A Comparison of Forecasting Techniques in Bike Sharing Services

Luca Baggi

Nov 11, 2021

CONTENTS

Insert catchphrase

CHAPTER

ONE

INTRODUCTION

1.1 COVID-19 Effects and Mobility Habits in Italy

The COVID-19 pandemic set the world back in its pursuit of the seventeen Sustainable Development Goals (SDGs) (The 17 Goals, n.d.). This is also the case for Italy, as we can see from the 92 statistical measures collected by the Italian national institute for statistics (ISTAT, Istituto Nazionale di Statistica). When comparing 2019 with data from ten years prior, sixty percent of the measurements register an improvement, while twenty percent do not change and the remainder worsens. However, when comparing 2019 and 2020, only forty percent of the measurements improve, and almost the same amount displays a statistically significant drop (Dall'andamento degli obiettivi di sviluppo sostenibile alla mobilità, 2021).

To say that the pandemic brought about a lot of change would be a redundant understatement. Coming out of 2020, the Italian outlook is worse than the average of the (Western) European countries - as one can notice upon looking at macroeconomic variables such as the Gross Domestic Product (GDP). Part of the negative consequences could be offset, thanks to the government's stimulus packages, the European Union's (EU) funds and the European Central Bank's (ECB) Pandemic Emergency Purchase Program, or PEPP - a massive asset purchase program of 1,850 billion euros. From the beginning, it was clear that the pandemic would accelerate digitalisation (Härmand, 2021) - and, indeed, it has: see (Anthony Jnr & Abbas Petersen, 2021) for a meta-review or (Truant et al., 2021) for Italy. The generalised lockdowns reduced the greenhouse gases (GHG) emissions (see (Jephcote et al., 2021), (Singh & Chauhan, 2020), (Lian et al., 2020) and (Gualtieri et al., 2020)) but only temporarily and relatively to the severity of the restrictions put in place: the pandemic altered electricity consumption habits, but in countries like Sweden "the consumption even increased" compared to 2019 (Bahmanyar et al., 2020). Besides, this also came at the expense of our transportation habits and the mobility sector in general.

According to ISTAT, private cars are now used for half of all travels, up from 44 percent in the pre-pandemic period. Furthermore, the so-called "mobilità dolce" (the Italian translation of micro-mobility) "does not seem to take off" (Dall'andamento degli obiettivi di sviluppo sostenibile alla mobilità, 2021). In 2020, 30 percent of interviewed families stated that they had "some or a lot of difficulties in connecting with public transportation in the area where they live" - an improvement from the 33 percent of the previous year. However, it is enough to look at the regional level to realise that improvements are not spread evenly, and some areas of the country are actually worse off with respect to 2019. There is a great deal of heterogeneity: the share of families with troubles in accessing public transportation services is lower in the North (26 percent) compared to the South (36 percent), while in Campania more than half of the families are affected (51 percent). Furthermore, only 27 percent of students reach the places where they study with public means of transportation (and it's declining), while 75 percent reach the workplace with private means only (and the share is increasing) (Dall'andamento degli obiettivi di sviluppo sostenibile alla mobilità, 2021).

The only measurement that has been improving throughout the decade is that of air quality. However, these values remain significantly higher than the guidelines from the World Health Organisation (WHO) (Dall'andamento degli obiettivi di sviluppo sostenibile alla mobilità, 2021). Besides, on September 22, 2021 the WHO revised downwards their recommendations on air pollutants (the latest update dated back to 2005), and the European Commission (EC) declared that it will take this update into account when revising the guidelines for the upcoming year (Standards - Air Quality - Environment - European Commission, n.d.). Italy has always struggled to respect the European Directive on air pollution: on Novem-

ber 2020, the European Court of Justice (ECJ) judged that, from 2008 to 2017, Italy "systematically and continuously" violated the standards set by the Directive and failed in enacting countermeasures to avoid it ("Che Aria Respiriamo," 2021). On July 2021, the European Environmental Agency (EEA) ranked European cities according to the average level of $PM_{2.5}$ in the two previous years: out of 323 cities, Cremona (Lombardy) was second to last, Brescia and Pavia (both in Lombardy) were 315th and 314th respectively, while Venezia (Veneto) was 311st, Bergamo 306th and Milano 303rd (all in Lombardy) (European City Air Quality Viewer — European Environment Agency, 2021). Air pollution caused more than *52 thousands* premature deaths: almost 14 percent of the total premature death toll in Europe (Italy - Air Pollution Country Fact Sheet — European Environment Agency, 2020).

Unequal access to infrastructure (and, to a minor degree, air pollution) hinder the popularity of more sustainable mode of transport, such as biking and sharing services. The ISTAT report mentioned above estimated that in 2019 there were 30 million commuters in Italy (Gli spostamenti per motivi di studio o lavoro nel 2019 secondo il Censimento permanente della popolazione, 2021): more than two thirds (more than 20 million) need to reach their workplace, while the rest is made up of students (with a moderate variance: the share of students is higher in those regions where unemployment is higher, for example in Campania, where they make up 40% of commuters). However, the report did not present any data on biking habits or the use of sharing services. These can only be found in an earlier report, published in 2019 and referring to two years prior.

The 2019 report mentions two interesting statistics about commuting habits: "almost one in five [commuters] choose an 'active' mode of transport" - that is, walking or biking. However, most of the "active" commuters are actually walking to work (17,4%), whereas the bikers are only some 1,7%. In general, it is women, young and more educated people who use more public transportation means and bicycles, while private vehicles (the exclusive means of commuting for more than 73 percent of the employed) are spread among men between 25 to 44 and with an average level of education (Spostamenti quotidiani e nuove forme di mobilità, 2019). Car pooling is chosen by some 12 percent of the employed and 14,5 percent of students aged 18-24, while only less than half a million used bike sharing services at least once during the year (i.e., less than 2 percent). Such services are more popular across the young and more educated people, while the incidence is almost double the national average in metropolitan cities (Spostamenti quotidiani e nuove forme di mobilità, 2019).

1.2 Bike-Sharing Demand Determinants and the State of Public Transport Infrastructure in Italy

The picture drawn by the two ISTAT reports feels like a pool of untapped (if not wasted) potential. According to the most recent survey, in 2019 some 57,5% percent of commuters moved within the same municipality of residence. This value is driven up by the students, who make up almost 71 percent of commuters within the same municipality. However, even after taking them out of the computation, we still end up with an even figure: more than 51 percent of workers move within the same municipality (Gli spostamenti per motivi di studio o lavoro nel 2019 secondo il Censimento permanente della popolazione, 2021).

Sure, it would be naive to argue that commuters who work in the same municipality where they live could all bike to reach their destinations: after all, there is a great deal of heterogeneity across municipalities under several dimensions - like their sheer extension, morphology and, of course, infrastructure. There are many factors that affect bike sharing demand: the first one that comes to mind are the weather conditions: precipitations, humidity and seasonal patterns; the one that arguably plays the biggest role is the so-called "built environment" (Eren & Uz, 2020), i.e. infrastructure such as the availability of isolated or dedicated bike lanes instead of mixed ones, but also safe parking areas and bike racks for private bikes. The terrain clearly plays a role: slopes have a negative effect on bike usage (as one of the many examples, see (Bordagaray et al., 2016)), but incentive schemes can be devised to promote returning bikes to up-hill stations and even the least loaded ones (which also improves the overall efficiency of the system) (Fricker & Gast, 2016). Furthermore, in this scenario there is a positive effect of e-bikes. Besides, the literature also outlines the role of land use: pick-ups are more frequent in commercial areas and parks, compared to residential ones and, more broadly "the proximity to green spaces and recreation areas, schools, universities, museums, shopping centers, sports areas, restaurants, hotels, bus/subway/train/suburban/ ferry transit hubs has a positive effect on the use of BSP [Bike Sharing Programs]" (Eren & Uz, 2020).

The degree of integration with the public transportation is also important, as bike-sharing systems are found to be complementary means for "[bridging] the gap between multiple transit hubs" (Eren & Uz, 2020). But bike sharing can also be a substitute "especially when public transport is not available, between 22:00 pm [and] 06:00 a.m., they can encourage users to use BSP" (Eren & Uz, 2020). This implies that there is no single channel to promote bike-sharing services and that coordination across institutional players is crucial. Despite this, investments in bike-sharing services shall not fall in the background: their complementary role as "first/last mile solution" is recognised in the literature and enhance public transport as a whole.

Improving the public transport infrastructure is a priority for Italy. As always, there are territorial imbalances on two different dimensions: on the macro level, there is a clear divide between North and South, but there continue to be striking differences even within the wealthiest regions. Bike lanes have been increasing steadily: the total number of kilometres has grown by 15,5 percent since 2015, totalling approximately 4700 kilometres. However, the infrastructure is still far from adequate in most cities (Ambiente urbano, 2021).

The report from ISTAT outlines that public transport (or TPL, "trasporto pubblico locale") suffers both from lack of infrastructure and outdated fleets. As a starter, the TPL is over reliant on buses, which offer more than 55 percent of the number of seats per kilometre. However, once metropolitan cities are factored out, this figure skyrockets to well beyond 90-95 percent (Ambiente urbano, 2021). Only 32 percent of the bus fleets is in line with Euro 6 standard and some 34 percent belongs to the Euro 4 class - i.e., was deployed before 2008. Low emission buses make up some 28 percent of the total, but only slightly more than 3 percent are electric: the rest (almost 25 percent) is fuelled by natural gas. Unsurprisingly, the share of low-emission vehicles is higher in metropolitan cities.

Trolley buses are available in only 13 municipalities, trams in 11 and metropolitan trains in 7. However, there is a remarkable divide between the Italian champion, Milan, and the other cities. Tram network density in Milan is measured as 122 km per 100 squared kilometres; the silver medal is awarded to Turin, which has almost half the kilometres than Milan: 66. The average of the other cities is a mere 16 km. In general, while the supply of TPL (measured in seats per kilometre, per inhabitant) is on the rise, we are still far from the levels before the Great Recession (-7,3 percent compared to 2008). However, the supply in the North is 25 percent greater compared to the Centre and almost three times bigger than the South. Public demand for TPL is increasing in the North, stationary in the Centre and even declining in the South.

A minor and uneven push is provided by sharing services. Car sharing is available in 37 out of 107 "comuni capoluogo" (i.e., the "capital" of a province, corresponding to the NUTS3 classification), of which only 8 are in the South. Besides, only 26 percent of the fleets is composed of electric cars. Bike sharing services are present in 53 *capoluoghi*, registering an overall decline from 2015. Luckily, the number of bikes has more than tripled: from 6 to 19 bicycles per ten thousand inhabitants. The divide, as always, is quite stark: the number of bikes is 29 in metropolis compared to provinces, and these services are much more common in the North (32 bikes per ten thousand citizens) than in the Centre (17) and the South (just 2). Much of this success is to be attributed to the appearance of free-float systems, which require greater fleets (Ambiente urbano, 2021). The report does not provide information on electric scooters.

Restructuring the public transport will require extensive coordination between national, regional and municipal administrations, across multiple channels simultaneously. It is a widespread hope that many of this results can (only) be achieved via the Next Generation EU (NGEU), the 750 billion euros stimulus that will be financed by bonds from the European Commission. The NGEU is a bold and unprecedented move from the EU: the fund will be made up with up to 390 billion euros in subsidies, while up to 360 billions will be given out as loans with low interest rates. Italy is the first beneficiary in absolute terms for the main facilities of the NGEU: the country will receive more than 190 billion euros, to which the government will add 30 billions of its own. According to the so-called PNRR (*Piano Nazionale di Recupero e Resilienza*, i.e. National Plan of Recovery and Resilience), almost 25 billion euros will be invested in railways. However, less than one billion will be used to improve on the regional railways - the Achilles' heel of public transportation and the bane of commuters. In addition, more than 8,5 billion euros will be invested in TPL. But there's a catch: the NGEU grants will only be available for projects to be completed within the year 2026. Interviewed by *Il Post*, prof. Gabriele Grea from Bocconi University stated that these funds will mostly be awarded to projects in an already "advanced state": on one side, this will provide stronger guarantees about their completion, but will likely increase the inequalities across municipalities (Un po' di cose notevoli dentro il PNRR, 2021).

1.3 The Case for Promoting Bike Sharing

Reforming the public transport will be crucial to reach carbon neutrality and possibly promote economic growth. After all, transport accounts for as much as 27 percent of emissions in the EU (Bergantino et al., 2021) and while the overall greenhouse gases (GHG) emissions has been declining since the 1990s, the emissions from road transportation has nonetheless been increasing ever since (Annual European Union Greenhouse Gas Inventory 1990–2018 and Inventory Report 2020 — European Environment Agency, 2020). Besides, it is well-known that Italy chose to privilege rubber over railways to transport goods: Eurostat estimated that from 2000 to 2016 less than 10 percent of goods travelled by train, while the EU average was almost 18 percent (Milano ha un'occasione storica, 2017), so there seems to be much room to increase productivity. And, besides, despite the fact that emissions have been decreasing since the 1990s, Italy is not on the side of the achievers: since 1990, emissions in the country were reduced by 17,2 percent, compared to the EU28's 25,2 percent.

The Next Generation EU provides Italy with the perfect chance to narrow the divide with Western economies, while curbing emissions and finally improving the air quality. To promote more sustainable modes of transport, measures are needed on both the supply side (for example, by improving vehicle and fuel performance) and on the demand side by reducing demand for private transport, or at least increasing the demand for greener modes of transport (Bergantino et al., 2021). Investments in sharing services can and should play a role in this transition. Indeed, capital might be limited: after all, municipalities will receive a smaller share of the NGEU funds and upgrading their bus fleets seems more urgent. Besides, there are also the time constraints that need to be taken into account. However, biking infrastructure projects can be relatively cheaper compared to other TPL investments - especially if factoring in the presence of private entrepreneurs. Once infrastructure is in place, the costs of the service "only" amount to the human and technical cost to reallocate bikes to be at the right place at the right time.

This dissertation stems from the idea that bike sharing systems can be promoted with cheap measures. One of the most widely discussed problems in the literature is improving customer satisfaction through repositioning, i.e. forecasting the demand for bikes and "design efficient bike repositioning solutions" (Ghosh et al., 2019). The problem is ever more important since the introduction of free-float bike-sharing systems (FFBBS), as the bikes can be dropped anywhere and end up in sub-optimal places for the next customers. On the positive side, FFBBS do not require the upfront investments for building docking stations - which is necessary for station-based BSS (also known as SBSS). Furthermore, FFBSS "prevents bike theft", "by tracking bikes in real-time with built-in GPS", and "offers significant opportunities for smart management"(Pal & Zhang, 2017). This implies a greater satisfaction level for customers, "because obtaining and returning the bikes becomes much more convenient" (Pal & Zhang, 2017). However, all of this comes at the increased costs for bike rebalancing, because of how inefficient bike redistribution becomes and the and higher operating costs in terms of human and financial resources (Pal & Zhang, 2017). This has been recognised as "one of the main reasons why many FFBS enterprises lose money or even withdrawn from market."(Tian et al., 2021).

Most of the new approaches involve deep learning (DL) techniques, such as long-term short-memory (LSTM) neural networks, which usually outperform other statistical and machine learning approaches (Xu et al., 2018). Some are incredibly sophisticated: Convolutional LSTM are deep learning models "stacked and fused by multiple convolutional long short-term memory (LSTM) layers, standard LSTM layers, and convolutional layers […] to better capture the spatio-temporal characteristics and correlations of explanatory variables" (Ke et al., 2017). This, combined with external data such as "travel time rate, time-of-day, day-of-week, and weather conditions", results in an improvement of error metrics (RMSE) by almost 50 percent (Ke et al., 2017).

There is no way we could best the performance of such sophisticated models. However, this may not be the ultimate goal for policymakers. Despite their impressive performances, deep learning methods are hard to implement. They require a considerable amount of resources and, unlike simpler models, are much more difficult to visualise and interpret. For one, they require senior data scientists and access to a local server cluster with access to Graphic Processing Units (GPUs) or cloud computing platforms such as Google Cloud, Microsoft Azure or Amazon Web Services. This infrastructure needs time to set up and delays are inevitable, since the data manipulated by public administration deserves a much greater degree of privacy. Besides, such models require a long time to train - which translates to more expensive models.

Given the policymaker constraints, DL techniques may well be out of time, and budget. Our goal is to satisfy the constraints of a local planner with tight budget and more pressing issues, or the limited options of a private BSS company who needs to sustain high operational costs. We will develop two classes of models, univariate and multivariate, using

both statistical and machine-learning methods. We will also attempt to evaluate the usefulness of external data - which might be hard to require and process - and the performances of libraries such as Facebook's Prophet, which have been specifically designed for 'forecasting at scale', i.e. forecasting multiple time-series (in the order of the thousands) with little to no pre-processing, feature engineering and simple models.

This experiment has many drawbacks and limitations: as a starter, it does not include a proper spatial analysis, and does not explore Vector Auto Regressive (VAR) models, nor Bayesian models. Our goal is to prove that feature engineering and better data can do a better job at improving a model compared to more advanced techniques, especially under (hypothetically) tight budget constraints. In other words, we might want to get the feel for the marginally decreasing utility of accuracy improvements, which come at progressively greater computational costs.

1.4 The Open Source Stack, Transparency and Reproducibility

This is one of the many (millions?) projects to benefit from the existence of the open source community. This project was developed end-to-end using open source tools, starting from PostgreSQL to store the data and many Python libraries to train the models. Jupyter Notebooks have been the main developing tool (Perkel, 2018), and Jupyter Books (Executable Books Community, 2020) to convert the code into a LaTeXè publication, thanks to `pandoc` working in the background (Pandoc - About Pandoc, n.d.).

Zotero was used as a bibliography manager (Zotero | Your Personal Research Assistant, n.d.), and merits are due to several extensions for JupyterLab that made writing on JupyterLab possible: in particular, `jupyterlab-citation-manager` was used to insert citations inside notebooks via the official Zotero API, and `jupyterlab_spellchecker` helped in spotting typographic mistakes. Of course, version control with `git` and hosting on GitHub played a crucial role in project management. The code is free to see on the dedicated GitHub repository; however, data cannot be accessed due to the terms of the partnership between the provider and the University of Milan. We will try to find the space to introduce and contextualise all the other open source libraries that have been actively used.

There would be much to say about why using open source software, even (and especially) in economics. When it comes down to open source against proprietary software, the differences are not merely technical:

> There is an independent social dimension, where the metrics assess the interactions between people. Does it increase trust? Does it increase the importance that people attach to a reputation for integrity?

This is the Economics Nobel Prize Paul Romer (Romer, 2018), now 65, when comparing his experience with Jupyter and Mathematica notebooks. It goes on to make a bold claim:

> Jupyter exemplifies the social systems that emerged from the Scientific Revolution and the Enlightenment, systems that make it possible for people to cooperate by committing to objective truth; Mathematica exemplifies the horde of new Vandals whose pursuit of private gain threatens a far greater pubic loss – the collapse of social systems that took centuries to build.

When Romer tried to work with Mathematica notebooks and share their results, it became clear that "Wolfram made it hard to share a readable PDF version of a [Mathematica] notebook because it wanted someone like me to distribute content in its proprietary file format, the CDF". The conclusion of his articles are quite dramatic:

> The tie-breaker [between Wolfram and Jupyter, as well as proprietary and open source] is social, not technical. The more I learn about the open source community, the more I trust its members. The more I learn about proprietary software, the more I worry that objective truth might perish from the earth.

Jupyter Notebooks are being developed since 2001 and they are ever more popular. Perhaps they might even replace the scientific paper (Somers, 2018); what's sure is that they are ever more present. Indeed, Romer is not the only Nobel prize using open source software: Thomas Sargent, currently 78, uses Julia for his scientific research and launched a website, QuantEcon built with Jupyter Books, to teach computational economics in Python and Julia.

CHAPTER

TWO

BIKE SHARING AND BIKEMI: A LITERATURE REVIEW

2.1 A Brief History of Bike Sharing Services

The concept behind bike sharing systems (BSS) has been around since at least the 1960 (DeMaio, 2009). Perhaps unsurprisingly, the first BSS operator was Dutch: it was called Witte Fietsen, or White Bikes, and was deployed in Amsterdam, 1965. The bikes for the sharing service were painted in white, in order to distinguish them from private bicycles, and were made freely available with no locks. Even less surprisingly, "the total absence of security mechanisms led to theft and vandalism, and a rapid demise of Witte Fietsen", as pointed out by Fishman (Fishman, 2015).

Since this failed attempt, researchers have identified three further generations of bike sharing systems (Parkes et al., 2013), along the lines of their technological improvements. Being one of a kind, Witte Fietsen was basically the only representative of the first generations of bike sharing systems. According to DeMaio, the so-called second generation of bike sharing system was launched in the 1990s, in Denmark (1991 and 1993) and subsequently in the Netherlands (1995). These systems were designed with better insurances against frequent usage, as well as vandalism: "The Copenhagen bikes were specially designed for intense utilitarian use with solid rubber tires and wheels with advertising plates, and could be picked up and returned at specific locations throughout the central city with a coin deposit" (DeMaio, 2009). These incentives were not enough, as the time was not yet prime for better customer identification technologies as well as tracking systems.

New features such as "electronically-locking racks or bike locks, telecommunication systems, smart-cards, mobile phone access, and on-board computers" became the norm since 1996, following the example of services such as Bikeabout, developed by Portsmouth University in the United Kingdom (DeMaio, 2009). With Bikeabout, students could use a magnetic stripe card to rent a bike. Since then, a couple of third-generation bike sharing services launched every year across Europe, such as in France (Rennes, 1998) and Germany (Munich, 2000). These services managed to deter theft with "dedicated docking stations (in which bicycles are picked up and returned), as well as automated credit card payment and other technologies to allow the tracking of the bicycles" (Fishman, 2015). The peak of the third generation was reached around the second half of the 2000s, when Velo'v and its 1500-bike fleet were launched in Lyon in 2005, followed by the 7000 bikes deployed by Velib in 2007 in Paris.

Since then, bike-sharing spread to the rest of the world, and around the first half of the 2010s China established itself as a leader. In 2014, the global bike-sharing fleet was estimated at almost one million bicycles, three fourths of which where in China (Fishman, 2015); the country also had more than double the number of bike-sharing systems (237) compared to Italy (114) and Spain (113), while there were only 54 in the USA (Fishman, 2015). China is now the leader in sharing services, followed by Europe - which today still maintains its lead over the US, where "the adoption process it at an earlier stage and is gaining momentum" (Parkes et al., 2013).

This ongoing surge in popularity is once again due to technological breakthroughs: the fourth generation of bike sharing services exploits the Global Positioning System (GPS) and smartphones to deploy fleets of dock-less (or flee-floating) bikes and e-bikes. China could establish itself as leader also thanks to the rise of mobile technology and apps like WeChat, which was launched in 2011 and quickly became the most used platform in the country: "By the end of 2015, WeChat had 762 million monthly active users worldwide, and roughly 91% of them were from China; moreover, around 639 million users accessed WeChat on a smartphone" CHEN 2017. WeChat is ubiquitous in China and living without it nigh

impossible ("In Cina vivere senza WeChat è complicato," 2018), as the platform supports cash transfers and is used for shopping, tipping and paying services.

These new technologies drastically lowered the entrance barriers and the otherwise high upfront investments to set up a bike sharing network. One of the most popular BSS, ofo, was born in 2014 thanks to a collective of students at China's Peking University, who realised they could use the GPS on user's phone to track the bikes. They chose the name "ofo" because word itself resembled a biker and brought together some 2000 bikes to use on-campus (Schmidt, 2018). The service was so popular that they quickly made it into a company.

According to Samantha Herr, executive director of the North American Bikeshare Association in Portland, Maine, "large-scale venture capital and cheaper equipment are the game changers that propelled the explosive growth of dockless companies like Mobike and ofo". Dockless bikes are of "lower quality than their docked counterparts" and do not require expensive technologies to interface with a docking station: "That makes them cheaper to mass produce and drop off in new markets" (Schmidt, 2018). It is not by chance that most free-float BSS are completely private, while docked systems almost always feature a partnership between the private sector and local administrations.

Today, according to the estimates on the website Bike Sharing World Map, there appear to be almost 1900 bike sharing services in the world, with almost 1000 that have closed down and 300 to be opened. According to an interview to Russell Meddin, the website founder, in 2018 there were 16 to 18 million free-float bikes, plus 3,7 million docked bikes.

2.2 The Challenges of Sharing Services and the Sharing Economy

Sharing services are not without flaws. For example, Ma and their coauthors warn against some of the worse consequences of the sharing economy as a whole: "exploitative capitalism", labour precarity, widening income gaps and "platform capitalism" (Ma et al., 2018). The researchers make a dramatic claim: "Left unaddressed, these trade-offs risk becoming crippling contradictions to the potential of the sharing economy in promoting urban transformations to sustainability" (Ma et al., 2018). This situation represents a great challenge for public administrations, as the pace of innovation is always much faster than policy-making, and regulating the more controversial aspects of a rapidly growing and popular platform is easier said than done.

Lawmaking takes years: as an example, the two "champions" of the sharing economy, Uber and AirBnB, were valued almost 70 and 30 billion dollars in 2017 (Stone, 2017), despite the press already outlining some of the most controversial aspects of their business models. Uber could never really get past the scandals (Goggin & Taylor, 2019), especially those concerning the sexist culture of the company (Jackson, 2021), and this played a role in the weak IPO performance of the startup in 2019 ("Nel primo giorno di quotazione in borsa, le azioni di Uber hanno perso il 7,6 per cento," 2019). In the meantime, other competitors were facing accusations about the legal status of their "riders", and so did many other companies in the so-called gig economy, like Deliveroo (the renowned food delivery service). In countries such as Italy, this eventually led to the companies being ordered to recognise the employee status of the workers on the platform, further questioning not only the profitability but also the ethical basis of their business model (Secondo la procura di Milano, Uber Eats, Glovo, Deliveroo e Just Eat devono regolarizzare 60mila rider con contratti di collaborazione, 2021). Legal actions in Italy resulted in the company being placed under external management (Uber Italia è accusata di sfruttamento dei rider ed è stata commissariata, 2020), while in some US States, Uber and its primary competitor, Lyft, are now bound to observe a minimum wage (Seattle ha imposto una paga minima per gli autisti di Uber e Lyft, 2020).

However, these results only came several years later, and the same happened to AirBnB: despite a unexpectedly successful IPO ("La quotazione in borsa di Airbnb è stata un successo," 2020) in the midst of the pandemic ("La crisi della sharing economy," 2020), the company now also has to face tighter regulations that undermine their business model ("Airbnb Urges Housing Reform in Berlin after Court Overturns Permit Rejection," 2017) ("La nuova sentenza europea sugli affitti brevi," 2020) ("Le città europee contro gli affitti brevi," 2021).

To a lesser extent, this is a concern for bike-sharing services as well - in particular for dockless bike sharing systems, which have undergone an unforeseeable growth in the past five years. As Ma and their coauthors synthesise, the unregulated and unexpected growth of Mobike in Shangai results in hitting "a threshold of oversupply, under-distribution and user misbehaviour problems, which endanger[ed] the environmental and social sustainability of innovative urban mobility schemes" (Ma et al., 2018). They proceed to state that "[T]he social, political and infrastructural institutions in cities

have not developed adequate capacities and norms to respond, absorb and adapt to changes brought by under-regulated commercial and technological forces embodied in the modern sharing economy" (Ma et al., 2018).

The growth of Mobike was so fast that it also "exacerbated problems of user misbehaviour such as theft, vandalism and illegal parking, undermining the sharing values and public space resources that FFBSs require to operate efficiently" (Ma et al., 2018). Besides, this triggered venture capital funds (VC), which flocked to bike-sharing startups, leading to more than 1,7 million new bikes flooding the streets of Shanghai: "The oversupply of shared bikes created a serious strain on public resources. Massive numbers of bikes were dumped in public spaces, exacerbating existing crowding and stress on the city's roads and parking spaces" (Ma et al., 2018). The collaboration between private enterprises and public officers began to deteriorate: bikes were ordered to be removed, while new BSS startups "dropped their bikes without any notice in advance" - a strategy that was also used in other parts of the world. The authors claim that this "unintended tragedy of the commons" was due to "under-regulated FFBSs", which had to put their private interests (profitability) first if they wanted to survive the competition (Ma et al., 2018).

2.3 The Elephant on the Sidewalk: Electric Scooters

Nowadays, bike sharing systems contend roads (and sometimes even sidewalks) with new competitors: electric scooters, or moped, which have become surprisingly popular across the globe in recent years, displaying a stronger growth trend and adoption compared to BSS. The literature on the history of e-scooters services is still quite young, but authors generally trace the beginning of electric scooter systems (ESS) to 2017, specifically to the US (Yang et al., 2021). Scooters arrived in Europe, specifically in Brussels, during the summer of 2018 (Moreau et al., 2020).

E-scooters have since developed quite rapidly, even faster than dockless bike sharing services (Yang et al., 2021), and today it is estimated that they make up almost two thirds of the shared micromobility trips in the US, while almost one in four citizens in Paris tried one in 2019 (Yang et al., 2021). However, moped did not come without hassles. For one, "rented and privately-owned e-scooters suddenly became a conspicuous, controversial and disruptive presence in urban public space" (Tuncer et al., 2020). Besides, their road status is not quite clear, and legal frameworks are still not ready to adapt to include e-scooters, which "upset the normal order of traffic and public space" (Tuncer et al., 2020). While they are used at least as widely as bikes (if not more), they do not seem to belong to roads more than skateboards, not to mention sidewalks.

Their advantages are clear: being electric, they can travel longer distances with lesser effort. Besides, scooters are handier and more portable than bikes. There are some disadvantages, too: scooters are electric and thus require charging, which introduces another step in the reallocation procedure. Overall, this entails greater costs for the service providers: while manufacturing costs might be assumed to be equivalent, recharging the batteries and moving them to and from the charging station surely translates into higher operational costs (Zhu et al., 2020). Furthermore, some authors in the literature also find that scooters have a shorter life span and thus "at present, the use of e-scooters shows a higher impact than the transportation modes they replace" (Moreau et al., 2020). This calculation does not include "end-of-life treatment", which could positively affect their "GWP" or "Global Warming Potential" (Moreau et al., 2020). The authors suggest a list of measures that could reduce their impact, which include increased cooperation with the public sector: "new electric charging stations that are installed in the city could also include charging devices for e-scooters"; however, the supplier should also provide incentives, for example "a financial incentive for the users to drop the e-scooter off at charging areas and plug them in" (Moreau et al., 2020).

However, the literature is still too young to coherently assess the impact of e-scooters and their relationship with existing sharing service and public transport network. For example, it would be deemed reasonable that e-scooters might provide a good alternative to private transportation, yet "several studies suggest that they are frequently used instead of walking" (Tuncer et al., 2020), as it was found by (Laa & Leth, 2020), (Mitra & Hess, 2021), (Nikiforiadis et al., 2021) and (Sanders et al., 2020). Some authors specifically found that "People travelling with bicycle or motorcycle were not attracted by e-scooters novelty" (Nikiforiadis et al., 2021) or that are more likely to be employed for recreational purposes, "potentially filling a niche" (Sanders et al., 2020). One interesting trend that e-scooters display is that, after trying the service, some users do buy a scooter of their own (Tuncer et al., 2020), and e-scooters owners are more likely to use them as a replacement for their private cars (Moreau et al., 2020).

In other words, borrowing from economic jargon, it is still not clear why and when e-scooters are to be considered substitutes or complements of traditional modes of transport, and the same goes for bike sharing. On the one hand, it seems clear that e-scooters (and bikes) do not consistently replace private vehicles and cars in particular. By the way, this reinforces the idea that in order to transition towards a greener economy, promoting sharing services must go hand in hand with investments in public transportation networks, as well as improving existing biking infrastructure (Laa & Leth, 2020).

For one, it might be expected that adoption of e-scooters translates into a decline in bike rentals. These are the findings of Yang and their coauthors (Yang et al., 2021), who estimated that in Chicago weekly usage of bike sharing in e-scooter sharing operation area declined by slightly more than 10 percent. Specifically, the usage of bike-sharing service subscribers was down by 4 percent, while the drop across non-subscribers was as big as 34 percent (Yang et al., 2021). Indeed, this might reveal that subscribers choose bikes for endogenously different reasons - i.e., they might deem them better suited for commuting. After all, "bike sharing use during non-peak hours decreased but was not affected during peak hours" (Yang et al., 2021). The drop was more marked with short trips, for which bike usage was down by almost 11 percent - twice as much as the decline in medium-duration trips (5,5) but almost half as much as in short trips (20,5). In general, it seems that docked bikes are preferred for commuting (Reck & Axhausen, 2021) (Reck et al., 2021).

2.4 Sharing Services in Milan: BikeMi and its Competitors

BikeMi was introduced in December 2008 in a partnership between the City of Milan and Clear Channel Italia, a subsidiary of a global media and advertising group. In 2015, pedal assisted bicycles were introduced. Now the service counts 325 stations with 5430 bikes: 4280 are "classic", 1000 are e-bikes and 150 are "pedal-assisted" (i.e., electric) with a child seat (Who We Are - BikeMi, n.d.). Including private enterprises (fully free-float), there are a total of 15400 bikes up for sharing, of which 3500 are electric. Private companies were progressively introduced in 2017, after public procurement and a testing phase (Bike Sharing - Comune Di Milano, 2021).

Mobike was the first, in July 2017, deploying 4000 bikes across Milan and Florence (P.Sol, 2017). ofo joined around the same time and economic Newspaper *il Sole 24 ORE* reported that, by November 2017, MoBike had deployed 8000 bikes in Milan and 7000 more in the rest of Italy, while ofo had 4000 bikes (Magnani, 2017). This resulted in BikeMi losing some 5 percent of their subscribers. ClearChannel, the service provider, disclosed that in 2009 they had some 10700 subscribers, which became around six times as much by 2017. At the time, BikeMi was reported to have 4650 bikes, of which 1000 were electric (Magnani, 2017). Surprisingly, BikeMi was reporting a profit: 200 thousand euros. Operating costs amounted to 6 million euros, two of which were covered by subscriptions and the remainder by advertisement revenue (Magnani, 2017).

The article was also reporting rumours about a fusion between the two private operators, which were undergoing extensive losses and could not be seen reaching the profitability goals they set themselves for the following years. In 2018, ofo was already said to be close to failure, while Mobike was allegedly looking to sell their operations in Europe (Salvioli, 2018). Indeed, ofo failed in 2020, while the Italian Idri Bk (manager of Mobike fleets) bought the European branch of Mobike on November 2019. From the operation, the new sharing service Movi (now RideMovi) was born, after ofo had already withdrawn from the Italian and European market (Soldavini, 2019). Now in Milan there are three bike sharing services: BikeMi is the only docked one, with rentals services open from 7 a.m. to 1 a.m. and from 7 a.m. to 2 a.m. during summer, plus 24 hours on Fridays and Saturdays. The other two services, RideMovi and Lime, are free-floating and operating 24 hours a day (Bike Sharing - Comune Di Milano, 2021).

Electric-scooter sharing services arrived in Milan only on February 2020 (Monopattini in Sharing - Comune Di Milano, 2021), right before the first wave of the COVID-19 pandemic and following a one-year long public procurement. Each provider was allowed a fleet of 750 e-moped, for a total of 2250 vehicles and three providers. To face the mobility challenges of the pandemic, the cap was increased to 6000 and the number of providers was doubled.

2.5 Competitors and Data in Our Analysis

The history of BikeMi's competitors has several implications for our analysis. While we will be discussing the data and the choices for our project in the next chapter, it is worth anticipating some points. For one, we can already rule out electric scooters and their effects, as they only appeared in 2020 - a year for which we would have BikeMi data. However, given the pandemic break out, we chose not to forecast rentals in that period of time.

The situation becomes more complicated when it comes to the other dockless bike-sharing services. For one, we could not obtain data from any of the other providers, which either failed or sold all their activities to other private enterprises. Not having data about dockless services forces us to change our data strategy, but the literature comes to help. As we have outlined above, several authors found that docked bike-sharing systems are mostly used for commuting and are less sensitive to the introduction of free-floating BSS. Since ClearChannel only registered a 5 percent decrease in subscriptions since the FFBSS introduction, we feel reassured when accepting that we will only be able to model FFBSS effects with a dummy variable. After all, in just a few months private operators introduced at least as many bikes as BikeMi had: 4000 for ofo and 4000 for Mobike, which became twice as many in less than six months after the launch of the service. Given these numbers, a decline of only 5 percent does not seem worrisome.

If we buy the "docked bikes services are less sensitive to dockless ones" assumption, i.e. that docked bikes are preferred for commuting, it can be enough to forecast the number of bikes during peak hours. In the next chapter, we will see that peak commuting hours perfectly overlap with peak BikeMi usage. Finally, there is a considerable data gap from July 2018 to the end of that year, which forced us to analyse data from 2015 to that date only. All in all, the instability and novelty of free-floating bike sharing services has likely dampened their usage growth. For all the reasons outlined so far, we concluded that the lack of ways to account for ofo and Mobike data would not hinder our forecasts significantly. Besides, adopting the policymaker perspective, it would be quite difficult to have your competitor's data available.

2.6 BikeMi: a Review of the Literature

Saibene and Manzi (Saibene & Manzi, 2015) analyse survey data to evaluate the level of satisfaction for "all the actors involved": service management, city council and users. The area taken into consideration was inside the "Bastioni", i.e. inside of the historic hispanic walls. This area coincides with the so-called "Area C", where cars have restricted access since 2012 (and paid access since 2008 with the so-called "Ecopass"). Sorrentino, Manzi and Virili (Sorrentino et al., 2019) also investigate the "organisational identity" of Clear Channel Italia, the service provider, focusing on the normative and utilitarian dimensions that characterise a public service. In particular, they observe how the "publicness" and "privateness" of BikeMi interact and how public accountability and social impact interact with Clear Channel's profitability goals.

Toro and his coauthors (Toro et al., 2020) perform a similar analysis with data from June 2015 to December 2018 - the same period we analyse. They find that the service is "extensively used for commuting to work-related activities" and that "only strong meteorological conditions can impact the use of the service" (Toro et al., 2020). Indeed, it seems reasonable to assume that commuters are more inelastic to moderately adverse weather, as light rain or cold. The authors also implement a clustering algorithm to analyse bike-sharing services patterns, a widespread step in the literature - especially for forecasting about dockless services. The goal of this strategy is to "identify temporal-spatial patterns for specific users' typologies" (Toro et al., 2020). They dispose of 13,789,569 records for 3650 traditional bikes and 1150 electric bikes (150 with a child seat). The service operates from 7 a.m. to midnight.

The authors remove all records before the service working hours, as they might inadvertently capture "extraordinary schedules" or maintenance. They do the same for trips whose duration was shorter than a minute and "records whose arrival or departure did not match with the existing stations" (Toro et al., 2020). They identify two types of users, occasional and regular, according to the 75th percentile threshold, i.e. 36 trips, with findings consistent with the literature. Regular users use the service more during weekdays with two different peaks along the day, and occasional users during holidays and weekends, with larger activities at lunchtime and evening hours (Toro et al., 2020).

Through spatial analysis, the researchers confirm that traffic flows from the periphery to the city centre in the mornings and back in the evenings, with greater use in the proximity of train stations. These patterns are also captured by cluster

analysis. Looking at the weather, the usage starts to decline "dramatically" only around the 20mm threshold. Also Croci and Rossi (Croci & Rossi, 2014) find that their results are robust to "confounding factors such as weather conditions". Their research establishes that "the presence of metro and train stations, universities, museums, cinema and restricted traffic areas in correspondence of bike sharing stations significantly increase use. On the other hand the presence of tram and bus stops and theatres does not and has an opposite influence" (Croci & Rossi, 2014). Lastly, Cappozzo and their coauthors work on scrapped BikeMi data to classify stations to determine "the future *full*, *empty* or *not problematic*" state of each station (Cappozzo et al., n.d.).

CHAPTER

THREE

BIKEMI DATA

```
# path manipulation
from pathlib import Path

# data manipulation
import numpy as np
import pandas as pd
import geopandas

# plotting
import matplotlib.pyplot as plt
import contextily as cx
import seaborn as sns

# connecting to the local database with the data
import psycopg2

# to use pandas dtypes in matplotlib plots
from pandas.plotting import register_matplotlib_converters

register_matplotlib_converters()

# set settings for seaborn
sns.set_style(style="whitegrid", rc={"grid.color": ".9"})
sns.set_palette(palette="deep")

# customise matplotlib and sns plot dimensions
plt.rcParams["figure.figsize"] = [12, 6]
plt.rcParams["figure.dpi"] = 100
title_font = {"fontname": "DejaVu Sans Mono"}

# create paths
milan_data = Path("../data/milan")

# establish connection with the database
conn = psycopg2.connect("dbname=bikemi user=luca")
```

3.1 Data Ingestion

The data was made available thanks to a partnership established by Prof. Giancarlo Manzi of the University of Milan and Clear Channel Italia, the provider of the service. The data is comprised of all the individual trips performed by each client (`cliente_anonimizzato`). This includes the bike type (which can either be a regular bike or an electric bike), the bike identifier, the station of departure and arrival with the time, the duration of the trip `durata_noleggio` plus the total travel distance. We do not know how the total travel distance `distanza_totale` is computed. Here are a selection of fields for the first five rows of the source data (time features are rounded to the daily level to fit into the page):

```
def show_data_sample(connection) -> pd.DataFrame:
    query = """
        SELECT
            bici,
            tipo_bici,
            cliente_anonimizzato,
            DATE_TRUNC('day', data_prelievo)::date AS giorno_prelievo,
            nome_stazione_prelievo,
            DATE_TRUNC('day', data_restituzione)::date AS giorno_restituzione,
            nome_stazione_restituzione
        FROM bikemi_source_data
        LIMIT 5;
    """

    return pd.read_sql(query, connection)


show_data_sample(conn)
```

```
    bici tipo_bici  cliente_anonimizzato giorno_prelievo  \
0   8480      Bike                 47869      2015-07-01
1   6190      Bike                 74372      2015-07-01
2   6000      Bike                105372      2015-07-01
3  10538     eBike                103840      2015-07-01
4   1981      Bike                 57260      2015-07-01

       nome_stazione_prelievo giorno_restituzione nome_stazione_restituzione
0  Arco della Pace 2 - Pagano          2015-07-01       Vercelli - Cherubini
1                 XXV Aprile          2015-07-01                    Caiazzo
2                San Giorgio          2015-07-01                    Rosario
3                 XXV Aprile          2015-07-01                    Caiazzo
4                 XXV Aprile          2015-07-01              Sant'Ambrogio
```

The data available ranges from the first of June, 2015, to the first of October, 2020, totalling to 15.842.891 observations. Data was made available in Excel spreadsheets, following the Office Open XML SpreadsheetML File Format (the `.xlsx` file format). Python"s Pandas library has methods to read `.xlsx` files; however, given how big these files are, data manipulation would have proven unfeasible.

For this reason, we resorted to some useful and popular open source tools, which we used to build `bash` scripts and functions to automate conversion from `.xlsx` to `.csv` files, perform some elementary data cleaning and load the data into a local PostgreSQL database. Format conversion to Comma-Separated Values (`.csv`) was performed using `csvkit`, a Python package to perform basic operations on `.csv` files from the command line. Being written in Python, `csvkit` can be slow. However, as part of a major trend for several command-line applications, `csvkit` was rewritten in Rust, a fast and secure programming language whose popularity has been rising in the last couple of years (Perkel, 2020). Much alike Julia (Perkel, 2019), Rust is becoming a tool for data science, as well as scientific computing (for example in bio-statistics) as it is "a language that offer[s] the "expressiveness" of Python but the speed of languages such as C and C++" (Perkel, 2020).

The Rust port of `csvkit` is called `xsv`, and is blazing fast. Much alike `awk` (Gawk - GNU Project - Free Software Foundation (FSF), n.d.), `xsv` can perform filtering operations, but also joins and partitions, as well as computing summary statistics. `xsv` does not offer format conversion (yet), but was used to filter out a negligible number of invalid observations from each original `.xslx` files (after the conversion to `.csv`), and select only the columns that would enter the final dataset.

Finally, `psql` (PostgreSQL"s command line utility) was used to upload the "clean" data into a local database instance. PostgreSQL was also used to perform basic survey statistics, like computing the number of rows, and data aggregation (such as counting the number of observations by year). Looking at the frequency tables by year, there appears to be an oddly small number of observations from 2018. This is because there is indeed missing data from June 2018 until the end of the year. For this reason, we chose to work only with data from June 2015 to the end of May 2018.

```
def count_users_by_year(connection) -> pd.DataFrame:
    query = """
        SELECT
            EXTRACT("year" FROM b.data_prelievo) AS date,
            COUNT(b.bici)
        FROM bikemi_source_data b
        GROUP BY EXTRACT("year" FROM b.data_prelievo);
    """

    return pd.read_sql(query, connection).astype("int").set_index("date")


count_users_by_year(conn)
```

```
        count
date
2015  1971891
2016  4066783
2017  4272480
2018  1457631
2019  2830566
2020  1243540
```

3.2 Data Analysis

3.2.1 Remove Outliers and Select the Time Span

In addition to selecting only trips from June 2015 to June 2018, we also disregard all rentals whose duration is smaller than one minute - as previously done in the literature. This leaves us with more than 11,7 million observations. We store these in a materialised view:

```
materialized_view_query = """
    CREATE MATERIALIZED VIEW IF NOT EXISTS bikemi_rentals_before_2019 AS (
        SELECT
            b.tipo_bici,
            b.cliente_anonimizzato,
            DATE_TRUNC("second", b.data_prelievo) AS data_prelievo,
            b.numero_stazione_prelievo,
            b.nome_stazione_prelievo,
            DATE_TRUNC("second", b.data_restituzione) AS data_restituzione,
            b.numero_stazione_restituzione,
            b.nome_stazione_restituzione,
```

(continues on next page)

(continued from previous page)

```
            b.durata_noleggio
        FROM bikemi_source_data b
        WHERE
            EXTRACT("year" FROM b.data_restituzione) < 2019 AND
            durata_noleggio > interval "1 minute"
    );
"""
```

3.2.2 Top Users and Commuting Habits

```
def count_distinct_users(connection) -> pd.DataFrame:
    query = """
        SELECT
            COUNT(DISTINCT cliente_anonimizzato)
        FROM bikemi_rentals_before_2019;
        """

    return pd.read_sql(query, connection)


count_distinct_users(conn)
```

```
    count
0  192431
```

The service has almost 200 thousands unique subscribers in the time period. Then breakdown by year is the following:

```
def count_users_by_year(connection) -> pd.DataFrame:
    query = """
    SELECT
        EXTRACT("year" FROM data_prelievo) AS anno,
        COUNT(DISTINCT cliente_anonimizzato)
    FROM bikemi_rentals_before_2019
    GROUP BY EXTRACT("year" FROM data_prelievo);
    """
    return pd.read_sql(query, connection).astype({"anno": "int"}).set_index("anno")


count_users_by_year(conn)
```

```
      count
anno
2015  64079
2016  93239
2017  95931
2018  50860
```

The number of subscriptions is declining in 2015 and 2018, as there are observations for six months only. In particular, for the year 2018 the count is even lower because the autumn/winter months are missing.

It is also interesting to look at the top users:

```
def get_top_users(connection) -> pd.DataFrame:
    query = """
```

(continues on next page)

(continued from previous page)

```
        SELECT
            cliente_anonimizzato,
            COUNT(*) AS noleggi_totali
        FROM bikemi_rentals_before_2019 b
        GROUP BY
            cliente_anonimizzato
        ORDER BY noleggi_totali DESC
        LIMIT 10;
    """

    return pd.read_sql(query, connection).set_index("cliente_anonimizzato")


get_top_users(conn)
```

```
                       noleggi_totali
cliente_anonimizzato
40585                            3689
19515                            2859
38603                            2747
43969                            2736
23891                            2636
15912                            2565
56496                            2456
42815                            2451
17683                            2451
146640                           2417
```

But it might be of greater interest to look at the distribution by year:

```
def get_top_users_by_year(connection) -> pd.DataFrame:
    query = """
        SELECT
            cliente_anonimizzato,
            COUNT(*) AS noleggi_totali,
            EXTRACT("year" FROM data_prelievo) AS anno
        FROM bikemi_rentals_before_2019 b
        GROUP BY
            cliente_anonimizzato,
            EXTRACT("year" FROM data_prelievo)
        ORDER BY noleggi_totali DESC
        LIMIT 10;
    """

    return pd.read_sql(query, connection).astype({"anno": "int"}).set_index("cliente_
↪anonimizzato")


get_top_users_by_year(conn)
```

```
                       noleggi_totali  anno
cliente_anonimizzato
43969                            1478  2017
146640                           1355  2017
40585                            1290  2017
40585                            1263  2016
```

(continues on next page)

(continued from previous page)

```
188391                          1133  2017
165664                          1056  2016
147597                          1042  2016
19515                           1039  2017
97713                           1025  2017
90177                            988  2016
```

As expected, there are more observations from the years 2016 and 2017 as these are complete years. The great number of usage translates to an average of almost 4 trips per day - i.e., to reach the first train station and then the workplace.

3.2.3 Usage Patterns and Origin-Destination Matrix

As a starter, it is useful to look at the origin-destination (OD) matrix, to see which are the most popular starting and departure points. On the aggregate level, the most popular points on the OD matrix are the train stations and sightseeing places such as Duomo.

```
def get_top_stations(cols: list[str], connection, commuting_hours: bool = False) ->␣
↪pd.DataFrame:
    def _top_stations(colname: str, _connection, commuting: bool = commuting_hours) ->
↪ pd.DataFrame:
        if commuting:
            query = f"""
            SELECT
                {colname},
                COUNT(*) AS numero_noleggi
            FROM bikemi_rentals_before_2019
            WHERE
                EXTRACT("dow" FROM data_restituzione) BETWEEN 0 AND 4 AND
                EXTRACT("hour" FROM data_restituzione) BETWEEN 7 AND 10 OR
                EXTRACT("hour" FROM data_restituzione) BETWEEN 17 AND 20
            GROUP BY
                f{colname}
            ORDER BY numero_noleggi DESC
            LIMIT 10;
            """

        query = f"""
            SELECT
                {colname},
                COUNT(*) AS numero_noleggi
            FROM bikemi_rentals_before_2019
            GROUP BY
                {colname}
            ORDER BY numero_noleggi DESC
            LIMIT 10;
        """
        return pd.read_sql(query, _connection)

    return pd.concat([_top_stations(col, connection) for col in cols], axis=1)


get_top_stations(["nome_stazione_prelievo", "nome_stazione_restituzione"], conn)
```

```
  nome_stazione_prelievo  numero_noleggi nome_stazione_restituzione  \
0              Cadorna 3          234694                  Cadorna 3
```

(continues on next page)

(continued from previous page)

```
1                  Duomo          197573                     Duomo
2                Moscova          139561                   Moscova
3    Garibaldi - Sturzo           134243       Garibaldi - Sturzo
4  San Babila - RIMOSSA-          128261                XXV Aprile
5             XXV Aprile          127804     San Babila - RIMOSSA-
6              Cadorna 1          118692                  Cairoli
7                Cairoli          108717           Palazzo Marino
8             Centrale 1          107864                Cadorna 1
9     Coni Zugna Solari            99152                   Cavour

   numero_noleggi
0          234349
1          202773
2          140552
3          129611
4          129006
5          127974
6          113104
7          103253
8          102102
9           98083
```

The ranking is basically unchanged if we look only at data from Monday to Friday and within "core" commuting hours (say, from 7 to 10 and from 17 to 20). However, trips towards stations become more frequent and destinations such as Moscova are slightly less popular. Indeed, employees might straight bike to Moscova after work to take a sip from their Negroni.

```
get_top_stations(["nome_stazione_prelievo", "nome_stazione_restituzione"], conn,␣
↪commuting_hours=True)
```

```
  nome_stazione_prelievo  numero_noleggi nome_stazione_restituzione  \
0              Cadorna 3          234694                   Cadorna 3
1                  Duomo          197573                       Duomo
2                Moscova          139561                     Moscova
3    Garibaldi - Sturzo           134243       Garibaldi - Sturzo
4  San Babila - RIMOSSA-          128261                XXV Aprile
5             XXV Aprile          127804     San Babila - RIMOSSA-
6              Cadorna 1          118692                  Cairoli
7                Cairoli          108717           Palazzo Marino
8             Centrale 1          107864                Cadorna 1
9     Coni Zugna Solari            99152                   Cavour

   numero_noleggi
0          234349
1          202773
2          140552
3          129611
4          129006
5          127974
6          113104
7          103253
8          102102
9           98083
```

The behaviour changes if we look ad the individual trips, i.e. if we `GROUP BY` both departure and destination stations (`nome_stazione_prelievo` and `nome_stazione_restituzione`). To be exact, `Coni Zugna Solari`

is close to both the station in Porta Genova as well as the Navigli and Darsena, which partly explains its popularity.

```
def get_top_od(connection, commuting_hours: bool = False) -> pd.DataFrame:
    if commuting_hours:
        query = """
                SELECT
                    nome_stazione_prelievo,
                    nome_stazione_restituzione,
                    COUNT(*) AS numero_noleggi
                FROM bikemi_rentals_before_2019
                WHERE
                    EXTRACT("dow" FROM data_restituzione) BETWEEN 0 AND 4 AND
                    EXTRACT("hour" FROM data_restituzione) BETWEEN 7 AND 10 OR
                    EXTRACT("hour" FROM data_restituzione) BETWEEN 17 AND 20
                GROUP BY
                    nome_stazione_prelievo,
                    nome_stazione_restituzione
                ORDER BY numero_noleggi DESC
                LIMIT 10;
            """

    query = """
        SELECT
            nome_stazione_prelievo,
            nome_stazione_restituzione,
            COUNT(*) AS numero_noleggi
        FROM bikemi_rentals_before_2019
        GROUP BY
            nome_stazione_prelievo,
            nome_stazione_restituzione
        ORDER BY numero_noleggi DESC
        LIMIT 10;
    """

    return pd.read_sql(query, connection)


get_top_od(conn)
```

	nome_stazione_prelievo	nome_stazione_restituzione	numero_noleggi
0	Coni Zugna Solari	Napoli - Washington	14144
1	Napoli - Washington	Coni Zugna Solari	13256
2	Cadorna 3	Duomo	9363
3	Duomo	Cadorna 3	8903
4	Coni Zugna Solari	Savona - Tolstoj	8217
5	Palazzo Marino	Cadorna 3	7436
6	Bertarelli	Cadorna 3	6967
7	Vercelli - Piemonte	Roncaglia - Washington	6373
8	Piola	Gorini - Strambio	6336
9	Cadorna 3	Arcivescovado	6197

The same considerations apply when looking at the top trips only within the "core" commuting hours. This reinforces the conclusion that BikeMi is consistently used for commuting purposes.

```
get_top_od(conn, commuting_hours=True)
```

	nome_stazione_prelievo	nome_stazione_restituzione	numero_noleggi

(continues on next page)

(continued from previous page)

```
0      Coni Zugna Solari        Napoli - Washington           14144
1    Napoli - Washington          Coni Zugna Solari           13256
2              Cadorna 3                      Duomo            9363
3                  Duomo                  Cadorna 3            8903
4      Coni Zugna Solari           Savona - Tolstoj            8217
5         Palazzo Marino                  Cadorna 3            7436
6              Bertarelli                  Cadorna 3            6967
7    Vercelli - Piemonte     Roncaglia - Washington            6373
8                  Piola          Gorini - Strambio            6336
9              Cadorna 3              Arcivescovado            6197
```

3.3 Spatial Data Analysis

On its open data portal, the Comune di Milano publishes all sorts of open data - like the daily entrances in the so-called Area C, where access to private cars is limited. On this treasure trove of data, we can also find things like the location of column fountains and newsstands, but also the geo-referenced data of bike tracks and the location of BikeMi Stations. As noted previously, the service has some 320 stations, spread unevenly across the town, as it was outlined by previous studies (Saibene & Manzi, 2015). As a starter, we filter out all stations that have been introduced after 2019. This leaves us with 281 official stations, versus 279 in the historic data.

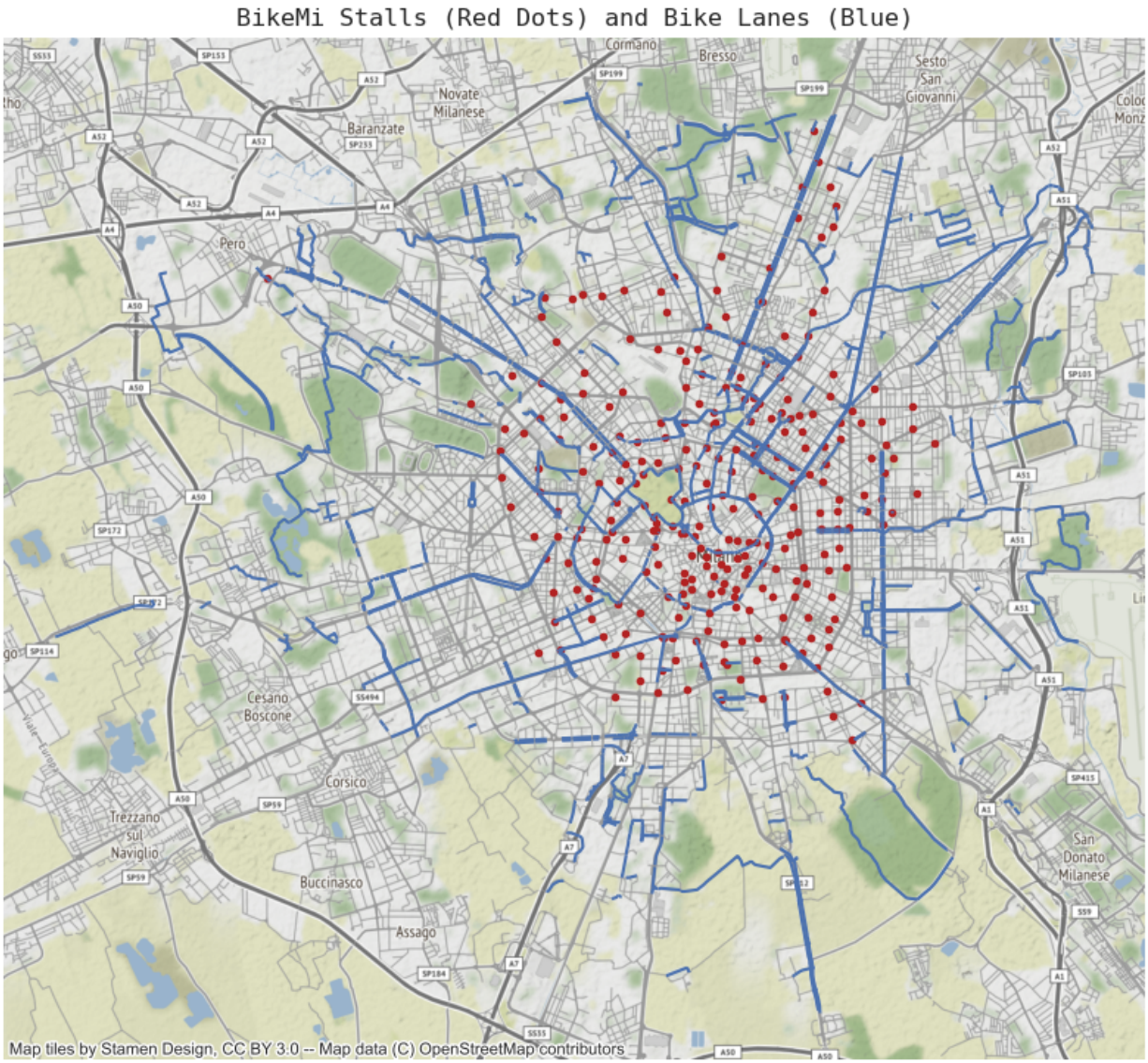
```
bikemi_stalls = (
    geopandas.read_file(Path(milan_data / "bikemi-stalls.geo.json"))
        .filter(["numero", "nome", "zd_attuale", "anno", "geometry"])
        .rename(columns={
        "numero": "numero_stazione",
        "zd_attuale": "municipio",
        "geometry": "stalls_geometry"
    })
        .set_geometry("stalls_geometry")
        .set_index("numero_stazione")
        .query("anno < 2019")
        .astype({"anno": "int"})
)

bikemi_stalls.head()
```

```
                                nome  municipio  anno  \
numero_stazione
001                           Duomo          1  2008
402                 San Babila Bis           1  2008
003                      Cadorna 1           1  2008
004                          Lanza           1  2008
005              Universita' Cattolica       1  2008

                         stalls_geometry
numero_stazione
001              POINT (9.18914 45.46475)
402              POINT (9.19725 45.46627)
003              POINT (9.17566 45.46800)
004              POINT (9.18197 45.47227)
005              POINT (9.17641 45.46312)
```

The data is represented with the (geographic) coordinate reference system (CRS) `EPSG:4326`, but in order to use the `contextily` library we need to cast it to the (projected) reference system `EPSG:3587`. As the map shows, the stalls

stretch to the North, towards Bicocca and Sesto San Giovanni. We can already see that the stalls distribution outside the city centre follows bike lanes (in blue).

```
bike_lanes = geopandas.read_file(Path(milan_data / "transports-bike_lanes.geo.json"))

bikes_dict = {"color": "firebrick", "marker": "."}

def plot_stalls_and_bikelanes():
    fig, ax = plt.subplots(1, 1, figsize=(10, 10))

    bikemi_stalls.to_crs(3857).plot(ax=ax, **bikes_dict)
    bike_lanes.to_crs(3857).plot(ax=ax)
    cx.add_basemap(ax)

    plt.axis("off")

    plt.title("BikeMi Stalls (Red Dots) and Bike Lanes (Blue)", **title_font)

    plt.show()

plot_stalls_and_bikelanes()
```



3.3.1 Inspecting Stations with Zero Daily Rentals

The stalls in the outer stations are not as used as the ones in the city centre, which come up as an abundance of zeros in the data aggregated at daily and hourly level. Some of these stations are practically unused. Besides, the great count of stations would represent a problem in the context of multivariate regressions (the so-called $p > n$ problem). Hence, we can drop the unused stations from our data to help us reduce the dimensionality.

We start by creating another two `materialised views` with the daily and hourly rentals, by station. Since the service is only active between 7 and 24, we just keep hourly observations within that time span. If we simply were to `GROUP BY` station names and time units, we would obtain series with gaps in the time index. We first create a table with all possible combinations of stations and dates using a `CROSS JOIN` and a common table expression (or `CTE`), then left join on this table the values obtained via the `GROUP BY` on the reference table. A similar query is used to obtain the hourly rentals, with hourly time intervals instead of daily ones.

```
# query with cross join used to create the second materialised view
# another similar one is used to create the view for hourly observations
```

(continues on next page)

(continued from previous page)

```
materialised_view_daily_rentals = """
    DROP TABLE IF EXISTS daily_rentals_all;

    CREATE MATERIALIZED VIEW IF NOT EXISTS daily_rentals_before_2019 AS (

        WITH cross_table AS (SELECT
            d.date AS data_partenza,
            s.nome AS stazione_partenza,
            s.numero_stazione
        FROM bikemi_stalls s
        CROSS JOIN (
           SELECT generate_series (timestamp "2015-06-01"
                                 , timestamp "2018-06-01"
                                 , interval  "1 day")::date
           ) d(date)
        ORDER BY nome, date ASC)

        SELECT
            c.data_partenza,
            c.stazione_partenza,
            c.numero_stazione,
            COUNT(b.*) AS noleggi_giornalieri
        FROM cross_table c
        LEFT JOIN bikemi_rentals_before_2019 b
            ON b.numero_stazione_prelievo = c.numero_stazione
            AND DATE_TRUNC("day", b.data_prelievo)::date = c.data_partenza
        GROUP BY
            c.data_partenza,
            c.stazione_partenza,
            c.numero_stazione
        ORDER BY stazione_partenza, data_partenza ASC
    );
"""

def retrieve_daily_rentals(connection) -> pd.DataFrame:
    query = """
        SELECT * FROM daily_rentals_all;
    """
    return pd.read_sql(query, connection).set_index("data_partenza")

daily_rentals = retrieve_daily_rentals(conn)
```

Then, we compute the number of null values in the data (`null_obs`), pivot the table to wider format and compute the percentage of missing values in each station (`null_obs`). Using Pandas"`cut()` method, we can convert this numerical column into a categorical variable (`null_obs_ranking`) and choose the number of levels - five, in our case - to split the column into even intervals with.

```
obs_number = daily_rentals.index.unique().shape[0]

stations_missing_obs = (
    daily_rentals[[col for col in daily_rentals.columns if "numero_stazione" not in␣
 ↪col]]
        .pivot(columns="stazione_partenza")
        .replace(0, np.nan)
        .isna().sum()
        .sort_values(ascending=False)
```

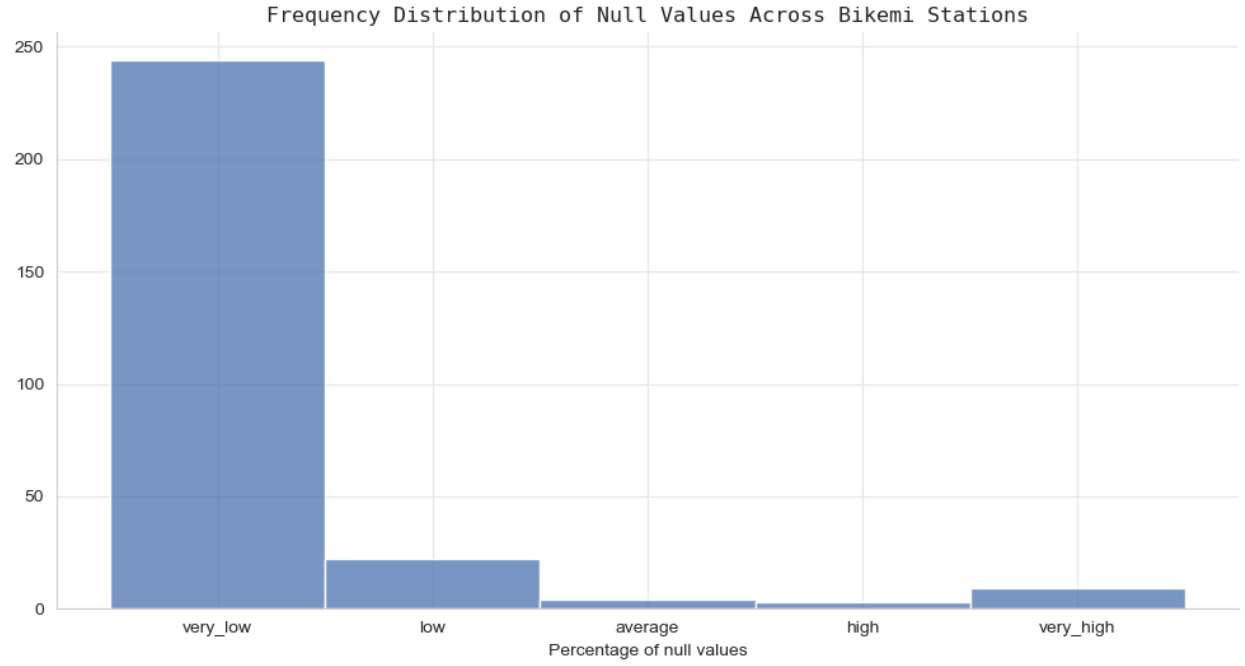
(continues on next page)

(continued from previous page)

```
        .pipe(pd.DataFrame)
        # drop the redundant index
        .droplevel(0)
        .rename({0: "null_obs"}, axis=1)
        # create the % by dividing the number of null values by the length of the␣
 ↪datetime idnex
        .assign(pct_null=lambda x: x.null_obs / obs_number)
)

# labels for the categorical variable, assumed to be ascending order
labels = ["very_low", "low", "average", "high", "very_high"]

stations_missing_obs["null_obs_ranking"] = pd.cut(
    stations_missing_obs["pct_null"], bins=5, labels=labels
)
```

In this way, we can see that the number of stations with an "unacceptable" number of missing values is smaller than thirty.

```
sns.histplot(stations_missing_obs, x="null_obs_ranking")

plt.xlabel("Percentage of null values")
plt.ylabel("")
plt.title("Frequency Distribution of Null Values Across Bikemi Stations", **title_
 ↪font)

sns.despine()

plt.show()
```



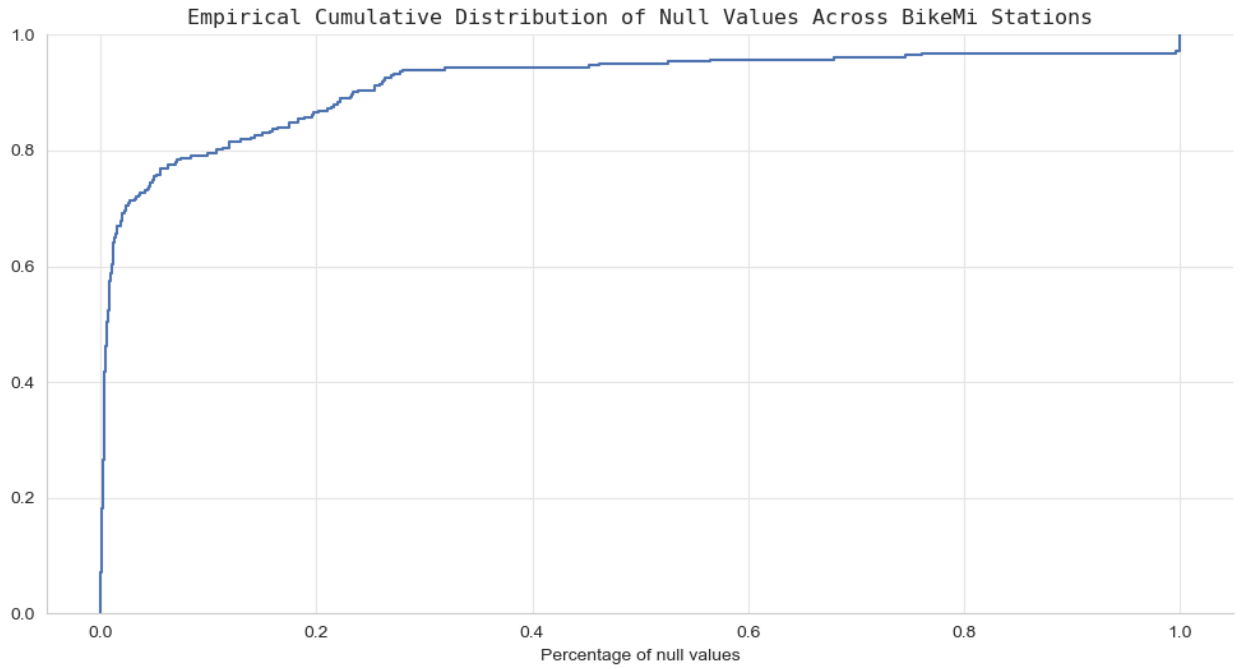
More than 80 percent of the stations display a count of null observations inferior to 20 percent.

```
sns.ecdfplot(stations_missing_obs, x="pct_null")
```

(continues on next page)

(continued from previous page)

```
plt.xlabel("Percentage of null values")
plt.ylabel("")
plt.title("Empirical Cumulative Distribution of Null Values Across BikeMi Stations",␣
 ↪**title_font)

sns.despine()

plt.show()
```



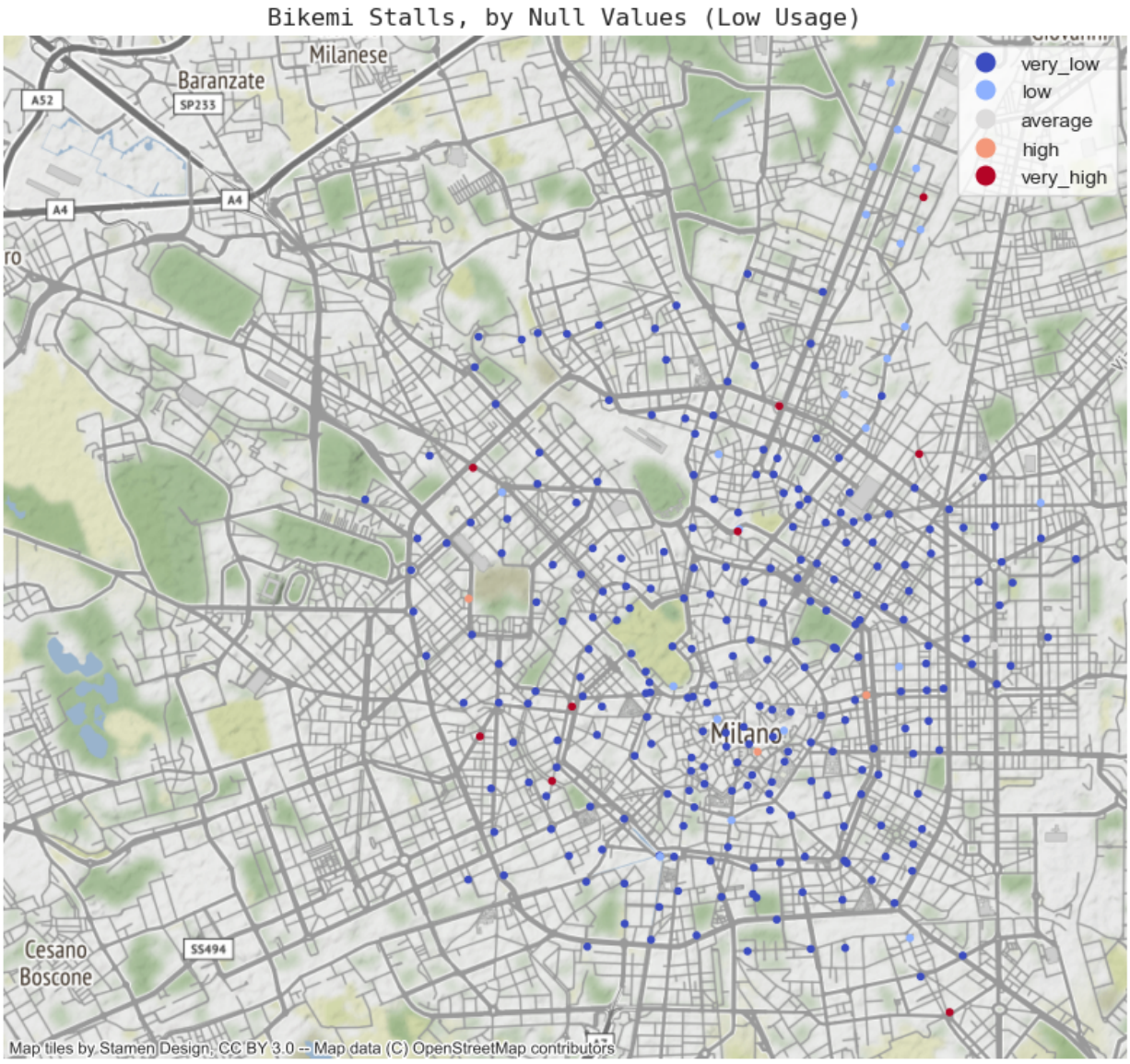
We can also join this data with the station spatial data to plot them. While this does not expose patterns that could be used to reduce the dimensionality of the data, it shows that there are some stations that need to be dropped off even from the most crowded areas in the city. This could not be spotted before as, to the best of our knowledge, no one in the BikeMi literature has ever found a way to take into account of how public work on roads and infrastructure had effects on the bike sharing service.

```
bikemi_stalls_nulls = (
    bikemi_stalls
        .merge(stations_missing_obs, left_on="nome", right_index=True)
)

def plot_stalls_missing_values() -> None:
    fig, ax = plt.subplots(1, 1, figsize=(10, 10))

    (
        bikemi_stalls_nulls
            .to_crs(3857)
            .plot(
                column="null_obs_ranking",
                cmap="coolwarm",
                marker=".",
                legend=True,
                ax=ax
```

(continues on next page)

(continued from previous page)

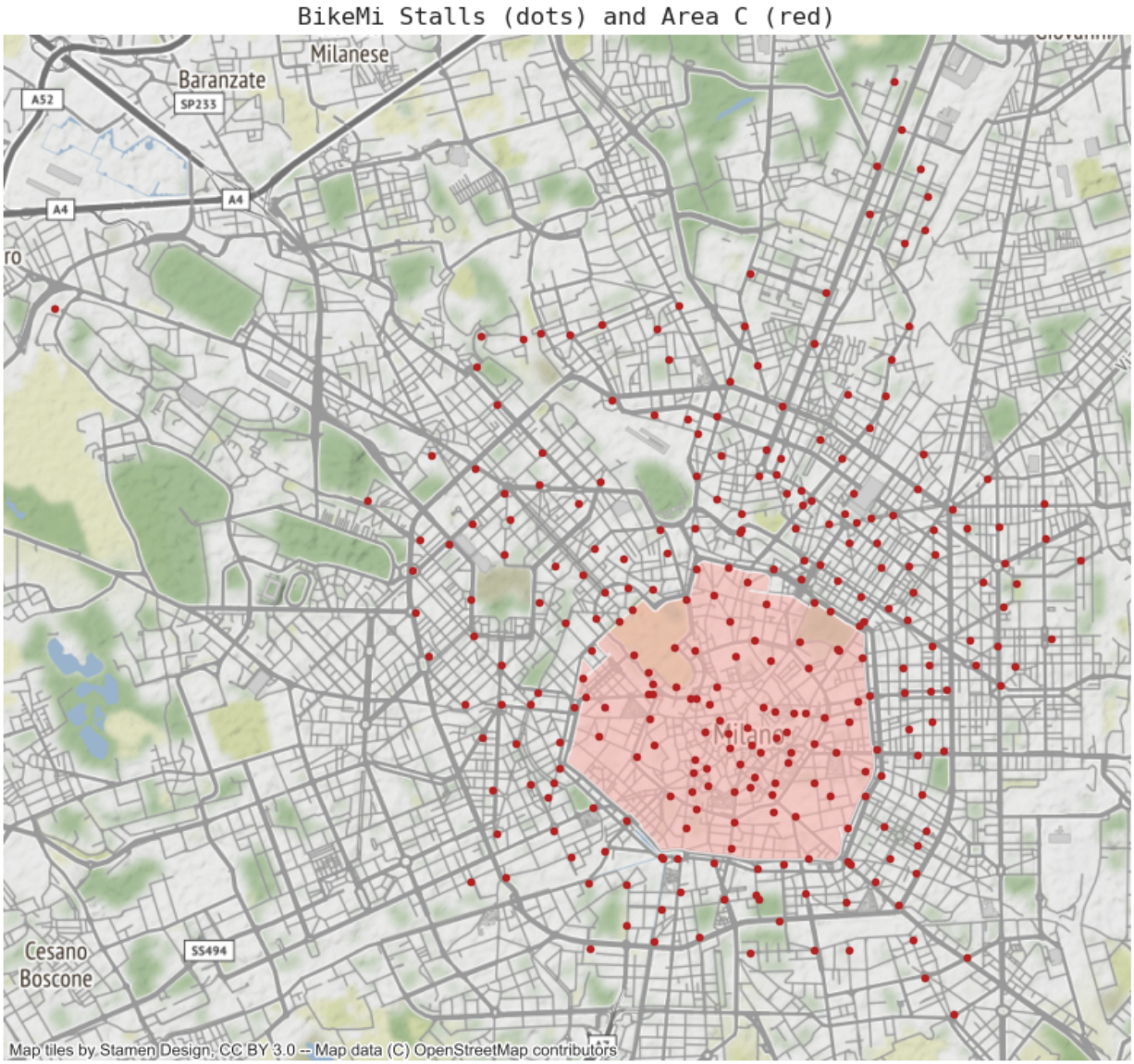
```
        )
    )

    cx.add_basemap(ax)

    plt.axis("off")

    plt.title("Bikemi Stalls, by Null Values (Low Usage)", **title_font)

    plt.show()

plot_stalls_missing_values()
```



3.3.2 Area of Analysis

Besides dropping the "emptiest" stations, we should also come up with a way to narrow down the geographic area inside which we perform the analysis. Previously, authors have chosen to analyse just the area inside the Bastioni, or "Area C" (Saibene & Manzi, 2015) or the whole set of stations (Toro et al., 2020). This was mainly due to the different purpose of their analysis and the data availability (Saibene and Manzi analyse data from 2008 to 2012). For our goal, as well as the policymaker perspective, this choice seems restricting, as it leaves out most of the train stations. It is worth noting that analysing the bike sharing traffic inside of Area C can be appropriate: for example, since, the municipality publishes the number of daily accesses to the area, which could be used to evaluate the effect of sharing services on traffic.

```
area_c = (
    geopandas
        .read_file(Path(milan_data / "administrative-area_c.geo.json"))
        .filter(["tipo", "geometry"])
        .query("tipo == 'AREA_C'")
)

def plot_area_c() -> None:
    fig, ax = plt.subplots(figsize=(10, 10))

    area_c.to_crs(3857).plot(ax=ax, cmap="Pastel1", alpha=0.6)
    bikemi_stalls.to_crs(3857).plot(ax=ax, **bikes_dict)
    cx.add_basemap(ax)

    plt.axis("off")

    plt.title("BikeMi Stalls (dots) and Area C (red)", **title_font)

    plt.show()

plot_area_c()
```



The area we propose is the road ring across Milan, informally referred to as Circonvallazione. This area encapsulates the city centre, as well as most of the train stations, and is represented by the green shade in the map below. This is arguably limiting, as the area does not take in Lambrate and Villapizzone/Bovisa train stations. However, one might argue that it is unlikely for to choose to go to Lambrate and then rent a bike to get to the city centre, as they can just get to Centrale; the same goes for the other stations. In other words, this area contains the sufficient amount of stations to provide a useful forecast for the policymaker, albeit neglecting the outer stalls.

This approach might just reinforce the bias towards the centre of the city, but given the technical constraints seems to be the more viable option. However, as a comparison, the Area C would only contain one train station, whereas this area contains all of the top rentals stalls from the origin-destination (OD) matrix.

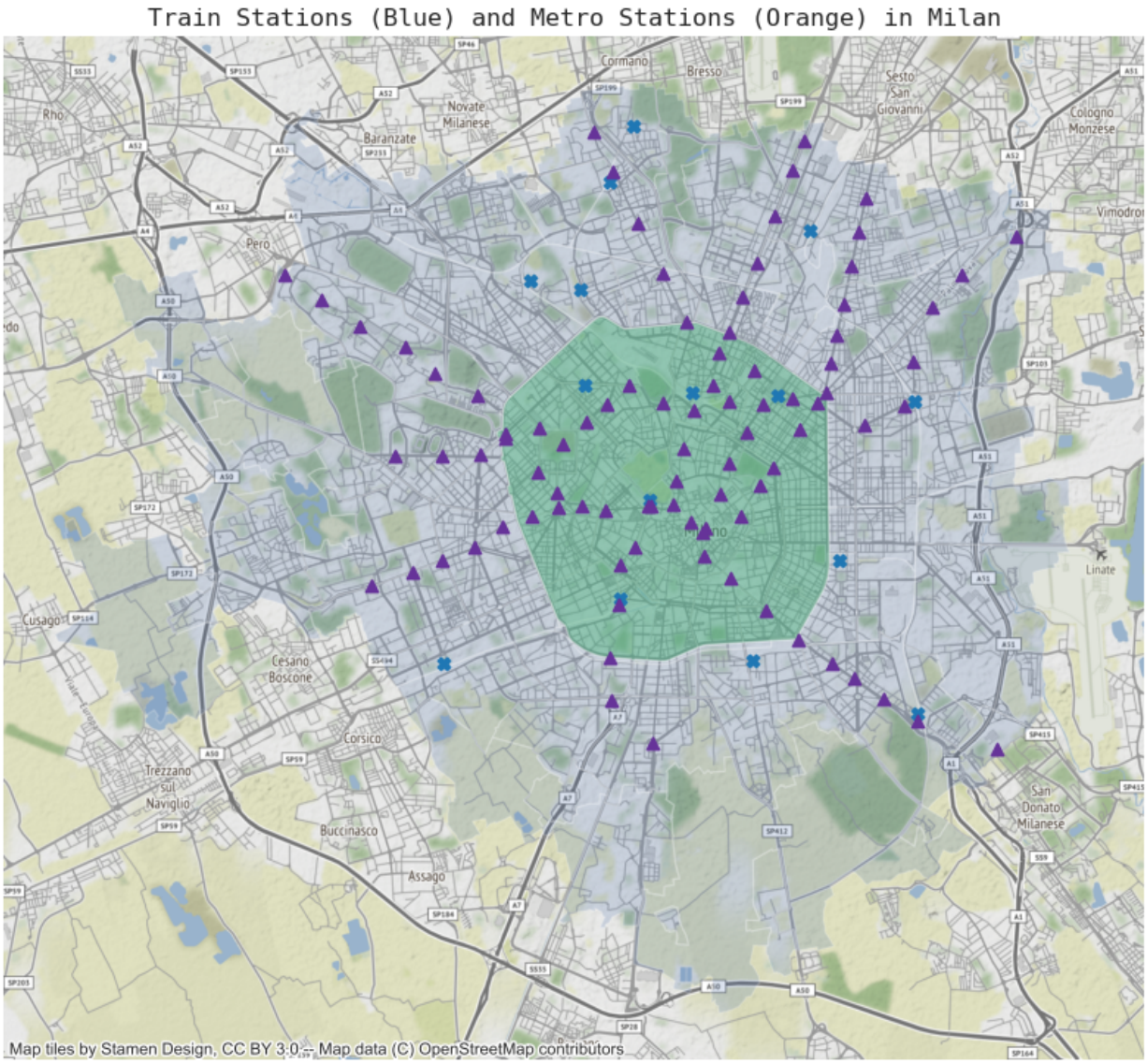
```
metro_stations = geopandas.read_file(Path(milan_data / "transports-metro_stops.geo.
↪json"))

# train_stations = geopandas.read_file(Path(milan_data / "transports-train_stations.
↪zip"))
# train_stations.to_file("../data/milan/milan-transports-train_stations.geo.json",␣
↪driver="GeoJSON")
```

(continues on next page)

(continued from previous page)

```
train_stations = (
    geopandas
        .read_file(Path(milan_data / "transports-train_stations.geo.json"))
        .rename(str.lower, axis=1)
        .rename({"geometry": "train_geometry"}, axis=1)
        .set_geometry("train_geometry")
)

municipi = (
    geopandas
        .read_file(Path(milan_data / "administrative-municipi.geo.json"))
        .rename(str.lower, axis=1)
        .filter(["geometry", "municipio"])
        .rename({"geometry": "municipi_geometry"}, axis=1)
        .set_geometry("municipi_geometry")
)

nil = (
    geopandas
        .read_file(Path(milan_data / "administrative-nil.geo.json"))
        .rename(str.lower, axis=1)
        .filter(["id_nil", "nil", "geometry"])
        .rename({"geometry": "nil_geometry"}, axis=1)
        .set_geometry("nil_geometry")
)

milan = municipi.dissolve().rename({"municipio": "milano"}, axis=1)
```

```
milan_train_stations = train_stations.to_crs(4326).sjoin(milan)
milan_metro_stations = metro_stations.sjoin(milan)

area_circonvallazione = geopandas.read_file(Path(milan_data / "custom-area_
 ↪circonvallazione.geo.json"))

trains_dict = {"color": "tab:blue", "marker": "X"}
metro_dict = {"color": "rebeccapurple", "marker": "^"}

def plot_stations() -> None:

    fig, ax = plt.subplots(1, 1, figsize=(10, 10))
    nil.to_crs(3857).plot(ax=ax, alpha=0.2)
    area_circonvallazione.to_crs(3857).plot(ax=ax, color="mediumseagreen", alpha=0.4)
    milan_train_stations.to_crs(3857).plot(ax=ax, **trains_dict)
    milan_metro_stations.to_crs(3857).plot(ax=ax, **metro_dict)
    cx.add_basemap(ax)

    plt.axis("off")

    plt.title("Train Stations (Blue) and Metro Stations (Orange) in Milan", **title_
 ↪font)

    plt.show()

plot_stations()
```



This leaves us with the following stations, which we store as a `.csv` file and then upload into our local PostgreSQL database with a Bash script.

```
all_stalls: geopandas.GeoDataFrame = (
    bikemi_stalls_nulls
    .sjoin(nil)
    .sort_values("numero_stazione")
    .drop("index_right", axis=1)
    .filter(["nome", "stalls_geometry", "anno", "null_obs_ranking", "nil", "id_nil",
↪"municipio"])
    .rename(columns={"nome": "nome_stazione"})
)

selected_stalls: geopandas.GeoDataFrame = (
    all_stalls
        .query("null_obs_ranking == 'very_low'")
        .sjoin(area_circonvallazione)
        .drop("index_right", axis=1)
```

(continues on next page)

(continued from previous page)

```
)
```

```
train_stations_circ: geopandas.GeoDataFrame = (
    train_stations
        .to_crs(4326)
        .sjoin(area_circonvallazione)
        .drop("index_right", axis=1)
)

metro_stations_circ: geopandas.GeoDataFrame = (
    metro_stations
        .sjoin(area_circonvallazione)
        .drop("index_right", axis=1)
)



def plot_final_data() -> None:
    fig, ax = plt.subplots(1, 1, figsize=(10, 10))

    train_stations_circ.to_crs(3857).plot(ax=ax, **trains_dict)
    metro_stations_circ.to_crs(3857).plot(ax=ax, **metro_dict)
    selected_stalls.to_crs(3857).plot(ax=ax, **bikes_dict)
    cx.add_basemap(ax)

    plt.axis("off")

    plt.title("BikeMi Stalls and Stations Inside the Circonvallazione", **title_font)

    plt.show()

plot_final_data()
```


```
def unpack_geometry(data: geopandas.GeoDataFrame) -> pd.DataFrame:
    df = data.copy()

    df["longitudine"] = df.geometry.x
    df["latitudine"] = df.geometry.y

    return df[[col for col in df.columns if col not in "stalls_geometry"]]

all_stalls.pipe(unpack_geometry).to_csv(milan_data / "bikemi-stalls-with_nils.csv")
selected_stalls.pipe(unpack_geometry).to_csv(milan_data / "bikemi-selected_stalls-
↪with_nils.csv")
```

CHAPTER

FOUR

BIKEMI STALLS *K*-MEANS CLUSTERING

```
# path manipulation
from pathlib import Path
from typing import Union

# data manipulation
import pandas as pd
import geopandas

# plotting
import matplotlib.pyplot as plt
import contextily as cx
import seaborn as sns

# connecting to a database
import psycopg2

from pandas.plotting import register_matplotlib_converters

register_matplotlib_converters()

# set settings for seaborn
sns.set_style(style="whitegrid", rc={"grid.color": ".9"})
sns.set_palette(palette="deep")

# customise matplotlib and sns plot dimensions
plt.rcParams["figure.figsize"] = [12, 6]
plt.rcParams["figure.dpi"] = 100
title_font: dict[str] = {"fontname": "DejaVu Sans Mono"}

# create paths
milan_data: Path = Path("../data/milan")

# establish connection with the database
conn = psycopg2.connect("dbname=bikemi user=luca")

# load stalls data
bikemi_selected_stalls: pd.DataFrame = pd.read_sql("SELECT * FROM bikemi_selected_
 ↪stalls", conn).set_index(
    "numero_stazione")

nils: geopandas.GeoDataFrame = (
    geopandas.read_file(milan_data / "administrative-nil.geo.json")
        .rename(str.lower, axis=1)
        .set_index("id_nil")
```

(continues on next page)

(continued from previous page)

```
        .filter(["nil", "geometry"])
)
```

4.1 *k*-Means Clustering Review

After our first data selection to remove outliers and restrict the spatial area in which we are conducting our analysis, we are still left with more than 200 stations, spread across 25 neighbourhoods out of 88 - identified by the acronym NIL, i.e. *nuclei d'identità locale*. This figure might still be too high, especially as far as multivariate models are concerned: indeed, shrinkage will be necessary in order to avoid highly correlated features (*multicollinearity*). However, it is still in our interests to reduce the number of series to model even for the univariate forecasting: fitting twenty or two-hundred series is a different task. Even inspecting the correlation across series becomes a daunting task with such a great number of features.

To do so, we will use *k*-means clustering - a popular method widely used in the sharing-services literature, especially to identify "virtual stations" in free-float services (Ma et al., 2018) or to "visualize the spatial distribution of DBS [Dockless Bike Sharing] and taxis around metro stations" (Li et al., 2019).

In a few words, with *k*-means clustering we "want to partition the observations into K clusters such that the total within-cluster variation [also known as *inertia*], summed over all K clusters, is as small as possible" (Sohil et al., 2021). The objective function to optimise is usually the squared Euclidean distance. Simply put, *k*-means "aims to partition n observations into K clusters, represented by their centres or means. The centre of each cluster is calculated as the mean of all the instances belonging to that cluster" (Li et al., 2019) and "is extremely efficient and concise for the classification of equivalent multidimensional data" such as sharing services data (Li et al., 2019).

The algorithm begins with randomly choosing clusters centres and, with each iteration, the centres are re-calculated to reduce the partitioning error - which decreases monotonically, as K increases. Basically, in this second step the algorithm "creates new centroids by taking the mean value of all of the samples assigned to each previous centroid [...] until the centroids do not move significantly" (Clustering, n.d.). However, despite similarity always increasing, values of K that are too great defy the purpose of this classification algorithm. For this reasons, practitioners and researchers have come up with more or less sophisticated ways to assess the optimal number of clusters, the most common of which is the so-called "Elbow method". In a few words, a chosen performance metric is computed for each number of clusters and the optimum is represented by the point at which the performance improvements start to marginally decline.

k-Means clustering scales well with the number of samples n, but assumes convex shapes (i.e., has worse performances where the "true" clusters have elongated or irregular shapes) (Clustering, n.d.). Besides, since the initial position of the cluster is random, it might take some attempt for the algorithm to converge. More importantly, however, *k*-Means is sensitive to the scales of the variables in the data, so normalising the feature matrix is a crucial step. Finally, *k*-Means assumes continuous data and is not designed to handle categorical features: "[T]he *k*-means algorithm only works on numerical data, i.e. variables that are measured on a ratio scale [...], because it minimises a cost function by changing the means of clusters. This prohibits it from being used in applications where categorical data are involved" (Huang, 1998).

Some workarounds have long been studied in the literature: as an example, by changing the distance metrics from the Euclidean distance to the Gower distance, which is a measure of similarity (Gower, 1971), *k*-Means can be adjusted to cluster categorical variables. More recently, new methods have been developed, such as the *k*-Modes (Huang, 1998). These alternative methods rose in popularity because the "quadratic computational costs" of similarity-based algorithms are not suited for the larger and larger datasets we have been dealing with, especially in the past years.

Our context does not warrant new, sophisticated clustering algorithms. The sample size of the spatial distribution of stalls would even be small enough to justify an attempt with similarity scores. Yet, to the best of our knowledge, this procedure has not been tried yet in the literature, and there are plenty of good reasons for doing so.

As a starter, the categorical variables we have - like the neighbourhood - are just another measure of geographic distribution. It would be much more meaningful to run the *k*-means algorithm with other, quantitative features, such as the number of stations or bike lanes in the proximity of the station. The nature and number of buildings, like the extension

of parks or traffic measurements, could be used, too. This procedure of data collection should take a lot of time and diversified skills and would primarily accomplish the goal to provide a better segmentation of the individual stations: as an example, it could represent an improvement over choosing neighbourhood fixed effects. While the level of accuracy of such an approach could be much greater, this is nothing that would not be similarly accomplished by "arbitrarily" selecting stations close to each other (say, within a radius of a couple of hundred metres, or that varies according to the population density of the area). Given the nature of our task - forecasting - we shall stick to the usage of *k*-means that is prevalent in the literature: identify virtual stations to reduce dimensionality and perhaps offer new insights as far as data analysis is concerned.

```
from sklearn.metrics import silhouette_score, davies_bouldin_score, calinski_harabasz_
↪score
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.pipeline import make_pipeline


def filter_coords(data: pd.DataFrame, cols=None) -> pd.DataFrame:
    if cols is None:
        cols = ["longitudine", "latitudine"]
    return data.filter(cols)


selected_stalls_lonlat: pd.DataFrame = bikemi_selected_stalls.pipe(filter_coords)
```

4.2 *k*-Means Clustering Evaluation Metrics

The main, if not only, hyperparameter to tune in *k*-means is, indeed, the number of clusters. As most machine learning methods, *k*-means is a quite old algorithm and has thus been widely studied. The first metric to look at is indeed the inertia, or *within-cluster sum-of-squares* (WSS), which is usually computed using the Euclidean norm:

$$\sum_{i=0}^{n} \underset{\mu_j \in C}{min}(||x_i - \mu_j||^2)$$

Other norms can be used; however, some other performance metrics can only be calculated with the Euclidean distance. While clustering is usually presented as an unsupervised machine-learning technique, a first set of metrics is actually computed comparing the clustering with the ground truths. This is necessary because the evaluation metrics should not take into account the absolute number of clusters, yet if this clustering defines "separations of the data similar to some ground truth set of classes" **SKLEARN**.

Among these metrics there are the *Rand index*, mutual information-based scores, measures of completeness and homogeneity (or both, such as the *V-measure*) and Fowlkes-Mallows scores. There is, however, another category of metrics that does not require a comparison with the ground-truth: the *silhouette coefficients*, the *Calinski-Harabasz index*, computed as and the *Davies-Bouldin index*. The dowside of these evaluation metrics is that they are not robust against "unconventional" shapes and tend to display "better" values when the clusters are convex.

These measures are computed for each set of data points and the labels assigned with the *k*-means. The silhouette score is computed as $s = \frac{b-a}{max(a,b)}$, where a is the mean distance between a sample and all points in the same class and b is the average distance between the sample and the points in the *next nearest* cluster. The silhouette coefficients for all samples are averaged and a higher silhouette coefficient relates to a model with better defined clusters **SKLEARN**. For this reason, the silhouette score is defined within $[-1; 1]$, and scores closer to -1 denote incorrect clustering and values near 0 indicate overlapping data points.

The Calinski-Harabasz Index is defined as:

$$s = \frac{\text{tr}(B_k)}{\text{tr}(W_k)} \times \frac{n_E - k}{k - 1}$$

Where B_k is the dispersion between clusters and W_k is the one within clusters and n is the number of observations in the dataset E. Because of this, the index is a raw number (i.e., it is not normalised between, say, 0 and 1). Since we want the clusters to be as far apart as possible and the points inside to be as close as possible, we will choose K such as it maximises the Calinski-Harabasz index.

Finally, the Davies-Bouldin index indicates a better separation between clusters the lower it is, and is computed as the average similarity across the two closer classes C_i and C_k. The similarity measure between C_i and C_k is called R_{ij} and is defined as $R_{ij} = \frac{s_i + s_j}{d_{ij}}$, where s is the average distance between each point of a cluster and its centre, and $d_{ij} is the distance between the two clusters' centroids. The Davies-Bouldin index is thus formally defined as such:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij}$$

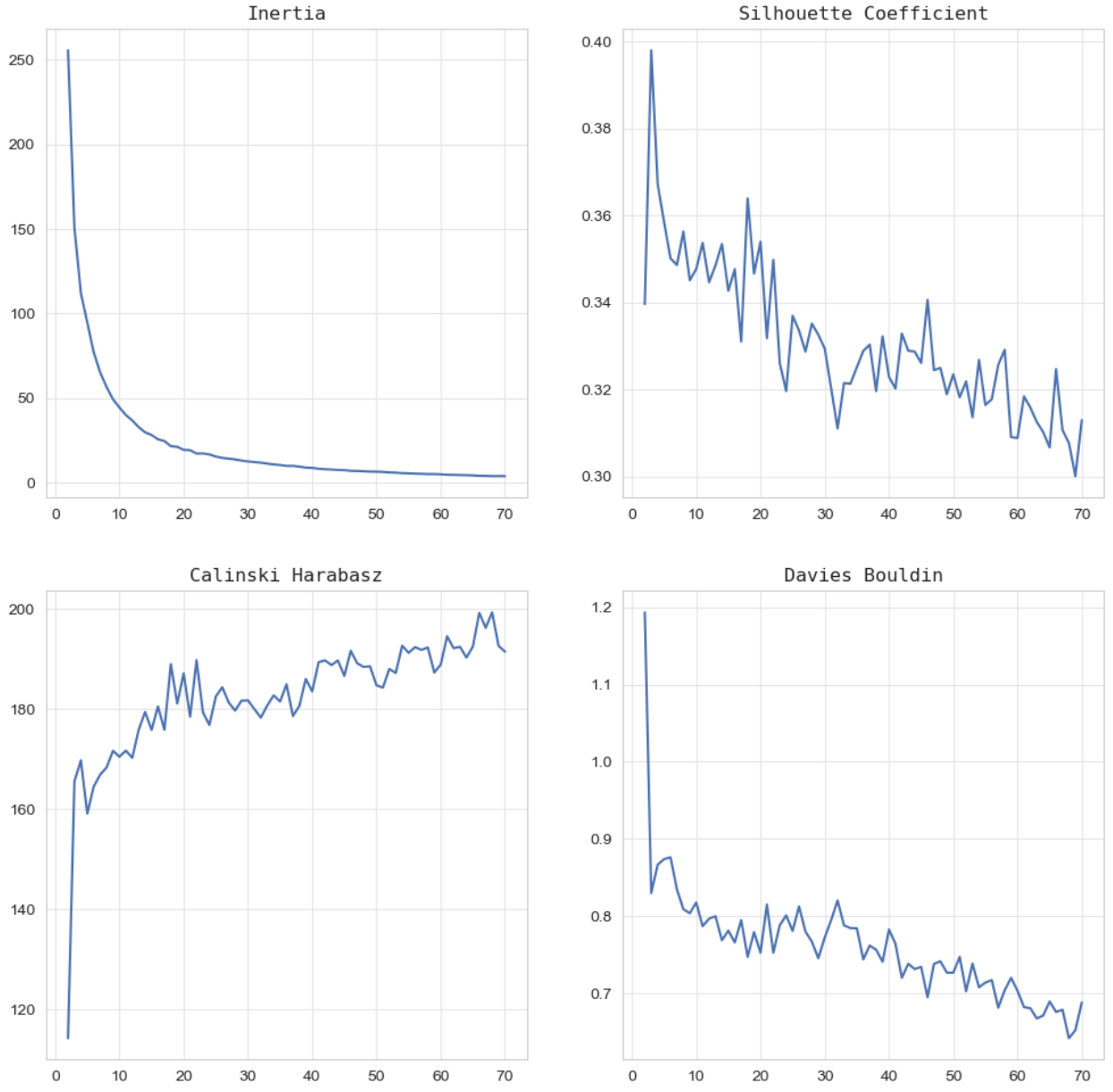
4.3 Fitting *k*-Means on BikeMi Stalls Geographic Data

```
def get_kmeans_metrics(data, k_max, random_state) -> pd.DataFrame:
    scaled_data = StandardScaler().fit_transform(data)
    k_range = range(2, k_max + 1)

    Metrics = tuple[float, float, float, float]

    def compute_scores(source_data, num_clusters, rand) -> Metrics:
        # fit kmeans
        kmeans = KMeans(num_clusters, random_state=rand).fit(source_data)

        # scores
        inertia = kmeans.inertia_
        silhouette_coefficient = silhouette_score(source_data, kmeans.labels_, metric=
↪"euclidean")
        calinski_harabasz = calinski_harabasz_score(source_data, kmeans.labels_)
        davies_bouldin = davies_bouldin_score(source_data, kmeans.labels_)

        return inertia, silhouette_coefficient, calinski_harabasz, davies_bouldin

    scores: list[Metrics] = [compute_scores(scaled_data, k, random_state) for k in k_
↪range]

    output: pd.DataFrame = pd.DataFrame(
        scores,
        index=k_range,
        columns=["inertia", "silhouette_coefficient", "calinski_harabasz", "davies_
↪bouldin"]
    )

    output.index.name = "k"

    return output


def plot_all_metrics(metrics_data: pd.DataFrame) -> None:
    def plot_metric(dataf: pd.DataFrame, col: str, _ax: plt.Axes):
        plot_title: str = col.replace("_", " ").title()
        _ax.plot(dataf[col])
        _ax.set_title(f"{plot_title}", **title_font)
```

(continues on next page)

(continued from previous page)

```
    fig, ax = plt.subplots(2, 2, figsize=(12, 12))

    for ax, metric in zip(ax.reshape(-1), metrics_data.columns):
        plot_metric(metrics_data, metric, ax)


selected_stalls_metrics = get_kmeans_metrics(selected_stalls_lonlat, 70, 42)

plot_all_metrics(selected_stalls_metrics)
```



The WSS or inertia is not really indicative of a significant value of K. The silhouette coefficient is greatest for a quite small number of clusters (smaller than 5) and in general not greater than 20. This appears to be the case for the Calinski-Harabasz index too and, while the same cannot be said of the Davies-Bouldin index, beyond said threshold the index is bound to improve. Our final choice should fall on 18, as we believe that clustering the whole set of stations into three

great clusters bears no practical utility for the purpose of the policymaker - forecasting the bikes demand in Milan.

However, as mentioned above, the total number of neighbourhoods inside our area of analysis is 25: at this point, the policymaker might just be better off aggregating the data by neighbourhood, without going through the hassle of computing the *k*-means. Indeed, the purpose of fitting a *k*-means should be to isolate a small amount of clusters inside of each neighbourhood, to aggregate together stations that are quite close to each other and that cannot all be used at once as independent variables for the forecasting models without injecting error in the estimates via multicollinearity.

```
def fit_kmeans(input_data, k, rand) -> object:
    return make_pipeline(StandardScaler(), KMeans(n_clusters=k, random_state=rand)).
 ↪fit(input_data)


def assign_clusters(input_data: pd.DataFrame, fitted_kmeans) -> pd.DataFrame:
    return input_data.assign(cluster=fitted_kmeans.labels_)


def make_geodataframe(
        data: pd.DataFrame,
        cols=None,
        crs: int = 4326
) -> geopandas.GeoDataFrame:
    if cols is None:
        cols = ["longitudine", "latitudine"]
    return data.pipe(
        geopandas.GeoDataFrame,
        geometry=geopandas.points_from_xy(
            data[cols[0]],
            data[cols[1]],
            crs=crs)
    )


def plot_clusters(
        geodata: geopandas.GeoDataFrame,
        cluster_col: str,
        layer: Union[geopandas.GeoDataFrame, None]
) -> None:
    fig, ax = plt.subplots(1, 1, figsize=(12, 12))

    ax.axis("off")

    num_of_clusters: int = geodata[cluster_col].nunique()
    ax.set_title(f"Bikemi Stalls, Result of KMeans Clustering ($k$ = {num_of_clusters}
 ↪)", **title_font)

    if layer is not None:
        layer.to_crs(3857).plot(ax=ax, cmap="summer", alpha=0.05)
    geodata.to_crs(3857).plot(column=cluster_col, ax=ax)
    cx.add_basemap(ax=ax)

    plt.show()
```

```
kmeans_18 = fit_kmeans(selected_stalls_lonlat, k=18, rand=42)

selected_stalls_18k: geopandas.GeoDataFrame = (
    selected_stalls_lonlat
        .pipe(assign_clusters, fitted_kmeans=kmeans_18[-1])
```

(continues on next page)

(continued from previous page)

```
        .pipe(make_geodataframe)
)

plot_clusters(
    selected_stalls_18k,
    cluster_col="cluster",
    layer=nils.sjoin(selected_stalls_18k)
)
```

Bikemi Stalls, Result of KMeans Clustering (k = 18)

If we sort the metrics matrix by the values of `sihlouette_coefficient`, the first number of clusters greater than 25 than we meet is 46. This, however, is only the 18th best value for K according to the `silhouette_score`.

```
def get_k_greater_than_25(metrics_data: pd.DataFrame, metric: str = "silhouette_
↪coefficient") -> pd.DataFrame:
```

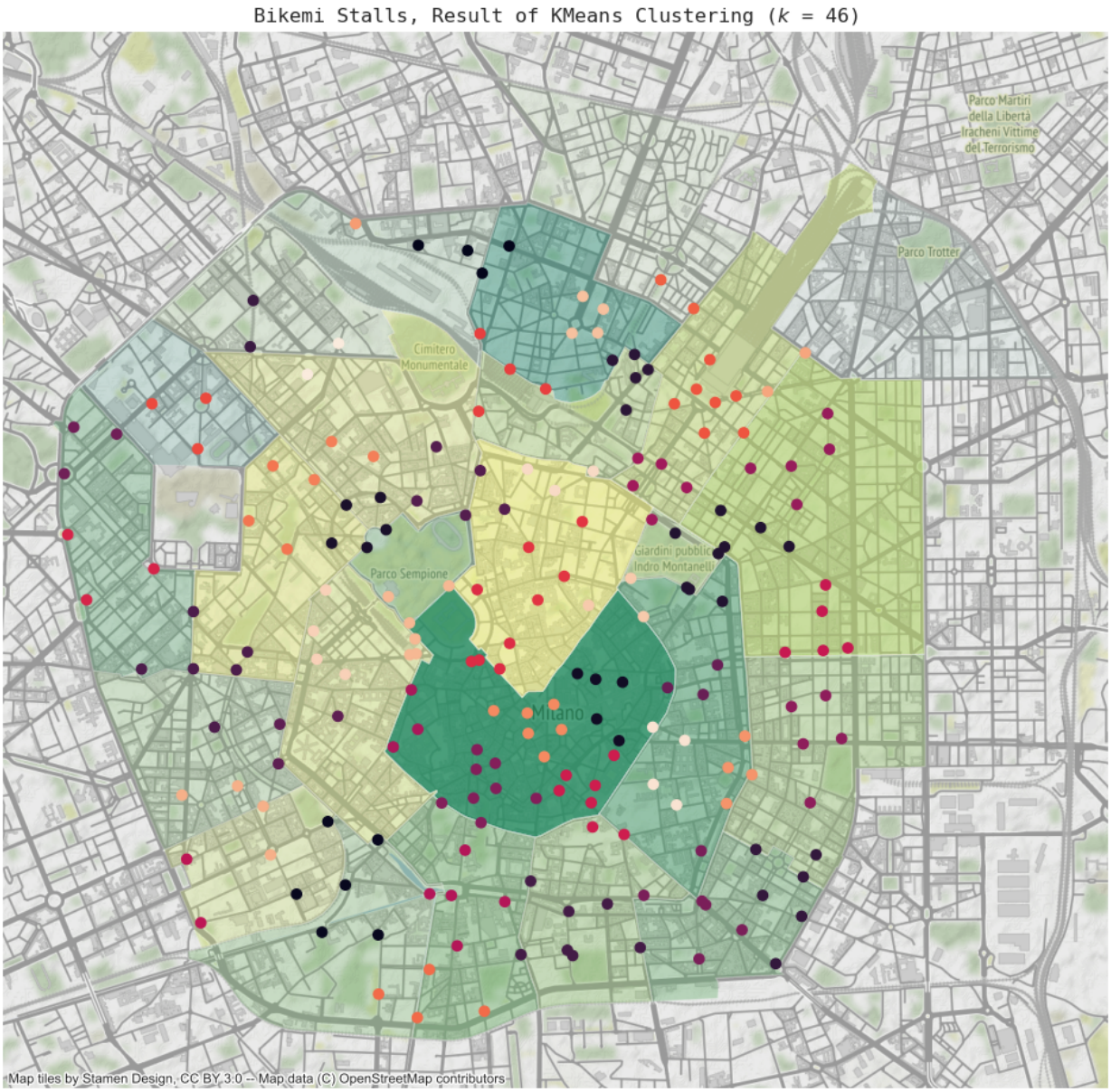
(continues on next page)

(continued from previous page)

```
    return metrics_data.sort_values(metric, ascending=False).
 ↪assign(ranking=range(metrics_data.shape[0])).query(
        "k > 25").head(1)


get_k_greater_than_25(selected_stalls_metrics)
```

```
     inertia  silhouette_coefficient  calinski_harabasz  davies_bouldin  \
k
46  7.098486                0.340648         191.620618        0.694719

    ranking
k
46       18
```

```
kmeans_46 = fit_kmeans(selected_stalls_lonlat, k=46, rand=42)

selected_stalls_46k: geopandas.GeoDataFrame = (
    selected_stalls_lonlat
        .pipe(assign_clusters, kmeans_46[-1])
        .pipe(make_geodataframe)
)

selected_nills_46k: geopandas.GeoDataFrame = nils.sjoin(selected_stalls_46k)

plot_clusters(
    selected_stalls_46k,
    cluster_col="cluster",
    layer=selected_nills_46k
)
```



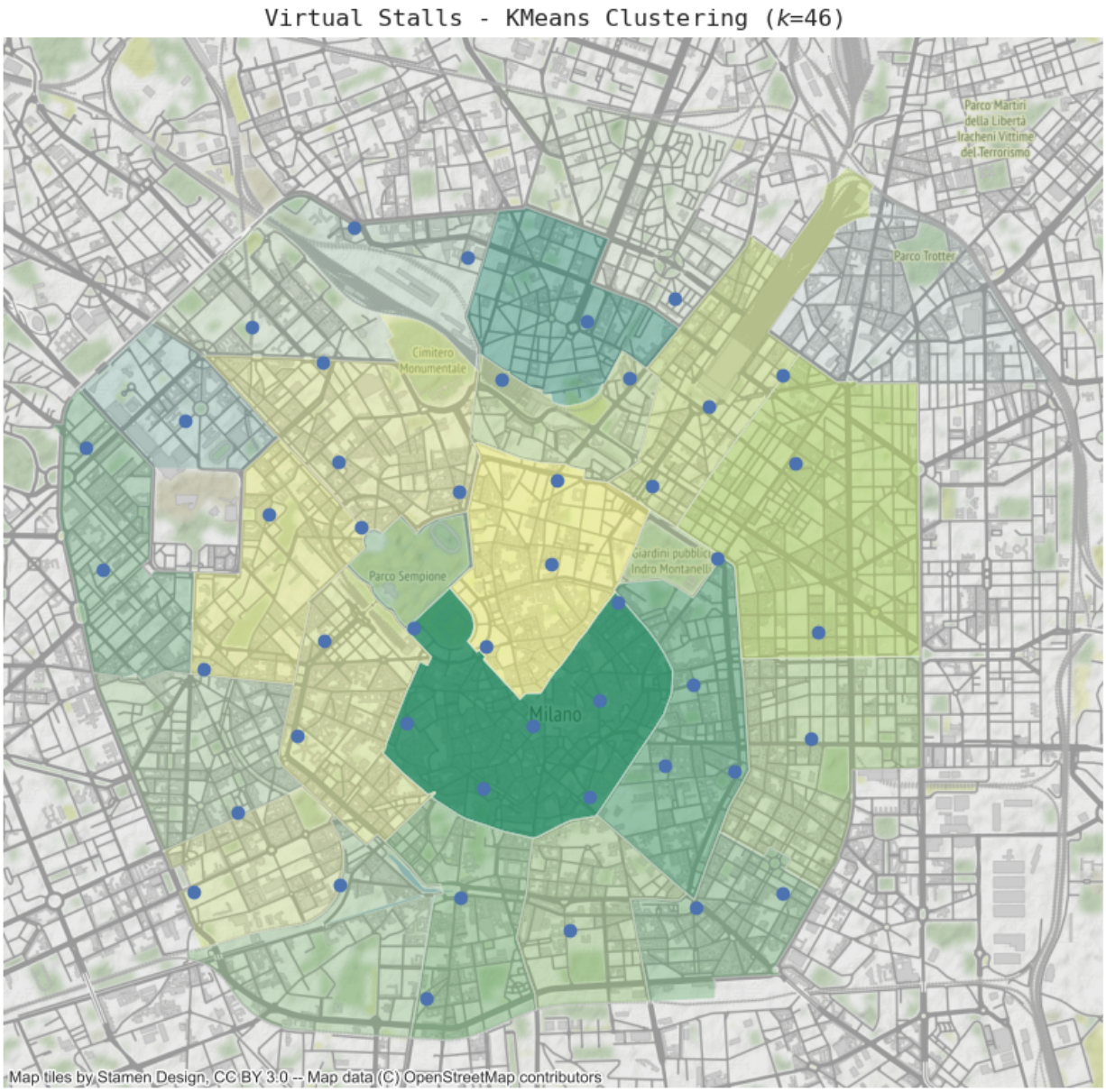
After choosing the value for K, stall data is aggregated to compute the coordinates of these new "virtual stalls" as the average of the coordinates of the stations inside the cluster. Because of how spatial joins work, one neighbourhood is not plotted, and there seems to be one cluster without left without a NIL. However, the identifier (`12`, i.e. Maciachini - Maggiolina) is still correctly assigned. As the last step, these virtual clusters are assigned with a join operation to the list of selected stalls, and exported.

```
def plot_virtual_clusters(
        virtual_clusters: geopandas.GeoDataFrame,
        layer: Union[geopandas.GeoDataFrame, None]
) -> None:
    fig, ax = plt.subplots(1, 1, figsize=(10, 10))

    if layer is not None:
        layer.to_crs(3857).plot(ax=ax, cmap="summer", alpha=0.05)
    virtual_clusters.to_crs(3857).plot(ax=ax)
```

(continues on next page)

(continued from previous page)

```
    cx.add_basemap(ax=ax)

    plt.axis("off")
    plt.title("Virtual Stalls - KMeans Clustering ($k$=46)", **title_font)

    plt.show()


virtual_stalls: geopandas.GeoDataFrame = (
    selected_stalls_46k
        .groupby("cluster")[["longitudine", "latitudine"]]
        .mean()
        .pipe(make_geodataframe)
        .rename({"longitudine": "lon_cluster", "latitudine": "lat_cluster"}, axis=1)
        .sjoin(nils, how="left")
        .rename({"index_right": "id_nil"}, axis=1)
)

plot_virtual_clusters(virtual_stalls, layer=selected_nills_46k)
```




```
clustered_selected_stalls: pd.DataFrame = (
    bikemi_selected_stalls.filter(["nome_stazione"])
        .join(selected_stalls_46k.drop(columns="geometry"))
        .join(virtual_stalls.drop(columns="geometry"), on="cluster")
        .rename(columns={"id_nil": "cluster_id_nil", "nil": "cluster_nil"})
)

clustered_selected_stalls.to_csv(milan_data / "bikemi-selected_stalls-clusters.csv",␣
↪index=True)
```