena, as an example disease cancer detection (Bell et al., 2006) at several sites, COVID19 spread in China (Li et al., 2020), PM pollution concentration in a North-Italian region Piemonte (Cameletti et al., 2012). Moreover house prices variation, as observed in Gómez Rubio (2020), where selling prices smoothly vary between closer neighborhoods. All the Examples taken before might document a spatial nature of data according to which closer observations can display similar values, this phenomenon is named spatial autocorrelation. Spatial autocorrelation conceptually originates from geographer Waldo Tobler whose famous quote, known as first law of geography, inspires geostatisticians:

> "Everything is related to everything else, but near things are more related than distant things"
>
> — Waldo R. Tobler

Spatial models are explicitly designed to take into account this behavior and can separate spatial patterns from simply random spatial variance. Spatial data can be partitioned into three spatial data type whose modeling tools are specific with respect to their category.

- Areal Data
- **Point Referenced Data**
- Point Pattern Data

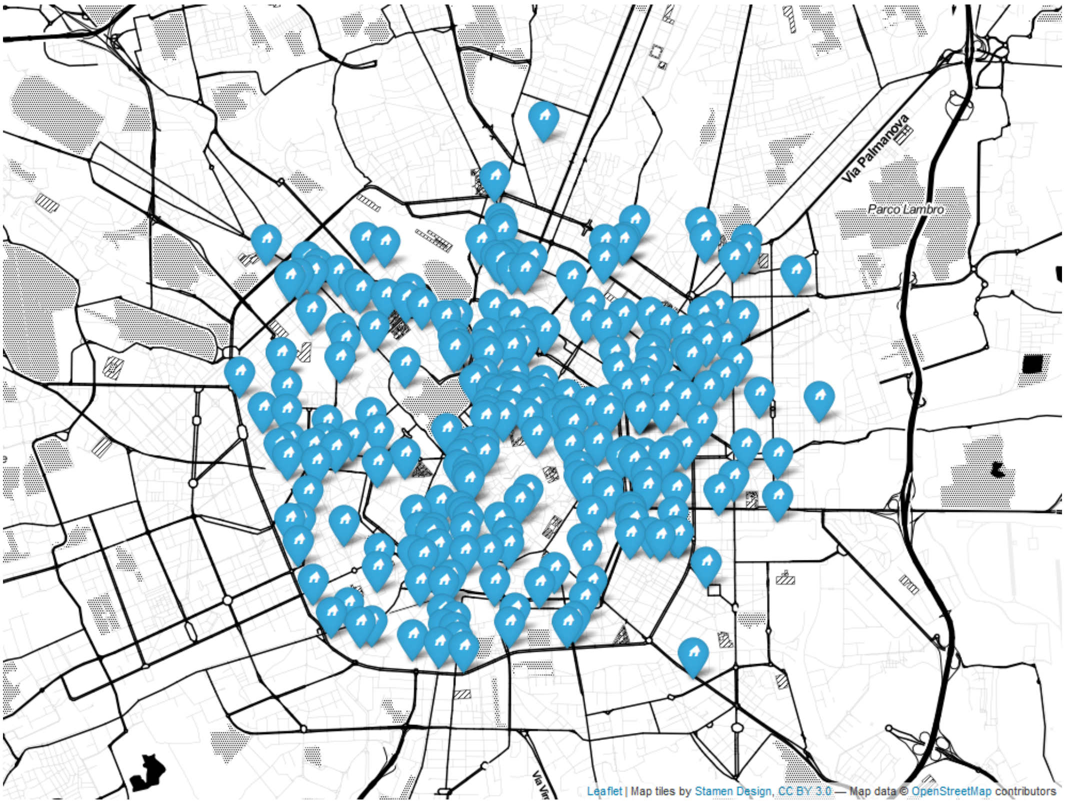

Via Palmanova
Parco Lambro
Via Vit
Leaflet | Map tiles by Stamen Design, CC BY 3.0 — Map data © OpenStreetMap contributors

Figure 5.1: point referenced data example, Milan Rental Real Estate, Author's Source

REST API designed in chapter 3 extracts point referenced data, so modeling methodologies described in this analysis will exclusively take into account point referenced techniques. In order to extend the notion from discrete measurements (i.e. point referenced) to a continuous spatial surface a stochastic process, namely Gaussian Process, has to be introduced and constrained to Stationarity and Isotropy. GP are then evaluated with a specific covariance function, i.e. Matèrn. The reason why Matérn is selected as candidate for covariance function will be much more clear in the next chapter 6. Hedonic Price Models are at first introduced and then a brief literature review is offered. Hedonic Prices brings to this work the theoretical basis but they do not suggest estimation methods, which are essentially the major issue in geostatistics. For this reason Hedonic Models are exploited into a spatial bayesian regression framework with the aim to apply INLA (seen in chpater 4) methodology. At first standard Bayesian regression is presented as introduction, then the spatial component in the form of a GP is added to the model. Many parameters are considered so far, as a consequence a hierarchy structure is imposed. To this extent an entire section is dedicated to hierarchy which simplifies model building and methodology understanding as well as allowing to bring in many different parameters that come from different levels through the exchangeability property. As a matter of fact parameters originate from the Gaussian latent field, but also from Matérn covariance function tuning hyper parameters. Then INLA is applied and a GMRF representation of GP is... Spatial kriging is essential to predict the process at new locations so that the spatial surface can be plotted and analyzed. In the end models have to be checked and verified with resampling schemes which are once again specific to the data type and the scope of the analysis.

(forse mettere alla fine come further developments) As a side note Spatial data can also be measured according to a further dimension which is the Time. Latest literature suggests that spatio temporal models are the most accurate, as a consequence it might be interesting to research time correlation between subsequent spatial data time points, a valuable reference is offered in Paci et al. (2017). This will not take an enormous effort due ti the fact that on a daily basis REST API generates data which are stored as .json file on a DB. Future research on this data might consider the idea to include the time component in the model.

5.1 Gaussian Process (GP)

For simplicity lets consider y point of interest observations $y\left(s_1\right), y\left(s_2\right), \ldots, y\left(s_n\right)$ from a random spatial process Y, such that: $Y\left(s_1\right), Y\left(s_2\right), \ldots, Y\left(s_n\right)$ observed at location $s_1, \ldots, s_n$. In the context of geostatistical data each observation has to be considered as a partial realization of an unobserved random spatial process. $\left\{Y(s): s \in D \subset \mathbb{R}^2\right\}$, where surface D is a subset of r-dimensional Euclidean space $\mathbb{R}^r$. Moreover When $r = 1$ it is the most simple stochastic process widely explored in literature i.e. time series process. However geostatistical data always have $r = 2$ (i.e. lat and long, eastings and northings) or eventually $r = 3$, when elevation data is available. The stochastic process Y is observed in a fixed set of "monitoring stations" and inference can be done regarding moments of the realized process. This information are essential to build a spatially continuous surface over the y-studied variable in order to predict the phenomenon at locations not yet observed.

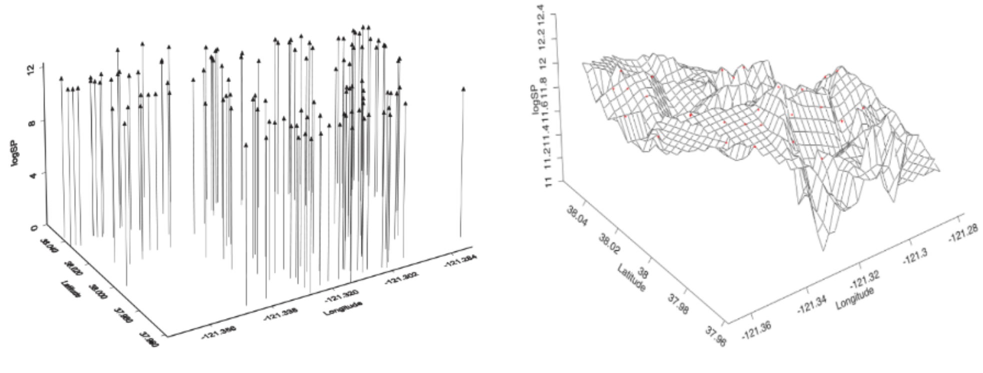


Figure 5.2: 3D scatterplot and surface, Stockton data.

Definition 5.1 (GP definition)**.** A collection of n random variables, such as $Y(s_1), Y(s_2), \ldots, Y(s_n)$ that are *valid* spatial processes are said to be a **GP**

if for any set of spatial index n and for each set of corresponding locations $\{y(s_1), \ldots, y(s_n)\}$ follows a multivariate *Gaussian* distribution with mean $\mu = \{\mu(s_1), \ldots, \mu(s_n)\}$ and covariance matrix $\mathbf{Q}_{i,j}^{-1}, \forall i \neq j$

Even tough sometimes it is more convenient to express the covariance matrix as its inverse i.e. precision matrix $Q_{i,j}$ (Marta Blangiardo, 2015). The covariance matrix relates each observation to each of the others through a covariance function defined as $\mathcal{C}(\cdot)$.

GP in the spatial context must check two important properties in order to exploit INLA, even though both of these assumptions can be relaxed:

- **Stationary**.
- **Isotropy**.

Stationarity in a stochastic process can be *strong*, *weak* or *intrinsic*. The strong property forces the distribution of the process $\{y(s_1), \ldots, y(s_n)\}$ for any given spatial index n and its correspondent location sets $s_{1,\ldots,n}$ to be the same as the one in $\{y(s_1 + h), \ldots, y(s_n + h)\}$, where h is a number belonging to R^2. On the other hand the weak property ensures that if the GP mean moment is constant over the study domain $\mu(\mathbf{s}) \equiv \mu$ (e.g. $E[Y(s)] = \mu, \forall s \in D$) then the covariance functions does depend only on the distance (euclidean $\|s_i - s_j\|$ distance) between each couple points. Weak stationarity consequences are the most interesting: It does not matter whether observations are placed either in a specific region, nor the direction towards they are oriented, the covariance functions $\mathcal{C}(h)$ can summarize the process through the separation vector $\mathbf{h}$ i.e. $\mathcal{C}(\mathbf{s}, \mathbf{s} + \mathbf{h}) = \mathcal{C}(\mathbf{h}), \forall \mathbf{h} \in \mathbb{R}^r$ (Banerjee et al., 2014). In other words weak stationarity in GP implies being invariant under *translation* (2019). The relationship between strong and weak is not bijective since being strong implies also being weak, but the opposite is not always true for non-Gaussian process. Furthermore through the intrinsic stationary property it is meant that $E[Y(\mathbf{s}+\mathbf{h}) - Y(\mathbf{s})] = 0$, the second moment of the latter expression can be written as $E[Y(\mathbf{s}+\mathbf{h}) - Y(\mathbf{s})]^2$ leading to $\operatorname{Var}(Y(\mathbf{s}+\mathbf{h}) - Y(\mathbf{s}))$. Last expression is called *variogram* and can be expressed with $2\gamma(\mathbf{h})$, even tough its half,i.e. $\gamma(\mathbf{h})$, is more interpretable, namely *semivariogram* (Cressie, 2015).

Semivariograms are characterized by mainly 3 tuning parameters:

- *range* σ^2: At some offset distance, the variogram values will stop changing and reach a sort of "plateau". The distance at which the effect occurs is called the range $\frac{\Delta\gamma(\mathrm{h})}{h} \approx 0$.
- *sill* τ^2: The "plateau" value at which the variogram stops changing $\frac{\Delta\gamma(\mathrm{h})}{h} = 0$.
- *nugget* $\tau^2 + \sigma^2$: The discontinuity at the origin. Although this theoretically should be zero, sampling error and short scale variability can cause it to be non-zero $\gamma(0)$.

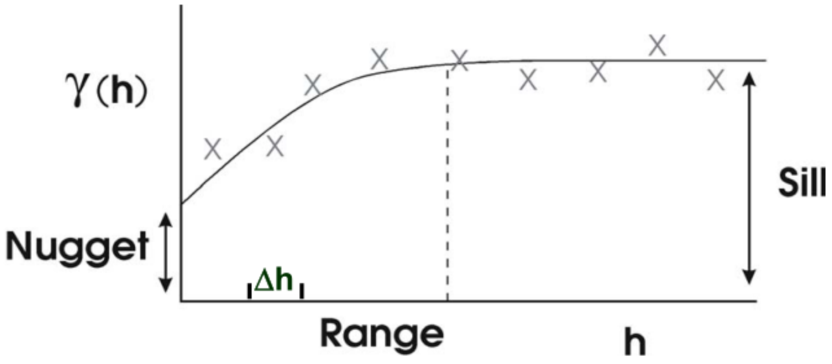


Figure 5.3: variogram example

presi i dati con le relative distanze euclidee a coppie di punti si binnano le distanze grazie ad un offset ottenendo i valori per il semivariogram. ottenuti i valori si fitta il semivargiogram a quei valori, un modo è la likelihood. A questo punto si calcolano le tre grandezze nugget sill e range per poi poter far uscire le funzioni di covarianza.

The process is said to be **Isotropic** if the covariance function depends only on the between-points distance $\|\mathbf{h}\|$ so it is invariant under *rotation* (2019). A further way of seeing the property is that Isotropy implies concentric decaying contours that resemble the vanishing of spatial dependence, and so covariance values too. then if the last assumption does not hold and direction towards point are distant from each other matters within the spatial domain D, then is said to be **Anisotropic**. Formalizing the results:

$$\mathcal{C}(\mathbf{h}) = \mathcal{C}(\|\mathbf{h}\|)$$

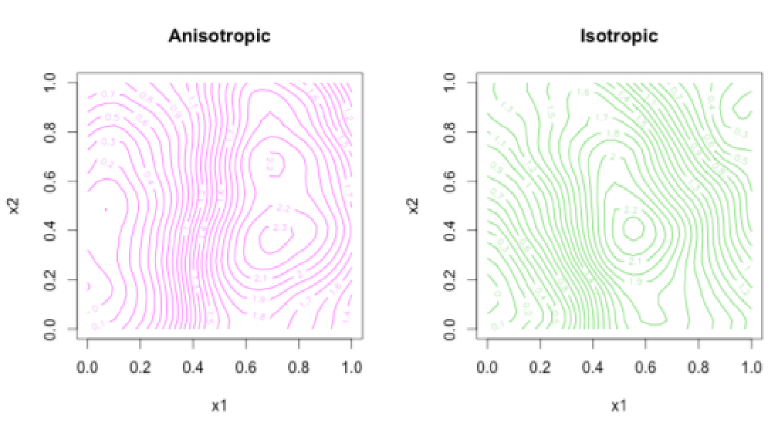


Figure 5.4: isotropy VS anisotropy, source Blanchet-Scalliet et al. (2019)

5.2 Spatial Covariance Function

The covariance function $\mathcal{C}(\cdot)$ ensures that all the values that are close together in input space will produce output values that are close together. $\mathcal{C}(\cdot)$ needs to inherits the *validity* characteristics from the random spatial process, furthermore it has to be *positive definite*. In addition covariance function must share characteristic properties of functions, such as:

(cerca di capire queste...)

- Multiply valid covariance functions (summing independent random variables)

- Mixing covariance functions (mixing distributions)
- Convolving covariance functions, this will be very important ...

Covariance functions under stationary and isotropic GPs displays two important properties: they are constant in mean within D i.e. $\mathcal{C}(\mathbf{s}, \mathbf{s}+\mathbf{h}) = \mathcal{C}(\mathbf{h}), \forall \mathbf{h} \in \mathbb{R}^r$ and they depends on distance vector $\mathbf{h}$, not direction i.e. $\mathcal{C}(\mathbf{h}) = \mathcal{C}(\|\mathbf{h}\|)$ There are many covariance functions and ways to relate distant points on a spatial domain D. Typically the choice of the Covariance can depend either on data or the scope of the analysis. Covariance functions are wrapped into special hyper parameters which are mainly three:

1. *Range*: At some offset distance, the variogram values will stop changing and reach a "plateau". The distance at which this occurs is called the range.
2. *Sill*: The "plateau" value at which the variogram stops changing.
3. *Nugget*: The discontinuity at the origin. Although this theoretically should be zero, sampling error and short scale variability can cause it to be non-zero

(espressione della covariance function insieme a alle σ^2 come: $C(\mathbf{s}+\mathbf{h}, \mathbf{s} \mid \theta) = \sigma^2 \mathbf{R}(\|h\|; \phi))$ spiega anche queste due sotto

$$\mathbf{w} = \left(w\left(\mathbf{s}_1\right), \ldots, w\left(\mathbf{s}_n\right)\right)' \sim \mathrm{N}\left(\mathbf{0}, \sigma^2 \mathbf{R}(\phi)\right) \text{ where } \mathbf{R}(\phi)_{ij} = \rho\left(\left\|\mathbf{s}_i - \mathbf{s}_j\right\|; \phi\right)$$

$\Sigma_\theta = \sigma^2 \mathbf{R}(\phi) + \tau^2 I_n$

A summary of the most used covariance functions are presented below.

$$\text{Exponential} \quad \mathcal{C}(\mathbf{h}) = \begin{cases} \tau^2 + \sigma^2 & \text{if } h = 0 \\ \sigma^2 \exp(-\phi h) & \text{if } h > 0 \end{cases}$$

$$\text{Gaussian} \quad \mathcal{C}(\mathbf{h}) = \begin{cases} \tau^2 + \sigma^2 & \text{if } h = 0 \\ \sigma^2 \exp\left(-\phi^2 h^2\right) & \text{if } h > 0 \end{cases}$$

$$\text{Matérn} \quad \mathcal{C}(\mathbf{h}) = \begin{cases} \tau^2 + \sigma^2 & \text{if } h = 0 \\ \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)}(\phi h)^\nu K_\nu(\phi h) & \text{if } h > 0 \end{cases}$$

5.2.1 Matérn Covariance Function

Matérn is special since when it is used together with a stationary and isotropic GP, the SPDE approach can provide a GMRF representation of the same process, chapter 6 discloses this fundamental property. Matérn can also be accounted as the most used in geostatistics (Krainski et al., 2018) and (Gómez Rubio, 2020) and is tuned mainly by two parameters, a scaling one $\kappa > 0$, usually set equal to the range by the relation $\sigma^2 = \frac{\sqrt{8\lambda}}{\kappa}$) and a smoothing one $\nu > 0$. A *stationary* and *isotropic* Matérn covariance function has this form:

$$\mathcal{C}(\mathbf{h}) = \begin{cases} \tau^2 + \sigma^2 & \text{if } h = 0 \\ \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)}(\phi t)^\nu K_\nu(\phi t) & \text{if } h > 0 \end{cases}$$

$\Gamma(\nu)$ is a Gamma function depending on ν values, $K_\nu(\cdot)$ is a modified Bessel function of second kind. The smoothness parameter ν in the figure below takes 4 different values showing the potentiality of Matérn to relates distances to covariance values. When $\nu = 1$ ... When $\nu = 1/2$ it becomes the exponential covariance function, When $\nu = 3/2$ it uncovers a convenient closed form, when $\nu \approx \infty$, in this case for representation purposes $\nu = 80$ it becomes Gaussian covariance function.

ancora di più su matern, forse di più in spde

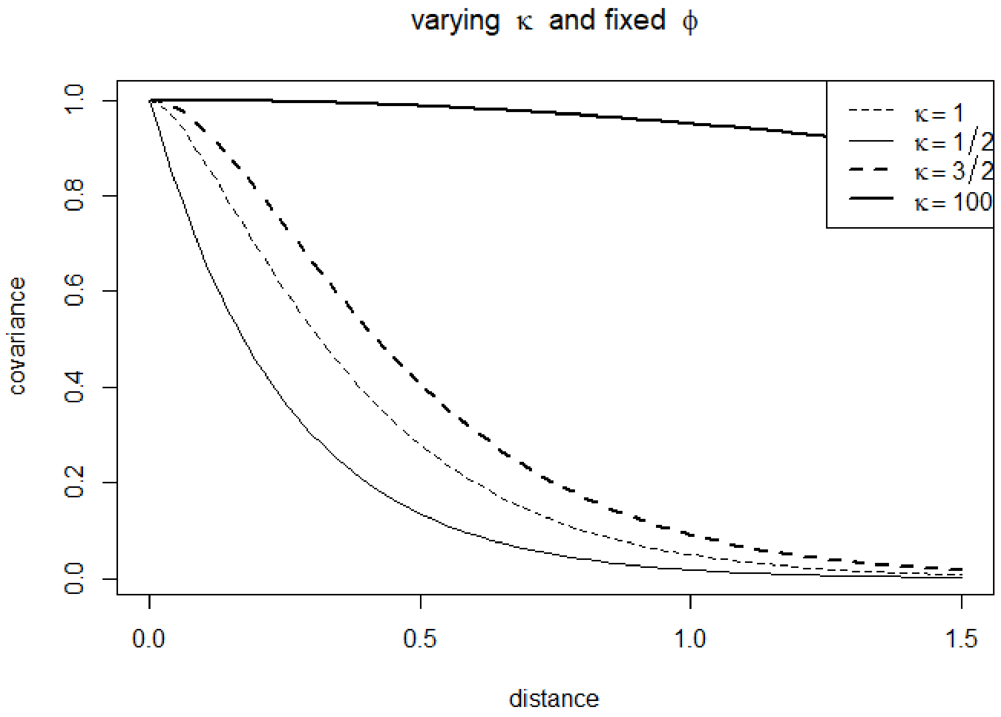


Figure 5.5: matern correlation function for 4 diff values of nu with phi fixed, author's source

5.3 Hedonic models Literature Review and Spatial Hedonic Price Models

The theoretical foundation of the Hedonic Price Models (from now on HPM) resides in the consumer utility theory of Lancaster (1966) together with Rosen (1974) market equilibrium. According to Lancaster the utility of a commodity does not exist by itself, instead it exists as the sum of the utilities associated to its separable characteristics. Integrating Lancater, Rosen introduces HPM and suggests that each separate commodity characteristics are priced by the markets on the basis of supply and demand equilibrium. Applying HPM to Real Estate in a market context, from the buy side house prices (but also rents) are set as the unit cost of each household attributes, conversely from the selling side the expenditures associated to build of each them. Formalizing the results, Hedonic Price P in Real Estate is expressed as a general f functional form that takes as input the house characteristics vector $\mathbf{C}$.

$$P = f\left(c_1, c_2, c_3, \ldots, c_n\right)$$

Vector $\mathbf{C}$ since now might contain a unidentified and presumably vast number of ungrouped characteristics. In this setting Malpezzi (2008) tried to organize house features by decomposing $\mathbf{C}$ into mutually exclusive and exhaustive subgroups. An overview of the vector components involved is given by Ling and Ling (2019) according to which P represents the house price, S is the structural characteristics of the house, N represents the neighborhood characteristics, L signifies the locational characteristics, C describes the contract conditions and T is time. β is the vector of the parameters to be estimated. Then

$$P = f\left(S, N, L, C, T, \beta\right)$$

Historically a first attempt to include spatial effect in urban economic literature is provided by *Alonso (1964) miss ref.* Its contribution was to raise voice on house prices (also rent) mainly depending on land price and a number of purely spatial covariates like CBD, the distance from City Business District. Other covariates were transport cost per kilometer and community income, even though they were defined also as spatial parameters through distances. The model proposed by Alonso is called monocentric since the centroid from which distances are calculated is only one. Moreover a first touch to spatial data thory was done since the CBD was defined as areal unit with well-defined boundaries of regular or irregular shape. However applications of the model were not convincing since empirical studies offered a different picture. Results instead displayed a Poly-centric areal structure (universities and Malls) which might be better explaining prices. The model also assumed that covariates like CBD are only informative within city center boundaries and then displayed no significance out of the core of the city. Poly-centric theory was also more coherent with the architectural and socio-economical evolution of cities during that times, therefore mono centric theory was then criticized and abandoned. Critics regarded also neighborhood quality measure and boundary problems *Dubin (1987) miss ref.* Dubin for these reasons developed a model including areal effects in the error term since handling these covariates was posing several hard challenges. Areal data choice for Dubin was forced since he was interested in land values, geostatics interest was not a focus also due to the difficulties in gathering accurate data. Coming to recent literature a change in focus has been made by switching from theory based model to estimation methods. As a consequence to the change in focus Ling and Ling (2019) said that practitioners should spend more time in variable selection and model specification with respect to their specific need. As Ling has observed the emerging trends are in the field of semi-parametric and non-parametric methods (2019). Historically semi-parametric regression considers models indexed by spatial coordinates *Pace RK (1995).* At the same time *Kammann and Wand (2003)* gave birth to geoadditive models where the spatial component is added as a covariate. [...]

A further aspect of the problem is posed by scholars that do not consider rents to be representative for the actual value of real estate. Nevertheless in empirical analysis rent value are considered a proxy for real estate pricing (Herath and Maier, 2011). A further argument to endorse this hypothesis is brought by Manganelli et al. (2013) considering housing a commodity, then the selling or the rental should be considered interchangeable economic actions with respect to same inner need to be satisfied. This is also truer to the thesis' extent since Manganelli, Morano, and Tajani have centered their analysis exactly on italian real estate data. Moreover Capozza and Seguin (1996) discussed on how much rent-price ratio predicts future changes both in rents and prices. Among all the other discussions raised they brought the decomposition of rent-price ratio into two parts: the predictable part and the unexplained residuals part. The predictable part was discovered to be negatively correlated with price changes, in other words cities in which prices are relatively high with respect to rents are associated with higher capital gains that might justify that misalignment. This is also true for the opposite, that is cities in which prices are lower with respect to the rents, and this effect can not be associated to any local condition, realize lower capital gains. A further argument is offered by Clark (Clark, 1995) which went after the Capozza and Seguin work. Rent-price ratio is negatively correlated with following future changes in rents. In other words prices are still higher when areas in which they are observed documents an increase in rent prices. All the literature review above is oriented to a long-run alignment of price and rent.

5.4 Point Referenced Regression for univariate spatial data

Since in HPM the relationships between the characteristics of the house, i.e. vector $\mathbf{C}$ and the price P is not in any case fixed by econometric literature it is possible to assume any f functional form. The open possibility to apply a wide range of relationship between covariates fit in the INLA setting, since Latent Gaussian Models are prepared to accept a any linear and non linear

f functions 4.1 through the f() method. Hedonic price models are, as a consequence, a subset of models that can be fitted into LGM and therefore by INLA method.

Moreover what the vast majority of econometric literature *(Greene, 2018)* suggest to apply a is log-linear / square root model. This is due to the fact that log transformation / square root smooths the skewness of prices normalizing the curve, leading to more accurate estimates. Having an exponential family generating process lowers even further computational cost for reasons linked to the $\tilde{\pi}(\psi)$ hyper param INLA approximation (Marta Blangiardo, 2015). Notation is taken from the previous chapter 4, for brevity purposes β $\mathbf{X}$ and y indicates vectors incorporating all their respective realizations and the s spatial component is left out in favor of the observation pedix i.
The simplest log-linear bayesian regression model assumes linear relationship between predictors and a Normal data generating process: (log has been taken out for simplicity, bu it will be then considered in the regression setting) (valuta l'idea che per interpretabilità di modellarla come Gamma exponential family anzichè tenerla normale)

$$\log\left(y_i\right) \sim \operatorname{Normal}\left(\mu_i, \sigma^2\right)$$

$$y_i = \mu_i + \varepsilon_i$$

then by the following relationship $E\left(y_i \mid \beta_0, \ldots, \beta_M, x_{i1}, \ldots, x_{iM}\right) = \beta_0 + \sum_{m=1}^{M} \beta_m x_{im}$ it is possible to specify a more general linear predictor (seen also in chapter 4) through an identity link function i.e. $\eta_i = g\left(\mu_i\right) = \mu_i$ obtaining:

$$\eta_i = \beta_0 + \sum_{m=1}^{M} \beta_m x_{mi} + \sum_{l=1}^{L} f_l\left(z_{li}\right)$$

Where, once again, the mean structure linearly depends on some $\mathbf{X}$ covariates,

β coefficients, $f_l(\cdot), \forall l \in 1 \ldots L$ are a set of random effects defined in terms of a z set of covariates $z = (z_1, \ldots, z_L)$ (e.g. rw, ar1) and ε_i white noise error. Priors have to be specified and a non informativeness for $\tau^2 = 1/\sigma^2$ and β is chosen, such that $\pi(\tau^2) \propto 1$ and $\pi(\beta) \propto 1$. As a consequence the conditional posterior for the parameters of interest β is:

$$\beta \mid \sigma^2, y, X \sim \text{MVNormal}\left(\left(X'X\right)^{-1} X'y, \sigma^2 \left(X'X\right)^{-1}\right)$$

where the mean structure corresponds to the OLS estimator: $\left(X'X\right)^{-1} X'y$ for β and then to obtain the marginal posterior for β it is needed to integrate with respect to σ^2.

In order to engage the spatial coordinate components into the regression setting w_i has to be added to the equation. w_i is set as a stationary and isotropic GP with mean 0 and variance as covariance function expressed as Matérn. Recall that GP The new regression setting integrates the *spatial error* part in the name of w_i and a *non-spatial error* part ε_i distributed normally with mean 0 and variance τ^2 ,i.e. $\mathrm{N}\left(0, \tau^2\right)$, which offers its contribution error to the nuggets via the covariance function. Consequently there is one more parameter to estimate. It is worth mentioning that the distribution of w_i at a finite number of points is considered a realization of a multivariate Gaussian distribution. In this case, the likelihood estimation is possible and it is the multivariate Gaussian distribution with covariance Σ.

$$\log(y_i) = \beta_0 + (\mathbf{X})'\beta + w_i + \varepsilon_i$$

The covariance of the marginal distribution of y_i at a finite number of locations is $\Sigma_y = \Sigma + \tau^2\mathbf{I}$, where $\mathbf{I}$ denotes the indicator function (i.e., $\mathbf{I}(i = i') = 1$ if $i = i'$, and 0 otherwise). This is a short extension of the basic GF model, and gives one additional parameter to estimate

5.5 Hierarchical Bayesian models

Spatial Models are characterized by many parameters which in turn are tuned by other hyper-parameters. Traditionally Bayesian hierarchical models are not widely adopted since they have high computational burdens, indeed they can handle very complex interactions via random components, especially when dealing with spatio temporal data Ling and Ling (2019). Blangiardo e Cameletti (2015) tried to approach the problem from a different angle offering an intuitive solution on how hierarchy relates different levels parameters. This is done by reversing the problem and starting from data back to parameters, instead the other way round. So taking a few steps back the problem can be reformulated by starting from grouping observation into categories and then trying to impose a hierarchical structure on data based on the categories. As a result observations might fall into different categories, underlining their natural characteristics, such as: some of them might belong to category *levels* like males or females, married or not-married. Moreover diving into the specific problem house prices can be faceted by which floor they belong or whether they are assigned to different energy classes and many others more. As an example Blangiardo and Cameletti example consider grouping data according to just a single 9 *levels* category. Data for the reasons stated before can be organized such that each single observation (squares in figure below) belongs to its respective mutually exclusive and collectively exhaustive category (circles in figure).

Furthermore data can be partitioned into two meta-categories, *fist level* and *second level*, highlighting the parameter and hyper paramter chain roles. *First level* are identified by sampling observations which are drawn by the same probability distribution (squares) . *Second level* (circles) are categories and might be associated to a set of parameters $\theta = \{\theta_1, \ldots, \theta_J\}$. Since the structure is hierarchical, a DAG (Directed Acyclical Graph) (2015) representation might sort out ideas. If categories are represented by different θ_j nodes and edges (arrows in the figure) are the logical belonging condition to the category then

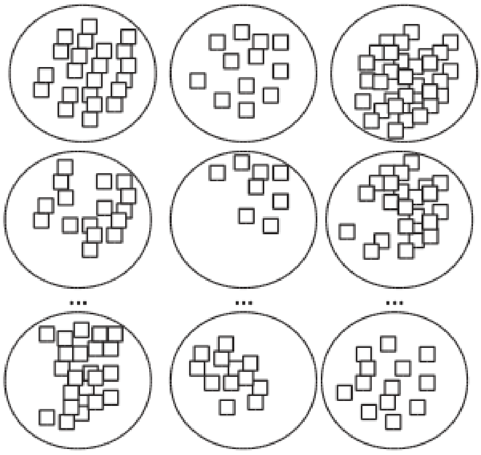
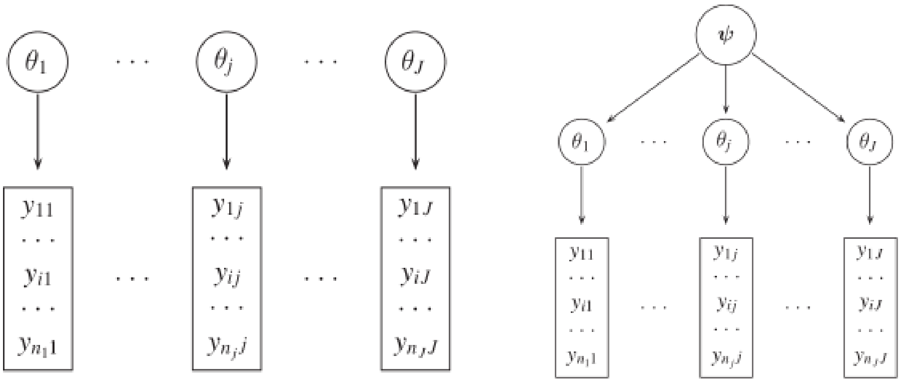


Figure 5.6: 9 levels cat vs observaitions, source Marta Blangiardo (2015)

a single parameter θ model has the right figure form:



To fully take into account the hierarchical structure of the data the model should also consider further lelvels. Since $\{\theta_1, \dots, \theta_J\}$ are assumed to come from the same distribution $\pi(\theta_j)$, then they are also assumed to be sharing information (Marta Blangiardo, 2015), (left figure). When a further parameter $\psi = \{\psi_1, \dots, \psi_K\}$ is introduced, for which a prior distribution is specified, then the conditional distribution of θ given ψ is:

$$\pi\left(\theta_1, \dots, \theta_J \mid \psi\right) = \int \prod_{j=1}^{J} \pi\left(\theta_j \mid \psi\right) \pi(\psi) \mathrm{d}\psi$$

This is possible thanks to the conditional independence property already encountered in chapter 4, which means that each single θ is conditional independent given ψ This structure can extended to allow more than two levels of hierarchy since the marginal prior distributions of θ can be decomposed into the product of their conditional priors distributions given some hyper parameter ψ as well as their prior distribution $\pi(\psi)$.

$$\pi(\theta)=\int \pi\left(\theta \mid \psi_1\right) \pi\left(\psi_1 \mid \psi_2\right) \ldots \pi\left(\psi_{L-1} \mid \psi_L\right) \pi\left(\psi_L\right) \mathrm{d} \psi_1 \ldots \mathrm{d} \psi_L$$

ψ_l identifies the hyper pram for the l_{th} level of hierarchy. Each further parameter level ψ is conditioned to its previous in hierarchy level $l-1$ so that the parameter hierarchy chain is respected and all the linear combinations of parameters are carefully evaluated. The *Exchangeability* property enables to have higher H nested DAG (i.e. add further L levels) and to extend the dimensions in which the problem is evaluated, considering also time together with space. From a theoretical point of view there are no constraints to how many L levels can be included in the model, but as a drawback the more the model is nested the more it suffers in terms of interpretability and computational power. Empirical studies have suggest that three levels are the desired amount since they offer a good bias vs variance trade-off.

5.6 INLA model through spatial hierarchical regression

INLA model seen in section 4.1 can be rearranged according to the hierarchical structure considering also the regression settings for point referenced data stated in the previous section 5.4.

As an initial step it is required to specify a probability distribution for $y = (y\left(s_1\right), \ldots, y\left(s_n\right)) = (y_1, \ldots, y_n)$, this is a mandatory step looking at the 4.3.2 methods needed to compute the $\operatorname{inla}()$ function. A Normal distribution for simplicity is chosen.

As *first level* is picked up an **exponential family** sampling distribution (i.e. Normally distributed, Gamma one other choice), which is *exchangeable* with respect to the $\theta = \{\beta_0, \beta, f\}$ *latent field* and hyper parameters ψ_1, which includes also the ones coming from the latent Matérn GP process w_i. The Spatial Guassian Process is centered in 0 and with Matérn covariance function as τ^2. w_i addresses the spatial autocorrelation between observation through a Matérn covariance function $\mathcal{C}(\cdot|\psi_1)$ which in turn is tuned by hyper param included in ψ_1. Moreover the w_i surface has to be passed in the formula method definition 4.3.2 via the f() function, so that INLA takes into cosideration the spatial component.

$$y \mid \theta, \psi_1 \sim \mathrm{N}\left(\beta_0 + (\mathbf{X}_i)'\beta + w_i, \tau^2 I_n\right) = \prod_{i=1}^{n} \mathrm{N}\left(y_i \mid \theta_i, \psi_1\right)$$

Then at the *second level* the latent field θ is characterized by a Normal distribution given the remaining hyper parameters ψ_2, recall the covariance matrix $Q^{-1}(\psi_2)$, depending on ψ_2 hyperparameters, is handled now by a Matérn covariace function depeding on its hyperparamter. This is done in order to map the GP spatial surface into a GMRF by SPDE solutions.

$$\theta \mid \psi_2 \sim \mathrm{N}\left(0, \mathcal{C}(\cdot, \cdot \mid \psi_2)\right)$$

In the end a *third level* hyper parameters $\psi = \{\psi_1, \psi_2\}$ having some specified prior distribution i.e. $\psi \sim \pi(\psi)$,

5.7 Spatial Kriging

In Geostatistics the main interest resides in the spatial prediction of the spatial latent field pr the response variable at location not yet observed. Assumed the model in the previous section, suppose that $y^\star$ is not a observed occurrence of the response variable at location s_0 (not in the data) of the GP w_i spatial surface estimated through observed refereced points in y. As a consequence of exchangeability (first step previous section 5.6) then $y^\otimes = \{y, y^\star\}$. Then considering INLA notation it is obtained:

$$\begin{aligned}
\pi\left(y^{\star} \mid y\right) &= \frac{\pi\left(y, y^{\star}\right)}{\pi(y)} \text{ from the conditional probability} \\
&= \frac{\int \pi\left(y^{\star} \mid \theta\right) \pi(y \mid \theta) \pi(\theta) \mathrm{d} \theta}{\pi(y)} \text{ by exchangeability} \\
&= \frac{\int \pi\left(y^{\star} \mid \theta\right) \pi(\theta \mid y) \pi(y) \mathrm{d} \theta}{\pi(y)} \text{ applying Bayes' theorem} \\
&= \int \pi\left(y^{\star} \mid \theta\right) \pi(\theta \mid y) \mathrm{d} \theta
\end{aligned}$$

A DAG representation might offr the intuition behind Prediction in spatial models:

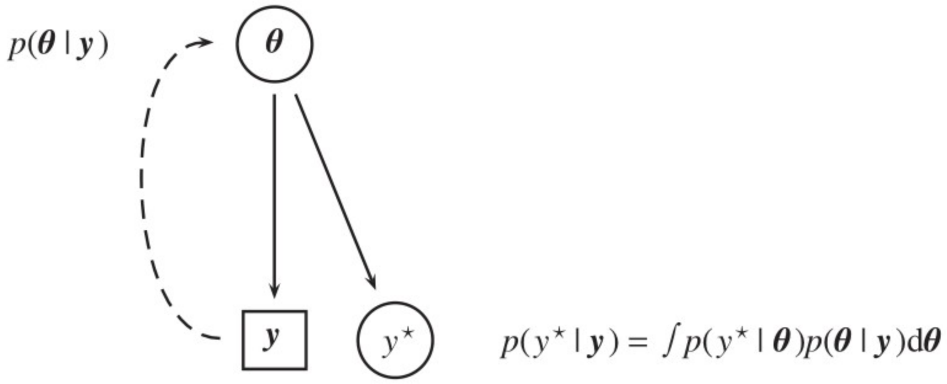

$p(\boldsymbol{\theta} \mid \boldsymbol{y})$
$\boldsymbol{\theta}$
$\boldsymbol{y}$
$y^{\star}$
$p(y^{\star} \mid \boldsymbol{y}) = \int p(y^{\star} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{y}) \mathrm{d}\boldsymbol{\theta}$

Figure 5.7: Spatial prediction representation through DAG, source Marta Blangiardo (2015)

where $\pi\left(y^{\star} \mid y\right)$ is said predictive distribution and it is meaningful only in the Bayesian framework since the posterior distribution is treated as a random variable, which is totally not true in frequentist statistics.

5.8 Model Checking

(Incrociarlo con altri tesi)

Once the model is set up and fitted a resampling scheme has to be chosen in order to evaluate the model performance. One of the most used method to assess beyasian model quality is LOOCV cross validation and defualt choice fo R-INLA package. From data is left out one single observation and so that the Validation set is $y_v = y_{-i}$ and the Assessement set is a $y_a = y_i$ the rest of the observations. Two KPI are assumed to be representative:

- CPO conditional predictive ordinate (pettit, 1990): $CPO_i = \pi(y^\star \mid y_v)$
- PIT probability integral tranform (dawid, 1984): $PIT_i = \pi(y^\star < y_i \mid y_v)$

These quantities are used by default by setting control options in the inla(control.compute = list()) list object by setting them equal o TRUE. Inla also provides an inner method to authomatically handlee failing in computing those two quantities, leadind to values of 1 when predictions are not reliable and the ipposite for 0.Moreover the empirical distribution of the PIT can be used to asses predictive performance: if it is Uniform, so there are not values that strongly differ from the others then the model is correctly checked. Otherwise if the dostribtuon almost approxiamtes any of the other possibles then the Cross validation assessement prediction has led incorrectly predict the “out of the bag” validation sample.

Posteerior checking method exploits a full cross validation where $y_a = y_v$ and it is called predictive checks. Th assessement set now is equal to the validation set,a s a consequence all the observation are evaluated twice. 4 quantities are driver to model estimate quality:

- the *posterior predictive distribution*: $\pi(y^\star \mid y) = \int \pi(y^\star \mid \theta_i)\pi(\theta_i \mid y)\mathrm{d}\theta_i$ which is the likelihood of a replicate observation. When values are small that indicates that are those values are coming from tails, since the area under the curve (i.e. probability) is less. If this happens for many observation then outliers are driving the model leading to poor estimates
- the *posterior predictive p-value* whose math expression is:$\pi(y^\star \leq y_i \mid y)$ for which values near to 0 and 1 indicates poor perfomances.
- *Root Mean Square Predictive Error RMSE*: $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i^\star)^2}$
- R^2

R-INLA has already antiticipated in chapter 4 section4.3.2 have designed function to compute statistics on posterior distribution as inla .pmarginal() returning the cumulative density distribution.

5.9 Prior Specification

Chapter 6

SPDE approach

Observations in the spatial problem setting are considered as realizations of a stationary, isotropic unobserved GP $w(s)$ that we aim to estimate (5.1). Before approaching the problem with SPDE, GPs were treated as multivariate Gaussian densities and Cholesky factorizations were applied on the covariance matrices and then fitted with likelihood. Matrices in this settings were very dense and they were scaling with the order of $O\left(n^3\right)$, leading to obvious big-n problem. The breakthrough, came with Lindgren et al. (2011) that proves that a stationary, isotropic GP with Matérn covariance can be represented as a GMRF using SPDE solutions by finite element method (Krainski, 2019). In other words given a GP whose covariance matrix is Q, SPDE can provide a method to approximate Q without computational constraints. As a matter of fact SPDE are equations whose solutions are GPs with a chosen covariance function focused on satisfying the relationship SPDE specifies. Benefits are many but the most important is that the representation of the GP through a GMRF provides a sparse representation of the spatial effect through a sparse precision matrix $Q^{-}1$. Sparse matrices enable convenient inner computation properties of GMRF that can be exploited with INLA. Bayesian inference on GMRF can take advantage of lower computational cost because of these properties stated before leading to a more feasible big-O $O\left(n^{3/2}\right)$. The following chapter will provide a intuition on SPDE oriented to practitioners.

The chapeter once again will follow the track of Krainski & Rubio (2019) and Blangiardo and Cameletti (2015) works, together with the street-opener paper from Miller et al. (2019) as compendium. SPDE might be complex for those who are not used to applied mathematics and physics making it difficult not only to grab the concept, but also to find its applications. One more obstacle regards SPDE software implementation, since without deep technical expertise it might be difficult to customize code with the aim to extend the methodology to different models. For a gentle introduction on what a SPDE is from a mathemathical perspective a valuable reference is Miller et al. (2019) in secction 2.1, then also its application to Matérn in 2.3.

6.1 Set SPDE Problem

Given the statistical model already encountered in chapter 5.4:

$$y\left(\mathbf{s}_{i}\right)=\mathbf{x}\left(\mathbf{s}_{i}\right)^{\prime} \beta_{j}+w(\mathbf{s})+\varepsilon\left(\mathbf{s}_{i}\right)$$

where $\eta(\mathbf{s}_i) = g(\mathbf{x}\left(\mathbf{s}_{i}\right)^{\prime} \beta_{j})$ is the linear predictor, whose link function $g(\cdot)$ is identity (can be also extended to GLM), where $w(\mathbf{s})$ is a Gaussian Process with mean structure 0 and $C(\cdot)$ covariance structure (where Q is the covariance matrix and Q^{-1} precision matrix). Then $w(s) \sim MV\mathcal{N}(0, Q_{i,j}^{-1})$ and where $\varepsilon(\mathbf{s}_i)$ is white noise error such that $\varepsilon(\mathbf{s}_i) \sim \mathcal{N}\left(0, \tau^{2}\right)$. Comprehending w in the model brings two major issues, specify a covariance function for observations as well as how to fit the model. Among all the possible reachable solutions including the SPDE, the common goal is to define covariance function between locations by approximating the precision matrix Q^{-1}, since they are an effective tool to represent covariance function as in section 4.1. For those reasons SPDE approach implies finding an SPDE whose solution have the precision matrix, that is desired for w. Lindgren et al. (2011) proves that an approximate solution to SPDE equations is to represent w as a sum of basis function multiplied by coefficients (2019). Moreover the basis function coefficients are in reality a GMRF (for which fast method comptations already exists).

6.2 SPDE within R-INLA

First point addresses the assumption that a GP with Matérn covariance function and $\nu > 0$ is a solution to *SPDE* equations. Second point addressed the issues of solving SPDE when grids are irregular, as opposite with the one seen in first point (regular grid for irregular distribution. In here comes FEM used in mathematics and engineering application with the purpose to solve differential equations. Notation is kept coherent with the one for the previous chapter.

6.3 First Point Krainsky Rubio TOO TECHNICAL

A regular 2D grid lattice is considered with infinite number of location points as vertices.

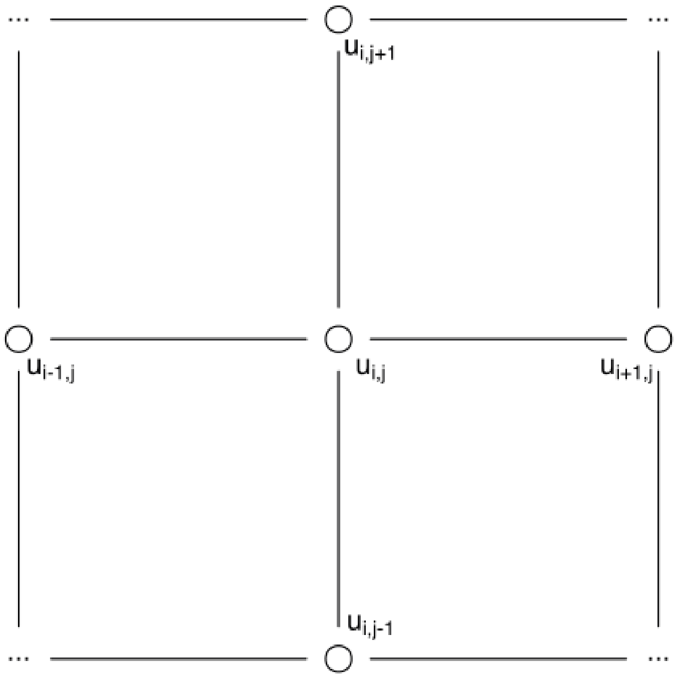


Figure 6.1: lattice 2D regular grid

Chapter 7

Exploratory Analysis

Data flows out the REST API end point ∗/complete in a .json format. Data can be filtered out On the basis of the options set in the API endpoint argument body. Some of the options supplied to the API, as in section **??**, might regard the real estate city interested, npages as the number of pages to scrape, type as the choice between rental of selling market. Since to the analysis extent data should come from the same geographic area API, city and filter parameters are kept permanent (e.g. Milan rental real estate within "circonvallazione" approximated geo-borders). As a consequence a dedicated endpoint . thesis parameter is passed in the argument body. By setting the option equal to to TRUE the API caller requests thesis data. In other words the latter option under the hood secures to specify to the API an already composed query url to be passed to the scraping endppoint, which corresponds to precise zones imposed while searching for advertisemnts on Immobiliare.it. To help figure out the idea behind the operation it can be thought as refreshing everyday the same immobiliare.it url on their website looking for accomodations within a specified zone. Parameters specified are also npages = 120, leading to to 3000 observations. The ∗ refers to the EC2 public DNS.

http://∗/complete/120/milano/affitto/. thesis =true

As a further data source is available a mondgoDB ATLAS cluster which, because of the scheduler, stocks daily .csv information from Milan real estate.

Credentials have to be supplied. For run time reasons also related to the bookdown files continuous building the API endpoint is not called and code chunks outputs are cached due to heavy computation. Instead data is extracted from the MongoDB cluster. A summary table of the columns involved is presented with the goal to introduce the reader to API incoming data. Data needs some heavy preprocess steps to get modeled which is briefly covered in the data preparation part in section 7.1. Data coming from the /complete enndpoint has a geo-statistical spatial component and consequently a spatial representation of the dataset is showed. One further plot points out that geographic coordinates are non-linearly related 7.2 to the price response variable so dedicated techniques are required. Exploration starts with factor counts evidencing a "Bilocale" prevalence which is then compared to other cities. This suggest some critical Milan real estate market demand information and consequently reflections on the offer. Heating and cooling systems, two of the covariates extracted, are grouped and then arranged by descending order prevalence. They both do not display any significative price change but they bring to the surface an important environmental concern. The same is done by highlighting ridges distribution for other two newly engineered covariates. Data displays bimodality in prices distribution for different n-roomed accommodations and the model should take account of the behavior. Then a piece-wise linear regression is fitted for each n-roomed accommodation sub-group whose single predictor is the square meter footage. The analysis emphasize some valuable economic consequences both for investors interested into property expansions and for tenants that are planning to partition single properties into rentable sub-units. The previous analysis brings along a major question which regards the most valuable properties per single square meter surface and a answer based on data is given. Then a further log linear regression setting is proposed to evaluate the impact of some other presumably important covariate. A Tie Fighter plot displays for which coefficient, associated to each dummyied predictor, there are surprisingly high prices compared to the effect of the square meter footage expansion. A partial conclusion is that having 2 or 3 bathrooms truly pays an extra monthly gain, also caused by the number of tentants the accomodations could host. Then missing assessement and imputation takes place. At first is made a brief a revision of missing and randomnes by Little and Rubin (2014), then theory is applied by visualizing missing in combination with heat-map and co-occurrence plot. Combined missing observation test is able to detect whether data is missing because of inner scraping faiilures or simple rarity in data appereance. Then for the observations that passed the test imputation is made through INLA posterior expectation. That is the case of data lost in predictors so the missing covariates (*condominium*) are brought into a model as response variable whose this time predictors are explanatory ones. Through a method specified within the INLA function the posterior statistics are computed and then finally imputed in the place of missing ones.

Visualisations are done with ggplot2 in a Tidyverse approach. Maps are done with ggplot2 too and Leaflet, together with its extensions. A preliminary API data exploratory analysis evidences 34 covariates and 250 rows, which are once again conditioned to the query sent to the API. Immobiliare.it furnishes many information regarding property attributes and estate agency circumstances. Data displays many NA in some of the columns but georeference coordinates, due to the design of scraping functions, are in any case present.

name	ref
ID	ID of the apartements
LAT	latitude coordinate
LONG	longitude coordinate
LOCATION	the complete address: street name and number
CONDOM	the condominium monthly expenses
BUILDAGE	the age in which the building was contructed
FLOOR	the property floor
INDIVSAPT	indipendent property type versus apartement type
LOCALI	specification of the type and number of rooms
TPPROP	property type residential or not

STATUS	the actual status of the house, ristrutturato, nuovo,
HEATING	the heating system Cen_Rad_Gas (centralizzato a
AC	air conditioning hot and cold, Autonomo, freddo/ca
PUB_DATE	the date of publication of the advertisement
CATASTINFO	land registry information
APTCHAR	apartement main characteristics
PHOTOSNUM	number of photos displayed in the advertisement
AGE	real estate agency name
LOWRDPRICE_ORIGINAL_PRICE	If the price is lowered it flags the starting price
LOWRDPRICE_CURRENT_PRICE	If the price is lowered it flags the current price
LOWRDPRICE_PASSED_DAYS	If the price is lowered indicates the days passed sinc
LOWRDPRICE_DATE	If the price is lowered indicates the date the price ha
ENCLASS	the energy class according to the land registers
CONTR	the type of contract
DISP	if it is still avaiable or already rented
TOTPIANI	the total number of the building floors
PAUTO	number of parking box or garages avaibable in the p
REVIEW	estate agency review, long chr string
HASMULTI	it if has multimedia option, such as 3D house vitual
PRICE	the monthly price <- response
SQFEET	square meters footage
NROOM	the number of rooms in the house, and their types
TITLE	title of published advertisement

7.1 Data preparation

Data needs to undergo to many previous cleaning preprocess steps, this is a forced stage since API data comes in human readable format, which is not prepared to be modeled. Cleaning steps mainly regards:

- encoding from UTF-8 to Latin due to Italian characters incorrectly parsed.
- *floors* covariate needs to be separated by its *ascensore* and *accdisabili* components, adding 2 more bivariate covariates.
- *locali* needs to be separated too. 5 category levels drain out: *totlocali*, *camereletto*, *altro*, *bagno*, *cucina*. *nroom* is a duplicate for *totlocali*, so it is discarded.
- *aptchar* is a character strign column that contains a various number of different features per house. The preprocess steps include cleaning the string from unnecessary characters, then finding the whole set of unique elements across the character column by splitting on a regex pattern, in the end recoding newly created bivariate columns "yes" or "no" accoeding to a matching pattern whether the feature appears in the string not. A slice from the API output APTCHAR is:

fibra ottica videocitofono impianto di allarme porta blindata reception balcone portiere intera giornata impianto tv centralizzato parzialmente arredato esposizione doppia

7.1.1 Maps and Geo-Visualisations

Geographic coordinates can be represented on a map in order to reveal first symptoms of spatial autocorrelation. Observations are spread almost equally throughout the surface even though the response var *price* indicates unsurprisingly that higher prices are nearer to the city center. The map in figure @ref(fig:leaflet_visuals) is a leaflet object, which needs to be overlapped with layers indicating different maps projections. This is interactive in the .html version, and static is proposed in the .pdf output version. The map object takes a input the latitute and longitude coordinates coming from THE API, and they do not need any CRS (Coordinate Reference System) projection since leaflet can accept the data type.

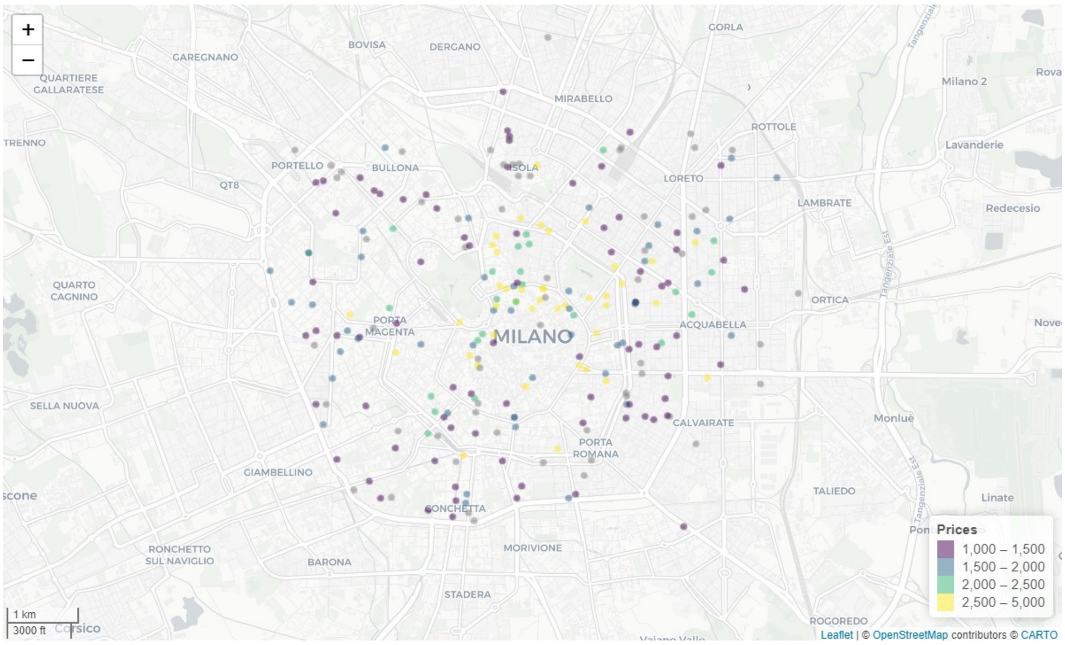


Figure 7.1: Leaflet Map

Predictors, in this case latitude and longitude appear to have nonlinear relationships with the outcome price. The relationship appears to be Gaussian whose mean points to the city center, red dashed line represent latitude and longitude coordinates for the Dome of Milan. Non linearities can be treated with regression splines

ggplot2 visualzitaion matt dancho inspiration::

7.2 Counts and First Orientations

Arranged Counts for categorical columns can give a sense of the distribution of categories across the dataset suggesting also which predictor to include in the model. The visualization in figure 7.3 offers the rearranged factor *TOTLOCALI*. Bilocali are the most common option for rent, then trilocali comes after. The intuition behind suggests that Milan rental market is oriented to “lighter” accommodations in terms of space and squarefootage. This should comes natural since Milan is both a vivid study and working area, so short stayings are warmly welcomed.

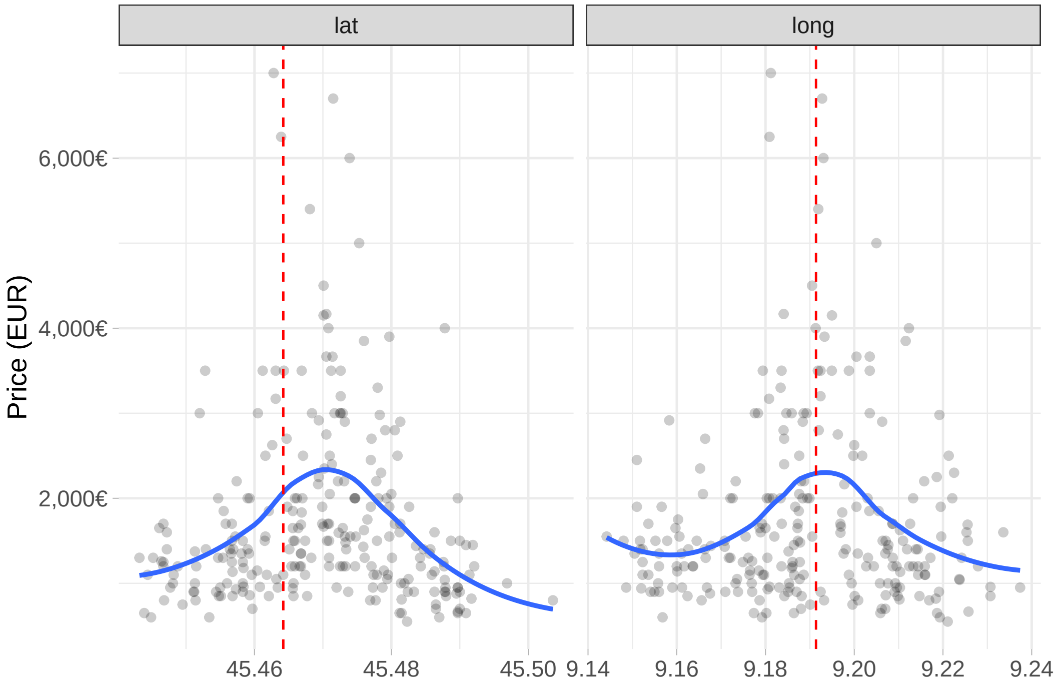


Figure 7.2: Non Linear Spatial Relationship disclosed

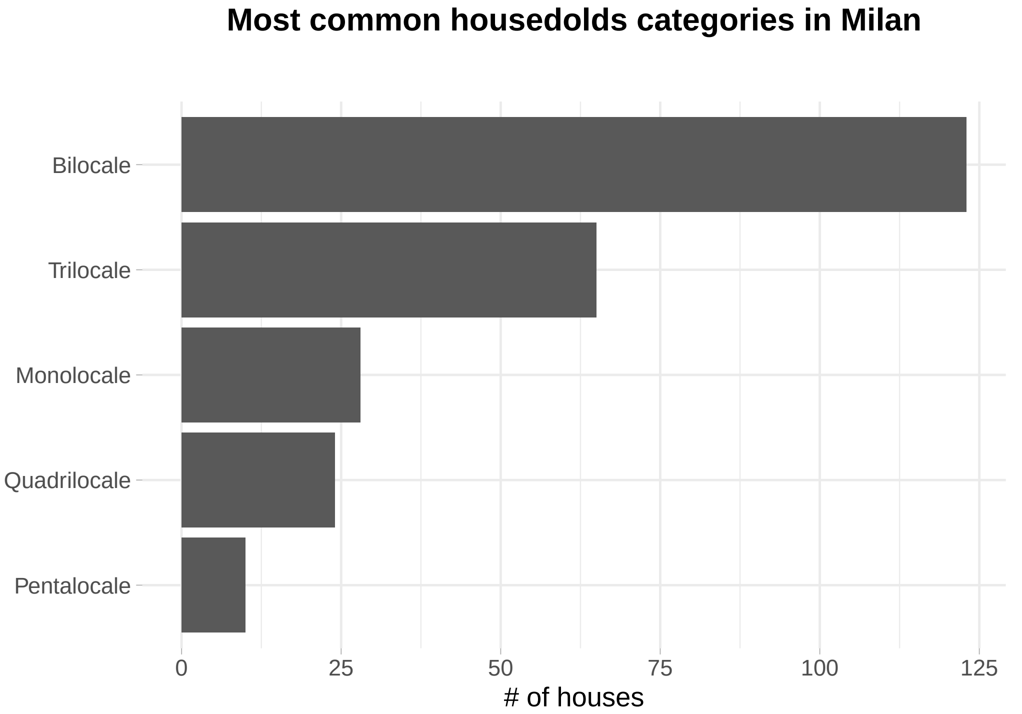


Figure 7.3: Most common housedolds categories

Two of the most requested features for comfort and livability in rents are the heating/cooling systems installed. Moreover rental market demand, regardless of the rent duration, strives for a ready-to-accomodate offer to meet clients needs. In this sense accomodation coming with the newest and most techonological systems are naturally preferred with respect the contrary. x-axis in figure 7.4 represents log_10 price for both of the two plots. Logarithmic scale is needed to smooth distributions and the resulting price interpretation have to considered into relative percent changes. Furthermore factors are reordered with respect to decreasing price.
y-axis are the different level for the categorical variables recoded from the original data due to simplify lables and to hold plot dimension. Moreover counts per level are expressed between brackets close to their respective factor. The top plot displays the most prevalent heating systems categories, among which the most prevalent is “Cen_Rad_Met” by far. This fact is extremely important since metano is a green energy source and if the adoption is wide spread and pipelines are well organized than it brings enormous benefit to the city. As a consequence one major concern regards that for many years policies have been oriented to reduce vehicles emission (euro1 euro2...) instead of focusing on house emissions. This was also a consequence of the lack of house data especially in rural areas. According to data there are still a 15% portion of houses powered by oil fired. Then in bottom plot Jittering is then applied to point out the number of outliers outside the IQR (Inter Quantile Range) .25 and their impact on the distribution. A first conclusion is that outliers are mainly located in autonomous systems, which leads of course to believe that the most expensive houses are heated by autonomoius heating systems. Indedd in any case this fact that does not affect monthly price. The overlapping IQR signifies that the covariates levels do not impact the response variable.

this visualization intersects allows to discover bimodality in the response variable. Log scales was needed since they are all veru skewd and log scale then is needed also in the model.

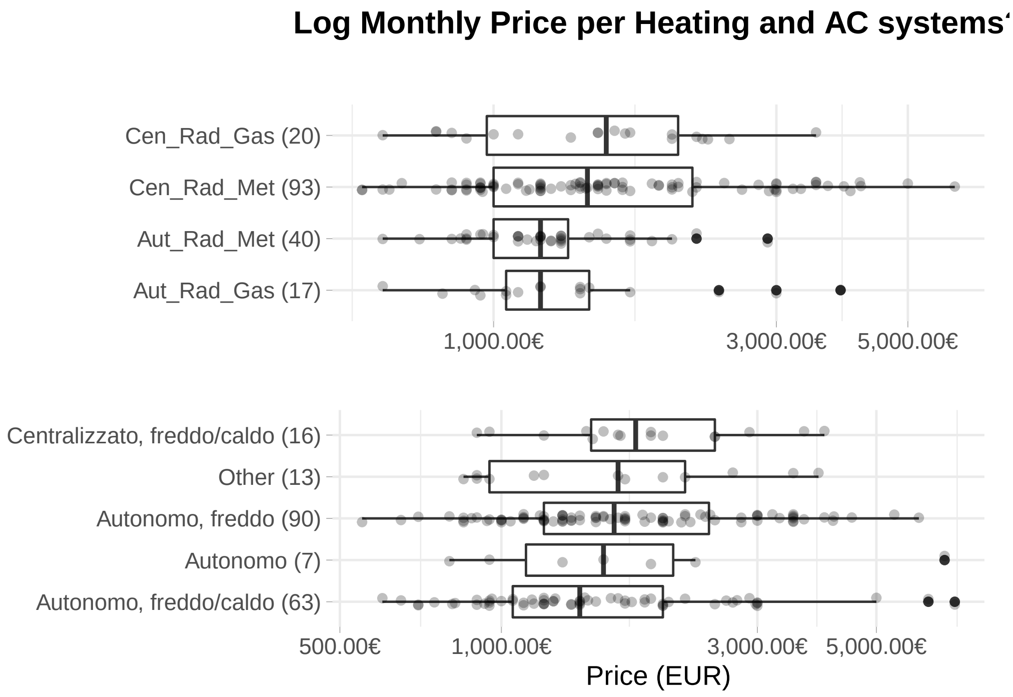
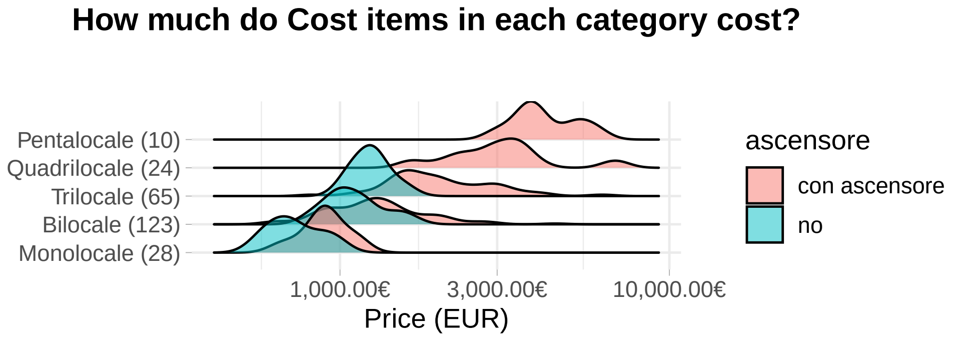
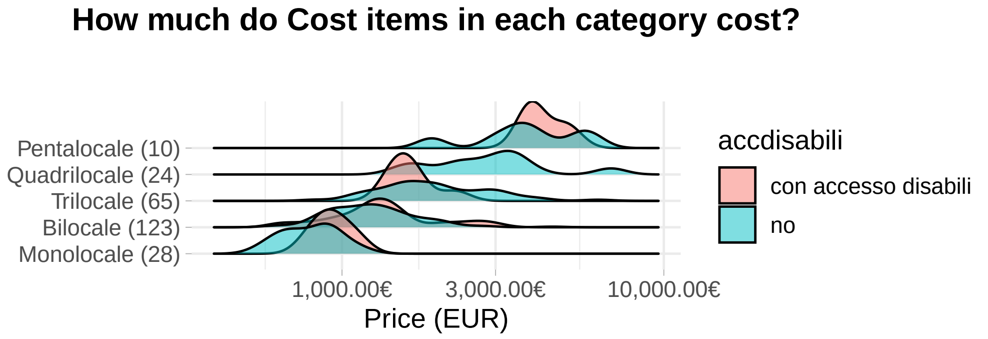


Figure 7.4: Log Monthly Price per Heating/Cooling system?

(qui ci puoi mettere a confronto per variabile bianria, così vedi cosa includere nel modello esempio sotto dove commentato,)





What it might be really relevant to research is how monthly prices change with respect to house square footage for each house configuration. The idea is to asses how much adding a further square meter affetcs the monthly price for each n-roomed flat. One implication is how the property should be developed in order to request a greater amount of money per month. As an example in a situation in which the household has to lot its property into different sub units he can be helped to decide the most proficient choice in economic terms by setting ex ante the square footage extensions for each of the sub-properties. A further implication can regard economic convenience to enlarge new property acquisitions under the expectation to broadened the square footage (construction firms). Some of the potential enlargements are economically justified, some of the other are not. The plot 7.5 has two continuous variables for x (price) and y (sqfeet) axis, the latter is log 10 scaled due to smoothness reasons. Coloration discretizes points for the each j household rooms totlocali. A sort of overlapping piece-wise linear regression (log-linear due to transformation) is fitted on each totlocali group, whose response variable is price and whose only predictor is the square footage surface (i.e. $\log_{10}(\mathbf{price_j}) \sim +\beta_{0,j} + \beta_{1,j}\mathbf{sqfeet_j}$). Five different regression models are proposed in the top left. The interesting part regards the models slopes $\hat{\beta}_{1,j}$. The highest corresponds to “Monolocale” for which the enlargement of a 10 square meters in surface enriches the apartment of a 0.1819524% monthly price addition. Almost the same is witnessed in “Bilocale” for which a 10 square meters extension gains a 0.1194379% value. One more major thing to notice is the “ensamble” regression line obtained as the interpolation of the 5 plotted ones. The line suggests a clear slope descending pattern (logarithmic trend) from Pentalocale and beyond whose assumption is strengthened by looking at the decreasing trend in the $\hat{\beta}_1$ predictor slopes coefficients. Furthermore investing into an extension for “Quadrilocale” and “Trilocale” is *coeteris paribus* an interchangeable economic choice.

In table (...) resides the answer to the question “which are the most profitable properties per month in terms of the price per square meter footage ratio”. The covariate floor together with the totpiani are not part of the model, in-

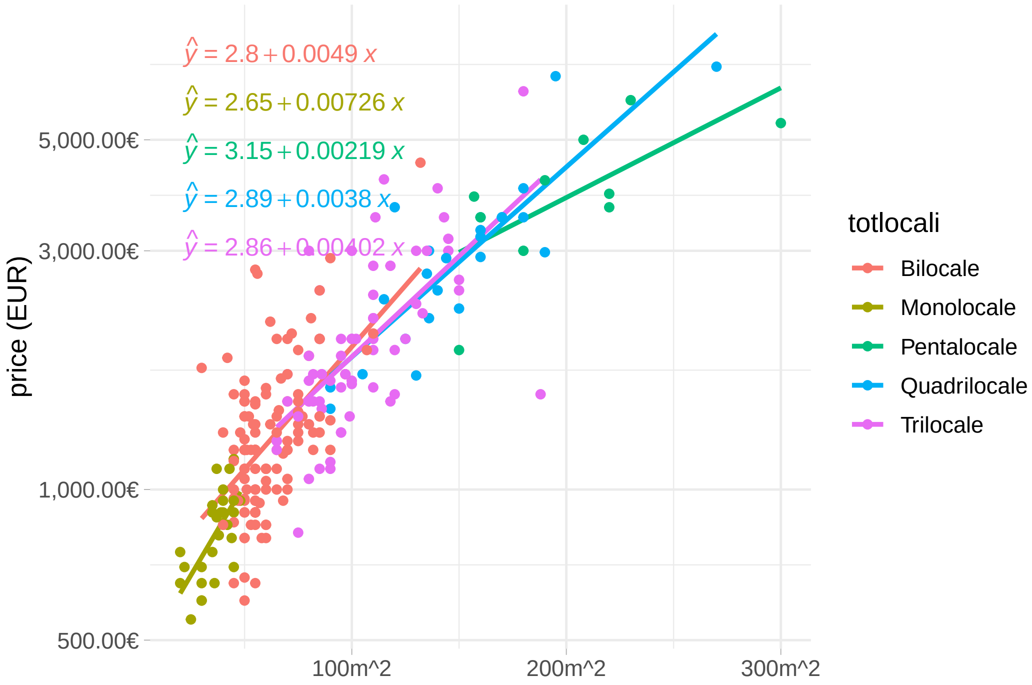


Figure 7.5: Monthly Prices change wrt square meters footage in different n-roomed apt

deed they can explain the importance and the height of the building justifying extraordinary prices. The first 4 observations are unsurprisingly “Bilocale”, the spatial column location, not a regressor, can lend a hand to acknowledge that the street addresses point to very central and popular zones. The zones are, first City Life, second Brera and third Moscova, proving that in modeling real estate rents the spatial component is fundamental , even more in Milan.

location	totlocali	price	sqfeet	floor	totpiani	abs_pri
viale cassiodoro 28	Bilocale	1750	30	9 piano	10 piani	58.333
via della spiga 23	Bilocale	2750	55	2 piano	4 piani	50.000
corso giuseppe garibaldi 95	Bilocale	2700	56	2 piano	5 piani	48.214
piazza san babila C.A.	Bilocale	1833	42	4 piano	4 piani	43.642
ottimo stato piano terra, C.A.	Trilocale	3000	80	Piano terra	3 piani	37.500
via federico confalonieri 5	Monolocale	750	20	1 piano	3 piani	37.500

Then as a further point it might be important to investigate a linear model whose response is price and whose covariates are the newly created abs_price and some other presumably important ones e.g. floor, bagno, totpiani. The model fitted is log2(price) ~ log2(abs_price) + bagno + floor + totpiani. The plot in figure 7.6 has the purpose to demonstrate how monthly price is affected by covariates conditioned to their respective square meter footage. The interpretation of the plot starts by fixing a focus point on 0, which is the null effect highlighted by the red dashed line. Then the second focus is on house surface effect (i.e. House Surface (doubling) in the plot, the term log2(abs_price) has been converted to more familiar House Surface (doubling)), which contributes to increase the price of an estimated coefficient of ≈ .6 for each doubling of the square meter footage. Then what it can be noticed with respect to the two focus points are the unusual effects provoked by the other predictors t the right of the house surface effect and to the far left below 0. “2 and 3 bagni” are unusually expensive with respect to the square meter footage increment, on the other hand” al piano rialzato” and “al piano terra” are undervalued with respect to their surface. The fact that 2 and 3 bathrooms can guarantee a monthly extra check is probably caused to a minimum rent plateau requested for each occupant. the number of bathrooms are a proxy to both house extension since normally for each sleeping room there also exist at least 1 bathroom as well as prestigious houses dispose of more than 1 toilette services. So the more are the occupants regardless of the square meter footage dedicated to them, the more the house monthly returns,

it can be noticed is that ultimo piano, otgether with 2 abagni ad 3 bagni are unusually expensive with respect to their proper square meter footage. On the other hand the piano rialzato and piano terra are unusually undervalued given their surface.

In other words the to help with the interpretation. The fact that 2 and 3 bathrooms can guarantee a monthly extra check is probably caused to a minimum rent plateau requested for each occupant. So the more are the occupants regardless of the square meter footage dedicated to them, the more the house returns. The conclusion

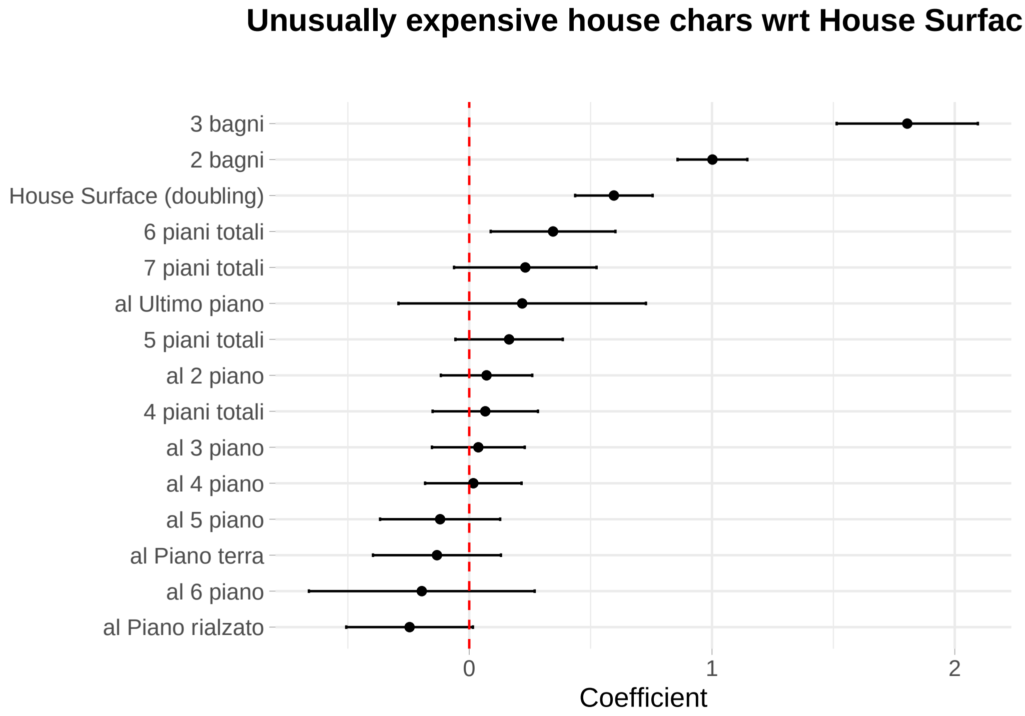


Figure 7.6: Coefficient Tie fighter plot for the linear model: log2(price) ~ log2(abs_price) + condom + other_colors

7.3 Text Mining in estate Review

The word network in figure **??**fig:WordNetworkgraph) tries to summarize relevant information from real estate agency review into each advertisement. avg_totprice expresses the sum of the price per month plus the condominium in order to fully integrate inner property characteristics together with building exclusivity. Tokenized words are then filtered with “stopwords-iso” italian dictionary. Nodes associated with hotter colours are also associated to more expensive in and out-house characteristics. The size of nodes keeps track of the number of reviews in which the specific word appears. A table of the most common words can help highlight both the real estate jargon as well as words that brings up house values.

word	count	reviews	avg_totprice
bagno	249	192	1888.622
cucina	247	190	2088.814
ingresso	194	173	1964.062
soggiorno	182	159	1872.500
camera	200	158	1936.945
piano	197	157	1982.234
arredato	184	152	1744.614
composto	158	146	1758.911
riscaldamento	171	144	1877.404
zona	282	139	1930.213

Furthermore it is possible to grossly divide the plot in figure **??**fig:WordNetworkgraph) into 3 sub-groups of nodes, each of which addresses a specific part of the house comprehensive evaluation. In the far right side of the plot are considered the external appliances like neighbor stores, subway stations and services and are associated to mean prices. The correspondent number of reviews are not justifying by any type of price increasing effect. Whereas slightly moving the view to the left, the area centered in portineria evidences a sub-groups of nodes associated to relatively higher avg-totprice. Some of them are servizio signorile palazzo. The previous set of nodes indicates services that are proper to the building can lead to some sort of extra payment. Then still moving Possiamo immaginare di dividere il network in 3 raggruppamenti di nodi, ognuno dei quali parla di uni specifico tema. nella parte alta sinistra csi parla delle circostanze estenre dell'appartamente, i negozi i mezzi serizi la metri, i prezzi evidenzziati dal colore nei nodi sono neutri, indicando che non impattano il prezzo in maniera significativa. poco più sotto è possibile vedere un altro centroide verso il quale puntano una serie di edges peritenti che riguardano i servizi interni al building come la portinerua, l'ingresso, il palazzo. in questo caso i colori sono più caldi e i servizi sembrano essere pagati di più. successivamente sosptandoci veros il centro del'network si nota un nodo di gravità attorno alquale si trovamo molti outgoing edges, che riscaldamento. Attorno a riscaldamento che vista la grandezza ricorre spesso nelle recensioni, si sviluppano tutti i servizi non descritti da immobiliare all'interno della casa, insiema a tutte le caratteristiche cbe distinguono la casa revisionata dalle altre. i colori degradano spostandosi da sinistra verso destea, accanto a riscaldmaento si nota un cluster che associati a prezzi minoro come spese condominiali e arredato arredato. nel caso delle spese condominiali i cluster sono associati a prezzi minori perchè il prezzo del conominio spesso non è commisurato al prezzo nè al prestigio dell'appartamento. Speso infatti include costi variabili come utenze gas e luce, o acqua che vengono scontati con prezzi più bassi di affitto. la somma di condominio e prezzo offrirebbe un panorama più chiaro.

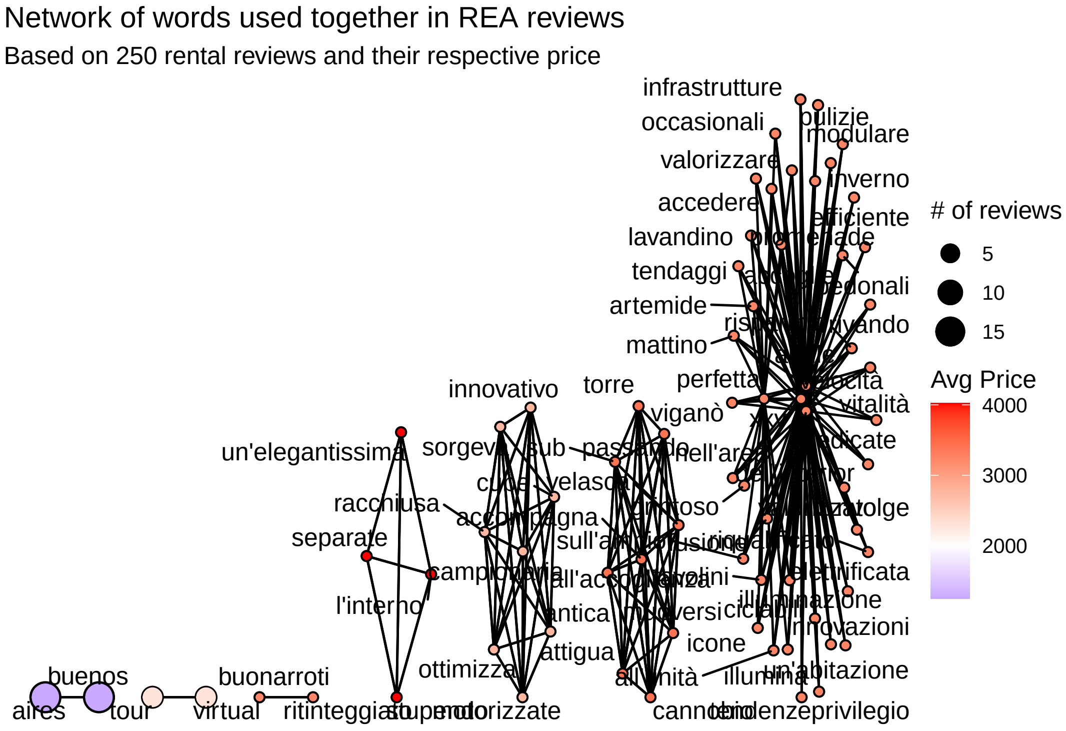


Figure 7.7: Word Network Graph for 250 Estate Agencies Review

7.4 Missing Assessement and Imputation

As already pointed out some data might be lost since immobiliare provides the information that in turn are pre filled by estate agencies or privates through standard document formats. Some of the missing can be reverse engineered by other information in the web pages e.g. given the street address it is possible to trace back the lat and long coordinates. Some other information can be encountered in .json files hidden inside each of the single web pages. The approach followed in this part is to prune redundant data and rare predictors trying to limit the dimensionality of the dataset.

7.4.1 Missing assessement

The first problem to assess is why information are missing. As already pointed out in the preliminary part as well as in section **??** many of the presumably important covariates (i.e. price lat, long, title ,id ...) undergo to a sequence of forced step inside scraping functions with the aim to avoid to be lost. If at the end of the sequence covariates are still missing, the correspondent observation is not considered and it is left out of the resulting scraped dataset. The choice originates from empirical missing patterns suggesting that when important information are missing then the rest of the covariates are more likely to be missing to, as a consequence the observation should be discarded. The missing profile is crucial since it can also raise suspicion on the scraping failures. By Taking advantage of the missing pattern in observations the maintainer can directly identify the problem and derivatives and immediately debug the error. In order to identify if the nature of the pattern a revision of missing and randomness is introduced by Little and Rubin (2014). Missing can be devided into 3 categories:

- *MCAR* (missing completely at random) likelihood of missing is equal for all the information, in other words missing data are one idependetn for the other.

- *MAR* (missing at random) likelihood of missing is not equal.
- *NMAR* (not missing at random) data that is missing due to a specific cause, scarping can be the cause.

MNAR is often the case of daily monitoring clinical studies (Kuhn and Johnson, 2019), where patient might drop out the experiment because of death and so all the relating data starting from the death time +1 are lost. To identify the pattern a *heat map* plot 7.8 clarifies the idea:

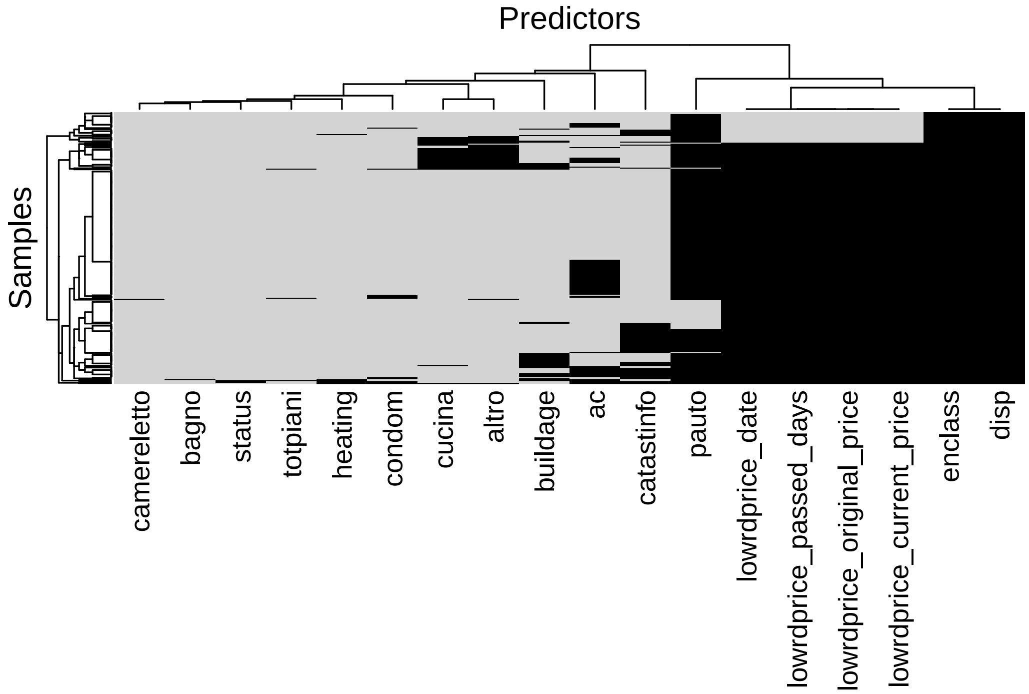


Figure 7.8: Missingness Heatmap plot

Looking at the top of the heat map plot, right under the "Predictor" label, the first tree split divides predictors into two sub-groups. The left branch considers from *TOTPIANI* to *CATASTINFO* and there are no evident patterns. Then missingness can be traced back to MAR. Imputation needs to be applied up to *CONDOM* included, the others are discarded due to rarity: i.e. *BUILDAGE*: 14% missing, *CATASTINFO*: 21% and *AC*: 24%. Moreover *CUCINA* and *ALTRO* are generated as "childred" of the original *LOCALI* variable, so it should not surprise that their missing behavior is similar ,whose prevalence is respectively 13% and 14%, for that reason are discarded. In the far right hand side *ENCLASS* and *DISP* data are completely missing and a pattern seems to be found. The most obvious reason is a scraping fail in capturing data. Further inspection of the API scraping functions focused on the two covariates is strongly advised. From *LOWRDPRICE.* covariates gorup class it seems to be witnessing a missing underlining pattern NMAR which is clearer by looking at the co_occurrence plot in figure 7.9. Co-occurrence analysis might suggest frequency of missing predictor in combination and *LOWRDPRICE.* class covariates are displaying this type of behavior. *PAUTO* is missing in the place where *LOWRDPRICE.* class covariates are missing, but this is not happening for the opposite, leading to the conclusion that *PAUTO* should be treated as a rare covariate MAR, therefore *PAUTO* is dropped. After some further investigation on *LOWRDPRICE.*, the group class flags when the *PRICE* covariate is effectively decreased and this is unusual. That is solved by grouping the covariate's information and to encode it as a two levels categorical covariate if lowered or not. Further methods to feature engineer the *LOWRDPRICE.* class covariates can be with techniques typical of profile data, further references are on Kuhn and Johnson (2019).

7.4.2 Covariates Imputation

A relatively simple approach to front missingness is to build a regression model to explain the covariates that have some missing and plug-back-in the respective estimates (e.g. posterior means) from their predictive distributions Little and Rubin (2014). This approach is fast and easy to implement in most of the cases, but it ignores the uncertainty behind the imputed values (Gómez Rubio, 2020). However it has the benefit to be a more than a reasonable choice with respect to the number of computation required, especially with INLA and in a spatial setting. That makes it the first choice method to follow since imputation regards also a small portion of data and predictors. At first it is considered the predictor *condominium* for which some observation are missing.

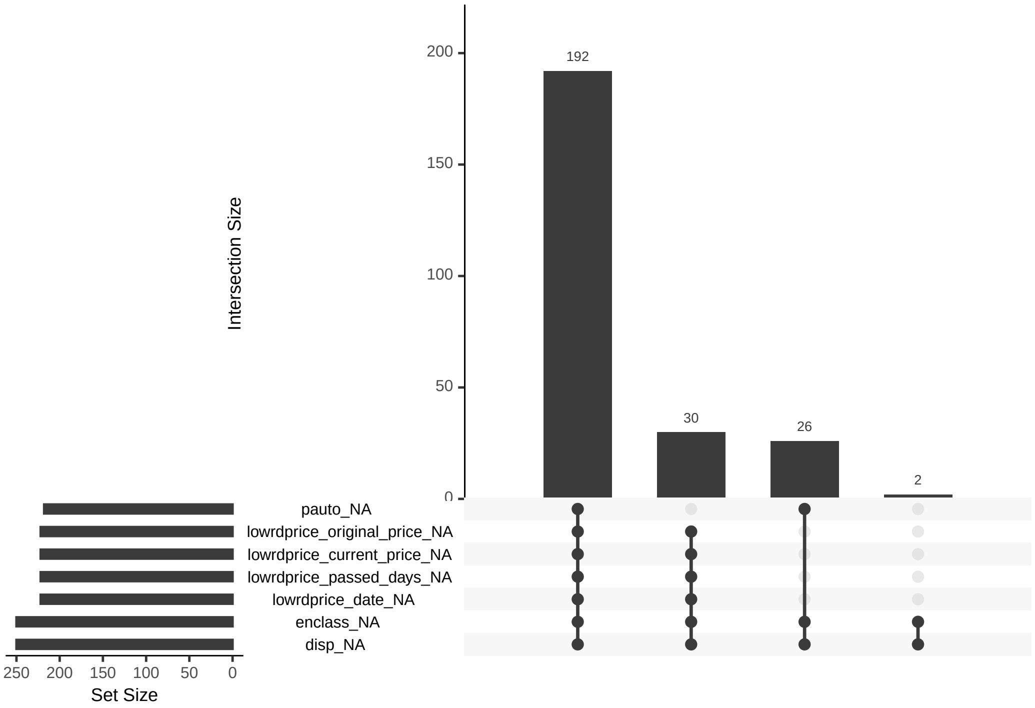


Figure 7.9: Missingness co-occurrence plot

Indices are:

```
##  [1]  19  74  77  90  99 113 116 120 179 249
```

A model is fitted based on missing data for which the response var is condominium and predictors are other important explanatory ones, i.e.condom ∼ 1 + sqfeet + totlocali + floor + heating + ascensore. In addition to the formula in the inla function a further specification has to be provided with the command compute = TRUE in the argument control.predictor. The command compute estimates the posterior means of the predictive distribution in the response variable for the missing points. The estimated posetior mean quantities are then imputeda are in table @red(tab:CondomImputation)

	mean	sd
fitted.Predictor.019	198.11095	19.67085
fitted.Predictor.074	162.96544	13.29456
fitted.Predictor.077	99.38197	32.34108
fitted.Predictor.090	331.73519	33.05035
fitted.Predictor.099	170.54068	12.30267
fitted.Predictor.113	196.61593	15.86545
fitted.Predictor.116	108.40482	20.79689
fitted.Predictor.120	162.86977	25.61622
fitted.Predictor.179	165.03632	20.53485
fitted.Predictor.249	117.24234	30.80290

A further method for imputation has been designed by *Gómez-Rubio, Cameletti, and Blangiardo 2019) miss lit* by adding a sub-model for the imputations to the final model through the inla function. This is directly handled inside the predictor formula adding a parameter in the latent field. However the approach makes the model more complex with a further layer of uncertainty to handle. At first the additive regression model with all the covariates is called including the covariates with missing values. The response variable *PRICE* displays no missing values and the model fitted is:

7.5 Model Specification

7.6 Mesh building

PARAFRASARE The SPDE approach approximates the continuous Gaussian field w_i as a discrete Gaussian Markov random field by means of a finite basis function defined on a triangulated mesh of the region of study. The spatial surface can be interpolated performing this approximation with the inla.mesh.2d() function of the R-INLA package. This function creates a Constrained Refined Delaunay Triangulation (CRDT) over the study region, that will be simply referred to as the mesh. Mesh should be intended as a trade off between the accuracy of the GMRF surface representation and the computational cost, in other words the more are the vertices, the finer is the GF approximation, leading to a computational funnel.

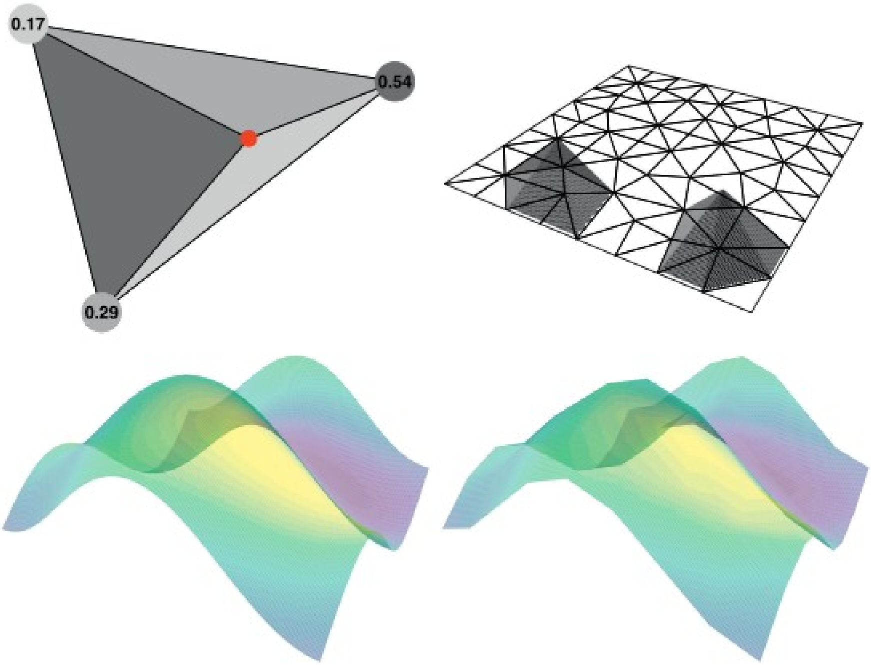


Figure 7.10: Traingularization intuition, Krainski (2019) source

Arguments can tune triangularization through inla.mesh.2d() :

- loc:location coordinates that are used as initial mesh vertices
- boundary:object describing the boundary of the domain,
- offset : argument is a numeric value (or a length two vector) and it is used to set the automatic extension distance. If positive, it is the extension distance in the same scale units. If negative, it is interpreted as a factor relative to the approximate data diameter; i.e., a value of -0.10 (the default) will add a 10% of the data diameter as outer extension.
- cutoff : points at a closer distance than the supplied value are replaced by a single vertex. Hence, it avoids small triangles

- max.edge: A good mesh needs to have triangles as regular as possible in size and shape.
- min.angleargument (which can be scalar or length two vector) can be used to specify the minimum internal angles of the triangles in the inner domain and the outer extension

A convex hull is a polygon of triangles out of the domain area, in other words the extension made to avoid the boundary effect. All meshes in Figure 2.12 have been made to have a convex hull boundary. If borders are available are generally preferred, so non convex hull meshes are avoided.

7.6.1 Shinyapp for mesh assessment

INLA includes a Shiny (Chang et al., 2018) application that can be used to tune the mesh params interactively

The mesh builder has a number of options to define the mesh on the left side. These include options to be passed to functions inla.nonconvex.hull() and inla.mesh.2d() and the resulting mesh displayed on the right part.

7.6.2 BUilding SPDE model on mesh

7.7 Spatial Kriging (Prediction)

QUI INCERTEZZE

Chapter 8

Model Selection & Fitting

8.1 Model Criticism

evaluation of the variables to include in the mode,, assumptions of the model i.e. exchangeablity and independence prior distribution to assign to parameters and hyper parameters.

8.2 Spatial Kriging

8.3 Model Checking

if (models > 2){

```
## Model Selection
```

```
IDEA: proporre due modelli uno più interepretabile
    con distribuzione normale, e un ò'altro con sempre
      exponential family ma con Gamma distribution
      function , ora vedo se riesco a fare tutto . dovrei
      sacrificare applicazione .
}
```

Chapter 9

Shiny Application

with UI build with free tool for front end design ion shiny fomantic-ui[1]. prendi shiny app e rifai interface. in questo blog vedi Hacaton tirato e vincitori blog[2].

Senno app paula moraga che ha già simil modello dentro,

senno flexdashboard paula moraga.

this inspiration[3]

[1] https://fomantic-ui.com/

[2] https://blog.rstudio.com/2020/11/10/the-appsilon-shiny-semantic-pocontest/

[3] https://demo.appsilon.ai/apps/polluter/

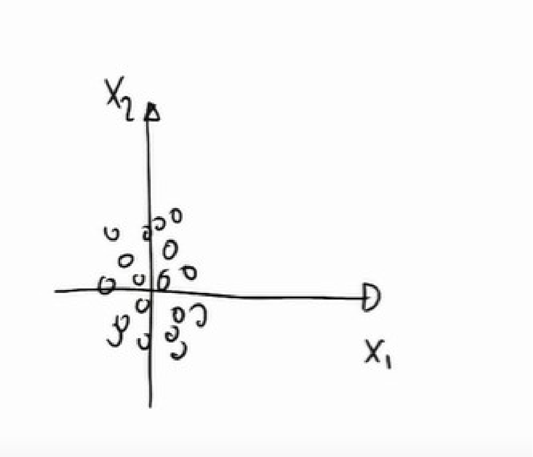
Appendix

9.1 GP basics

Nando de Freitas 1[4]

Nando de Freitas 2[5]

lets say there are a cloud of points represented by two variables x1 and x2. the cloud of points describes a realization of this two variable i.e. height and weight and then you just plot it , you might get measurement like that,



or:

each circle is a mesuduraments. now when we use multivariate gaussian we fit gaussian to data, the process of learning is to fit a gaussian to data, the ultimate goal is to describe the data, the smartest gaussian in the first image is to center the mean in the 0 and the draw a cricle containin all the other observation. Instaed for the second image it is still centering the mean in 0

[4] https://www.youtube.com/watch?v=4vGiHC35j9s&t=164s

[5] https://www.youtube.com/watch?v=MfHKW5z-OOA

but now it is an ellipse describing the variability, the size of the elipse descrube the variability of the data. the center is a vector μ_i that it is beacuse we have two components x_1 and x_2 whhose mean is 0 for each of the other. This is true for all the observation which have two coordinates too x_1 and x_2. in vector notations we have for the mean:

$$\mu = \begin{bmatrix} \mu_{x_1} \\ \mu_{x_2} \end{bmatrix}$$

for each of the points, e.g. for point 1:

$$\mathbf{x_1} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

the can be neagtive positive, the Real numbers, usually we have $\mathbb{R}^2$ extending from - infinity to + infinity, to the power of two because we have 2 dimensions, a Real plane.

any point is gaussian distributed when with mean .. an variance. how we explain covariance, thorugh *correlation.* we do it by correlation with its noraml forms. the covariance is the term that goes insisde the matrices in the upper right of the matrxi we have the expectation of x_1 times x_2, like $\mathbb{E}(x_1 \cdot x_2)$, where the extactation in the gaussian case is the mean which is 0, so the corresponding values is 0. the covariance essentially is the dot product ref dot product[6] of x_1 and x_2 variable, so what happens when you take the dot product of vectors, if for example you take a vector that looks like 1 and 0 and you take the dot product of one other vecto 1 and 0, so that:

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = 1$$

You will end up with 1, recall dot productm first element first vcetor times first element second vectro and second element first vector times second element

[6] https://mathinsight.org/dot_product_matrix_notation

secon vector. So identical vector will get a high dot product value leading to a high similarity measure. Dot product can be indeded as a similarity measure. ... But if you take two different vector as 1 0 and 0 1 then:

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \end{bmatrix} = 0$$

This time the multiplication leads to 0 value, as a matter of fact they are different. They are no similar. IF two points are closed the dot product will be high in 2D. What the covariance should be? if variances are assumed to be 1 then in this case i qould expect to be 0, i.e. covariance matrix is:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{cov_{plot1}}(x_1, x_2)$$

because I can picka poin tin two pointa in this cloud. Suppose i increase x1 then my chance of getting a x2 point that is positive or negative is the samee, knowing somthin about x1 give nothign about x2. no information is proivede. On the other hand i the second plot knowing a positive value of 1 can suggest with a certain probability that x2 will be positive (great proabibility. So some information is provided), e.g.

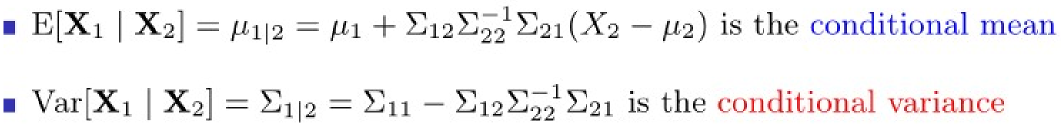
$$\begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix} = \mathbf{cov_{plot2}}(x_1, x_2)$$

Some positive number idicates that i expect a positive inc rease iwhen boht of the two are increasing singularly. thsi is what the correlation, the basis to do linear regresssion and non linear- thei is a bivariate gaussian. If the entri3es are means that they are uncoorellated, if they are non-zero then they are correlated, theby can be both positive or negative (correlatiob)

now lets generate a gassiian distrivution so x_1 and x_2 in 2D and then a third dimension hwere we express probability, this is said joint distribution. So i am going to cu this gaussian at certain point for x_1 and cut a plain rigght thgouth this gauissan imagine to ahava cake and then taka kkniw and cut it.(see the image)

form the man perspective you are goin to see a gaussian distribution, you will be lookong at x_ and you will be seeing a gaussain plot in green. this is the probability of x_1 gievn x_1. also said “conditioned” probabolity. This gaussian has a mean like the one alreasdy seen and this is the center of the gaussian, we can rewrite the mena and variance of the multivariate gaussian describing the cloud of points. sigma are the covaraince martix sigma.

... sigma 1 and signa 2 if you have 1 d varibale the widjth has to be postive, for mulitvariate gaussian equl so here positive definitness: covariance mateix symetric.
... any artibitray variable transposed x time the covarince matrix nedds to be positive. what is the mean of this gaussian i might want to know what is the widht of this gaussian would it be great if there is a formula that guven the cloud of point and likelihood estimation. we coilf obtain the red bell in figure. Compute the green curve how it is done? this requires some work and it is said matrix *invesrion lemma*, this is foudamental for machine learnign. let’s assume it. The theorem says that the the mean fof the gaussian is the mean of x_1 and then some other operation with sigma, see below from paci (miss ref)



- $E[\mathbf{X}_1 \mid \mathbf{X}_2] = \mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}(X_2 - \mu_2)$ is the conditional mean
- $\mathrm{Var}[\mathbf{X}_1 \mid \mathbf{X}_2] = \Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$ is the conditional variance

the theorem says toi ocnsider a multivariate gaussian a vector 1 and a vector 2 each vecto compinent has a mena and a covarianc matrix, this by lemma gives us the expression and the math behind is no tremendous, but it is long. What it is important is to undestand fto go from a joont to a conditional distribution in our case. thats i the value od the theorem.

One background further thing: assume that we have a gaussian variable distribution that we want to sample fromm, we had now ewe are going to do the opposite, before we had poitns and we tried to figure out the curve, now we have the curve and we are gointg to try to rpoduce data. I need to be able to draw sample froma gaussian distribution. i will assume that i have a meachnism that produces a uniform samples, so you have a random numebr generatior with equal probabolity from 0 to 1, I assume a also the cumulative aof a gaussian.

the cumulative of a gaussian is what you get if you syrta summing the area under the curve of the gaussian as you move from the left. value after valure you can plot the cumulative ahead (see figure) the point where there is a flex point is the mean beacuse tha gassias is symmetric. The asymptot is 1 becuase the are under the curve sumes to 1. If i can draw a random number form Uniform and the project it to thre cumulative and then finally projct it back to the gaussian distribution. Inverse cumulative mapping. If oyu do this multiple times you are going to have many sample palced next to the mean and as sparse as the variance. in this process of sampling try to sample a point i froma gaussian that has mean 0 a variance 1, now letes try to draw a point from a gaussian with mean mu and variance sigma. ...

In the multivariate case suppose that we have evctor with two variables how do i draw a vector from a multivariate gasussian with 0 means and plot 1 covarianc ematrix. the theormeem also says that the marginal distribution can be seen by civariance matrix , fist take the men_1 and take upper left element from the covariance matrix obtaining the marginal rpobabiloty for x_1, i.e.

$$\pi(x_1) = \mathbf{N}(\mu_1, \Sigma_{11})\pi(x_2) = \mathbf{N}(\mu_2, \Sigma_{22})$$

then in our problem:

$$\pi(x_1) = \mathbf{N}(0, 1)\pi(x_2) = \mathbf{N}(0, 1)$$

Then for simplicity we can simplyfy by groupign vector into: (vectore exoression multivariate)

I need a wau to take square trotto of matrices, if x come sfroma MVG

35:01–

Bibliography

(2004). Test driven development.

(2014). *Scraping the Web*, chapter 9, pages 219–294. John Wiley & Sons Ltd.

(2018).

(2020). Css.

(2020). Html.

(2020). What is parallel computing.

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., and Iannone, R. (2020). *rmarkdown: Dynamic Documents for R*. R package version 2.3.

Baker, H. C. and Hewitt, C. (1977). The incremental garbage collection of processes. *SIGPLAN Not.*, 12(8):55–59.

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2014). *Hierarchical Modeling and Analysis for Spatial Data*. Chapman and Hall/CRC.

Barney, B. (2020). Introduction to parallel computing.

Bell, B. S., Hoskins, R. E., Pickle, L., and Wartenberg, D. (2006). *International Journal of Health Geographics*, 5(1):49.

Bengtsson, H. (2020a). *future.batchtools: A Future API for Parallel and Distributed Processing using 'batchtools'*. R package version 0.9.0.

Bengtsson, H. (2020b). A unifying framework for parallel and distributed processing in r using futures.

Bengtsson, H. (2020c). A unifying framework for parallel and distributed processing in r using futures.

Bengtsson, H. (2020d). A unifying framework for parallel and distributed processing in r using futures.

Blanchet-Scalliet, C., Helbert, C., Ribaud, M., and Vial, C. (2019). Four algorithms to construct a sparse kriging kernel for dimensionality reduction. *Computational Statistics*, pages 1–21.

Cameletti, M., Lindgren, F., Simpson, D., and Rue, H. (2012). Spatio-temporal modeling of particulate matter concentration through the SPDE approach. *AStA Advances in Statistical Analysis*, 97(2):109–131.

Cantino, A. and Maxwell, K. (2013). Selectorgadget: point and click css selectors. (Accessed on 12/16/2020).

Capozza, D. and Seguin, P. (1996). Expectations, efficiency, and euphoria in the housing market. *Regional Science and Urban Economics*, 26:369–386.

Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. (2020). *shiny: Web Application Framework for R*. R package version 1.4.0.2.

Cisco, W. P. (2020). Cisco annual internet report - cisco annual internet report (2018–2023). (Accessed on 12/28/2020).

Clark, T. E. (1995). Rents and prices of housing across areas of the United States. A cross-section examination of the present value model. *Regional Science and Urban Economics*, 25(2):237–247.

Colin Fay, S. R. (2020).

Contributors (2004). Beautiful soup: We called him tortoise because he taught us. (Accessed on 01/02/2021).

contributors, W. (2020a). Clean url — Wikipedia, the free encyclopedia. Online; accessed 14-December-2020.

contributors, W. (2020b). Denial-of-service attack — Wikipedia, the free encyclopedia. Online; accessed 4-December-2020.

contributors, W. (2020c). Fork (system call) — Wikipedia, the free encyclopedia. Online; accessed 5-December-2020.

contributors, W. (2020d). Hiq labs v. linkedin — Wikipedia, the free encyclopedia. [Online; accessed 2-January-2021].

contributors, W. (2020e). Http client hints — Wikipedia, the free encyclopedia. Online; accessed 4-December-2020.

contributors, W. (2020f). Javascript — Wikipedia, the free encyclopedia. [Online; accessed 28-December-2020].

contributors, W. (2020g). Json — Wikipedia, the free encyclopedia. [Online; accessed 28-December-2020].

contributors, W. (2020h). Url — Wikipedia, the free encyclopedia. Online; accessed 14-December-2020.

contributors, W. (2020i). User agent — Wikipedia, the free encyclopedia. Online; accessed 4-December-2020.

contributors, W. (2020j). Xml — Wikipedia, the free encyclopedia. [Online; accessed 28-December-2020].

Corporation, L. (2017). Letter from linkedin to hiq labs. (Accessed on 01/02/2021).

Corporation, M. and Weston, S. (2020). *doParallel: Foreach Parallel Adaptor for the 'parallel' Package*. R package version 1.0.16.

Cressie, N. (2015). *Statistics for Spatial Data*. Probability and Mathematical Statistics. Wiley-Interscience, revised edition edition.

de Rencontres Mathématiques, C. I. (2018). Håvard rue: Bayesian computation with inla. (Accessed on 12/28/2020), speech originally.

Densmore, J. (2019). Ethics in web scraping.

Doepud (2010). Anatomy of a url. Accessed on 12/14/2020.

Dogucu, M. and Cetinkaya, M. (2020). Web scraping in the statistics and data science curriculum: Challenges and opportunities. *Journal of Statistics Education*, pages 1–24.

Eddelbuettel, D. (2020). Parallel computing with r: A brief review.

Edward G. Black, P. J. R. (2017). hiq labs, inc. v. linkedin corp.: A federal court weighs in on web scraping, free speech rights, and the computer fraud and abuse act | casetext. (Accessed on 01/02/2021).

Google (2020). Presentazione dei file robots.txt - guida di search console.

Gómez Rubio, V. (2020). *Bayesian Inference with INLA*. Chapman and Hall/CRC.

Hadley Wickham, G. G. (2017). *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly Media, 1 edition.

Herath, S. and Maier, G. (2011). Hedonic house prices in the presence of spatial and temporal dynamics. *Territorio Italia - Land Administration Cadastre, Real Estate*, 1:39–49.

immobiliare.it (2020). Condizioni generali - immobiliare.it. (Accessed on 01/02/2021).

Inc., D. (2020a). Get docker.

Inc., D. (2020b). Use volumes | docker documentation. (Accessed on 01/03/2021).

Khalil, S. (2018). *Rcrawler: Web Crawler and Scraper*. R package version 0.1.9-1.

Krainski, E., Gómez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., and Rue, H. (2018). *Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA*. Chapman and Hall/CRC.

Krainski, E. T. (2019). *Advanced spatial modeling with stochastic partial differential equations using R and INLA*. Chapman and Hall/CRC.

Krotov, V. and Silva, L. (2018). Legality and ethics of web scraping.

Krotov, V. and Tennyson, M. (2018). Tutorial: Web scraping in the r language.

Kuhn, M. and Johnson, K. (2019). *Feature Engineering and Selection*. Chapman and Hall/CRC.

LaFrance, A. (2017). The internet is mostly bots - the atlantic. https://www.theatlantic.com/technology/archive/2017/01/bots-bots-bots/515043/. (Accessed on 12/29/2020).

Lancaster, K. J. (1966). A new approach to consumer theory. *Journal of Political Economy*, 74(2):132–157.

Li, H., Li, H., Ding, Z., Hu, Z., Chen, F., Wang, K., Peng, Z., and Shen, H. (2020). Spatial statistical analysis of coronavirus disease 2019 (covid-19) in china. *Geospatial Health*, 15(1).

Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between gaussian fields and gaussian markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(4):423–498.

Ling, Y. and Ling, Y. (2019). Time, space and hedonic prediction accuracy evidence from the corsican apartment market. *The Annals of Regional Science*, page 28.

Lionel Henry RStudio, H. W. R. (2020a). Capture side effects. — safely • purrr. Accessed on 12/05/2020.

Lionel Henry RStudio, H. W. R. (2020b). Create delaying rate settings - rate-helpers. Accessed on 11/05/2020.

Little, R. and Rubin, D. (2014). *Statistical Analysis with Missing Data, Second Edition*, pages 200–220.

Maheedharan, V. (2016). A detailed overview of web crawlers. (Accessed on 12/29/2020).

Malpezzi, S. (2008). *Hedonic Pricing Models: A Selective and Applied Review*, pages 67–89.

Manganelli, B., Morano, P., and Tajani, F. (2013). Economic relationships between selling and rental prices in the italian housing market.

Marta Blangiardo, M. C. (2015). *Spatial and Spatio-temporal Bayesian Models with R-INLA*. Wiley.

Media, A. (2017). What courts have said about the legality of data scraping. (Accessed on 12/29/2020).

Meissner, P. (2020).

Meissner, P. and Ren, K. (2020). *robotstxt: A 'robots.txt' Parser and 'Webbot'/'Spider'/'Crawler' Permissions Checker.* R package version 0.7.7.

Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2.

MERRIAM-WEBSTER (2018). Kitchen-sink | definition of kitchen-sink by merriam-webster. (Accessed on 12/29/2020).

Microsoft (2018). Gateway api - azure architecture center | microsoft docs. (Accessed on 12/23/2020).

Microsoft and Weston, S. (2020). *foreach: Provides Foreach Looping Construct.* R package version 1.5.0.

Miller, D. L., Glennie, R., and Seaton, A. E. (2019). Understanding the stochastic partial differential equation approach to smoothing. *Journal of Agricultural, Biological and Environmental Statistics*, 25(1):1–16.

Moraga, P. (2019). *Geospatial Health Data*. Chapman and Hall/CRC.

NGINX (2014). What is a reverse proxy server? | nginx. (Accessed on 12/22/2020).

Nolis, J. (2020). R docker faster. suddenly r docker images build much… | medium. (Accessed on 12/24/2020).

Paci, L., Beamonte, M. A., Gelfand, A. E., Gargallo, P., and Salvador, M. (2017). Analysis of residential property sales using space–time point patterns. *Spatial Statistics*, 21:149 – 165.

Perepolkin, D. (2019). *polite: Be Nice on the Web*. R package version 0.1.1.

Peterson, R. A. and Merino, M. C. (2003). Consumer information search behavior and the internet. *Psychology & Marketing*, 20(2):99–121.

Rosen, S. (1974). Hedonic prices and implicit markets: Product differentiation in pure competition. *Journal of Political Economy*, 82(1):34–55.

Rue, H. and Held, L. (2005). *Gaussian Markov Random Fields*. Chapman and Hall/CRC.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2):319–392.

Santiago, J. (2020). *plungr: Opinionated Framework for Developing Plumber APIs*. https://ozean12.github.io/plungr.

Shah, T. (2018). *ratelimitr: Rate Limiting for R*. R package version 0.4.1.

SMARTBEAR (2019). Rest api documentation tool | swagger ui. (Accessed on 12/27/2020).

Trestle Technology, LLC (2018). *plumber: An API Generator for R*. R package version 0.4.6.

UserAgentString.com (1999). Useragentstring.com - chrome version 81.0.4044.110. (Accessed on 12/21/2020).

User:Jallan (2011). Definition of user agent - wai ua wiki. Accessed on 12/04/2020.

Vaughan, D. and Dancho, M. (2018). *furrr: Apply Mapping Functions in Parallel using Futures*. R package version 0.1.0.

WhoIsHostingThis.com (2020). User agent: Learn your web browsers user agent now.

Wickham, H. (2011). testthat: Get started with testing. *The R Journal*, 3:5–10.

Wickham, H. (2019). *rvest: Easily Harvest (Scrape) Web Pages*. R package version 0.3.5.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., Fran?ois, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., M?ller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.

Wickham, H. and Bryan, J. (2020). *usethis: Automate Package and Project Setup*. R package version 1.6.3.

Wickham, H., Hester, J., Müller, K., and Cook, D. (2017). *memoise: Memoisation of Functions*. R package version 1.1.0.

Xie, Y. (2016). *bookdown: Authoring Books and Technical Documents with R Markdown*. Chapman and Hall/CRC, Boca Raton, Florida. ISBN 978-1138700109.