Technische Universität München
Department of Civil, Geo and Environmental Engineering
Chair of Cartography
Prof. Dr.-Ing. Liqiu Meng

Combining Open-source Programming Languages with GIS for Spatial Data Science

Maja Kalinic

Master's Thesis

Duration: 01.04.2017 - 30.09.2017

Study Course: Cartography M.Sc.

Supervisor: Dr. Mathias Jahnke (TUM)
Prof. Dr. Menno-Jan Kraak (ITC)
Dr. Jan Wilkening (Esri Deutschland GmbH)

Cooperation: Esri Deutschland GmbH, Kranzberg

2017

Table of Contents

DECLARATION OF AUTHORSHIP

I hereby certify that this thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information, have been specifically acknowledged.

Date: Signature:

22.09.2017. Maja Kalinic

ACKNOWLEDMENTS

After an intensive period of six months, today is the day: writing this note of thanks is the finishing touch of my master thesis work. It has been a period of intense learning for me, not only in the scientific domain, but also on a personal level. I would like to reflect on the people who have supported and helped me so much throughout this period.

I would like to express my sincere gratitude to my supervisors, Dr. Mathias Jahnke, Prof. Dr. Menno-Jan Kraak and Dr. Jan Wilkening, for the continuous support during my thesis research, for your patience, motivation, and immense knowledge. Your guidance helped me in all aspects of this research and writing of this thesis. I could not have imagined having a better supervisors for my master thesis work.

Spatial thanks to Corné van Elzakker, Master's thesis semester coordinator, for his insightful comments and encouragement, but also for the motivation questions which stimulated me to widen my research from various perspectives.

My special thanks go to my friend, muse and one of the greatest person that I have ever met, Juline Cron, our program coordinator. Thank you for all your support, patience, generosity and kindness. All this would never be possible without you being always there for me, saying the right words in right time and making me believe in myself to the very end.

I would like to thank all my friends, my boyfriend and my loved ones, who have supported me throughout entire process, both by keeping me harmonious and helping me putting pieces together. I will be grateful forever for your love and support. Special thanks to my friend Victoria Curl, for proofreading and inspirational words. Final special thanks go to my mom, who has been with me in the hardest moments and inspired me to never give up.

Thank you all for being with me on this journey.

Maja Kalinic

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Extracting knowledge from data has always been a challenge for a wide range of professionals which includes scientists, statisticians, programmers and many others. Information visualization and communication is, on the other hand, a challenge for Cartographers and GIS professionals since their aim is to visually deliver useful information to aid humankind and their sustainability. To achieve this aim theories and techniques from broad professional fields such as maths, statistics, probability, computer science, as well as knowledge from machine learning, classification, cluster analysis, data mining, databases, and visualization had to be unified.

This master thesis aims to meet these challenges by investigating which tools and software are suitable for solving the afore mentioned tasks. Therefore, three main objectives are explored. The first refers to a decision of which programming languages, among a variety, are suitable for integration with GIS software for enhancing spatial analysis and predictions. The second deals with parsing the work between chosen tools in the most efficient way. The third focuses on an easy to understand real world example whose implementation either supports or denies the made choices.

This thesis consists of two main parts. Theoretical part which gathers available literature in the domain of interest and which delivers conclusions upon which software and tools to use (in this case those were Python, R and ArcGIS Pro). Practical part which takes out real world data about San Francisco crimes and through exhaustive and massive analysis delivers conclusions suitable for police departments to deploy while relocating their forces.

Python and R stand-alone scripts (Appendix 1 and 2) are used for exploratory analysis over the San Francisco criminal data. These analyses show which crimes are the most prominent ones in San Francisco, which district is the most dangerous/safest one, which month/day/hour of the week has the highest criminal activity, percentage of crime resolution and possible dependency between crime occurrence and variables from the dataset. ArcgGIS Pro is used for inferential analysis and predictions where crimes are most likely to occur. These analyses enable insight into areas with persistent and intensifying crime occurrence and areas with unusually high numbers of crimes while accounting for the population. Further on, R-ArcGIS Bridge gives potential explanations about correlations between income, population and median house value which can be used as input variables while building a predication model. Additionally, Python is used for building a predictive model considering prominent variables from the dataset. However, results were not satisfying, thus, this part of the study is left to be explored in more details for some future work or research.

The conducted work suggests that combination and integration of programming languages, such as Python and R, with the GIS software, in this case ArcGIS Pro, leverages and provides information extraction and visualisation. For further work, cooperation between Cartographers, Data Scientist and Crime Analysts would be highly productive, beneficial and recommended.

CHAPTER

I INTRODUCTION

1.1. Motivation and problem statement

There is GIS, specifically designed software for spatial data handling, there are programming languages, a set of instructions forwarded to a machine for getting desired outputs, and it is a challenge to know how, but important to link them.
The volume of data grows rapidly (80% of data possesses a geographic reference (Dempsey, 2012)) and the urge to retrieve information from it increases significantly. Thus, the need for a rich and broad array of open source tools for data headlining is emerging. This master thesis aims to investigate different open source programming languages regarding their usefulness for spatial data handling. How can these languages be combined with different Desktop or Web GIS applications to gain insights from large volume of spatial data? And how can the work efficiently be divided between the open-source programming languages and the GIS? Which languages are relevant and why?

These questions have been a subject of discussion among different professionals including scientists, statisticians, programmers and many others. Data Science term has emerged recently to specifically designate a new profession that is expected to make sense of the vast stores of big data. According to (Hayashi, 1998) "Data Science intends to analyse and understand actual phenomenon with data. In other words, the aim of Data Science is to reveal the features or the hidden structure of complicated natural, human and social phenomena with data from a different point of view from the established or traditional theory and method". To achieve this aim theories and techniques from broad professional fields such as maths, statistics, probability, computer science, as well as knowledge from machine learning, classification, cluster analysis, data mining, databases, and visualization had to be unified.

At this point, one can ask what does all this have to do with GIS? Why should anyone use GIS when there are all these open-source programming languages?
The fact is that the GIS community (whose knowledge domain is in the geographic data) is growing rapidly and it ranges from indigenous people, communities, research institutions, environmental scientists, health organisations, government agencies, etc. But, it is often the case that GIS users do not possess programming skills and the open source programming languages are not providing the comfort of a user-friendly interface as the GIS software does. However, open source programming languages communities are even larger and grow way faster than GIS's, due to the number of their users and contributors, ranging from programmers, developers, data scientist, analysts, etc. Therefore, significant amount of open access materials is provided and at service for usage, editing, redistributing for anyone who needs to employ the power of these tools. On behalf, of that, combined GIS and open-source programming languages provide access to variety of mapping and modelling tools for solving challenging tasks despite the data, and despite the analysis.

Further consideration goes towards the data itself. Where is the data that is relevant and worth processing? To which fields is data related? The emphasis in this thesis work will be on spatial data whose application varies among different domains (among which Public Safety domain). Additionally, spatial data presents a new set of challenges and opportunities for Cartography. This explicitly covers applications in technical, methodological and artistic domains since Cartography as an art and science (Krygier, 2013) constantly seeks to improve in finding solutions for problems important for humankind and its sustainability.

1.2. Research identification

The overall objective of this research is to identify and evaluate the most relevant open-source programming languages as well as how they can be combined with GIS. The aim is to help cartographers to more easily understand and use available tools such as programming languages libraries and packages, so that they can analyze and visualize (big) spatial data sources available nowadays, and map results.
The scope of this thesis is on identifying the two most important open-source programming languages and one GIS platform for testing purposes. Considering that, fallowing individual objectives must be defined:

1. Decision upon which open-source programming language/(s) is/are the most prominent one(s) to be combined with GIS for the analysis and interpretation of spatial data.
2. Decision upon how it/they should be combined with GIS. The most efficient way of dividing the work between open-source programming languages and GIS.
3. Decision upon which dataset to use and methodology to develop real world example scenario to prove the afore mentioned choices.

1.3. Research questions

To meet the above set objectives, many accompanied questions must be answered.

- What are open-source programming languages, and which of those can be combined with GIS?
- What would be the most efficient way of combing these languages with GIS?
- How to divide the work between programming languages and GIS platform in a reasonable way?

Additionally, it is very important and necessary to have clear idea of:

- Where to get proper datasets for testing purposes and what type of data that should be considering that the case study should be an easy-to-understand real-world example which supports the research objectives.
- Which work is done by Data scientists and which by GIS professionals, and are there any interactions among them?

This master thesis consists of six chapters. Graphical representation of the thesis structure is shown on the Figure 1.

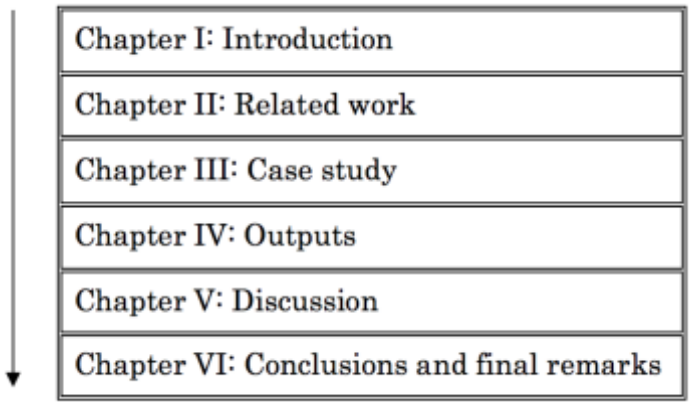

Figure 1: Thesis structure

First chapter starts with the topic introduction. It covers the problem statement, motivation and research objectives accompanied with the further research questions. Second chapter gives an overview of the related work. It is divided into four main sections (Open-source, Programming languages, GIS and Data Science). Moreover, it covers the literature findings about Crime analysis and application of the chosen tools/software in this domain. *Third chapter* is a case study and it is seen as a practical part of this thesis. It introduces the data, the methodology and the workflow in accordance to methodology steps. *Fourth chapter* gives an outputs overview having in mind research objectives and research questions. *Fifth chapter* covers the discussion of the findings. *Sixth* chapter includes conclusions, encountered limitations and introduces ideas for the future work.

II RELATED WORK

In order to accomplish research objectives and thesis demands, it was necessary to investigate existing work in this domain and develop scenarios for practical examples according to met discoveries. Due to the diversity of included fields (GIS, Programming languages, Data Science) found literature was partitioned and every section was described individually. The chapter *Related work* consists of four sections. *Open-source* gives a clear and coherent explanation of the term itself, a description of what it mean for any software or platform in general, and how it is applicable to programming languages. *Programming languages* goes through a variety of offered languages with the emphasis on GIS oriented ones and their application in specific domains and fields. *GIS* gives an overview of the most widely used geographic information software, either in commercial, science or community environments, as well as examples of their widely-used applications in the domain of Data Science. Additionally, this chapter explains how some of the prominent languages are being paired with GIS. *Data Science* gives an overview of the term itself, its application, its relation to GIS and its involvement in Big Data handling.

Due to the necessity to develop the case study scenario, as mentioned in the beginning of this chapter, an additional literature review was explored and added respectively. The prerequisite for this task was to focus on finding data which was, later on, used within the case study, its source and background as well as relationship with the research objectives. After describing the data, the literature review continued towards finding relevant tools and functionalities which supported case study analysis.

2.1. Open-source

The development and popularity of Free and Open Source Software (FOSS) has grown rapidly in past few years. In the GIS domain, this is measured either by the number of projects started in the past few years, financial support from government organisations, download rate, or increasing number of use cases noted by accompanying web sites and organisations (Steiniger & Bocher, 2009). Additional contributions to the popularity rise are research publications on this topic, as well as research work published under the open licence.

It is important to distinguish some of the most common used terms and domains such as “proprietary”, “free” and “open source”. These definitions have been set up by the Free Software Foundation (FSF) for free software, the Open Source Initiative (OSI) for open source and the third domain covers software which are excluded from two afore mentioned. According to (Steiniger & Bocher, 2009), the software is free if it grants four freedoms:

- The freedom to run the program for any purpose.
- The freedom to study how the program works and adopt it to your needs.
- The freedom to redistribute copies so you can help your neighbours.
- The freedom to improve the program, and to release your improvements to the public so that the whole community benefits.

However, these four freedoms do not indicate whether the software is offered for free or can be sold. For this reason, the opposite domain of free software is proprietary which stands for ownership. Very often free and open source terms are being used synonymously which should not be the case. First because the term open-source is insufficient, it does not explain the freedoms of modification and redistribution. Second, the term has been proposed by Open Source Initiative as a kind of a brand, but to gain certification it must conform to definitions similar to the four freedoms.

While open-source technology trends keeps growing, many have discussed advantages and disadvantages of adopting such systems compared to propriety. (Donnelly, 2010) tests and compares FOSS GIS applications with Esri's ArcGIS Desktop to determine the suitability of adopting FOSS GIS in libraries to serve the needs of beginning and intermediate GIS users. According to the users' feedback, FOSS GIS still cannot match with ArcGIS Desktop.

Based on the project needs, users are often encouraged to choose a hybrid model which represents the combination of open and closed source technology. This way they can incorporate GIS in any application on a variety of desktops, servers, and mobile platforms and use geographic information stored in almost any format, accessed from a variety of databases, or delivered as a web service.

2.2. Programming languages

Currently, the number of known programming languages counts more than six hundred (Robinson, Hardisty, Dutton, & Chaplin, n.d.). Naturally, it is not the case that all these are being used daily, nor that anyone should know how to use them all (at least not one person). There are always several leaders which determine the usage, the application, the necessity to master them and, of course, the overall objective of their existence.

According to the PopularitY of Programming Language (PYPL)[1] index, published on September 2017, top ten languages are Java, Python, PHP, C#, JavaScript, C++, C, R, Objective-C and Swift. This index is created by analysing how often language tutorials are searched on Google and as such is seen as a leading indicator of current trends and user's needs.

The history of programming languages dates back to the 20th century and from that time till now hundreds of programming languages have been designed and used for different purposes. Therefore, the work that has to be performed influences the language of choice. Additionally, knowing the main language purpose, whether it is built for functional or declarative programming, web applications development or web services design, supports the choice further. Criteria and languages considered in (Li, Rabah, Liu, & Lai, 2010) assisted in this master thesis language evaluation. Among ten described languages (C++, C#, Java, Groovy, JavaScript, PHP, Schalar, Scheme, Haskell and AspectJ.), their native design and main performance, none have shown to be suitable for the role

[1] PYPL INDEX, Accessed 10.09.2017.

(to be combined with GIS software for Spatial Data Science). A similar approach to language categorization was seen in (Ciolobac, 2016). Additionally, he suggested Python and R as prompted languages for GIS scripting and applications, data processing, analysis and modelling, and geospatial libraries.

(Robinson, Hardisty, Dutton, & Chaplin, n.d.) proposed a moderately different, GIS oriented, language list. Excluding the above-mentioned ones, the highlighted languages were Python, R, PHP and Ruby. This research group focused their language choice based on the fact that the essential element in customizing, designing or developing any geospatial system is the choice of what programming language to use.

Further language categorization was done by considering languages which are supported and used by Spatial Data Science. This overview was given by (Piatetsky, 2017) from KDnuggets™, which is a leading site on Business Analytics, Big Data, Data Mining, Data Science, and Machine Learning. The 18th annual KDnuggets Software Poll included participation from Analytics and Data Science communities and vendors, attracting about 2,900 voters for this survey. The results showed that the most prompted Data Science programing languages in 2017 are Python, R, SQL, Spark and Tensorflow.

(Szczepankowska, Pawliczuk, & Źróbek, 2013)compared and stated advantages of Python over the other listed programming languages:

- Object-oriented scripting language adjustable to any platform and given environment (Windows, Linux or Mac).
- Short, simple and easy to read format.
- Python automatically identifies and keeps track of data type so that data does not have to be explicitly described.
- Python is a modular language. A variety of modules written by other programmers can be imported and customised. It can handle multiple threads to synchronize inputs and outputs and, it can run multiple operations.
- Open-source, high speed, low cost and supported by the programming community.
- Python scripts automate programming functionalities in the GIS environment.
- Integration with both commercial and open-source GIS software is possible.

Further contribution to Python comes from (De Sarkar, Biyahut, Kritika, & Singh, 2012) who developed GRASS Python interface for monitoring environmental data as a key point in environmental assessment. Python scripts enabled storing data into the database, while the interface allows visualisation and positive outputs in results implementation. Similarly, (Silva & Taborda, 2013) developed an applicable tool which simplifies dataflow effort, reduces the human error and provides a dynamic visualization of the modelling results. This interface was developed using Python scripting language, taking advantage of its object oriented and cross-platform capabilities and its flexibility, as well as strong integration with ArcGIS Desktop.

(Zhang, Yue, & Guo, 2014) described Python as an interpreted, portable, object-oriented, high level language with readability, extensibility, ease of programming, free and open-source, platform independent with a variety of standard extensions. The development of geospatial applications with Python is simpler, faster and more efficient by integrating existing components and tools. As such, Python represents the perfect candidate for the role of geospatial scripting language for GIS programming, GIScript. Its role would be highly appreciated in geoprocessing and map generation automation.

Further significant interest within this master thesis goes towards R programming language, as it showed to be as popular as Python when it comes to the support from both the GIS and Data Science community. (Henshaw, et al., 2004) demonstrated joint use of ArcGIS Desktop and R as an approach for comprehensive geostatistical analyses. Proposed integration helps accurate characterization and evaluation of environmental contamination and therefore plays an important role in protecting the public's health.

Based on the feedback[2] from ArcGIS Desktop and R users about their needs and techniques for integrating ArcGIS Desktop and R, a community was formed for developing and sharing useful tools and promoting learning and collaboration. The community collects a repository of free, open-source R scripts, geoprocessing tools, and tutorials. This helps both ArcGIS Desktop and R users to be successful in combining these technologies and therefore solve problems in specific domains. A great example is the marine ecology community at Duke University[3]. They have moved forward with integrating R and ArcGIS Desktop for some time now and proved to have success in implementing Marine Geospatial Ecology Tools for a variety of marine research, conservation, and spatial planning problems. Another great example is shown in (Allen, Tsou, Aslam, Nagel, & Gawron, 2016). The authors propose reliance on GIS and machine learning (approachable through programming languages such as R) for highlighting the role of using Tweeter data in studying disease outbreaks. This enhances Big data information extraction and points toward great potential in monitoring the outbreaks for different illnesses.

2.3. GIS software

A GIS represents a modern extension of traditional cartography with one fundamental similarity and two essential differences. The similarity lies in the fact that both a cartographic document and a GIS contain examples of a base map to which additional data can be added. The first difference is that there is no limit to the amount of additional data that can be added to a GIS map. Secondly, the GIS uses analysis and statistics to present data in support of particular arguments which a cartographic map cannot do.
Cartographic maps are often simplified as there are limits to the amount of data that can be physically and meaningfully stored on a small map. Compared to traditional Cartography, GIS appears to be way more steps ahead, but what happens when tasks cannot be solved with GIS either? Which other tools can help

[2] Bringing the R and ArcGIS Communities Together, Accessed 18.07.2017.
[3] Marine Geospatial Ecology Tools, Accessed 18.08.2017.

and how is this achieved? Which tools seem to match best and what are the advantages and disadvantages of stepping into such integrations? Some of the authors came across examples and offered their solution, which led to a decision upon which GIS software to use for the purpose of this master thesis.

Rapid growth of data very often forces traditional methods of data analysis and management to be traded for the velocity of information production. (Li, et al., 2016). Traditional GIS software is becoming incapable of solving Big data challenges which is why there is an urgent need for software improvement. However, it is necessary to conduct detailed research on how this can be achieved and what are the critical issues in the next generation of GIS. The development of GIS took place in the past sixty years, but with data growth coming, it must be continued. New data-aware GIS needs to have the ability to deal with massive geospatial data management, processing, analysis and visualization. Such powerful tools can overcome many issues, originating from storing and computing infrastructure to the front-end service architecture. (Yue & Jiang, 2014)discuss one such approach by giving an overview of existing GIS possibilities, and presenting the concept of, so called, BigGIS as well as its future research agenda.

Another alternative would be to empower the standard GIS functionalities with existing tools such as a scripting languages and known libraries. What is crucial and necessary is establishing proper methods for addressing the theory and extruding useful policy advice out of it. That means doing such tasks as accessing geographic data, both commercial and open-source, doing quality revision (how reliable the information is, how to get rid of irrelevant data) and doing information extraction (how to extract useful information out of the dataset or combined data sources, how to deal with conflicting data sources). The further question is which technologies can help manage these? Best practices and alternatives for visualizing geographic data are GIS and maps. However, the rise of information technology provides the capacity for standardised maps to be enhanced towards dynamic mapping and GIS to be combined with a variety of open-source tools for analysis and visualization. Maps and additional tools are very useful to support decision making. However, they cannot replace critical thinking, since they can be manipulated with different effects and therefore lead to uncertainty (Wang, Wang, & Alexander, 2015). The emphasis remains on data analysts or other experts who works with the data to extract what is useful and use available tools to visualize it (Kwakkel, Carley, Chase, & Cunningham, 2014).

(Wang S. , 2016) introduces new, sophisticated GIS for Big (spatial) Data handling called Cyber GIS. Big Data trigger existing scientific and technological approaches for detecting and extracting spatial relationships between people and places. Data collected from numerous sources can give insight into significant phenomena, which is why it is of great importance to access, analyse and synthesize it. However, due to their volume and complexity, advanced actions such as bringing together various data streams at scale and harmonizing diverse spatial datasets are necessary. This approach of dealing with Big Data has its roots in GIS and Spatial Data Science and tends to drive future development of both.

Significant support comes, additionally, from the open-source movement which has yielded a large array of free software tools that can serve as building blocks for more advanced analysis (Anselin, 2012). This primarily refers to the R environment and its packages as a sophisticated way to deal with spatial statistical functionalities and geospatial data structures. R also has several links with open source GIS, as well as ArcGIS Desktop.

The list of GIS software might seem endless, but as with Programming languages there are several competitors which are highly ranked and widely used. A GIS Geography[4] site ranks GIS software according to four specifications:

- most required GIS software by employers,
- most used GIS software in research and publications,
- most searched GIS software, quantified by Google Trends,
- most discussed GIS software in GIS communities such as GIS Subreddit and GIS Stack Exchange.

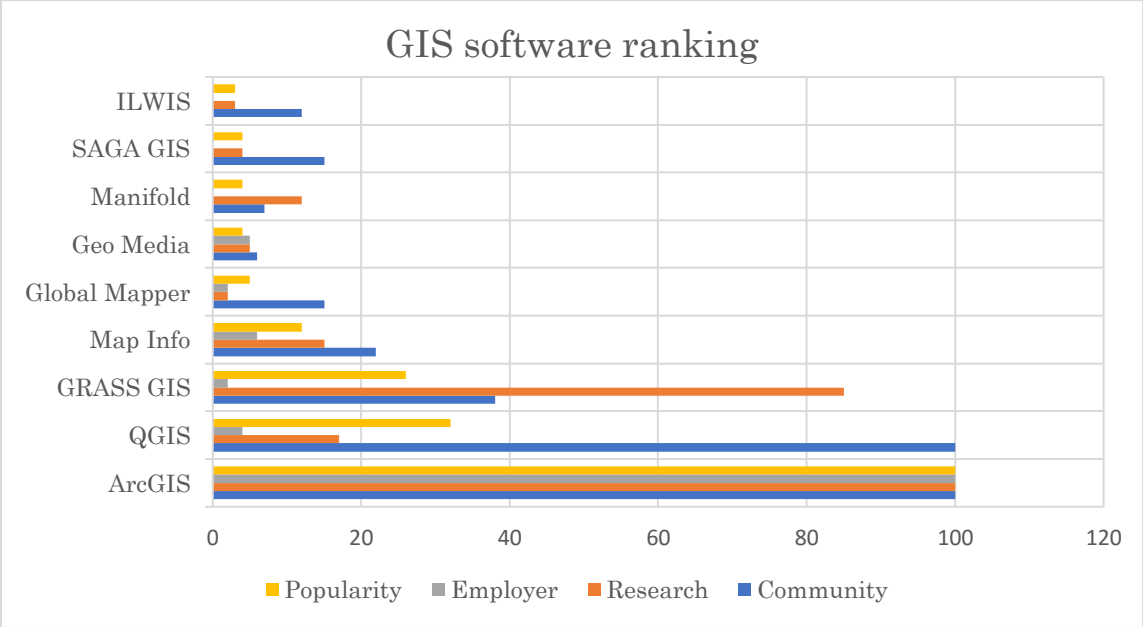

Figure 2: GIS software overview and ranking

The ranking was done such that 40% of all votes were based on which GIS software was required by employers, 20% was based on which software was being used in research and publications, 20% was based on which GIS software was being searched for and the last 20% was based on which GIS software was being discussed in community forums.

2.4. Data Science and Big Data

In the past all available data that represented the natural world were mapped and stored for future use. This approach changed significantly when experts started

[4] http://gisgeography.com/mapping-out-gis-software-landscape/

analysing data and searching for phenomena and rules of the data themselves. This evoked a transition towards the a new science called Data Science. (Zhu & Xiong, 2015)define Data Science as a novel research method which goes beyond computer science in researching data. Data Science is a new kind of science in which the scientists role is to focus on improving definitions of Data Science, exploring and explaining differences with other related fields, building up topics, directions, theories and methodologies, involving more scientific institutions and professionals into this field and trying to promote it via different methods and platforms so that more interested parties get involved. Data Science has become more important because of Big Data. Big Data (Dedic & Stainer, 2017) is being described as large, complex, growing datasets coming from different sources such as sensors, mobile devices, social media, internet, etc. It is characterised with volume, variety and velocity known as three Vs of Big Data. They are important if they can be analysed and provide useful information and competitive knowledge.

Integrating knowledge from Data Science and GIS can help develop much more sophisticated tools and techniques which congregate the data, set appropriate models and in return give useful results (Goodchild, 2016). Data itself provokes new GIS development, and new research opportunities, tools and products (Crampton, et al., 2013).

One of the main challenges of Data Science is how to translate data into actionable knowledge. The National Institutes of Health developed so called BD2K (Big Data to Knowledge) initiative which plays an important role in social and behaviour sciences (Margolis, et al., 2014). This role reflects an inexhaustible source of behaviour constructs coming from Big Data and possibilities to recalibrate existing mechanisms of scientific enterprise to be more transparent, cumulative, integrative, cohesive, rapid, relevant and responsive. Through a working partnership between and among the relevant stakeholders, useful solutions will emerge and lead to vibrant and sustainable models for translating data into new knowledge. Therefore, this is a nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data (Cios, Swiniarski, & Pedrycz, 2007). Moreover (Hesse, Moser, & Riley, 2015)propose several ways of accelerating this knowledge discovery, including, but not limited to:

- Open access to published material and research to support science through electronic networks.
- Open access to data.
- Improve cumulative knowledge building by making available research fully transparent.

The literature review from 2.1, 2.2., 2.3., and 2.4. focused on giving clear and understandable definitions on every "term" mentioned in the thesis title. As discussed before, this was done to better understand and approach each of these specific fields and to see if they work in junction, and how. Considering the choice of technologies, tools and other technical needs is influenced by type, size and availability of data, the next step was to decide upon which data to use for the practical part of this master thesis. The requirement was to choose the dataset or datasets that are free and open-source. Having this in mind, several open data

portals were examined and a decision was made that the practical part of this master thesis will be focused on criminal data, analysis, pattern recognition and discovery, visualisation and prediction. This decision was supported by the fact that many local, regional and national security authorities are turning towards new decision support tools such as GIS and predictive models to find better solutions for crime prevention (Ferreira, João, & Martins, 2012). Additional support came from the fact that 88% of police departments use GIS software for crime mapping purposes (Harries, 1999, p. 94). The growing potential of GIS can be employed at different levels to support operational policing, tactical crime mapping, detection, and wider-ranging strategic analyses (Chainey & Ratcliffe, 2005).

According to the above mentioned studies, GIS has strong and powerful influence in criminal data handling. It was necessary to investigate which programming languages tend to be the most prominent ones in crime analysis.
To support decision making one should think of the role of crime analysts, data analysts or GIS professionals in using the mentioned tools and retrieving useful information out of data?
To aid the decision process, an overview of the most prominent GIS software and programming languages was created, supported by findings from the sections 2.1, 2.2, 2.3 and 2.4. Table 1 shows the most widely used GIS software ranked (orderly) by popularity, employer, research, community (section 2.3, Figure 2) and police departments (Harries, 1999). Table 1 also shows the most used programming languages ranked by PYPL, GIS and the Data Science community ((Robinson, Hardisty, Dutton, & Chaplin, n.d.), (Piatetsky, 2017) and the number of searched tutorials and available packages for building crime prediction models.

Table 1: Overview of the GIS software and Programming languages

GIS software by:			**Programming languages by:**	
Popularity	ArcGIS Desktop QGIS		PYPL	Java Python
Employer	ArcGIS Desktop Map Info		GIS scripting	Python R
Research	ArcGIS Desktop QGIS		Data processing and analyses	Python R
Community	ArcGIS Desktop QGIS		Geospatial libraries	Python R
Police departments	Map Info ArcGIS Desktop		Data Science	Python R

According to Table 1, ArcGIS Desktop showed to be the most prominent GIS software within each criteria, with an exception in Police departments usage where MapInfo overtook by 10% (Harries, 1999, p. 94). In Programming language evaluation, Python and R led over all other competitors, excluding the PYPL index where Java showed to be the most popular language. This approach was significant and necessary for further work within this master thesis. Having decided upon which tools and data to use, one could concentrate on developing a real world

scenario which supports joint usage of Python, R and ArcGIS Desktop. The first step in this task was to get familiar with crime analyses, investigate what ArcGIS Desktop can do in this domain and what can be done with Python and R. The following sections describe each of these respectively.

2.5. Crime analysis - overview

Criminological theory says that three things are needed for crime to occur: a motivated offender, a suitable target and a location (Rossmo & Summers, 2015). When the first two pieces of this triangle are in place, the opportunity for criminal acts to happen is set. The role of analysts is to find a way to map crime locations. The initial step in understanding crime is putting dots of criminal events on the map. However, the context of the criminal event must be analyzed and compared with other crime data to derive meaning from it, to detect patterns in the data and to grasp the bigger picture. There are several steps for mapping crime:

- Aggregating crime data.
- Identifying patterns and clusters.
- Exploring the relationship between crime and other types of datasets.
- Assessing the effectiveness of crime-reduction strategies.

Crime mapping has long been an integral part of the process known today as crime analysis (Harries, 1999). Decades have passed from historical "pin" mapping and nowadays(Ratcliffe, 2007) describes crime analysis as the systematic study of crime and disorder problems as well as other police-related issues including sociodemographic, spatial, and temporal factors. Moreover, these analyses assist the police in criminal apprehension, crime and disorder reduction, crime prevention, and evaluation of police efforts. (LeBlanc, et al., 2014) list four major types of crime analyses, ordered from specific to general:

- Crime intelligence analysis,
- Tactical crime analysis,
- Strategic crime analysis, and
- Administrative crime analysis.

The purpose of *intelligence analysis* is to assist sworn personnel in the identification of networks and apprehension of individuals to subsequently prevent criminal activity. These analyses are used to relate elements such as companies, agencies, people, times and days, to crimes and places (Ahmadi, 2003). Therefore, these analyses aim to identify correlations between crimes and possible influencers of unlawful behaviour. *Tactical analyses* seek to identify trends and patterns in crimes that are not easily linked together. Pattern detection and analysis can be used to efficiently manage police operations by deploying resources to criminally active areas and may increase the rate of apprehension, reduce or displace crime, and assist in identifying other factors that may contribute to criminal events (Ioimo, 2016). *Strategic crime* analyses concentrate on criminal activity over a period of time. These analyses are often called temporal analyses since they analyse data in relation to time (year, month, day of the week, part of the day and hour of occurrence). Furthermore, (Perry, McInnis, Price, & Smith, 2013) point out

that there are methods which aim to prevent crime, predicting when and where crime is most likely to occur, which type of the crime is most likely to occur, and what influences criminal behaviour. This type of analyses (Perry, McInnis, Price, & Smith, 2013) is called predictive policing and is described as a means of forecasting and predicating crimes using sophisticated techniques that require computer analyses. This includes computer-assisted queries, statistical modelling, spatiotemporal analyses and advanced hot spot identification models to perform crime linking. *Administrative* crime analyses deal with long-range comparisons (quarterly, semi-annually or annually).

Crime analyses require an investment in training, hardware, software, and personnel in order to function at higher levels beyond basic management statistics (Ratcliffe, 2007). The usage of GIS and crime mapping is vital for analysing and identifying crime patterns and trends. In this respect, GIS and crime mapping support a broad variety of problem solving and spatial decision-making applications in crime and crime locations. GIS lets agencies utilize more information more intelligently (Dağlar & Argun, 2016). ArcGIS Desktop, Python and R functionalities in this domain range from vary basic to very complex analyses and sophisticated visualisations. An overview of these functionalities is given in the sections that follow.

2.5.1. ArcGIS Desktop in Crime analyses

ArcGIS Desktop provides a dedicated suite of tools to look at **what** crimes have occurred **where** and **when**. (Kraak & Koussoulakou, 2005) argue that the Space-Time-Cube offers good visual opportunities to study the relationship between time and space and additional variables. For an effective interpretation of spatiotemporal patterns of crime clusters/hotspots,(Nakaya & Yano, 2010) explore the possibility of three-dimensional mapping of crime events in a Space Time Cube with the aid of space-time variants of kernel density estimation and scan statistics. Creating a Space Time Cube by aggregating points is most common when the point data represents incidents, such as crimes, and there is a need to aggregate those incidents into either a grid or a set of polygons representing police beats. What is considered important for getting valubale results is setting up the structure of the cube, both spatial and temporal as well as time step alignment. In ArcGIS Desktop the Space Time Cube tool belongs to the Space Time Pattern Mining toolbox. More information about steps in creation, interpretation and visualisation of the Space Time Cube is given in Appendix 7.

The use of Emerging Hot Spot analysis is an additional innovation of applied geographical concepts, as spatiotemporal analyses have previously been more cumbersome to deploy. Other communities and researchers may use this method to understand patterns in crime rates as well as other spatial phenomena, and may connect this analysis to their own intervention programming to determine how and where best to intervene in community issues (Sadler, Pizarro, Turchan, Gasteyer, & McGarrell, 2017). An emerging hot spot analysis is conducted to investigate trends over space in addition to trends over time. Clustering patterns of point densities or summary fields across the study area are analysed. The Emerging Hot Spot analysis tool exists as part of ArcGIS Desktop basic functionalitis. It also

belongs to the Space Time Pattern Mining toolbox. The theory behind creation, input parameters and interpretation of the Emerging Hot Spot analysis is provided in Appendix 7.

Moreover, (Ferreira, João, & Martins, 2012) argue that police strategies consist of finding and identifying areas within the city with unusual amounts of crime to better respond to those specific areas. Traditional responses in hot spots are normally an increased police presence which leads to more arrests. ArcGIS Desktop Mapping Clusters toolset provides two of such tools. One is called Hot Spot Analysis and it is used to identify statistically significant spatial clusters of high values (hot spots) and low values (cold spots). The other one is called Optimized Hot Spot Analysis and it interrogates the data to automatically select parameter settings that will optimize the hot spot results. It will aggregate incident data, select an appropriate scale of analysis, and adjust results for multiple testing and spatial dependence. Both tools and their performance, as well as statistical background are explained in Appendix 7.

The last consideration about the application of ArcGIS Desktop in crime analysis is through R-ArcGIS Bridge. R- ArcGIS Bridge represents integration of statistical libraries and functions of R and the spatial capabilities of ArcGIS Desktop[5]. One such example is presented by (Veronesi, 2016) who built a tool for performing spatiotemporal point pattern analysis on London criminal data, by enhancing ArcGIS Desktop basic functionality with R. Another example comes from webinar materials provided by Cameron Plouffe[6] who also uses R-ArcGIS Bridge to show how easily and effectively it is to perform crime analysis and visualise the outputs. Marjean Pobuda[7] uses this software/tool integration to identify and visualize patterns and trends in criminal data that seem random and unconnected. Steps for R-ArcGIS Bridge installation as well as a link to installation are provided in Appendix 7.

2.5.2. Tools – Python and R for Crime Analysis

While there is a waging war[8] going on in the Data Science community about which language is better to use, either R or Python, the focus within this master thesis was more on how to combine them together to achieve desired outputs. At the time of writing, Python Package Index[9] counted for 117183 packages while CRAN[10] package repository features 11430 available packages. Table 2 shows basic information about each language, its purpose, user groups, and main pros and cons. Additionally, it lists some of the most used packages depending on their original purpose and performance (Theuwissen, 2015).

[5] Put the 'R' in ArcGIS, Accessed 25.08.2017.
[6] Integrating R with ArcGIS, Accessed 12. 07.2017.
[7] Analyze Crime Using Statistics and the R ArcGIS Bridge, Accessed 10.07.2017.
[8] Choosing R or Python for data analysis, Accessed 11.06.2017.
[9] Python Package Index, Accessed 15.09.2017.
[10] The Comprehensive R Archive Network, Accessed 15.09.2017.

Table 2: Python and R basics

	Python	R
History (established)	1991.	1995.
Purpose	Productivity, code readability	User-friendly data analysis, statistics, graphical models
Users	Statistics, research, data science	Engineering environment
Learning curve	Relatively low and gradual, good prognosis for new programmers	Steep at start, easier for experienced programmers
Environment	Spyder, IPython notebook, Rodeo	R Studio
Libraries/Packages	pandas, matplotlib, statsmodel, numpy, scipy, scikit-learn	dplyr, plyr, data.table, stringr, ggvis, lattice, ggplot2
Pros	Open- source, advanced tools, growing online community	
Cons	It does not offer an alternative to the hundreds of R packages, visualisation options are limited depending on the data size	Developed for and by statisticians, finding proper packages can be time consuming

Both Python and R offer very prominent packages for data manipulation and organization which is the first step in working with criminal data. (Saraswat, 2017) introduces *pandas* as the best choice for handling tabular data sets comprising different variable types and *numpy* as the most suitable for performing basic numerical computations such as mean, median, range, etc. to get a taste of data manipulation with Python. On the other hand, (Berhane, 2017) introduces *dplyr* as an elegant solution in data manipulation which resembles a natural language, and *data.table* where a lot can be done in just a single line. Further, data.table is, in some cases, faster and it may be a go-to package when performance and memory are constraints. Moreover, (Musings, 2017) adds *lubridate* for working with date variables in R and *reshape* for transforming different data forms according to needs. Lastly, *sp* package provides classes and methods for dealing with spatial data in R. The spatial data structures implemented include points, lines, polygons and grids; each of them with or without attribute data (Pebesma & Bivand, 2005).

A further approach in Crime analysis is to visualize the criminal data. Again, both languages offer a range of useful packages which deliver different visualisation outputs, from very basic graphs to sophisticated and interactive outputs. In the Python environment, *matplotlib* represents a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms (Hunter, Dale, Firing, & Droettboom, 2017). Visualizing spatial (among which criminal) data in R can be a challenging

task. (Kahle & Wickham, 2013) introduce *ggplo2*, a powerful and flexible data visualisation package for creating static graphics and *ggmap* for visualizing spatial data and models on top of static maps from various online sources. As all these graphics created with ggplot2 are static, they can easily be turned interactive by using the package *plotly.* Plotly is compatible with both Python and R.

Finally, both Python and R can serve in predictive modelling due to the variety of statistical models they have in their repositories. In general, every model has its own strengths and weaknesses and is best suited for particular types of problems. (Ray, 2017) gives an overview of commonly used algorithms with an example in both Python and R. Clearly, it is hard to decide which model and for which data type to choose. (Pluemacher, 2016) discuses three models which show decent performance in crime predication. These were Random Forest Classifier, Extra Trees Classifier and Gradient Boosting Classifier, all three belonging to the Python's *scikit-learn* package. (Ang, Wang, & Chyou, 2015) propose the usage of Random Forest, Logistic Regression and Naïve Bayes to build a model which makes use of both the time and location features present in the original dataset of San Francisco criminal data. (Ke, Li, & Chen, 2015) used and evaluated Naïve Bayes, k-Nearest Neighbour and Gradient Tree Boosting models in calculating probabilities of different categories of crime. The Gradient Boosting model performed best, but with the longest computing time. Important contributions in this sector has been made by (Bogomolov, 2014)where specific regions in London are predicted as crime hotspots using demographic data and behavioral data from mobile networks. (Yu, Ward, Morabito, & Ding, 2011) deploy classification algorithms such as Support Vector Machines, Naïve Bayes, and Neural Networks, to classify city neighborhoods as crime hotspots. (Toole, Eagle, & Plotkin, 2011)demonstrated the spatiotemporal significance of correlation in crime data by analyzing crime records for the city of Philadelphia. They were also able to identify clusters of neighborhoods affected by external forces. Significant contributions have also been made in understanding patterns of criminal behavior to facilitate criminal investigations. (Wang, Rudin, Wagner, & Sevieri, 2013)presents a procedure to find patterns in criminal activity and identify individuals or groups of individuals who might have committed particular crimes. The different types of crimes comprising traffic violations, theft, sexual crimes, arson, fraud, violent crimes and aggravated battery and drug offenses are analyzed using various data mining techniques such as association mining, prediction and pattern visualization.

Table 3 gives an overview of all Python and R packages listed in the above review, as well as their potential application in crime analysis depending on the type of the analysis. Detailed description about each package which was used within this master thesis is enclosed in Appendix 5.

Table 3: Python and R packages for Crime Analysis

	Package	Description	Crime Analysis
Python	*pandas*	data manipulation, data analysis	strategic
	numpy	data manipulation, scientific computing	strategic, intelligence
	matplotlib	data visualisation	visualisation
	scikit-learn	predictive modelling	predictive policing
R	*dyplr*	data manipulation, data analysis	strategic
	data.table	data manipulation, fast with large datasets	strategic, intelligence
	lubridate	data manipulation	strategic
	reshape	data manipulation	strategic
	sp	spatial data manipulation	tactical
	ggplot2	data visualisation	visualisation
	ggmap	mapping	administrative
	plotly	interactive data visualisation	administrative

In light of the preceding literature, it is stated that the joint combination of ArcGIS Desktop with Python and R has beneficial influence on analysing and visualising criminal data, as well as predicting where and when specific crimes have potential to occur. The following easy-to-understand-real-world-example presented in a Chapter III, as well as associated methodology, provides a research framework to address this statement.

III CASE STUDY

The second most important part of this thesis study was to develop an easy-to-understand-real-world-example which supports the theoretical findings from Chapter II. The idea was to show how ArcGIS Desktop combined or integrated with Python and R helps better, easier, faster, more effective and efficient data analysis and visualisation. For developing such a scenario, one should think of the data, methodology and accompanied workflow, as well as the steps in realisation depending on the software/tools used. In other words, the first step is to get to know the data and the basics about what information it carries. The second step, methodology, will concentrate on planning and explaining the steps of the case study. The final step, which is the workflow, will show the realisation of the planned methodology accompanied with theoretical explanations and graphical outputs.

3.1. Data

The choice of data for this case study has already been made in the Chapter II. This choice was supported by literature findings about significant work that has been performed in crime analysis, classification and prediction using ArcGIS Desktop, Python and R, either separately or combined. The dataset used was obtained from Kaggle's[11] platform and refers to *San Francisco Crimes*. The platform itself offers a variety of options to a user - from learning, working and playing with the data, to contributing, competing and receiving feedback from the community about provided work. The data is provided by the San Francisco Police Department Crime Incident Reporting System. All data provided by the Kaggle Datasets Platform is free and open data.

The *San Francisco Crimes* dataset consists of data from 01.01.2003 to 13.05.2015. The dataset size is 0.119GB, with the 878049 entries distributed in 9 columns. Table 4 shows the names of the dataset columns and their detailed description:

Table 4: Dataset description

Column name	Description
Dates	timestamp of the crime incident
Category	category of crime
Descript	detailed description of the crime incident
DayOfWeek	the day of week
PdDistrict	the name of the Police Department District
Resolution	describes the way how the crime was resolved
Address	the approximate street address of the crime incident
X	Longitude
Y	Latitude

[11] Kaggle

Table 4 additionally shows that the dataset carries both temporal (Dates, DayOfWeek) and spatial (PdDistrict, X, Y) information about crime occurrence. This information will be valuable in developing scenario steps for the case study.

3.2. Methodology

After being introduced to the dataset, one should think of what can be done with the data using the tools and software that are available, and how (ArcGIS Desktop, Python and R). The question of *what*, keeping in mind the nature of data, would mean getting to know the spatiotemporal distribution of the criminal activity, where the crime hot spots are, understanding the patterns of criminal behaviour, predicting crime occurrence, etc. The question of *how* would mean which of the tools (ArcGIS Desktop, Python, R) should be uded to get the answers to the above set of questions. Therefore, the precondition in developing methodology steps is to define the goals of the case study. Again, regards the nature of the data, the case study goals were to:

1. Analyse and visualize the spatial and temporal relationship of crimes.
2. Analyse how variables interact with each other and determine when and where arrests frequently occur.
3. Identify where crimes are emerging.
4. Identify areas with an unusually high number of crimes.
5. Investigate factors that might influence unlawful behavior.
6. Predict where higher numbers of crimes are likely to occur.

Figure 3 shows a slightly modified, classical data analysis pipeline (O'Neil & Schutt, 2013). The idea behind was to divide the case study scenario into several steps, name and describe each step with respect to crime analysis described in section 2.5 of Chapter II and suggest which tool to use for performing the analysis within each step.

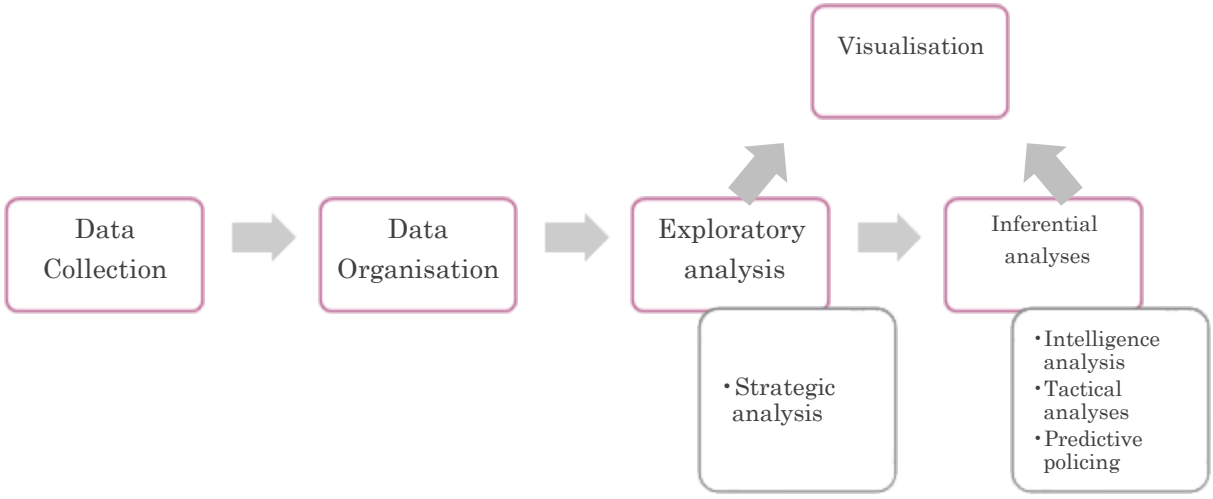

Figure 3: Data analysis pipeline

The first step in the process is, as described, *Data Collection*. Data for this master thesis was collected (downloaded) from Kaggle's open data platform. Data was already introduced in section 3.1 of this chapter.

After collecting the data, the next step was to do *Data Organisation* which is a process of detecting and removing errors and inconsistencies from the data in order to make it ready for the analysis (Rahm & Do, 2000). For the San Francisco dataset, this meant adding new columns into the dataset, removing unnecessary columns, searching, selecting and deleting incomplete entries, adjusting the timestamp column, and renaming some of the column names so that they are shorter and easier to embed within the code.

The next step in the data analysis pipeline was *Exploratory analysis* which is an approach in analysing datasets to summarize their main characteristics, often with visual methods such as box plots, scatter plots, histograms, charts, parallel coordinates, etc. (Tukey, 1977). This can be applied in strategic crime analysis (Ioimo, 2016) to get answers about spatiotemporal distribution of San Francisco crimes. This refers to understanding and delivering visual representations of where and when (yearly, monthly, weekly, daily, hourly counts of crimes) San Francisco crimes have occurred.

The simplest way of understanding the *Inferential analysis* is to think of it as of a set of analyses performed to get insights into what causes certain outcomes. In the crime analysis domain, this could be applied to Intelligence and Tactical analysis and Predictive policing. According to Chapter II, section 2.5, and applied to San Francisco crimes, intelligence analysis should give answers to crime rates depending on districts and day of the week, tactical analysis should show crime hot spots, places with unusually high numbers of crimes and crime dependency upon factors such as population, incomes, education, business, etc. Predicative policing should give the answer as to where crimes in San Francisco are likely to occur.

With the case study goals in mind and methodology steps described, one should think of the final step in realising an easy-to-understand, real-world-example and that is a decision upon which tool to use for a specific task. According to Chapter II, section 2.5.2, Table 4, both Python and R offer useful packages for data manipulation. Considering the very few pre-processing steps which were supposed to be done to prepare data for further analysis (Figure 3, Data Manipulation) a decision upon which one to use was influenced only by simplicity and readability of the code. Therefore, this step was performed in Python using pandas and numpy packages and was embedded within the Python stand-alone script enclosed in Appendix 1.
Strategic analysis, which focused on answering the first case study goal, was performed using Python stand-alone script (Appendix 1), Intelligence analysis, which focused on answering second case study goal, was performed using R stand-alone script (Appendix 2), Tactical analysis, which answered the third, fourth and fifth case study goal, was performed using the R-ArcGIS Bridge (Appendix 3) and the Predictive model was built in Python using the scikit-learn package (Appendix 4).

3.3. Workflow

The methodology section set up case study goals and described the necessary steps for realising these goals. Additionally, it mentioned the tools that were used for performing the analysis in every step. The workflow section describes each of these steps in more details. What was the goal, how it is achieved, which packages were used and how? Moreover, it includes visual outputs of performed analyses.

3.3.1. Strategic analysis in Python

Strategic analysis in Python had an aim to analyse and visualize the spatial and temporal relationship of San Francisco crimes. In other words, to visualise *where* and *when* crimes occur. Answering the question *where* starts with visualising the crimes according to the category they belong to. This way, it was possible to see which crimes were leading in this city (Figure 4). However, the question still remains as to where these 39 different crime categories are being reported. Figure 5 shows the answer to this question by isolating the Southern district as the one with the highest number of reported crimes.

Both Figure 4 and Figure 5 were produced using Python's numpy, pandas and matplotlib packages. For Figure 4, crimes were counted according to the category they belonged to using the *pandas unique value counts* and *numpy arrange* function which arranged the crime category counts in ascending order (from the highest to the lowest number of counted crimes). The visualisation was done using the *matplotlib* package which offers, among other visualisation outputs, the ability to create bar graphs with corresponding content. Figure 5 was created in a similar way, with the only difference being that crime counts were grouped by the districts using the *pandas group by* function and visualised the same way as in Figure 4.

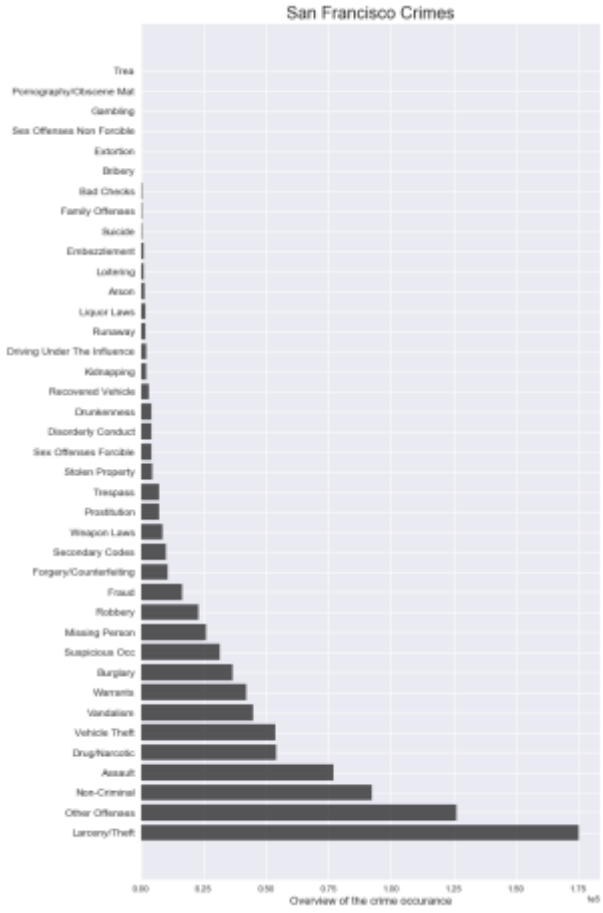

Figure 4: Overview of the crime occurrence in San Francisco

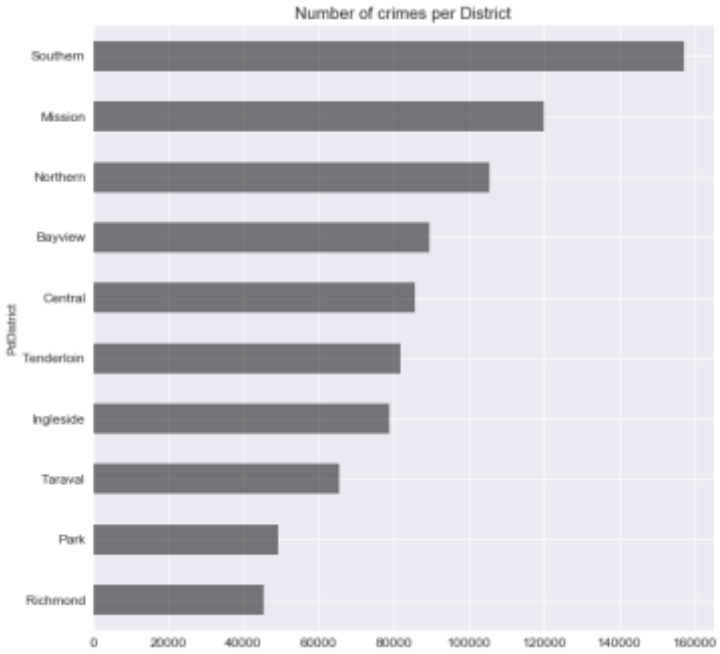

Figure 5: Crime counts per district

Answering the question *when* started by visualising the crime behaviour over the years obtained in dataset (Figure 6). The next visualisation output (Figure 7) shows the crime occurrences on a monthly level, Figure 8 on a weekly level, Figure 9 on a daily and Figure 10 on an hourly level.

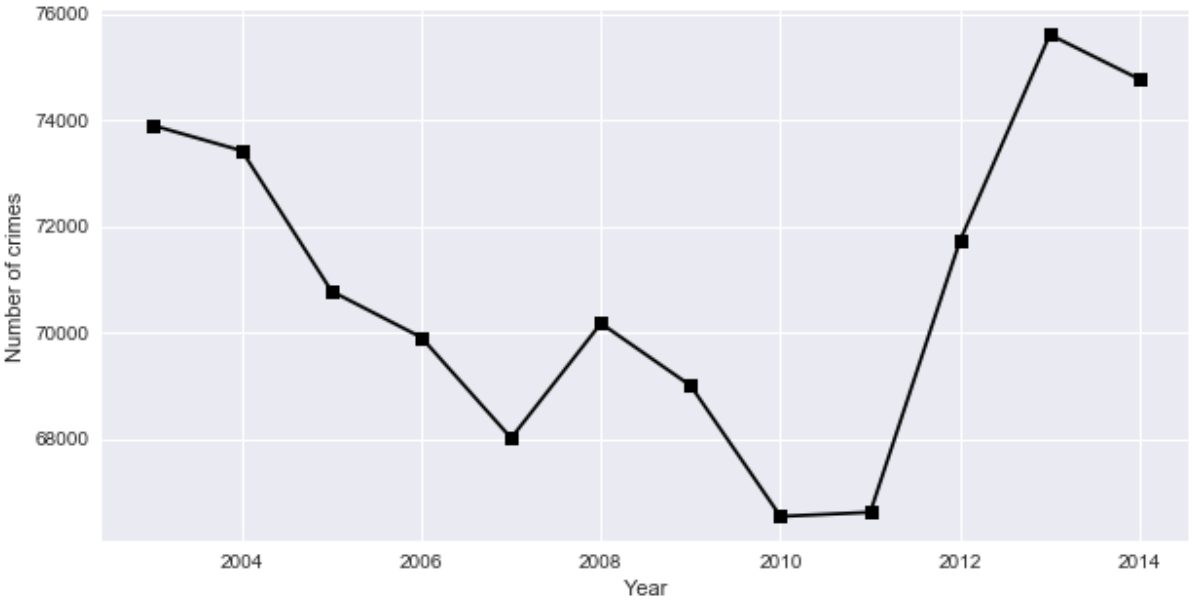

Figure 6: Number of reported crimes per Year

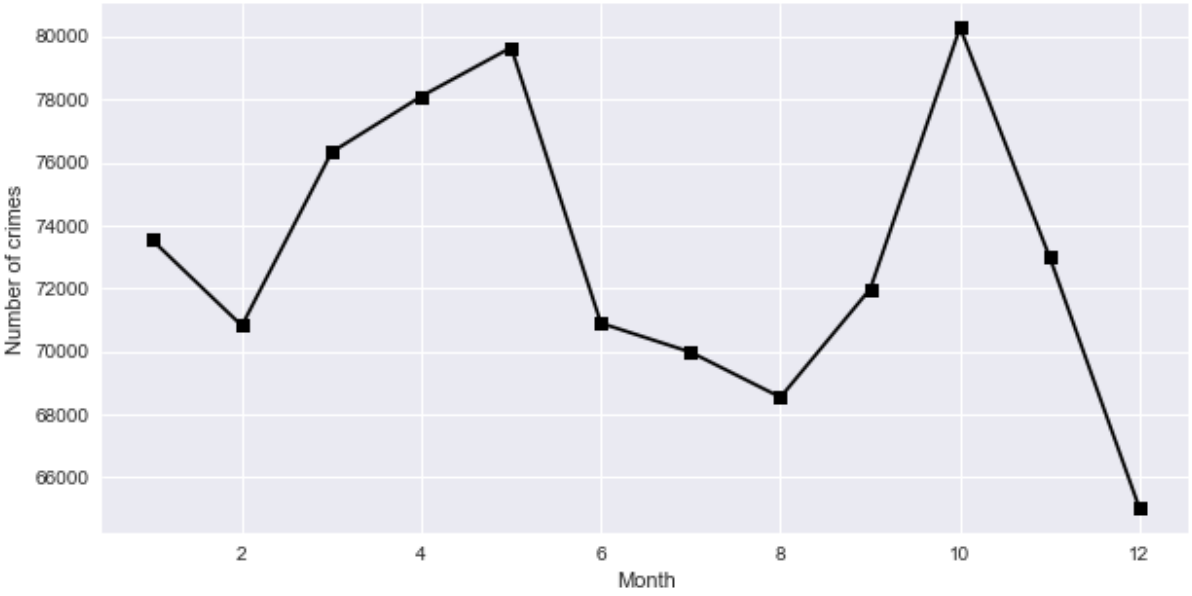

Figure 7: Number of reported crimes per Month

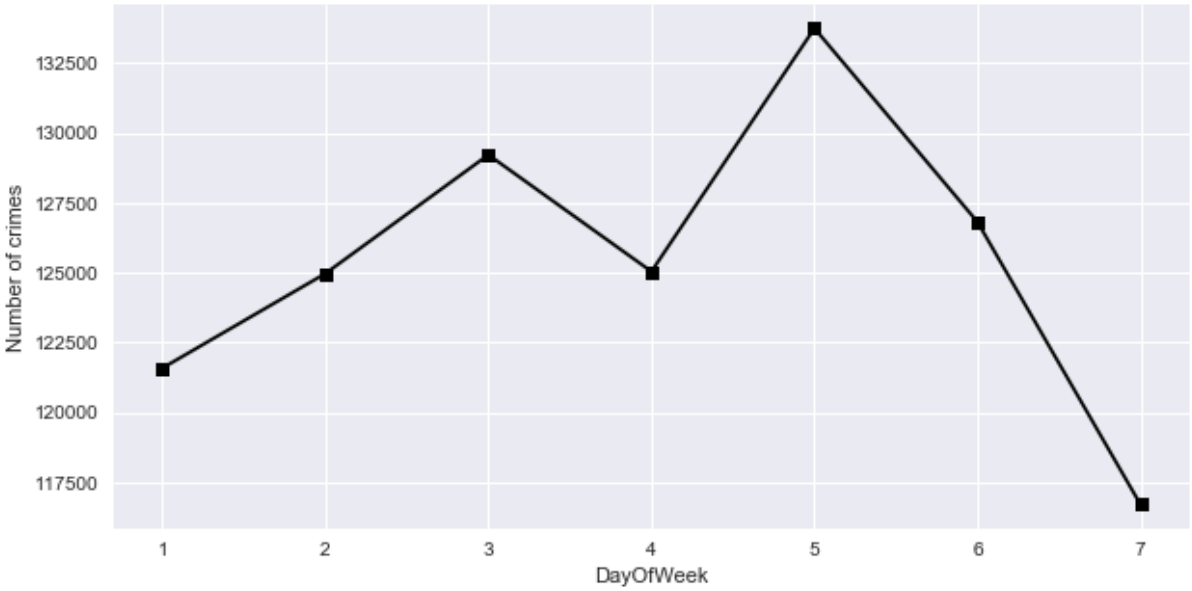

Figure 8: Number of reported crimes per Day of Week

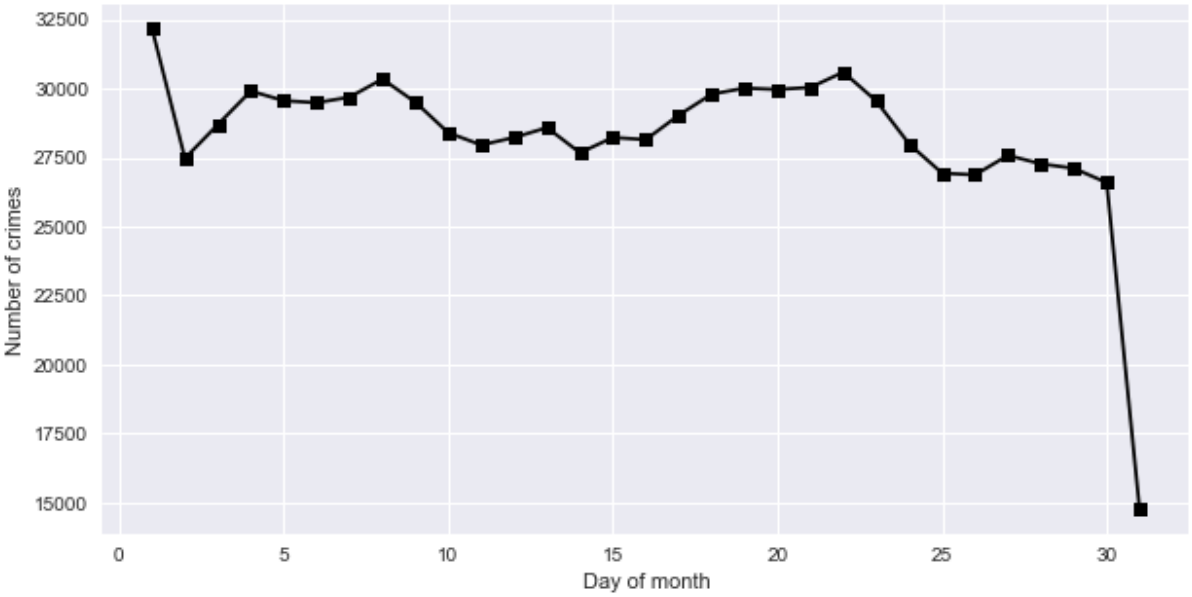

Figure 9: Number of reported crimes per Day of Month

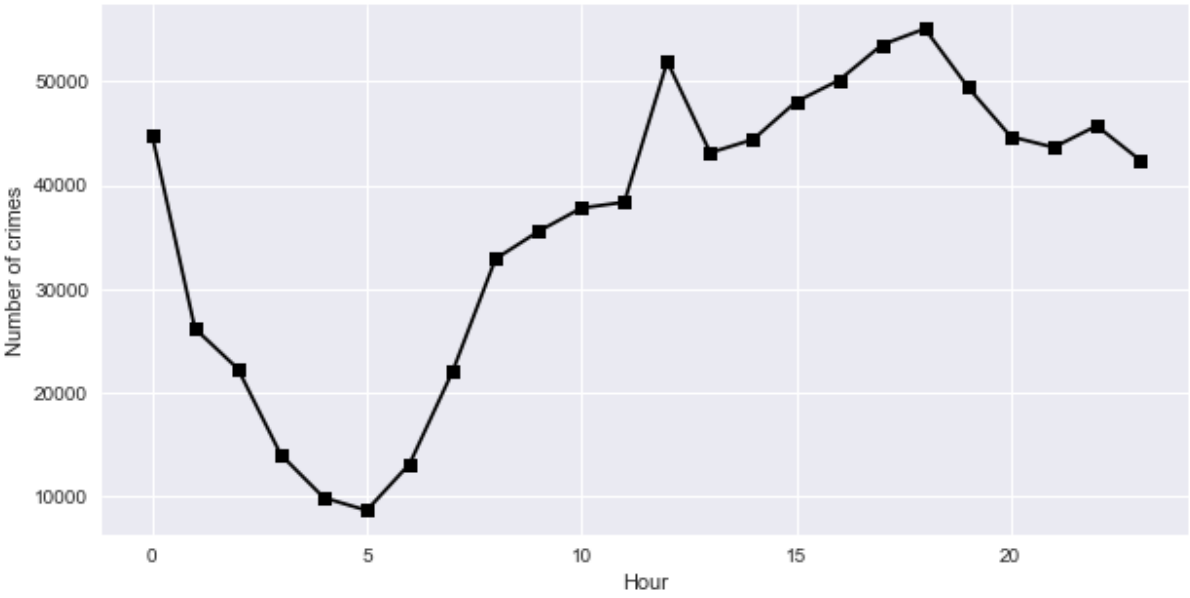

Figure 10: Number of reported crimes per Hour

The theory behind these plots is to split the timestamp column (provided within the data) into separate distinct columns showing respectively crimes which were recorded on a yearly, monthly, weekly, daily and hourly level. Python's pandas has the option of converting the datetime index into separate columns. Crimes were then grouped by each level individually and plotted using the matplotlib package. A line plots visualisation was chosen to show the overall distribution of criminal behaviour in contrast to time. Clearly, 2013. was the most dangerous year in this time frame of twelve years in which crimes were recorded, October was the most dangerous month, Friday the most dangerous day and 6pm the least safe time of the day to be outside on the San Francisco streets.

According to Figure 4, there are 39 different categories of crime listed within the dataset. The top six, out of these 39, crimes make up 65,8% of all crimes and these are larceny, assaults, drugs, vehicle theft, vandalism and warrants. Figure 11 shows relative numbers of top crimes depending on the week day. This visualisation was possible using the matplotlib package again, however some steps before were needed to prepare the data for plotting. Every individual crime from the top crimes list was grouped by the day of the week and the number of records was counted. This number was than divided with the sum of all top crimes for the week. This gave the relative number of that specific crime occurrence within the week. The same process was repeated for every crime from the top crimes list and the output was shown in Figure 11. The legend connects the type of crime with the colour represented on the graph and it was added manually while plotting the graph. The graph's x-axis was labelled manually so that it shows the names of the days of the week, instead of numbers as it produces by default.

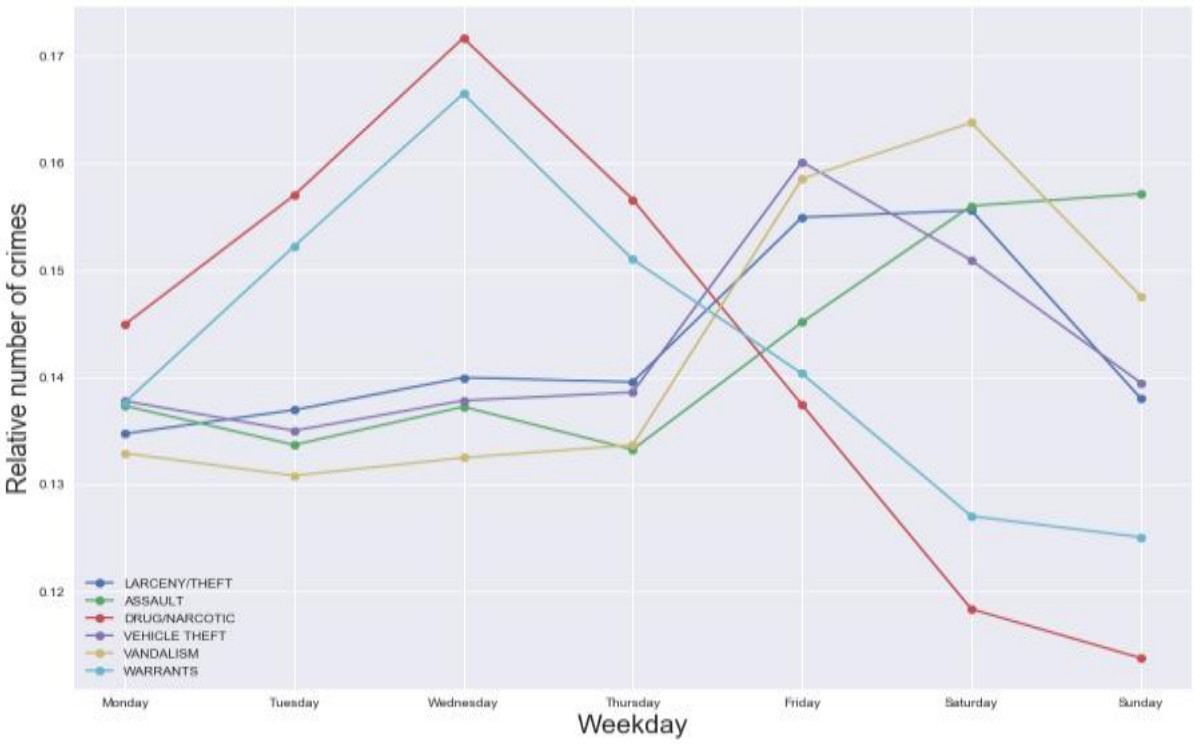

Figure 11: Relative number of top crimes per weekday

Further consideration was given to the top crimes and their behaviour on an hourly level. Figure 12 shows how each of these prominent crimes are being distributed over the whole day time slot. Considering the number of top crimes (which can be arbitrary as already mentioned) one should think of creating several plots which will show all top crimes on one graph. Something like this can be solved with matplotlib subplots or gridspec (Appendix 5). The option of creating multiple plots or subplots was used to produce Figure 12. Individual plots were arranged in a regular grid by defining the number of rows and columns. Rows and columns specified the size of the grid in which separate plots were stored. Additionally, for every subplot in Figure 12, it was necessary to set up the exact position, and the x and y label of the plot, for clearer and easier interpretation of the outputs.

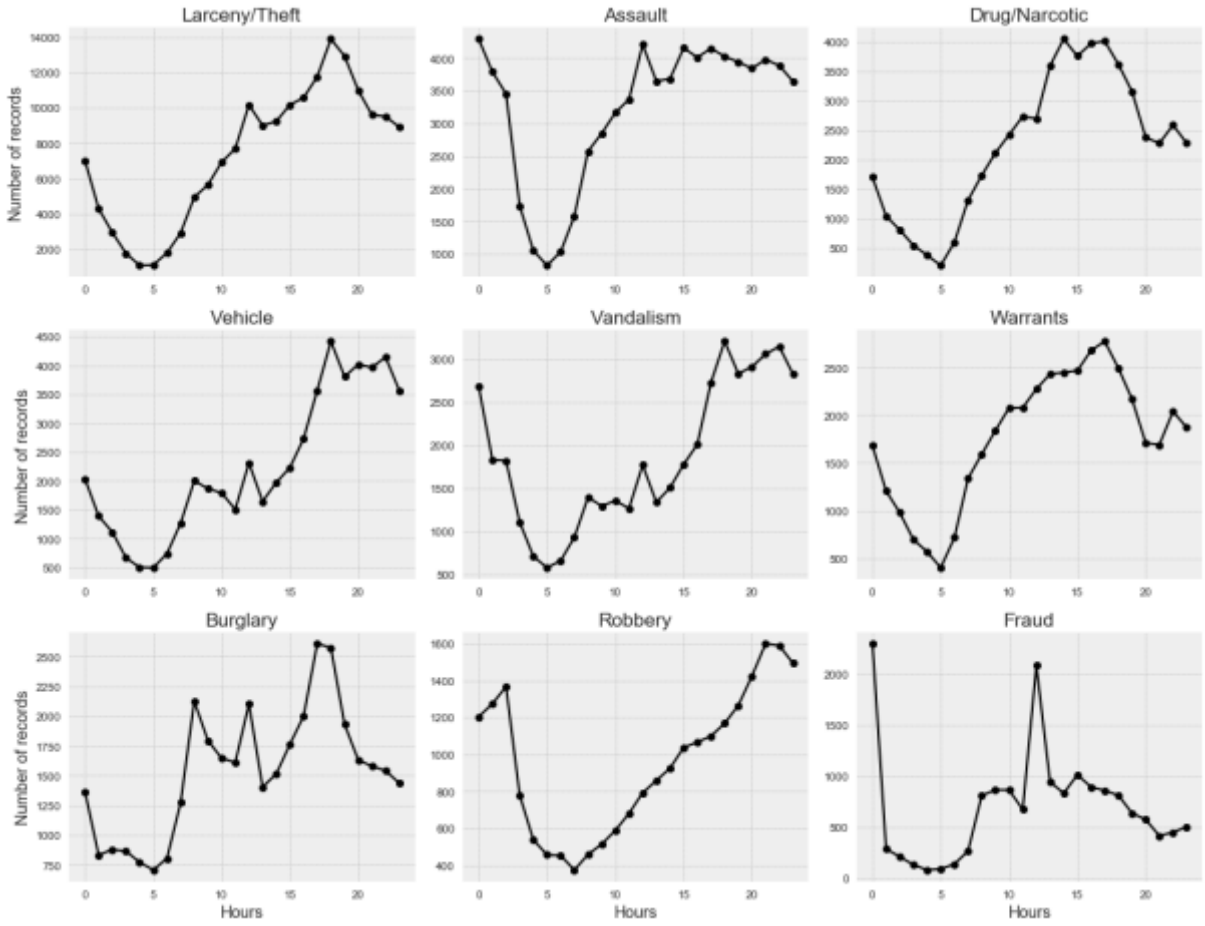

Figure 12: Crime occurrence by Hour

Strategic analysis with Python packages gave some interesting outputs which will be discussed in more detail in the Outputs and Discussion chapters. The code was enclosed in Appendix 1.

3.3.2. Intelligence analysis in R

As introduced in section 3.2 the intelligence analysis aims to discover relationships and dependencies among dataset variables. This means that not only are variables being analysed but their joint behaviour as well. This section aims to give answers to the second case study goal. In the San Francisco dataset, there are several variables whose dependencies can be examined in order to see whether the crimes are being influenced by these variables or not, such as districts, days of the week, months and even years.

But before starting any analysis, one should examine the distribution of the crimes over the city of San Francisco by taking a look at the map. This is, by far, the best and easiest way of getting a clear picture of what is happing where. R has the option of creating maps using the ggmap library and importing the map tiles either from google maps, open street maps or stamen maps. For both Figure 13 and Figure 14, maps were created by calling the google map tiles with the previously set up bounding box around the San Francisco area. The data, crimes per district,

were overlaid with the created map. In the first case (Figure 13), geom point produced the scatterplot which was overlaid with the map, while in the second case (Figure 14) the stat_density2d layer overlaid with the map give density map output. Both options are contained in the R's ggplot2 package.

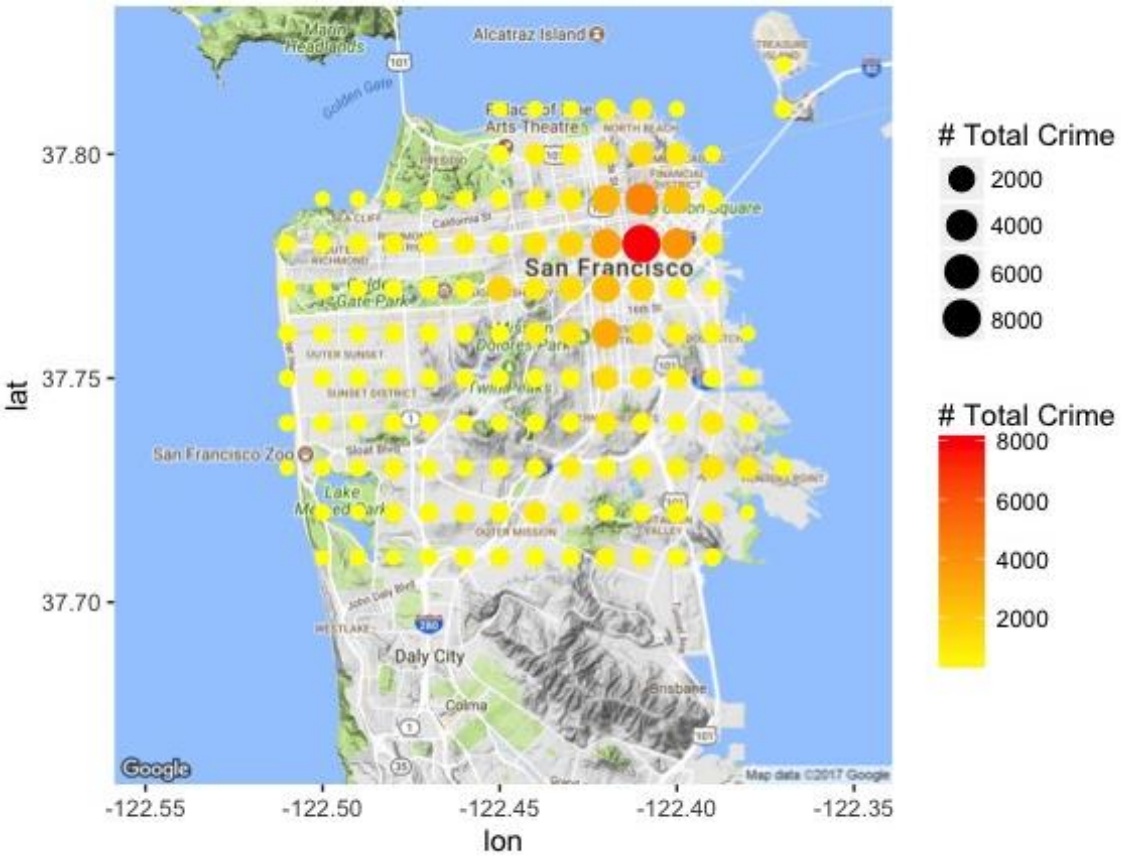

Figure 13: Map of crime occurrence per District

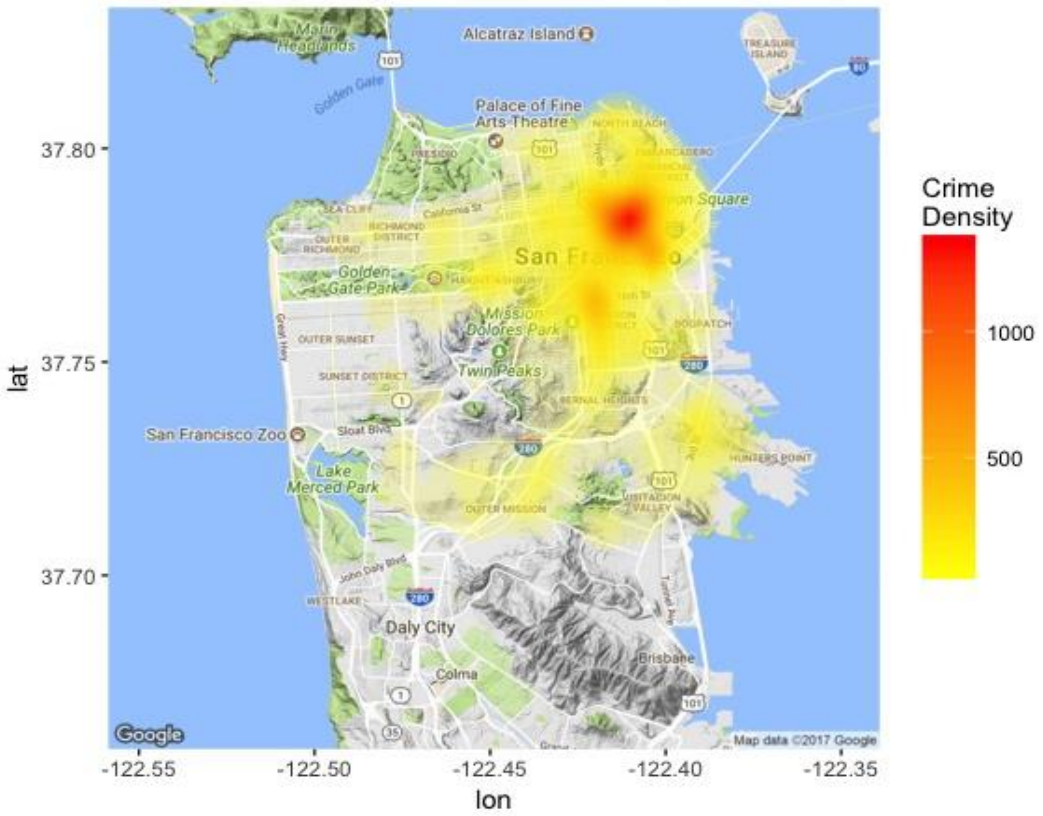

Figure 14: Density map of crime occurrence per District

Starting with the days of the week, the question is does the day affect crime type and volume? To answer this question, ggplot2 package offers an option of plotting a heat map with two dependent variables arranged on the x and y axes (x axis shows days of the week and y axis shows districts). Additionally, the data.frame package has a quick option (.N) of counting the number of crimes within the chosen variables (in this case those were districts and days of the week). This counts can then be converted into percentages by dividing the individual number of crimes with the sum of all crime counts. Figure 13 shows the influence of the week days on the type and volume of crimes.

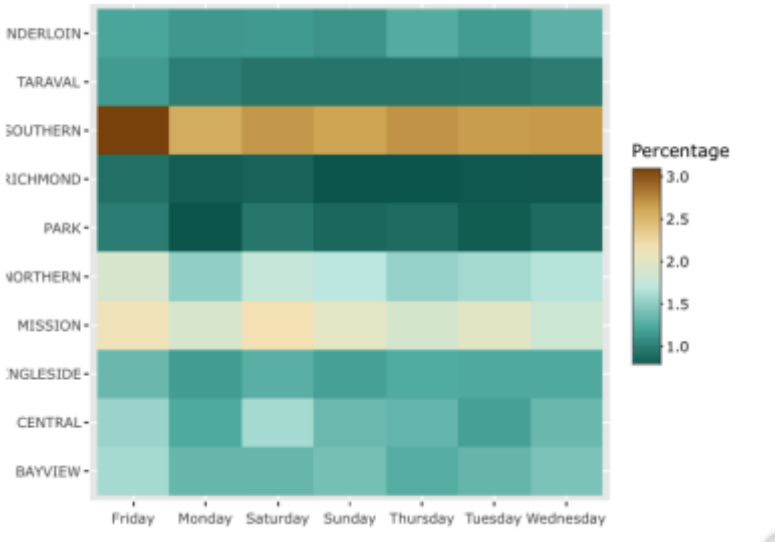

Figure 15: Relationship between day of the week and crime volume

Also, it was interesting to see whether on a given day, the crime rate differs across the district. Figure 14 was created following the same steps as with the previous step, with one distinction that crime counts were ordered by the district. That was done to get the answer to the above question whether on a given day, the crime rate differs across the districts. Moreover, this output showed whether on certain days, the ordering of districts changes or not. From Figure 14, the crime rate over the districts does not change with the day of the week, and is quite stable.

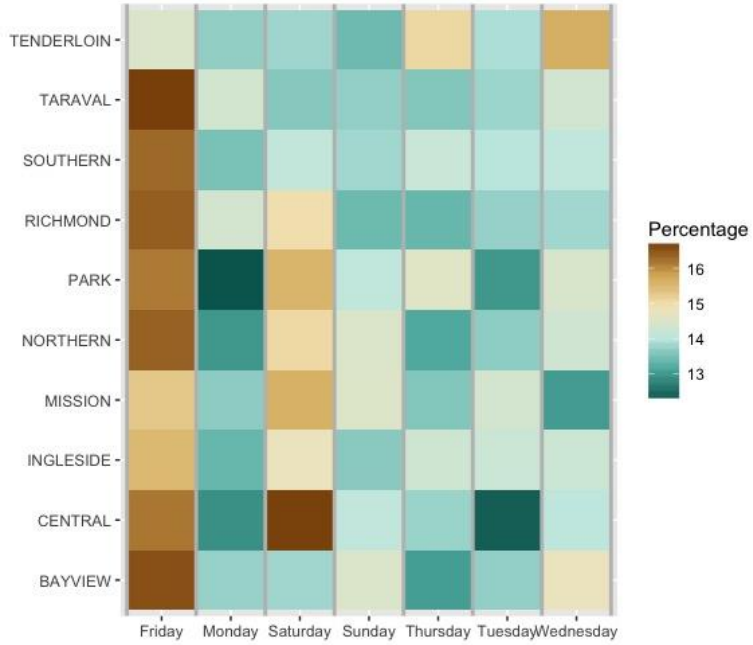

Figure 16: Relationship between crime rates and districts

Other variables that might show dependency were the day of the week and crime type (Figure 15), and crime type and months in which crimes occurred (Figure 16).

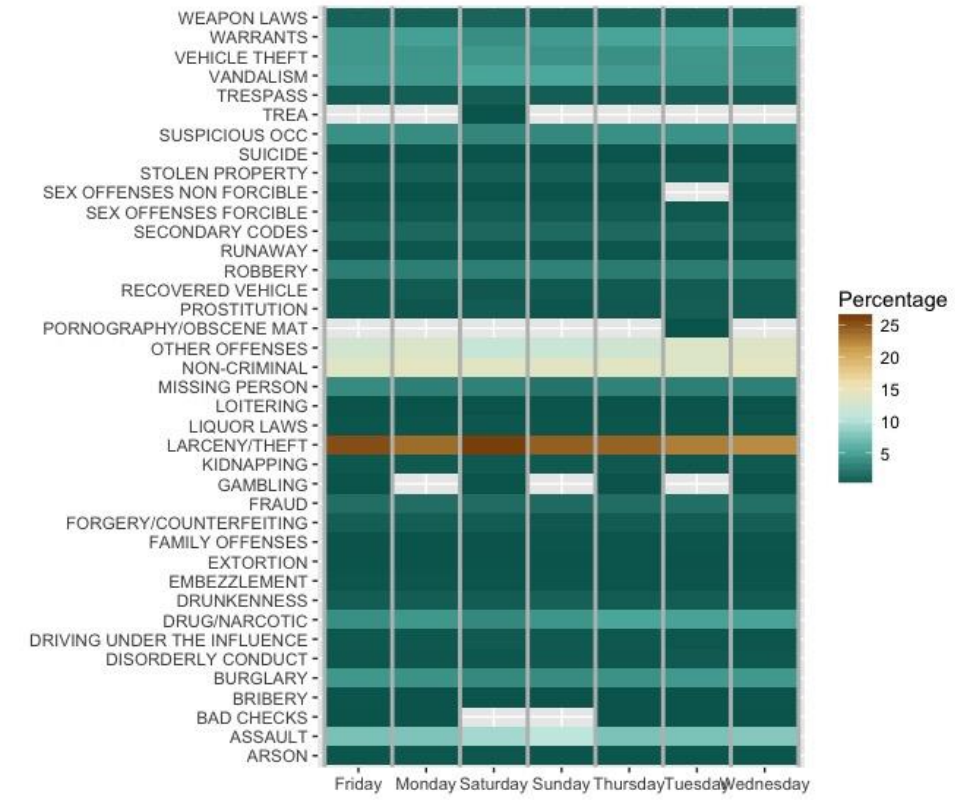

Figure 17: Relation between crime types and day of the week

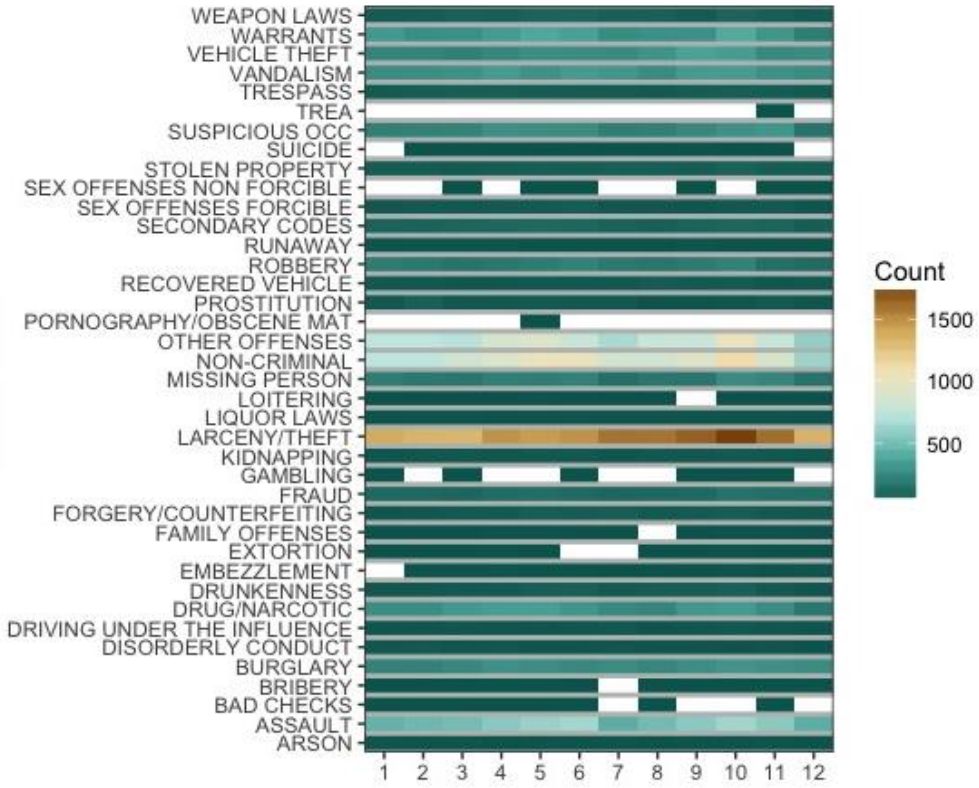

Figure 18: Relation between crime types and months

Going through the crime counts in the previous analysis, Larceny/Theft was consistently the most prominent crime in every combination of performed studies. Here, Figure 16 shows Larceny/Theft occurrence per district in contrast with day of the week. Figure 16 was created using the point geom from the ggplot2 package which created a scatterplot (bubble chart style) showing the relationship between three variables - districts, weekdays and larceny counts mapped to the size of points.

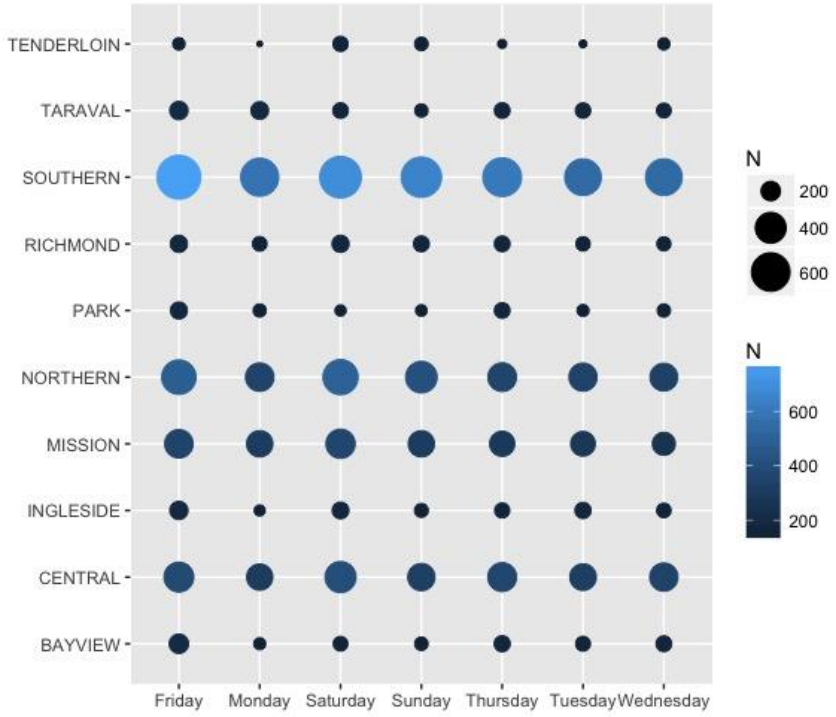

Figure 19: Larceny/Theft per District

Figures 19 and 20 show the distribution of Larceny/Theft on Friday and Saturday, as the days of highest criminal activity. Both figures were created in the same way as previously explained for Figures 13 and 14, with the distinction of using geom tiles rather than geom points, for showing the counted number of larceny occurrences on a chosen day.

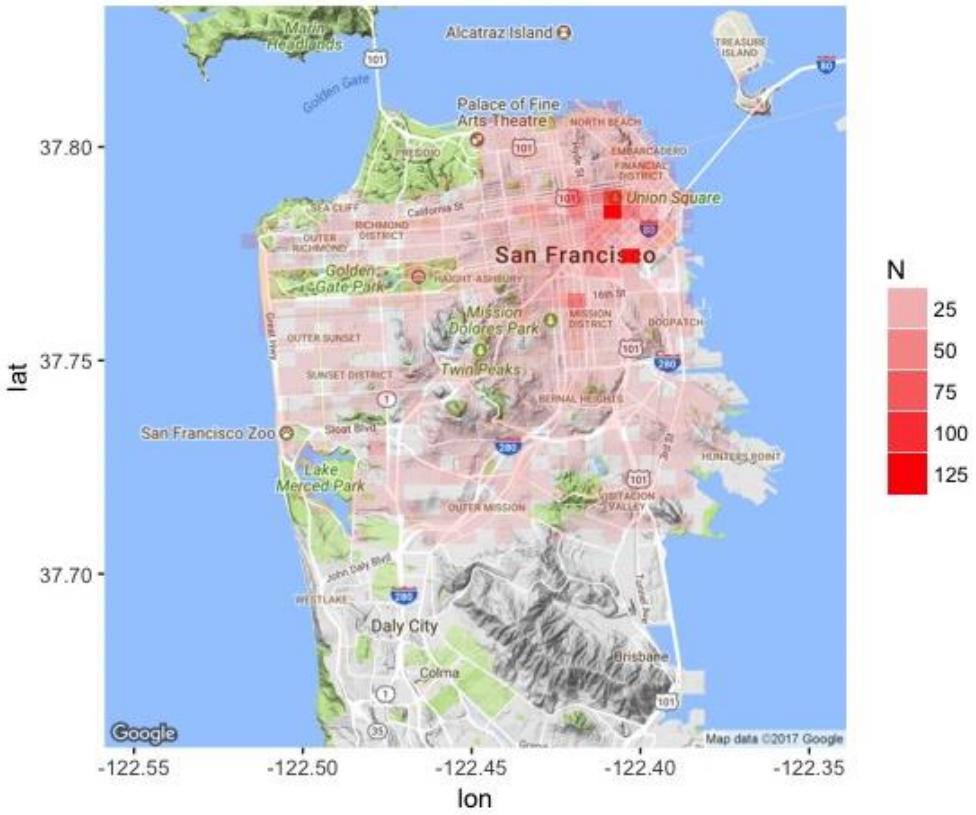

Figure 20: Larceny/Theft on Friday

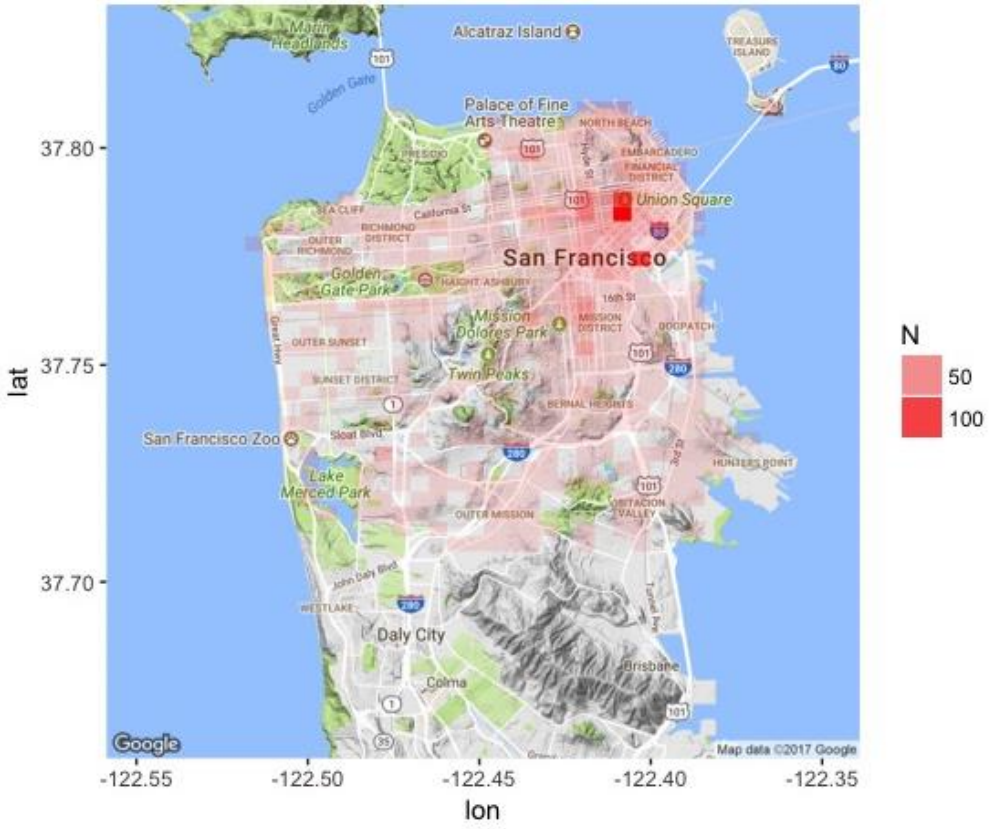

Figure 21: Larceny/Theft on Saturday

For accomplishing the second case study goal, within these intelligence analysis, one should focus on answering when and where arrests frequently occur. Analysis showed that almost 60% of crimes were not resolved. Figure 17 shows the counted crimes according to the resolution type. For creating this figure the number of crimes was counted by resolution type using the afore mentioned option of .N, delivered with the data.table package, and plotted as a bar graph using the ggplot2 package.

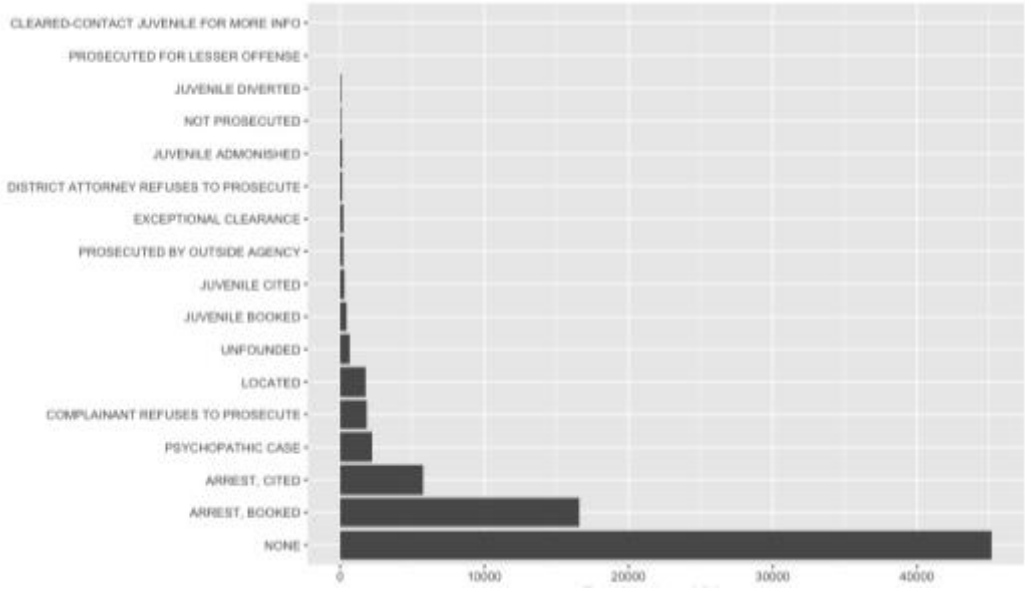

Figure 22: Total number of crimes per Resolution type

The next focus was on the Resolution type. Figure 18 gives an overview of each district's preferred resolution type. This gave the answer to the question of where the arrests frequently occurred.

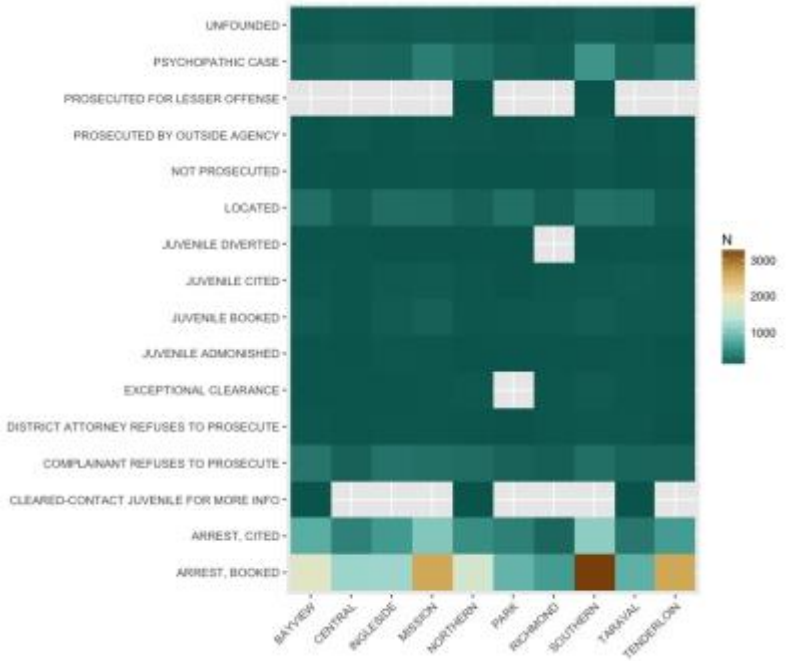

Figure 23: Crime Resolution overview per District

The next point of interest was to check these arrests counts depending on the day of the week and the time at which they happened, therefore, to answer the question *when* did the arrests most frequently occur.

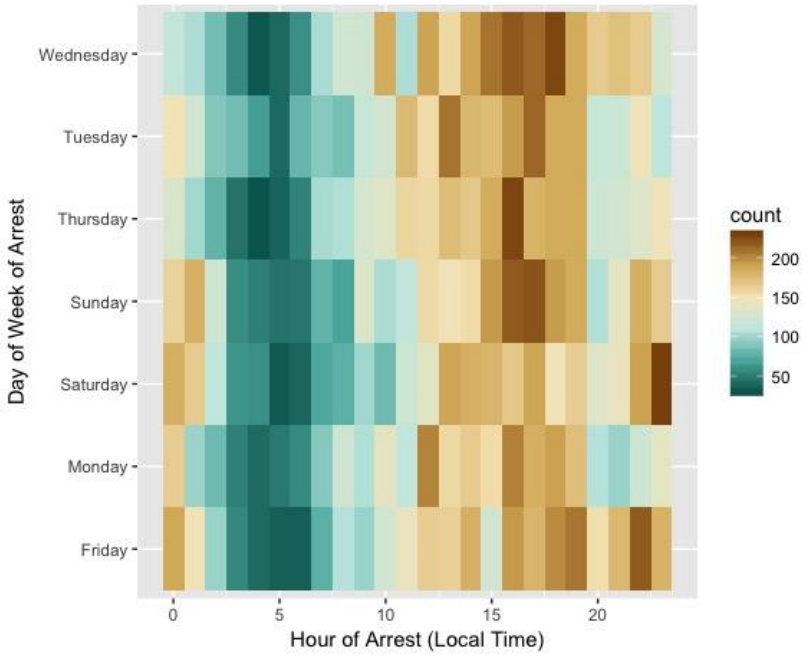

Figure 24:Arrest counts per Day of Week and Time

Both figures (Figure 18 and Figure 19) were created using the R package ggplot2 and outputted in a form of heat maps. For creating Figure 18, it was necessary to count the resolved crime cases by districts, excluding unresolved ones and plot the heat map with districts on the x axis, and resolution type on the y axis, displayed with resolution count values. For creating Figure 19, arrests were aggregated with the dplyr package by the day of the week and hour at which they occurred, and then summarized. A heat map was created the same way as Figure 18, changing the axis labels accordingly.

The intelligence analysis performed with R aimed to show potential relationships and dependencies among variables stored within the dataset, investigate when and where arrests happened, and offer different types of visualisation of results of the analyses.

3.3.3. Tactical analysis with R-ArcGIS Bridge

To be able to understand crime trends and discover patterns which are not obvious, to perform tactical analysis, one has to take advantage of all available sources and needs. In many cases, analysis is performed not only with one software, or one specific tool, but rather combined and integrated. In order to answer the third, fourth and fifth case study goals:

3. Identify where crimes are emerging,
4. Identify areas with unusually high number of crimes,

5. Investigate factors that might influence unlawful behavior,

one has to think of one such integration. Keeping in mind the statistical libraries and functions of R and the mapping, publishing, and sharing capabilities of ArcGIS Desktop (as well as ArcGIS Pro since that is the desktop version used for this part of the analysis) these tools make a perfect match for furthering analysis. The integration of these tools is possible through the R-ArcGIS Bridge, which enables easy transfer of the data from ArcGIS Pro to R and vice versa (installation steps and direct link to downloading the Bridge is enclosed in Appendix 6).

In its basic functionalities, ArcGIS Pro offers a tool called Emerging Hot Spot Analysis (Appendix 7). This tool would be capable of answering the question of where crimes are emerging (case study goal number 3). However, as an input feature, this tool requires the crime counts to be aggregated in space and time. That way, some spatiotemporal relationships might be revealed. These analyses were performed by running the ArcGIS Pro Space Time Cube with the Aggregating Points tool (Appendix 7). What is important here, is to properly set up the parameters of the cube because these define the shape and the size of the space-time bins. By default, after execution, the tool creates a netCDF[12] file and delivers a Message window (Figure 25) with some basic information. Additionally, Space Time Cube can be visualized in ArcGIS Pro either as in 3D or in 2D. Figure 26 shows snippets of a 3D scene captured on a monthly level, starting from January and ending with December.

Figure 25: Space Time Cube Message Window

[12] netCDF, Accessed on 15.09.2017.

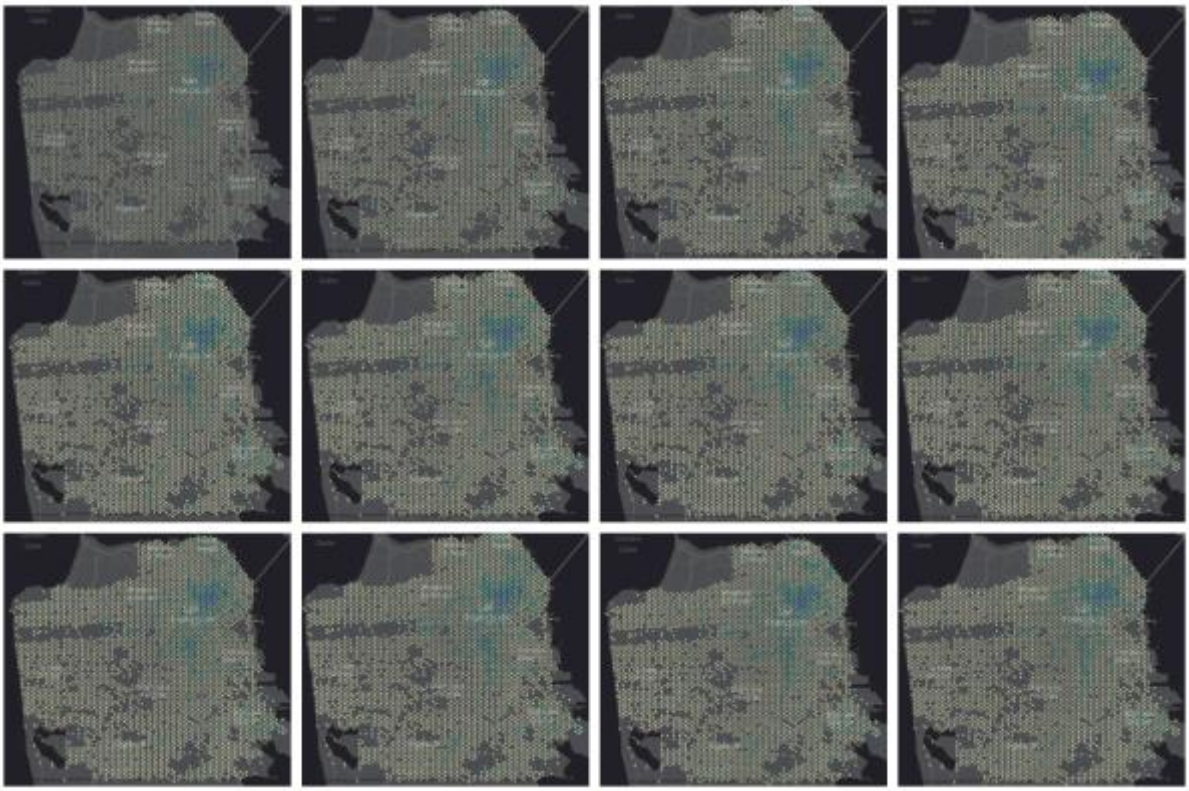

Figure 26: Visualisation of the Space Time Cube

As mentioned, the Space Time Cube output was necessary for performing the Emerging Hot Spot Analysis (Figure 27). Figure 27 shows trends in statistically significant hot and cold spots. Red areas indicate high number of crime clustering over time, while blue represents low numbers of crime. Dark red areas represent persistent hot spots which do not have discernible increase or decrease in the intensity of the crime clustering over time. On the contrary, light red hexagons represent intensifying hot spots where intensity of crime counts clustering is increased over time, and this increase is statistically significant. On the other side, dark blue areas represent cold spots where crimes are statistically less prevalent. Light blue bins stand for intensified clusters of low crime counts which means that cold spots are getting colder. What grabs the attention on this map are locations with persistent and intensifying crimes.

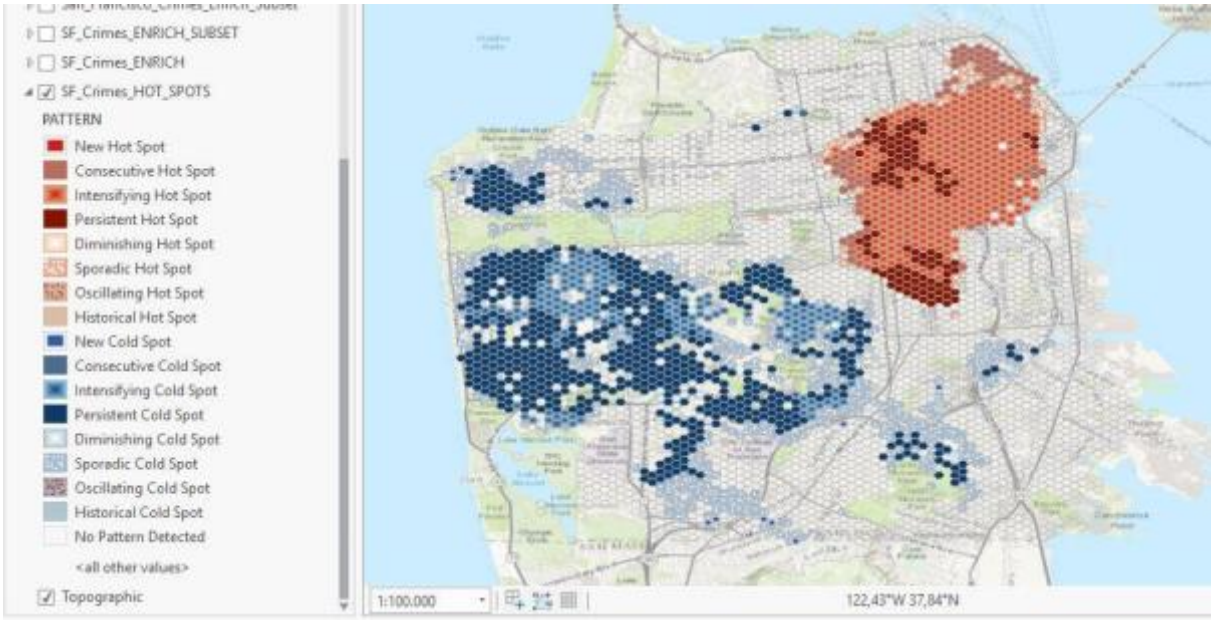

Figure 27: San Francisco Crimes Hot Spots

The next task was to identify which areas of San Francisco have an unexpectedly high number of crimes given the number of people (case study goal number 4). Again, ArcGIS Pro in its basic functionality has Optimized Hot Spot Analysis, capable of performing this task, but again there was a pre-step which needed to be handled prior to the tool execution and that was to calculate crime rates.

A crime rate describes the number of crimes reported to law enforcement agencies for every 100 000 persons within a population. It is calculated by dividing the number of reported crimes by the total population and multiplying the result by 100 000. This step was could be performed using the attribute table's Field Calculator, however to ensure that calculated crime rates are statistically robust, the Empirical Bayes smoothing method was used. Empirical Bayes method is the procedure for statistical inference in which the prior distribution is estimated from the data which is in contrast to standard Bayesian methods, for which the prior distribution is fixed before any data is observed (Petrone, Rizzelli, Rousseau, & Scricciolo, 2014). In this case, population in each of the bins will be a measure of confidence in the data. This means, the higher the population the higher the confidence. Additionally, it adjusts the areas with lower confidence towards the mean which gives the stability to the calculated crime rates. The Empirical Bayes method is contained in R's *spdep* package and as such it leverages the calculation of the crime rates. That means that the data was transferred from ArcGIS Pro to R using the previously mentioned and installed R-ArcGIS Bridge. Further calculations can be continued in the R Studio environment. The code enclosed in Appendix 3 (bullet point number 7) calculates the crime rates and writes the results in a new column. This output was then transferred back into ArcGIS Pro, again using the R-ArcGIS Bridge. With the calculated crime rates, Optimized Hot Spot analysis was performed using the crime rates as an input features. Figure 28 shows areas with intense clustering of high values (with 99% confidence). These areas stand for unusually high number of crimes occurring while accounting for

population. What is additionally interesting is that, when the population of each area is considered, there is no clustering of low crime counts.

Figure 28: Optimized Hot Spot Analysis

Knowing where crimes are occurring drives the question of why they are occurring. One of the potential answers can be the relationship between crimes and population, income, gender, or business (case study goal number 5).

Since the original San Francisco dataset does not contain information about population, income, education, business nor any other potential influencer of unlawful behaviour, it was necessary to pair the external datasets with the original one or use the functionality of ArcGIS Pro again and enrich the original data with the desired variables. ArcGIS Pro Enrich Layer tool (Appendix 7) enriches the data by adding chosen demographic variables (such as population, household, income, housing, mortgage status, family, race and ethnicity) about the people and places that surround or are inside data locations. The result of this tool execution was the dataset needed for performing further analysis. As mentioned earlier, the newly created dataset contained information about the old data and information that were added as variables prior to tool execution.

For identifying possible correlations between crime rates and enriched variables, one can switch back to the R Studio environment and use the exploratory data analysis tool called Correlation matrix. The matrix is presented as a grid that measures the level of association between the added attributes and desired variable and with one another. Attributes with higher correlation will be more prompt as predictors when trying to find the best fitting predictive model for the dataset. The matrix can also serve to identify possible multicollinearity between predictors. The matrix was created using the cormat function from the R package corrplot. The full code description for calculating the matrix coefficients and creating the matrix was enclosed in the Appendix 3. The correlation matrix was visualised using the R' package corrplot. The chosen visualisation method was

circle, the layout was upper triangular matrix and the colour scheme is left as default. Figure 29 shows the matrix with calculated correlation coefficients proportional to the circle size.

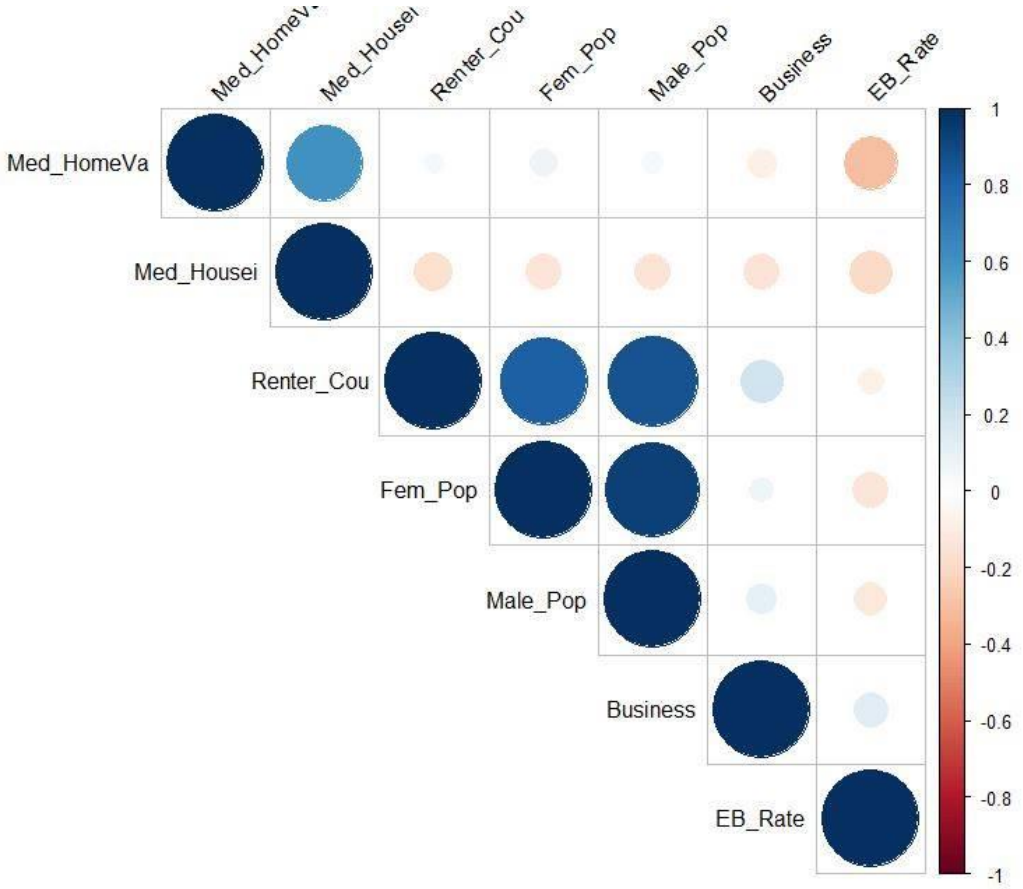

Figure 29: Correlation matrix

The higher the coefficient is, the stronger the relationship is. Correlation can be positive and negative. In the case of positive correlation between predictor and dependant value, the magnitude of both predictor and dependent variable increases. On the other hand, with a negative correlation coefficient - while the magnitude of the predictor increases, the magnitude of the dependent value decreases. Multicollinearity measures the similarity of two predictors and it is considered as a valuable measure in investigation of high crime rate influencers.

Positive correlation is displayed in blue, while negative is in red. Colour intensity and the size of the circle is proportional to the value of the correlation coefficient. The legend on the right side shows the correlation coefficients and corresponding colours. Most of the circles are small in size and light in colour which indicates that there is no strong correlation among these attributes. This type of visualisation can be of great use when there is a need to choose which predictors to include in a predictive model. However, this process requires advanced statistical methods to account for possible non-linear terms, spatial trends and other factors.

3.3.4. Predictive policing with Python scikit-learn

The final step of this case study scenario was to build a model which would predict where higher numbers of crimes were likely to occur. For building such a model some dataset pre-processing was necessary. As the original San Francisco dataset contained unnecessary information for building the prediction model, some columns were dropped and renamed and the original dataset was split into the training (Figure 25) and testing (Figure 26) set. The next step was to select training and validation sets and to recode categories from both testing and training datasets (these were Category and PDistrict) into numerical values.

```
In [179]: df_train.head(5)
Out[179]:
       Category                                          Descript  DayOfWeek  \
42538        32                          SUSPICIOUS OCCURRENCE          4
48588         4  BURGLARY,STORE UNDER CONSTRUCTION, FORCIBLE ENTRY          6
32652        20                      AIDED CASE, MENTAL DISTURBED          2
63917        36                                STOLEN TRUCK          1
2805         16                      PETTY THEFT FROM LOCKED AUTO          0

       PdDistrict        Resolution           X          Y
42538          1              NONE -122.418498  37.796824
48588          4              NONE -122.433622  37.800270
32652          7  PSYCHOPATHIC CASE -122.391425  37.777873
63917          0              NONE -122.409627  37.732755
2805           0              NONE -122.400589  37.738988
```

Figure 25: Train data

```
In [177]: df_test.head(5)
Out[177]:
        ID  DayOfWeek  PdDistrict           X          Y
0  102350          3           4 -122.423750  37.776729
1  102351          3           4 -122.435053  37.790659
2  102352          3           4 -122.419295  37.787805
3  102353          3           7 -122.401692  37.786065
4  102354          3           0 -122.398441  37.719339
```

Figure 26: Test data

Training and validation tests were then split for the purpose of creating a design matrix and response vector of the training data and a design matrix for the test data (Appendix 4).

The results were used for creating Logistic classifier, AdaBoost Logistic classifier and Random Forest Classifier (all three classifiers are contained in Python's scikit-learn package). Additionally, mean accuracy of each was calculated with results presented in Table 5.

Table 5: Mean accuracy of Classifiers

	Mean accuracy
Logistic Classifier	0.2388
AdaBoost Classifier	0,2043
Random Forest Classifier	0,2100

It is worth noting that in all the models, the classification accuracy was low, less than 25% even in the best model.

IV OUTPUTS

Keeping in mind the main thesis objectives presented in Chapter 1, section 1.2 one should think of the thesis outputs rather than results. Starting from the choice of tools which influenced the choice of dataset, over the developed methodology whose realization gave the outputs. This chapter gives an overview tasks performed, accompanied with descriptions of tools which were used to accomplish these tasks and theoretical explanations of given outputs. Graphical representation of the outputs overview is given in the Table 6.

Table 6: Outputs overview

Task	Tools	Outputs
Strategic analyses (criminal activity over time)	Python packages: pandas, numpy, matplotlib	**Bar** and **Line Graphs** showing where and when crimes occur
Intelligence analysis (variable dependencies)	R libraries: data.table, lubridate, dplyr, ggplot2, ggmap	Heat maps, scatterplot, bar graph, density map
Tactical analyses (trends and patterns)	ArcGIS Pro: Emerging Hot Spot Analyses	**Map** showing Hot and Cold spots of crime occurrence
	ArcGIS Pro: Optimized Hot Spot Analyses	**Map** with unusually high number of crimes
	R libraries: corrplot	Correlation matrix
Predictive policing (where crimes are likely to occur)	Python model: scikit-learn	Classifiers accuracy: Logistic, AdaBoost Logistic Random Forest

As seen in Chapter III, section 3.3.1 the first set of analysis, called strategic analysis was performed using the Python stand-alone script. Python packages pandas, numpy and matplotlib were used for getting the desired answers (case study goal number 1). Pandas and numpy packages were used for manipulating with the data (such as counting the number of crimes or calculating the relative time occurrence), while matplolib package delivered visualization outputs in the form of bar graphs, line graphs and combined line graphs (Figure 12).

The next set of analyses, called intelligence analysis, was performed within R stand-alone script. R libraries, including data.table, lubridate and dplyr were used

for data manipulation (similar as with Python's pandas and numpy) and prepared the data for visualization. Visualization was done using the ggplot 2 library which gave different outputs in the form of heat maps, scatter plot and bar graphs which showed the relationship between dataset variables (case study goal number 2). Moreover, ggmap enabled creation of the static map (from the online source which in this case was Google maps) and added the density layer on top of the static map for a final output.

The third set of analyses, called tactical analyses were performed using the joint forces of R and ArcGIS Pro. Powerful ArcGIS Pro tools, such as Create Space Time Cube, Emerging Hot Spot analysis, and Optimized Hot Spot Analysis gave informative outputs in the form of maps, for answering case study goals number 3 and 4. However, for running these tools, additional data manipulation was necessary, such as calculating the crime rates which was done in R using the package spdep. This output was in the form of numerical columns which was stored within the original dataset. Further contribution from ArcGIS Pro comes from the Enrich layer tool which output a new, enriched, dataset. This output was at the same time input for calculating the correlation coefficients between dataset variables (case study goal number 5). For accomplishing this task, R library corrplot was used. The final output of this calculation was a visual representation of the Correlation matrix.

The final set of analyses, called predictive policing, were performed using the Python's sikit-learn package which provides different algorithms for building the prediction model (case study goal number 6). Prediction model output gave numerical information about classifiers accuracy which did not provide satisfying results, therefore further explanations were intentionally excluded from the Outputs chapter. Due to the complexity of building a prediction model this part of the thesis referred mostly to existing literature and work performed by others in this field.

V DISCUSSION

Having a table of all the outputs listed in Chapter 5, one can discuss the findings in relation to the research objectives from Chapter 1 as well as reviewed literature from Chapter 2. Additionally, some interesting findings with respect to the case study goals, about San Francisco crime data were added as a final section of the discussion chapter.

5.1. Discussion of the findings

The discussion of the finding was conducted by referencing the research objectives. Starting from the decision as to which open-source programming languages to use, combined with GIS, to perform special analyses and data interpretation. According to (De Sarkar, Biyahut, Kritika, & Singh, 2012) , (Zhang, Yue, & Guo, 2014) and (Szczepankowska, Pawliczuk, & Źróbek, 2013)Python has proven to be the best choice for writing scripts to automate programming functionalities in the GIS environment. Whether it is combined with open-source or propriety GIS software, the authors mentioned descibe Python as an interpreted, portable, object-oriented, high level language with readability, extensibility, ease of programming, free and open source, platform independent, with a variety of standard extensions which makes development of geospatial applications simpler, faster and more efficient. In relation to this thesis case study, Python showed fast and straight-forward performance in the scope of exploratory (strategic) analysis. Python packages pandas, numpy and matplotlib enabled easy and quick data pre-processing, handling and 2D visualisation which gave insights and more information about the entire dataset, as well as spatial and temporal distribution of the data. What was especially interesting was the matplotlib package which was used to produce different types of visualisation outputs. The package itself has rich open-source documentation which offers the ability of understandable manipulation over the code functionalities. That way, the graphics are produced with the desired appearance (with or without figure title, renaming the axis, background colour, special graphic theme, type of the graph, etc.).

On the other hand, (Allen, Tsou, Aslam, Nagel, & Gawron, 2016) showed how GIS combined with machine learning (approachable through programming languages such as R) can enhance existing methods of information extraction from the data. (Anselin, 2012)describes the R environment (combined with GIS) and its packages as a sophisticated way to deal with spatial statistical functionalities and geospatial data structures. (Ciolobac, 2016) categorized programming languages, depending on their native purpose and performance, selecting Python and R as GIS Scripting and applications oriented, data processing, analysis, and modelling capable. The set of packages used in R environment, among which data.table, lubridate, dplyr, ggplot2, ggmap, provided deeper understanding of the information stored within the dataset and their mutual relationship (intelligence analysis). Data.table, lubridate and dplyr enabled counting, ordering, grouping, filtering and aggregating of the data. That meant fast and one-line-code data manipulation for desired visualisation which was done using the ggplot2 and ggmap packages. These packages, which by default come with embedded plotting options, enabled different visualisation outputs (heat maps, scatter plots, bar graphs) through easy

manipulation and modification of the options mentioned. Moreover, the ggmap package offered an option of creating maps with R (density map for example), which were overlaid with specific layers (such as geom point and geom tile) from ggplot2. That provided direct interpretation of specific crime occurrence and overall crime distribution. However, for adding essential elements on the map (such as legend, north arrow, scale) one must invest more coding effort which would be time consuming for this scope of analysis.

When it comes to ArcGIS Pro performance, the tools used (Space Time Cube, Emerging Hot Spot Analysis, Optimised Hot Spot Analysis) gave necessary answers to the case study goals. However, some calculations (which preceded running these tools) such as crime rate calculations which had to be statistically robust, were performed faster and easier in R using the spdep package embedded functions. Other than that, ArcGIS Pro offers flexible and easily-manageable manipulation for map creation. Once created, maps can be customized using the wide range of options provided, such as symbolization, style, base map, adding secondary content, etc.

The second research objective focused on which dataset to use and the methodology to develop real world example scenarios to prove the afore mentioned choices. The decision regarding which data to use was supported by the fact that many local, regional and national security authorities are turning towards new decision support tools such as GIS and predictive models to find better solutions for crime prevention (Ferreira, João, & Martins, GIS for Crime Analysis - Geography for Predictive Models, 2012). Additional support came from the fact that 88% of police departments use GIS software for crime mapping purposes (Harries, Mapping Crime and Geographic Information Systems, 1999, p. 94). GIS growing potential can be employed at different levels to support operational policing, tactical crime mapping, detection, and wider-ranging strategic analyses (Chainey & Ratcliffe, 2005). Methodology steps were defined by the type of the crime analysis which corresponded to the classical data analysis workflow (Figure 3). Together with the six case study goals these enabled developing an easy-to-understand, real- world-example in which the utility of the chosen tools was demonstrated.

The third research objective referred to how to combine these programming languages with GIS, and how to, most efficiently, divide the work among them. Referring to existing literature, there are several ways in which programming languages, such as Python and R can be integrated with ArcGIS. When it comes to Python, this integration can be done either through stand-alone scripts, ArcPy references or ArcPy API's. Due to the nature of the data and planned tasks (crime analysis), stand-alone script was used for this thesis study case. Strategic analyses performed in Python were executed in the Spyder environment (as it was seen as a structured and organized environment due to the possibility of writing a code, executing it directly in a console and visualizing the outputs in the same window). When it comes to R, the integration with ArcGIS Pro is possible and was done through R-ArcGIS Bridge. That way, the statistical power of R, enforced with the spatial capabilities of ArcGIS Pro gave significant answers to the study goals. Other than that, R stand-alone script was used for performing strategic analyses on the data and discovering hidden information by providing different

visualization outputs and interpretation options. The idea of this case study was to conduct, using Python packages and R libraries, statistics and visualization outputs in programming languages and prepare data for ArcGIS Pro where further, spatial, analyses were performed. Additionally, the idea was to use only the ArcGIS Pro and its basic functionalities. That way, police departments or any other party interested in extracting information out of criminal data, would be able to avoid additional license extension.

An additional question came along with the research objectives and it referred to the scope of work to be done either by GIS professionals or Data Scientists. Considering the fact that Cartographers are familiar with GIS, and not necessary with programming skills, the case study was done so that it included a portion of both (coding and GIS) in delivering imagined output. On the other hand, Data Scientists skills would be of great use for interpreting and building predictive models which was done theoretically for most part, within this thesis work. Interaction or at least cooperation, together with crime analysts, would be of great benefit for deeper understanding of unlawful behaviour as well as predicting where and which crimes are most likely to occur in the future, based on historical records.

5.2. Finding about San Francisco crime data

Crime analysis performed with Python, R and ArcGIS Pro gave some interesting findings about crime occurrence in San Francisco. Over the years Larceny/Theft was constantly the most prevalent crime in San Francisco, Southern district appeared to be, consistently, the most dangerous district while Richmond was the safest. Fridays were noted as the days with the highest criminal activity while Sundays had the lowest. Furthermore, the highest criminal occurrence was at 6pm and the lowest was at 5 am. On a monthly basis, the highest criminal occurrence was in October whereas the lowest criminal occurrence was in December. For annual distribution, variation among years was low. What was noticed was that the number of crimes increased for 2013. Analysis also showed that 60% of crimes were not resolved at all, while only 30% ended up with arrests which leads one to think that police districts should work more proactively to reduce a such high percentage of unresolved cases.

VI CONCLUSIONS AND FINAL REMARKS

The last chapter of this thesis was reserved for making conclusions from the given outputs and performed works. Additionally, limitations and possibilities for future work will be discussed.

6.1. Summary

The broad aim of this thesis was to explore, examine and discuss whether the combination of GIS software with programming languages leverages custom work in Spatial Data Science. In order to answer this question, as well as more specific ones described as research objectives and research questions, one had to concentrate on two main tasks. The first was related with to an extensive literature research and discovering what is being done in this field, what is not covered and who can give the most suitable answers to questions of interest. The second was to develop a case study scenario in which such integration of tools makes sense and gives outputs which would be harder to obtain in other circumstances. Additionally, the second task involved finding the proper dataset upon which all analyses were performed.

6.2. Conclusions

This thesis aimed to combine basic knowledge from the programming world, with the spatial tools and capabilities of GIS, as well as a cartographer's tendency to visualize all information gained from the data, in the form of maps, graphs, plots and other informative content.

All that has been done, described and discussed in the previous chapters suggests that Python and R, as open-source programming languages, combined with ArcGIS Pro, as a GIS software offer the opportunities to explore, test and extract useful information out of datasets. This trio, showed to be effective, efficient and fast in response with the chosen dataset.

Python and R libraries (such as pandas, numpy, matplotlib, data.frame, dplyr, ggplot2, ggmap, corrplot) showed efficient performance in identifying and visualizing crime spatiotemporal relationships, crime trends and patterns which gave insights into the entire dataset. ArcGIS Pro has very useful tools for crime analysis in its basic functionality, such as Emerging Hot Spot Analyses, Optimised Hot Spot Analyses, Space Time Cube and Enrich Layer. Combined, these tools enable information extraction and visualisation of criminal data. This knowledge aids decision making, problem solving and prevention. Thus, Cartography with additional side resources is useful in crime analysis and criminology and identifying patterns in crime for police departments to more accordingly place personnel and patrol vehicles.

With this thesis, more understanding of the local security environment of San Francisco city was achieved, partly from informative visualizations and partly

from the predictive analyses of ArcGIS Pro. More importantly, similar methods can be applied to future data or crime data from other cities. A larger population can hopefully benefit from a better security environment with the help of these analyses. For future work, more interesting questions can be studied further. Some examples are mentioned in the section that follows.

6.3. Limitations and Further study recommendations

Within the scope of this thesis, only two programming languages and one software were considered. That influenced decision making regarding tools and software choice as well as the extent of analyses and cooperation with other experts in fields of interest, mostly referring to Data Scientist and their support in choosing and interpreting prediction models. However, significant work has been done and many notes were saved for some potential future work.

Among further study recommendations, the first one would concentrate on programming language choice. The list of two should be extended which would give a possibly to perform evolution upon which one showed best performance. The same goes for GIS software. It would be beneficial to have more than one software involved in the case study. That way, it would be possible to say which one performed better in any given way and discuss further about options of integrating programming languages with the software itself. Lastly, considering the crime dataset used in this case study, it would be beneficial to concentrate on potential crime influencers and the process of finding good predictors in a given data set for a particular response variable (other than a correlation matrix approach used here). This can be tricky and can require advanced statistical methods to account for possible non-linear terms, spatial trends, and other factors.

Some interesting questions that may be handled in future work might refer to finding out why October has the highest reported criminal activity, and December the lowest? Why Larceny/Theft persists to be the most prominent crime over the years in San Francisco? It would be interesting to see how joint forces of professionals from different field such as, Cartography, GIS, Data Science, and Crime Analyst, can help answer these and many more interesting questions that will come along the way in future analyses or to include Big Data while this was not covered within this thesis scope. Also, it would be interesting to know how confident SFPD feel in regards to these analyses and if they would apply the given outputs to enforce enhanced crime control.

BIBLIOGRAPHY

Ahmadi, M. (2003, 02). Crime Mapping and Spatial Analysis. Enschede, Netherlands.

Allen, C., Tsou, M., Aslam, A., Nagel, A., & Gawron, J. (2016). Applying GIS and Machine Learning Methods to Twitter Data for Multiscale Surveillance of Influenza. *PLoS ONE , 11*(7), 10.

An overview of the Space Time Pattern Mining toolbox—ArcGIS Help | ArcGIS Desktop. (n.d.). Retrieved 08 20, 2017, from ArcGIS Pro: http://pro.arcgis.com/en/pro-app/tool-reference/space-time-pattern-mining/an-overview-of-the-space-time-pattern-mining-toolbox.htm#GUID-937C56DF-E112-4ABE-924F-E892A2371884

Ang, S., Wang, W., & Chyou, S. (2015). *San Francisco Crime Classification.* UCSDCSE.

Anselin, L. (2012). From SpaceStat to CyberGIS. *International Regional Science Review, 35*(2), 131-157.

Berhane, F. (2017, 05 17). *R Bloggers.* Retrieved 08 28, 2017, from Best packages for data manipulation in R: https://www.r-bloggers.com/best-packages-for-data-manipulation-in-r/

Bivand, R., Pebesma, E., & Gómez-Rubio, V. (2008). Hello World: Introducing Spatial Data. In R. Bivand, E. Pebesma, & V. Gómez-Rubio, *Applied Spatial Data Analysis with R* (p. 371). New York: Springer.

Bogomolov, A. L. (2014). Once upon a crime: towards crime prediction from demographics and mobile data. *Proceedings of the 16th international conference on multimodal interaction* (pp. 427-434). Istanbul, Turkey: ACM.

Chainey, S., & Ratcliffe, J. (2005). *GIS and Crime Mapping.* Wiley.

Chen, H., Chung, W., Xu, J., Wang, G., Qin, Y., & Chau, M. (2004). Crime data mining: A general framework and some examples. *Computer, 37*(4), 50-56.

Ciolobac, F. (2016, 11 16). What are the Top Programming Languages in the GIS World. *Geomatic Canada Magazine*.

Cios, K., Swiniarski, R., & Pedrycz, W. (2007). The Knowledge Discovery Approach. In K. Cios, W. Pedrycz, R. Swiniarski, & L. Kurgan, *Data Mining* (p. 606). Springer.

Coleridge, S. (2016). The role of data wrangling. In B. Boehmke, *Data Wrangling with R* (p. 237). Dayton, USA: Springer.

Crampton, J., Graham, M., Poorthuis, A., Shelton, T., Stephens, M., Wilson, M., & Zook, M. (2013). Beyond the geotag: situating 'big data' and laveraging the potential of the geoweb. *Cartography and the Geographic Information Science, 40*(2), 130-139.

Dağlar, M., & Argun, U. (2016). Crime Mapping and Geographical Information Systems in Crime Analysis. *Journal of Human Sciences, 13*(1), 14.

De Sarkar, A., Biyahut, N., Kritika, S., & Singh, N. (2012). An environment monitoring interface using GRASS GIS and Python. *Third International Conference on Emerging Applications of Information Technology* (pp. 235-238). Kolkota, India: IEEE.

Dedic, N., & Stainer, C. (2017). Towards Differentiting Bussiness Intelligence, Big Data, Data Analytics and Knowledge Discovery. *Inovations in Enterprise Information Systems Management and Enginering: 5th International Conference* (pp. 114-122). Hagenberg, Austria: Springer International Publishing AG.

Dempsey, C. (2012, 10 28). *Where is the Phrase "80% of Data is Geographic" From*. Retrieved 05 25, 2017, from GIS Lounge: https://www.gislounge.com/80-percent-data-is-geographic/

Donnelly, F. P. (2010). Evaluating open source GIS for libraries. *Library Hi Tech, 28*(1), 131-151.

Ferreira, J., João, P., & Martins, J. (2012). GIS for Crime Analysis - Geography for Predictive Models. *The Electronic Journal Information Systems Evaluation, 15*(2), 36-49.

Ferreira, J., João, P., & Martins, J. (2012). GIS for Crime Analysis - Geography for Predictive Models. *The Electronic Journal Information Systems Evaluation, 15*(1), 36-49.

Goodchild, M. (2016, April 25). GIS in the Era of Big Data. Cybergeo : European Journal of Geography . Retrieved May 09, 2017, from http://cybergeo.revues.org/27647

Grolemund, G., & Wickham, H. (2011). Dates and Times Made Easy with lubridate. *Journal of Statistical Software, 40*(3), 25.

Harries, K. (1999). Mapping Crime and Geographic Information Systems. In K. Harries, *Mapping Crime: Principle and Practice* (pp. 91-126). Washington DC: National Criminal Justice.

Harries, K. (1999). Mapping Crime: Principle and Practice. In K. Harries, *Mapping Crime: Principle and Practice.* Washington DC: National Institute of Justice.

Hayashi, C. (1998). What is Data Science ? Fundamental Concepts and a Heuristic Example. In C. Hayashi, K. Yajima, H. Bock, N. Ohsumi, Y. Tanaka, & Y. Baba, *Data Science, Classification, and Related Methods* (p. 780). Japan: Springer.

Henshaw, S., Curriero, F., Shields, T., Glass, G., Strickland, P., & Breysse, P. (2004). Geostatistics and GIS: Tools for Characterizing Environmental Contamination. *Journal of Medical Systems, 28*(4), 335-348.

Hesse, W., Moser, R., & Riley, W. (2015). From Big Data to Knowledge in the Social Sciences. *The ANNALS of the American Academy of Political and Social Science, 659*(1), 16-32.

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

Hunter, J., Dale, D., Firing, E., & Droettboom, M. (2017, 05 10). *matplotlib*. Retrieved 09 09, 2017, from matplotlib: http://matplotlib.org

Ioimo, R. (2016). Crime Analyses and Crime Mapping. In R. Ioimo, *Introduction to Criminal Justice Information Systems* (p. 335). Boca Raton, Florida: CRC Press.

Iqbal, R., Murad, M., Mustapha, A., Panahy, P., & Khanahmadliravi, N. (2013). An experimental study of classification algorithms for crime prediction. *Indian Journal of Science and Technology, 6*(3), 4219-4225.

Kahle, D., & Wickham, H. (2013). Spatial Visualization with ggplot2. *The R Journal , 5*(1), 144-161.

Ke, J., Li, X., & Chen, J. (2015). *San Francisco Crime Classification.*
Kraak, M., & Koussoulakou, A. (2005). A Visualization Environment for the Space-Time-Cube. *Developments in Spatial Data Handling* (pp. 189-200). Berlin: Springer.
Krygier, J. (2013). Cartography as an art and a science. *The Cartographic Journal, 32*(1), 3-10.
Kwakkel, J., Carley, S., Chase, J., & Cunningham, S. (2014). Visualizing geo-spatial data in science, technology and innovation. *Technological Forecasting and Social Change, 81*, 67-81.
LeBlanc, J., Elder, J., Bruce, C., Cook, T., Rodriguez, E., & Steiner, F. (2014, 10 02). *Definiton and Types of Crime Analysis.* Retrieved 06 25, 2017, from IACA - International Association of Crime Analysts: http://www.iaca.net/Publications/Whitepapers/iacawp_2014_02_definition_types_crime_analysis.pdf
Li, J., Rabah, S., Liu, M., & Lai, Y. (2010). Comparative Studies of 10 Programming Language within 10 Diverse Criteria. *a Team 7 COMP6411-S10 Term Report*, 139.
Li, S., Dragicevic, S., Castro, F., Sester, M., Winter, S., Coltekin, A., . . . Cheng, T. (2016). Geospatial big data handling theory and methods: A review and research challenges. *ISPRS Journal of Photogrammetry and Remote Sensing, 115*, 119-133.
Lugmayr, A., Stockleben, B., Scheib, C., & Mailaparampil, M. (2017). Cognitive Big Data - Survey and Review on Big Data Research and its Implications: What is Really New in Big Data? *Journal of Knowledge Management, 21*(1), 197-212.
Margolis, R., Derr, L., Dunn, M., Huerta, M., Larkin, J., Sheehan, J., . . . Green, E. (2014). The National Institutes of Health's Big Data to Knowledge (BD2K) initiative: capitalizing on biomedical big data. *Journal of the American Medical Informatics Association , 21*(6), 957-958.
Musings, M. (2017, 06 23). *Great R packages for data import, wrangling and visualization.* Retrieved 09 02, 2017, from Computer World: https://www.computerworld.com/article/2921176/business-intelligence/great-r-packages-for-data-import-wrangling-visualization.html
Nakaya, T., & Yano, K. (2010). Visualising Crime Clusters in a Space-time Cube: An Exploratory Data-analysis Approach Using Space-time Kernel Density Estimation and Scan Statistics. *Transactions in GIS, 14*(3), 223-239.
O'Neil, C., & Schutt, R. (2013). *Doing Data Science.* O'Reilly Media.
Pebesma, E., & Bivand, R. (2005, 02). *Cran R project.* Retrieved 09 03, 2017, from Classes and Methods for Spatial Data: the sp Package: https://cran.r-project.org/web/packages/sp/vignettes/intro_sp.pdf
Perry, W., McInnis, B., Price, C., & Smith, S. H. (2013). *Integrated Intelligence and Crime Analysis: Enhanced Information Management for Law Enforcement Leaders.* Rand Corporation.
Petrone, S., Rizzelli, S., Rousseau, J., & Scricciolo, C. (2014). Empirical Bayes methods in classical and Bayesian inference. *Metron, 72*(2), 201-215.
Piatetsky, G. (2017, 05). *KDnuggets*. Retrieved 06 06, 2017, from New Leader, Trends, and Surprises in Analytics, Data Science, Machine Learning Software Poll: http://www.kdnuggets.com/2017/05/poll-analytics-data-science-machine-learning-software-leaders.html

Pluemacher, M. (2016, 06 15). *San-Francisco-crimes*. Retrieved 09 09, 2017, from GitHub: https://github.com/MichaelPluemacher/San-Francisco-crimes

Rahm, E., & Do, H. (2000). Data Cleaning: Problems and Current Approaches. *Data Engineering, 23*(4), 11.

Ratcliffe, J. (2007). *Integrated Intelligence and Crime Analysis: Enhanced Information Management for Law Enforcement Leaders.* Washington, DC: Police Foundation.

Ray, S. (2017, 09 09). *Essentials of Machine Learning Algorithms (with Python and R Codes).* Retrieved 09 15, 2017, from Analytics Svidhya: https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

Robinson, A., Hardisty, F., Dutton, J., & Chaplin, G. (n.d.). *Overview of Programming Languages for GIS*. Retrieved 05 19, 2017, from Penn State: https://www.e-education.psu.edu/geog583/node/67

Rossmo, D., & Summers, L. (2015). Routine Activity Theory in Crime Investigation. In M. Andresen, & G. Farrell, *The Criminal Act* (pp. 19-32). London: Palgrave Macmillan.

Sadler, R., Pizarro, J., Turchan, B., Gasteyer, S., & McGarrell, E. (2017). Exploring the spatial-temporal relationships between a community greening program and neighborhood rates of crime. *Applied Geography, 83*, 13-26.

Saraswat, M. (2017). *Programming Tutorials and Practice Problems.* Retrieved 07 05, 2017, from Practical Tutorial on Data Manipulation with Numpy and Pandas in Python: https://www.hackerearth.com/practice/machine-learning/data-manipulation-visualisation-r-python/tutorial-data-manipulation-numpy-pandas-python/tutorial/

Silva, A. M., & Taborda, R. P. (2013). Integration of beach hydrodynamic and morphodynamic modelling in a GIS environment. *Journal of Coastal Conservation, 17*(2), 201-210.

Skipper, S., & P, J. (2010). Statsmodels: Econometric and Statistical Modeling with Python. *Proceedings of the 9th Python in Science Conference*, (pp. 57-61). Austin, Texas.

Steiniger, S., & Bocher, E. (2009). An overview on current free and open source desktop GIS developments. *International Journal of Geographical Information Science, 23*(10), 1345-1370.

Szczepankowska, K., Pawliczuk, K., & Źróbek, S. (2013). The Python programming language and its capabilities in the GIS environment. *International Geographic Information Systems Conference and Exhibition GIS Odyssey* (pp. 213-222). Zagreb, Croatia: Croatian Information Technology Society – GIS Forum.

Theuwissen, M. (2015, 05). *R vs Python for Data Science: The Winner is ….* Retrieved 07 07, 2017, from KDnuggets: http://www.kdnuggets.com/2015/05/r-vs-python-data-science.html

Toole, J., Eagle, N., & Plotkin, J. (2011). Spatiotemporal correlations in criminal offense records. *Transactions on Intelligent Systems and Technology, 2*(4), 18.

Tukey, J. (1977). *Exploratory Data Analysis* . Pearson.

Varoquaux, G. (n.d.). *Statistics in Python*. Retrieved 09 08, 2017, from Scipy-lectures: http://www.scipy-lectures.org/packages/statistics/index.html

Veronesi, F. (2016, 07 19). *Spatio-Temporal Point Pattern Analysis in ArcGIS with R.* Retrieved 08 10, 2017, from R-bloggers: https://www.r-bloggers.com/spatio-temporal-point-pattern-analysis-in-arcgis-with-r/

Wang, L., Wang, G., & Alexander, C. (2015). Big Data and Visualization: Methods, Challenges and Technology Progress. *Digital Technologies, 1*(1), 33-38.

Wang, S. (2016). Cyber GIS and spatial data science. *GeoJournal, 81*(6), 965-968.

Wang, T., Rudin, C., Wagner, D., & Sevieri, R. (2013). Learning to Detect Patterns of Crime. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 515-530). Berlin, Heidelberg: Springer.

Wickham, H. (2011). The Split-Apply-Combine Strategy for Data Analysis. *Journal of Statistical Software, 40*(1), 29.

Yu, C., Ward, M., Morabito, M., & Ding, W. (2011). Crime Forecasting Using Data Mining Techniques. *11th International Conference on Data Mining Workshops* (pp. 779-768). Vancuver, Canada: IEEE.

Yue, P., & Jiang, L. (2014). BigGIS: How big data can change shape next-generation GIS. *The Third International Conference on Agro-GeoInformatics* (pp. 1-6). Beijing, China: IEEE.

Zhang, M., Yue, P., & Guo, X. (2014). Towards an interoperable geospatial scripting langauge for GIS programming. *The Third International Conference on Agro-Geoinformatics* (pp. 1-5). Beijing, China: IEEE.

Zhu, Y., & Xiong, Y. (2015). Towards Data Science. *Data Science Journal , 14*(8), 1-7.

ABBREVIATIONS

BD – Big Data

GIS – Geographic Information System

CRAN – Comprehensive R Archive Network

IT – Information Technology

SFPD – San Francisco Police Department

SFPS – San Francisco Police

APPENDIX 1

Python stand-alone script (developed with the help from package online tutorials)

```
# -*- coding: utf-8 -*-
""" Created on Thu Jul 27 19:24:28 2017   @author: maka  """

##Import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import string

##1.  Load the SF_Crimes dataset
sf=pd.read_csv("C:/Users/user/Desktop/San_Francisco_Data/SF_AllCrimes.csv",
parse_dates=['Dates'])

##2. Examine the dataset
print(list(sf.columns.values)) #print the column name
print(sf.dtypes)              #print the data type of the columns
sf.shape                  #check the dimension of the dataset
sf.head(5)                #print first 5 rows and all columns to see the how the data looks like
sf.tail(5)
sf.isnull().values.any()      #check if there are any missing values within the data frame

##3. Check descriptive statistics of all columns
sf.describe(include="all")


##4. Check and remove duplicates from the dataset
sf_remove_duplicates=sf.drop_duplicates()
sf_remove_duplicates.shape

##5. Taking care of dates
sf['Date'] = pd.to_datetime(sf['Dates'].dt.date)
sf['Year'] = sf['Dates'].dt.year
sf['Month'] = sf['Dates'].dt.month
sf['Day'] = sf['Dates'].dt.day
sf['DayOfYear'] = sf['Dates'].dt.dayofyear
sf['Time'] = sf['Dates'].dt.hour + sf['Dates'].dt.minute/60
sf['Hour'] = sf['Dates'].dt.hour
sf['Minutes'] = sf['Dates'].dt.minute
del sf['Dates']
print(list(sf.columns.values))

# 6. Add column with named days of the week as number:

sf.loc[sf['DayOfWeek'] == 'Monday', 'DOW'] = 1
sf.loc[sf['DayOfWeek'] == 'Tuesday', 'DOW'] = 2
sf.loc[sf['DayOfWeek'] == 'Wednesday', 'DOW'] = 3
sf.loc[sf['DayOfWeek'] == 'Thursday', 'DOW'] = 4
sf.loc[sf['DayOfWeek'] == 'Friday', 'DOW'] = 5
sf.loc[sf['DayOfWeek'] == 'Saturday', 'DOW'] = 6
sf.loc[sf['DayOfWeek'] == 'Sunday', 'DOW'] = 7


# replace the name of X and Y column additionaly with the words longitude/Latitude
sf.rename(columns={'ID':'OBJECTID','X': 'Longitude', 'Y': 'Latitude'}, inplace=True)

# 7. Check all crime categories, years of occurance and districts where these were reported
Ncat=len(pd.unique(sf.Category.ravel()))
print ("All "+ str(Ncat)+ " categories:")
print (pd.unique(sf.Category.ravel()))

Nyears=len(pd.unique(sf.Year.ravel()))
print ("All "+str(Nyears)+" years:")
print (pd.unique(sf.Year.ravel()))

Ndist=len(pd.unique(sf.PdDistrict.ravel()))
print ("All "+str(Ndist)+" districts:")
print (pd.unique(sf.PdDistrict.ravel()))

## 8. Overview of occurrence of different crimes.
crime_rate=sf['Category'].value_counts()
y_pos = np.arange(len(crime_rate[0:39].keys()))

plt.figure(figsize=(10,15))
plt.barh(y_pos,crime_rate[0:39].get_values(),  align='center', alpha=0.4, color = 'black')
plt.barh(y_pos, crime_rate[0:39].get_values(),  align='center', alpha=0.4, color = 'black')
plt.yticks(y_pos, map(lambda x:x.title(), crime_rate[0:39].keys()), fontsize = 12)
plt.xlabel('Overview of the crime occurance', fontsize=13)
plt.title('San Francisco Crimes', fontsize = 20)
plt.ticklabel_format(style='sci', axis='x', scilimits=(0,0))
plt.tight_layout()
plt.show()

## 9. Where do crimes occur?
column='PdDistrict'
by_col = sf.groupby(column)
cf = by_col.size()
cf.index = cf.index.map(string.capwords)
cf.sort_values(ascending=True, inplace=True)
Ntot=sum(cf)

cf.plot(kind='barh',
        fontsize=12,
        figsize=(10,10),
        stacked=False,
        width=0.5,
        color='black',
        alpha= 0.5)
plt.title('Number of crimes per District',size=15)

## 10. When do crimes occur?
column='Year'
by_col = sf.groupby(column)
cf = by_col.size()
plt.figure(figsize=(10,5))
plt.plot(cf.index[:Nyears-1],cf[:Nyears-1], 'ks-')
plt.xlabel('Year')
plt.ylabel('Number of crimes')
column='Month'
by_col = sf.groupby(column)
cf = by_col.size()
plt.figure(figsize=(10,5))
plt.plot(cf.index,cf, 'ks-')
plt.xlabel('Month')
plt.ylabel('Number of crimes')

column='DOW'
by_col = sf.groupby(column)
cf = by_col.size()
plt.figure(figsize=(10,5))
plt.plot(cf.index,cf, 'ks-')
plt.xlabel('DayOfWeek')
plt.ylabel('Number of crimes')

column='Day'
by_col = sf.groupby(column)
cf = by_col.size()
plt.figure(figsize=(10,5))
plt.plot(cf.index,cf, 'ks-')
plt.xlabel('Day of month')
plt.ylabel('Number of crimes')

column='Hour'
by_col = sf.groupby(column)
cf = by_col.size()
plt.figure(figsize=(10,5))
plt.plot(cf.index,cf, 'ks-')
plt.xlabel('Hour')
plt.ylabel('Number of crimes')

# Relative frequency of Top Crimes per weekday
top_crime=("LARCENY/THEFT","ASSAULT","DRUG/NARCOTIC","VEHICLE THEFT",
    "VANDALISM","WARRANTS","BURGLARY","ROBBERY","FRAUD")
plt.figure(figsize=(10,5))
ax=plt.subplot()
for i in top_crime[0:6]:
    sf['event']=1
    tt=sf[sf.Category==i]
    dist_events = tt[['DOW','event']].groupby(['DOW']).count().reset_index()
    dist_events.event = dist_events.event/dist_events.event.sum()
    plt.plot(dist_events.DOW,dist_events.event,'o-',label=i)
    plt.legend(loc=3)
plt.ylabel("Relative number of crimes",size=20)
plt.xlabel("Weekday",size=20)
ax.set_xticklabels(('','Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'))
plt.show()

# Top crime occurrence on a daily level
larceny = sf[sf['Category'] == "LARCENY/THEFT"]
assault = sf[sf['Category'] == "ASSAULT"]
drug = sf[sf['Category'] == "DRUG/NARCOTIC"]
vehicle = sf[sf['Category'] == "VEHICLE THEFT"]
vandalism = sf[sf['Category'] == "VANDALISM"]
warrants = sf[sf['Category'] == "WARRANTS"]
burglary = sf[sf['Category'] == "BURGLARY"]
robbery = sf[sf['Category'] == "ROBBERY"]
fraud = sf[sf['Category'] == "FRAUD"]

with plt.style.context('bmh'):
    fig=plt.figure(figsize=(15,15))
    ax2 = plt.subplot2grid((4, 3), (1, 0))
    ax2.plot(larceny.groupby('Hour').size(), 'o-', color='black')
    ax2.set_title('Larceny/Theft')
    ax2.set_ylabel('Number of records')

    ax3 = plt.subplot2grid((4, 3), (1, 1))
    ax3.plot(assault.groupby('Hour').size(), 'o-', color='black')
    ax3.set_title('Assault')

    ax4 = plt.subplot2grid((4, 3), (1, 2))
    ax4.plot(drug.groupby('Hour').size(), 'o-', color='black')
    ax4.set_title('Drug/Narcotic')

    ax5 = plt.subplot2grid((4, 3), (2, 0))
    ax5.plot(vehicle.groupby('Hour').size(), 'o-', color='black')
    ax5.set_title('Vehicle')
    ax5.set_ylabel('Number of records')

    ax6 = plt.subplot2grid((4, 3), (2, 1))
    ax6.plot(vandalism.groupby('Hour').size(), 'o-', color='black')
    ax6.set_title('Vandalism')

    ax7 = plt.subplot2grid((4, 3), (2, 2))
    ax7.plot(warrants.groupby('Hour').size(), 'o-', color='black')
    ax7.set_title('Warrants')

    ax8 = plt.subplot2grid((4, 3), (3, 0))
    ax8.plot(burglary.groupby('Hour').size(), 'o-', color='black')
    ax8.set_title('Burglary')
    ax8.set_ylabel('Number of records')
    ax8.set_xlabel('Hours')

    ax9 = plt.subplot2grid((4, 3), (3, 1))
    ax9.plot(robbery.groupby('Hour').size(), 'o-', color='black')
    ax9.set_title('Robbery')
    ax9.set_xlabel('Hours')

    ax10 = plt.subplot2grid((4, 3), (3, 2))
    ax10.plot(fraud.groupby('Hour').size(), 'o-', color='black')
    ax10.set_title('Fraud')
    ax10.set_xlabel('Hours')
plt.tight_layout()
```

APPENDIX 2

R stand-alone script (developed with the help from package online tutorials)

```
library(data.table)
library(ggplot2)
library(lubridate)
library(plotly)
library(dplyr)
library(ggmap)

#Read the 2013 data
path <- "/Users/MajaKalinic/Desktop/MasterThesis/San_Francisco_Data/SFCrimes_2013.csv"
sf <- fread(path)

#Investigate basic info about the dataset
sf %>% head(10)
sprintf("# of Rows in Dataframe: %s", nrow(sf))
sprintf("Dataframe Size: %s", format(object.size(sf), units = "MB"))

#Check the basics about the data structure and summaries
str(sf)
summary(sf)
table(sf$Category)

#Investigate crimes on the steet level
sf[, .N, by = Address][order(N, decreasing = T)][1:10]
#By Descript
descript= sf[, .N, by = Descript][order(N, decreasing = T)]
#By Resolution
resolution= sf[, .N, by = Resolution][order(N, decreasing = T)]

#Investigate total crime count in 2013 per each category
a=sf[, .N, by = Category][order(N, decreasing = T)]
ggplot(a, aes(x = reorder(Category, -N), y = N)) +
  geom_bar(stat = 'identity') +
  coord_flip() +
  labs(x = '', y = 'Total Number of Crimes in 2013', title = 'Total Count of Crimes per Category')

#Print the total crimes in 2013 by Reesolution type
# grepl() is the best way to do in-text search
sf_arrest <- sf %>% filter(grepl("ARREST", Resolution))
sf_arrest %>% head(10)
sprintf("# of Rows in Dataframe: %s", nrow(sf_arrest))
sprintf("Dataframe Size: %s", format(object.size(sf_arrest), units = "MB"))

c = sf[, .N, by = Resolution][order(N, decreasing = T)]

ggplot(c, aes(x = reorder(Resolution, -N), y = N)) +
  geom_bar(stat = 'identity') +
  coord_flip() +
  guides(fill = F) +
  labs(x = '', y = 'Total Number of Crimes', title = 'Total Number of Crimes per Resolution Type')

#Basic statistics about arrests
#Daily arrests
sf_arrest_daily <- sf_arrest %>%
  group_by(Date) %>%
  summarize(count = n()) %>%
  arrange(Date)

sf_arrest_daily %>% head(10)

summary(sf_arrest_daily)

#Aggregate counts of arrests by Day-of-Week and Time
sf_arrest_time <- sf_arrest %>%
  group_by(DayOfWeek, Hour) %>%
  summarize(count = n())

sf_arrest_time %>% head(10)

ggplot(sf_arrest_time, aes(x = Hour, y = DayOfWeek, fill = count)) +
  geom_tile() +
  labs(x = "Hour of Arrest (Local Time)", y = "Day of Week of Arrest", title = " ") +
  #scale_fill_gradient(low = "white", high = "#27AE60")
  scale_fill_distiller(palette = 'BrBG')

#Further step will be to investigate how variables interect with each other
#1. How crimes are resolved (without Resolution=None) per each district

d = sf[, .N, by =.(PdDistrict, Resolution)][order(N, decreasing = T)][i= !Resolution %in% c('NONE')]
ggplot(d, aes(x = PdDistrict, y = Resolution, fill = N)) +
  geom_tile() +
  scale_fill_distiller(palette = 'BrBG') +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1))

#2. How crimes are related with the day of the week
e = sf[, .N, by =.(PdDistrict, DayOfWeek)][,Percentage:=N/sum(N)*100]
g<- ggplot(e, aes(x = DayOfWeek, y = PdDistrict, fill = Percentage)) +
  geom_tile() +
  scale_fill_distiller(palette = 'BrBG') +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank())

#3. How crime rates are distributed on a daily basis depending on the district
f = sf[, .N, by =.(PdDistrict, DayOfWeek)][,Percentage:=N/sum(N)*100, by = PdDistrict]
m<-ggplot(f, aes(x = DayOfWeek, y = PdDistrict, fill = Percentage)) +
  geom_tile() +
  geom_vline(xintercept=seq(0.5, 10, by=1), size = 1, color = 'grey') +
  scale_fill_distiller(palette = 'BrBG') +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank())

#4. Relation between crime types and day of the week
g = sf[, .N, by =.(Category, DayOfWeek)][,Percentage:=N/sum(N)*100, by = DayOfWeek][order(N, decreasing = T)]
ggplot(g, aes(x = DayOfWeek, y = Category, fill = Percentage)) +
  geom_tile() +
  geom_vline(xintercept=seq(0.5, 10, by=1), size = 1, color = 'grey') +
  scale_fill_distiller(palette = 'BrBG') +
  theme(axis.title.y = element_blank(),
        axis.title.x = element_blank())

#5. Lets check the change in crime categories over the whole year
incidents_by_monthofyear <- aggregate(sf$Category, by = list(sf$Category, sf$Month), FUN = length)
names(incidents_by_monthofyear) <- c("Crime_Category", "Incident_Month", "Count")

ggplot(incidents_by_monthofyear, aes(x= factor(Incident_Month), y= Crime_Category)) +
  geom_tile(aes(fill= Count)) + scale_x_discrete('', expand = c(0,0)) +
  geom_hline(yintercept=seq(0.5, 40, by=1), size = 1, color = 'grey')+
  scale_y_discrete("Month of Year", expand = c(0,-2)) +
  scale_fill_distiller(palette = 'BrBG')+
  theme_bw() + ggtitle("Crimes by months of year 2013") +
  theme(panel.grid.major = element_line(colour = NA), panel.grid.minor = element_line
        (colour = NA))

# Larency tends to show as the most prominent crime during te whole week
#Lets check this trend depending on the district

l = sf[i = Category == 'LARCENY/THEFT',j = .N, by = .(DayOfWeek, PdDistrict)]
ggplot(l, aes(y = PdDistrict, x = DayOfWeek)) +

  geom_point(aes(size = N, col = N))+
  scale_size(range=c(1,10)) +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())
#ggplotly(g, tooltip = c('x','y','colour'))

#Lets play with maps now
callmap = get_map(location = c(lon = -122.449067, lat = 37.746330), zoom = 12, maptype = 'terrain')
map = ggmap(callmap)
map

# Lets plot most prominent crimes per districts
BinnedCounts = sf[, .(N), by = .(Long = round(Longitude,2), Lat = round(Latitude,2))][order(N, decreasing = T)]
str(BinnedCounts)
map +
  geom_point(data = BinnedCounts, aes(x = Long, y = Lat, color = N, size=N)) +
  scale_colour_gradient(name = '# Total Crime', low="yellow", high="red") +
  scale_size(name = '# Total Crime', range = c(2,6))

#or some denisty map
map +
  stat_density2d( data = sample_frac(sf, 0.2), aes(x = Longitude, y = Latitude, fill = ..level.., alpha = ..level..), size = 1, bins = 50, geom = 'polygon') +
  scale_fill_gradient('Crime\nDensity', low = 'yellow', high = 'red') +
  scale_alpha(range = c(.2, .3), guide = FALSE) +
  guides(fill = guide_colorbar(barwidth = 1.5, barheight = 10))

#let's analzye the theft only by plotting it on the map for Friday and Saturday.
bbox = data.table(left = -122.5164, bottom = 37.7066, right = -122.3554, top = 37.8103)

x_length = abs(bbox[, left - right])/30
y_length = abs(bbox[, bottom - top])/30
sf[, LatBinned := round(Latitude/y_length)*y_length]
sf[, LongBinned := round(Longitude/x_length)*x_length]

f = sf[i = Category %in% c('LARCENY/THEFT'), .N, keyby = .(DayOfWeek, LatBinned, LongBinned)][
  order(LatBinned, LongBinned, N, decreasing = T)]
f1 = f[i = DayOfWeek == 'Friday']
f2 = f[i = DayOfWeek == 'Saturday']
map +
  geom_tile(data = f1, aes(x = LongBinned, y = LatBinned, alpha = N), fill = 'red') +
  ggtitle('Theft Crime Friday')

# and Saturday
map +
  geom_tile(data = f2, aes(x = LongBinned, y = LatBinned, alpha = N), fill = 'red') +
  ggtitle('Theft Crime Saturday')
```

APPENDIX 3

R-ArcGIS script [ArcGIS, (2017, Feb 8), Analyze Crime Using Statistics and the R ArcGIS Bridge, Video file, Retrieved from https://www.youtube.com/watch?v=ZGj-XX3GDP8&t=79s]

```
##Install and load all necessary packages

library(arcgisbinding)
arc.check_product()
install.packages("sp")
library(sp)
install.packages("spdep")
library(spdep)
install.packages("reshape2, dependencies = TRUE")
library (reshape2)
install.packages("ggplot2")
library (ggplot2)
install.packages("ggmap")
library (ggmap)

##1. Using arc.open() function load the data from ArcGIS Pro to a RStudio(load SF_Crimes_Enrich_Subset layer)
enrich_df <- arc.open(path = 'C:/Users/user/Desktop/San_Francisco_Data/SF_Crimes_Enrich_Subset.shp')

##2. Select the attributes from the enrich_df object for using in further analysis
enrich_select_df <- arc.select(object = enrich_df, fields = c('FID', 'SUM_VALUE', 'TOTPOP_CY', 'MEDVAL_CY', 'MEDHINC_CY', 'RENTER_FY', 'FEMALES_CY', 'MALES_CY', 'N01_BUS'))

##3. Using the arc.data2sp() function convert R data frame into a spatial data frame object
enrich_spdf <- arc.data2sp(enrich_select_df)

##4. Replace the attribute labels from the data with more human readable one
col_names <- c("OBJECTID", "Crime_Counts",
               "Population", "Med_HomeValue",
               "Med_HouseIncome", "Renter_Count", "Fem_Pop",
               "Male_Pop", "Business")

##5. Assign the new attribute labels with the help of colnames() function
colnames(enrich_spdf@data) <- col_names

##6. Check how does it look like now
head(enrich_spdf@data)

##7. Preform Empirical Bayes smoothing on the data to calculate crime rates for each hexagon bin. This is done using
##   the function EBest(), contained in spdep package. The following lines of code are doing calculation.

 n <- enrich_spdf@data$Crime_Counts
 x <- enrich_spdf@data$Population
 EB <- EBest (n, x)
 p <- EB$raw
 b <- attr(EB, "parameters")$b
 a <- attr(EB, "parameters")$a
 v <- a + (b/x)
 v[v < 0] <- b/x
 z <- (p - b)/sqrt(v)

##8. Create a new column within the spatial polygon data frame that contains the crime rate values calculated above
 enrich_spdf@data$EB_Rate <- z

##9. Convert the R spatial data frame object into the ArcGIS file type, to put back the results into ArcGIS Pro
 arcgis_df <- arc.sp2data(enrich_spdf)

##10. Write converted data frame object into ArcGIS project
arc.write('C:/Users/user/Desktop/San_Francisco_Data/SF_Crimes_Rates', arcgis_df, shape_info = arc.shapeinfo(enrich_df))

##11. Next step will be to investigate which of the chosen attributes may be influencing crime occurance in San Francisco.
##    One of the tools offered in R, for something like this is correlation matrix. Here, the grid of values shows
##    a level of association between added attributes and previously smoothed crime rates.

##11.a) Call back San Francisco crime rates shape file into the RStudio
rate_df <- arc.open('C:/Users/user/Desktop/San_Francisco_Data/SF_Crimes_Rates.shp')

##11.b) Select the atributes which will apear in correlation matrix
rate_select_df <- arc.select(rate_df, fields = c("OBJECTID_1", "Crime_Coun", "Population", "Med_HomeVa", "Med_Housei","Renter_Cou", "Fem_Pop", "Male_Pop", "Business", "EB_Rate"))

##11.c) Convert feature calss into a spatal data frame object
rate_spdf <- arc.data2sp(rate_select_df)

##11. d) Get lower triangle of the correlation matrix
get_lower_tri<-function(cormat) {
  cormat[upper.tri(cormat)] <- NA
  return(cormat)
}
## Get upper triangle of the correlation matrix
get_upper_tri <- function(cormat) {
  cormat[lower.tri(cormat)] <- NA
  return(cormat)
}
## Reorder correlation coefficients
reorder_cormat <- function(cormat) {
  # Use correlation between variables as distance
  dd <- as.dist((1-cormat) / 2)
  hc <- hclust(dd)
  cormat <- cormat [hc$order, hc$order]
}
## The following code creates the Correlation Matrix
corr_sub <- rate_spdf@data [ c ("Med_HomeVa", "Med_Housei","Renter_Cou", "Fem_Pop", "Male_Pop", "Business", "EB_Rate")]
cormax <- round (cor(corr_sub), 2)
upper_tri <- get_upper_tri (cormax)
melted_cormax <- melt (upper_tri, na.rm = TRUE)
cormax <- reorder_cormat (cormax)
upper_tri <- get_upper_tri (cormax)
melted_cormax <- melt (upper_tri, na.rm = TRUE)

## Visualising Correlation matrix, using corrplot
res<- cor(corr_sub)
install.packages("corrplot")
library(corrplot)
corrplot(res, type= "upper", order="hclust",
         tl.col= "black", tl.srt = 45)
```

APPENDIX 4

Prediction model in Python (developed with help of https://www.kaggle.com/rhoslug/sf-crime-prediction-with-scikit-learn?scriptVersionId=23855/code)

```
# -*- coding: utf-8 -*-
"""
Created on Sun Aug 13 18:59:05 2017

@author: maka
"""
##SF Crime Prediction with scikit-learn

from __future__ import print_function, division
import numpy as np

import pandas as pd

from patsy import dmatrices, dmatrix
from sklearn.calibration import CalibratedClassifierCV
from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier


    # Read in the training data
    df_test = pd.read_csv('C:/Users/user/Desktop/San_Francisco_Data/SF_prediction/SFCrimes_2013.csv', parse_dates=['Date'])

    # Delete unnecessary columns from the traing data and reename columns

    df_test.drop(df_test.columns[[0,2,3,6,7,10,11,12,13,14,15,16,17,18,19]], axis=1, inplace=True)

    df_test=df_test.rename(index=str, columns={"OBJECTID": "ID","Longitude": "X", "Latitude": "Y"})
    print(list(df_test.columns.values))
    df_test.head(5)

    # Read in testing data and perform the same as with train data, but keep necessary columns

    df_train = pd.read_csv('C:/Users/user/Desktop/San_Francisco_Data/SF_prediction/SFCrimes_2013.csv', parse_dates=['Date'])
    df_train.drop(df_train.columns[[0,1,7,10,11,12,13,14,15,16,17,18,19]], axis=1, inplace=True)
    df_train= df_train.rename(index=str, columns={"Longitude": "X", "Latitude": "Y"})
    print(list(df_train.columns.values))
    df_train.head(5)

    # Select training and validation sets
    inds = np.arange(df_train.shape[0])
    np.random.shuffle(inds)
    train_inds = inds[:int(0.2 * df_train.shape[0])]
    val_inds = inds[int(0.2 * df_train.shape[0]):]

    # Extract the column names
    col_names = np.sort(df_train['Category'].unique())

    # Recode categories to numerical
    df_train['Category'] = pd.Categorical(df_train['Category']).codes
    df_train['DayOfWeek'] = pd.Categorical(df_train['DayOfWeek']).codes
    df_train['PdDistrict'] = pd.Categorical(df_train['PdDistrict']).codes
    df_test['DayOfWeek'] = pd.Categorical(df_test['DayOfWeek']).codes
    df_test['PdDistrict'] = pd.Categorical(df_test['PdDistrict']).codes

    # Split up the training and validation sets

    df_val = df_train.iloc[val_inds]
    df_train = df_train.iloc[train_inds]

    # Construct the design matrix and response vector for the
    # training data and the design matrix for the test data
    y_train, X_train = dmatrices('Category ~ X + Y + DayOfWeek + PdDistrict', df_train)
    y_val, X_val = dmatrices('Category ~ X + Y + DayOfWeek + PdDistrict', df_val)
    X_test = dmatrix('X + Y + DayOfWeek + PdDistrict', df_test)

    # Create the logistic classifier and fit it
    logistic = LogisticRegression()
    logistic.fit(X_train, y_train.ravel())
    print('Mean accuracy (Logistic): {:.4f}'.format(logistic.score(X_val, y_val.ravel())))

    # Make predictions
    #predict_probs = logistic.predict_proba(X_test)

    # Create AdaBoost logistic classifier and fit
    adaboosttree = AdaBoostClassifier(DecisionTreeClassifier(max_depth=2))
    adaboosttree.fit(X_train, y_train.ravel())
    print('Mean accuracy (AdaBoost Logistic): {:.4f}'.format(adaboosttree.score(X_val, y_val.ravel())))

    # Fit the random forest
    randforest = RandomForestClassifier(n_estimators=11)
    randforest.fit(X_train, y_train.ravel())
    print('Mean accuracy (Random Forest): {:.4f}'.format(randforest.score(X_val, y_val.ravel())))
    #
    # # Make predictions
    predict_probs = randforest.predict_proba(X_test)

    # Add the id numbers for the incidents and construct the final df
    df_pred = pd.DataFrame(data=predict_probs, columns=col_names)
    df_pred['ID'] = df_test['ID'].astype(int)
    df_pred.to_csv('C:/Users/user/Desktop/San_Francisco_Data/SF_prediction/Prediction_output.csv', index=False)

    output = pd.read_csv('C:/Users/user/Desktop/San_Francisco_Data/output.csv')
```

APPENDIX 5

Within this thesis, several Python packages were used for data manipulation, visualization and building a predicative model:

<pandas[13]>

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal. *Pandas* is well suited for different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet.
- Ordered and unordered time series data.
- Arbitrary matrix data with row and column labels.
- Any other form of observational / statistical data sets.

Pandas is built on top of *numpy* and is intended to integrate well within a scientific computing environment with many other third party libraries. Here are just a few of the things that *pandas* does well:

- Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data.
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects.
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data in computations.
- Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data.
- Make it easy to convert ragged, differently-indexed data in other Python and *numpy* data structures into DataFrame objects.
- Intelligent label-based slicing, fancy indexing, and sub-setting of large data sets.
- Intuitive merging and joining data sets.
- Flexible reshaping and pivoting of data sets.
- Hierarchical labelling of axes (possible to have multiple labels per tick).
- Input/Output tools: loading tabular data from flat files (CSV, delimited, Excel 2003), and saving and loading *pandas* objects from the fast and efficient PyTables/HDF5 format
- Memory-efficient "sparse" versions of the standard data structures for storing data that is mostly missing or mostly constant (some fixed value).

[13] pandas online documentation, Accessed 12.08.2017.

- Moving window statistics (rolling mean, rolling standard deviation, etc.).
- Static and moving window.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: munging and cleaning data, analysing / modelling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. *Pandas* is the ideal tool for all of these tasks. *Pandas* is fast. However, as with anything else generalization usually sacrifices performance. *Pandas* is a dependency of *statsmodels*, making it an important part of the statistical computing ecosystem in Python.

<*numpy*[14]>

Numpy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Basic mathematical functions (such as trigonometric, hyperbolic, rounding, sums, products, differences, exponents and logarithms, arithmetic operations, handling complex numbers) operate elementwise on arrays, and are available both as operator overloads and as functions in the *numpy* module. Other than this, with *numpy*, it is possible to reshape or otherwise manipulate data in arrays such as copying values from one array to another, changing the shape of an array, transpose like operations, changing number of dimensions, changing the kind of an array, joining or splitting arrays, tiling arrays, adding and removing or rearranging elements.

<*matplotlib*[15]>

Matplotlib is probably the single most used Python package for 2D-graphics. It provides both a very quick way to visualize data from Python and publication-quality figures in many formats. *Matplotlib* comes with a set of default settings that allow customizing all kinds of properties. Manipulation can be done over defaults of almost every property in *matplotlib*: figure size and dpi, line width, colour and style, axes, axis and grid properties, text and font properties.

Within the figure there can be subplots. With subplots, plots can be arranged in a regular grid. The number of rows and columns and the number of the specific plot has to be specified. On the other side, the *gridspec* command is a more powerful alternative (compared to manual creation of subplots).

---

[14] numpy online documentation, Accessed 12.08.2017.
[15] (Hunter J. , 2007)

Matplotlib offers variety of plotting options, such as Regular plots, Scatter plots, Bar plots, Contour plots, Imshow, Pie Chart, Quiver plots, Grids, Multi plots, Polar Axis, 3D plots, Text plots.

<*scikit-learn*[16]>

Machine learning is about learning some properties of a data set and applying them to a new data. Common practice in machine learning, to evaluate an algorithm, is to split the data at hand into two sets, one that is called the training set on which data properties are examined and second one, called testing set, on which these properties are tested.

Learning problem considers a set of n samples of data and then tries to predict properties of unknown data. Learning problems can be separated into a few large categories:

- *supervised learning*, in which the data comes with additional attributes that we want to predict. This problem can be either:
  classification: samples belong to two or more classes and the task is to learn from already labeled data how to predict the class of unlabelled data,
  regression: if the desired output consists of one or more continuous variables, then the task is called regression.

- *unsupervised learning*, in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data, where it is called *clustering*, or to determine the distribution of data within the input space, known as *density estimation*, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization.

Supervised learning includes variety of models, among which Generalized Linear Model, Linear and Quadratic Discriminant Analysis, Kernel ridge regression, Support Vector Machines, Stochastic Gradient Descent, Nearest Neighbours, Gaussian Processes, Cross decomposition, Naïve Bayes, Decision Trees, Ensemble methods, Multiclass and multilabel algorithms, Neural network models.

Ensemble methods were used within this thesis to build a predication model. The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. AdaBoost Classifier (used within this thesis) fit a sequence of weak learners on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction. Second used classifier was a Random forest which is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive

[16] scikit-learn online documentation, Accessed 12.08.2017.

accuracy and control over-fitting. Third, Logical regression classifier, is a part of Generalised Linear Model, and represents generalized linear model using the same underlying formula, but instead of the continuous output, it is regressing for the probability of a categorical outcome. In other words, it deals with one outcome variable with two states of the variable - either 0 or 1.

R libraries used within this thesis were:

<*dplyr*[17]>

R package used to transform and summarize tabular data with rows and columns. The package itself contains a set of functions or "verbs" that preform common data manipulation operations such as filtering, selecting, re-ordering, adding new columns or summarizing data. Compared to traditional R functions, dplyr are easier to work with, more consistent in syntax and targeted for data analysis around data frames, not only vectors. Another interesting option that this package has to offer is the ability to import the *pipe operator %>%* (from another package) which will pipe the output from one function to the input of another function. That means that instead of function nesting, there is an option of reading the function from left to right.

<data.table[18]>

R provides a helpful data structure called the "data frame" that gives the user an intuitive way to organize, view, and access data. It works well with very large data files; it can behave just like a data frame; it offers fast subset, grouping, update, and joins; it makes it easy to turn an existing data frame into a data table. Within a data table the special variable .N represents the count of rows. If there is a group by index then it presents the number of rows within that grouping variable.

In general, data.table represents enhanced version of data.frame and it is being inherited from it. Aside mentioned functionalities, it offers option of sub-setting rows and columns; ordering variables in ascending/descending order; adding, updating, deleting a column or values in a data set; computing functions on variables based on grouping a column; sub-setting the data using keys.

<*lubridate*[19]>

R package that makes it easier to work with dates and times. Specifically, with *lubridate* it is possible to:

- identify and parse date-time data,
- extract and modify components of a date-time, such as years, months, days, hours, minutes, and seconds,
- perform accurate calculations with date-times and timespans
- handle time zones and daylight savings time.

---

[17] dplyr online documentation, Accessed 18.08.2017.
[18] data.table online documentation, Accessed 18.08.2017.
[19] (Grolemund & Wickham, 2011)

<ggplot2[20]>

R package for data visualisation. Compared to basic R graphics, ggplot2 is more verbose for simple (canned) graphics and less verbose for complex (custom) graphics, it does not have methods (data should always be in a data frame) and it uses a different system for adding plot elements. There are multiple advantages of ggplot2, such as:

- consistent underlying grammar of graphics,
- plot specification at a high level of abstraction,
- very flexible,
- theme system for polishing plot appearance,
- mature and complete graphics system,
- many users, active mailing list.

An example of creating a visualisation output with ggplot2 would include several steps: supply a dataset and aesthetic mapping, add on layers (such as geom_point or geom_histogram), scales, faceting specifications (such as facet_wrap) and coordinate systems.However, one should consider that this package is not suitable for 3D graphics, Graph-theory type graphs and Interactive graphics.

<ggmap[21]>

A collection of functions to visualize spatial data and models on top of static maps from various online sources. It includes tools common to those tasks, including functions for geolocation and routing. ggmap makes it easy to retrieve raster map tiles from popular online mapping services like Google Maps, OpenStreetMap, Stamen Maps, and plot them using the ggplot2 framework.

The basic idea driving ggmap is to take a downloaded map image, plot it as a context layer using ggplot2 and then plot additional content layers of data, statistics, or models on top of the map. ggmap has several utility functions which aid in spatial exploratory data analysis: geocode is a vectorized function which accepts character strings and returns a data frame of geographic information; revgeocode - in some instances it is useful to convert longitude/latitude coordinates into a physical address; mapdist - the ability to compute real distances in a spatial setting; route- the route function provides the map distances for the sequence of "legs" which constitute a route between two locations. Each leg has a beginning and ending longitude/latitude coordinate along with a distance and duration in the same units as reported by mapdist.

[20] ggplot2 online documentation, Accessed 18.08.2017.
[21] (Kahle & Wickham, 2013)

<*corrplot*[22]>

The *corrplot* package is a graphical display of a correlation matrix, confidence interval. It also contains some algorithms to do matrix reordering. In addition, corrplot is good at details, including choosing color, text labels, color labels, layout, etc. There are seven visualization methods in *corrplot* package, named circle, square, ellipse, number, shade, color, pie. There are three layout types named full (default), upper or lower, display full matrix, lower triangular or upper triangular matrix. Matrix reorder is very important for mining the hidden structure and pattern in the matrix. In *corrplot* there are four mathods to do that, named AOE, FPC, hclust, alphabet. AOE is for the angular order of the eigenvectors, FPC for the first principal component order, hclust for hierarchical clustering order, alphabet for alphabetical order. There is an option of specifying a colour spectrum, adding a colour legend and text legend.

[22] corrplot online documentation, Accessed 18.08.2017.

APPENDIX 6

Installation itself is straight-forward when certain pre-requirements are met:

- ArcGIS Pro 1.1 or later
- R Statistical Computing Software 3.1.0 or later
- 64-bit version for ArcGIS Pro

In the ArcGIS Pro Project environment, under the section Toolboxes, there is an option of Adding a new toolbox. After navigating to the location where the Python toolbox is stored, simple click on *Install R bindings* will do the work.

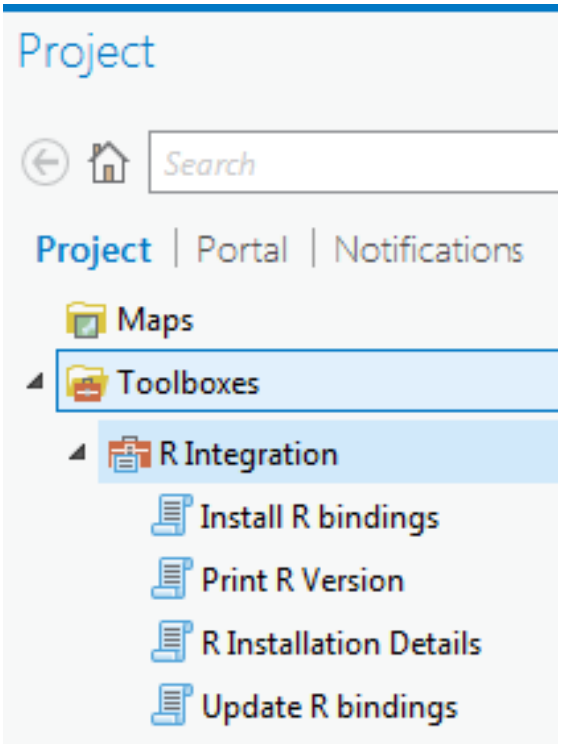

Figure 30: R-ArcGIS Bridge installation

This way, ArcGIS user can easily access R code through geoprocessing scripts, while R users can access organizations GIS' data, managed in traditional GIS ways. In other words, data stored in ArcGIS can be used in R and R objects can be sent back to ArcGIS in a native data types. Further considerations on spatial data conversion will be explained in more details in the workflow.

Direct link to installation steps and download: https://r-arcgis.github.io

APPENDIX 7

Space Time Cube[23]

The tool takes timestamped features and structure them into a netCDF data cube by generating space-time bins with either aggregated incident points or defined features with associated spatiotemporal attributes. If the timestamped point features need to be agregated spatially to understand spatiotemporal patterns at locations throughout the study area, than Create Space Time Cube By Aggregating Points tool should be used. This will result in either a grid cube (fishnet or hexagon) or a cube structured by the defined locations provided as aggregation polygons.

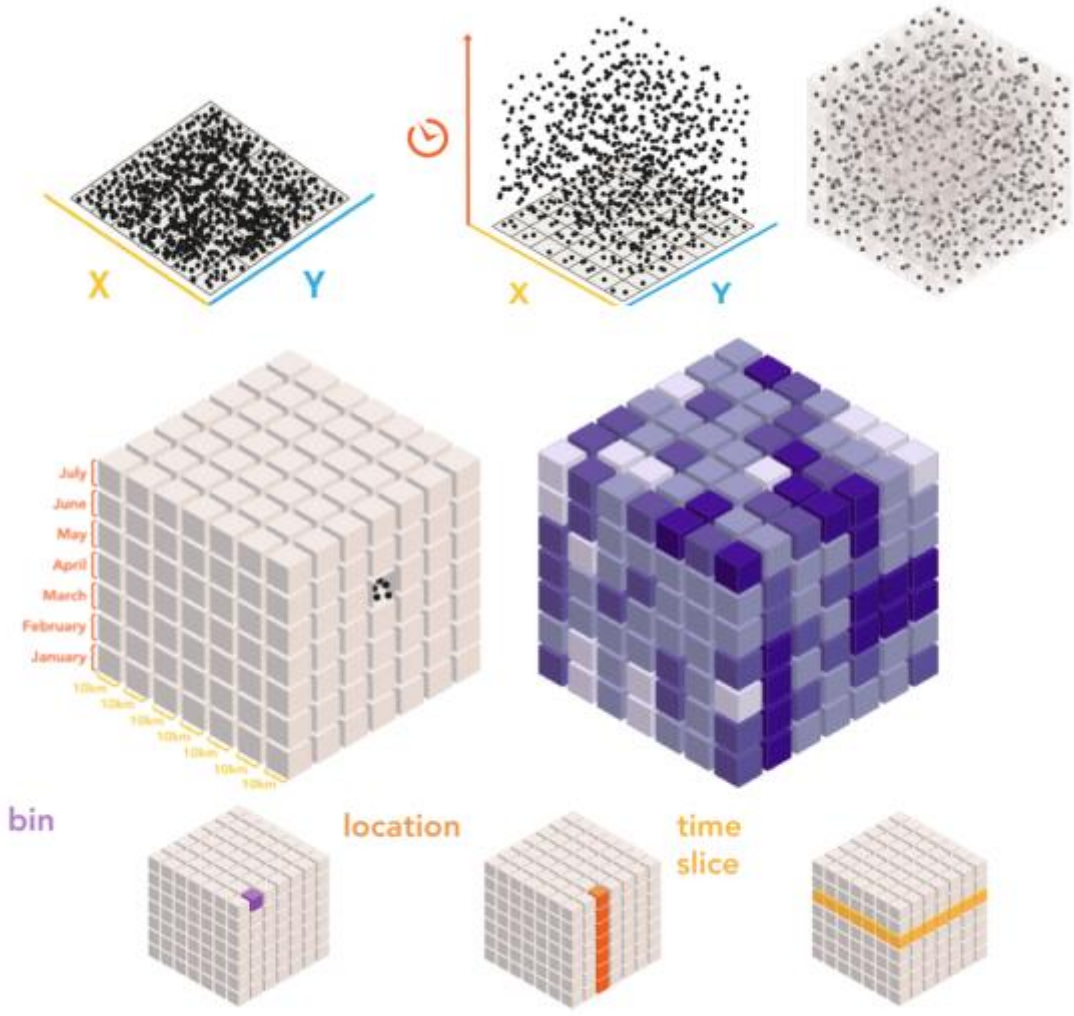

Figure 31: Creating a Space Time Cube

Within each bin of the cube, the points are counted, any Summary Field statistics are calculated, and the trend for bin values across time at each location is

[23] ArcGIS Pro Tool Reference - Space Time Cube, Accessed 01.09.2017.

measured using the Mann-Kendall statistic. The Mann-Kendall trend test is performed on every location with data as an independent bin time-series test. The Mann-Kendall statistic is a rank correlation analysis for the bin count or value and their time sequence. The bin value for the first time period is compared to the bin value for the second. If the first is smaller than the second, the result is a +1. If the first is larger than the second, the result is -1. If the two values are tied, the result is zero. The result for each pair of time periods compared are summed. The expected sum is zero, indicating no trend in the values over time. Based on the variance for the values in the bin time series, the number of ties, and the number of time periods, the observed sum is compared to the expected sum (zero) to determine if the difference is statistically significant or not. The trend for each bin time series is recorded as a z-score and a p-value. A small p-value indicates the trend is statistically significant. The sign associated with the z-score determines if the trend is an increase in bin values (positive z-score) or a decrease in bin values (negative z-score). Creating a space time cube by aggregating points is most common when the point data represents incidents, such as crimes, and there is a need to aggregate those incidents into either a grid or a set of polygons representing police beats.

What is considered important for getting valubale results is setting up the structure of the cube, both spatial and temporal as well as time step alignment. Finally, interpreting results can be done either through message which is delivered together with the tool execution or through 2D/3D Space Time Cube visualisation. The result needs to show the information about Overall Data Trend which is based on an aspatial time-series analysis. The question it answers is, overall, are the events represented by the input increasing or decreasing over time?

Emerging Hot Spot Analysis[24]

An emerging hot spot analysis is conducted to investigate trends over space in addition to trends over time. Clustering patterns of point densities or summary fields across the study area are analysed.

The Emerging Hot Spot Analysis tool takes the space time cube as an input and conducts a hot spot analysis using the Getis-Ord Gi* statistic for each individual bin. The Gi* statistic returned for each feature in the dataset is a z-score. For statistically significant positive z-scores, the larger the z-score is, the more intense the clustering of high values (hot spot). For statistically significant negative z-scores, the smaller the z-score is, the more intense the clustering of low values (cold spot).

[24] ArcGIS Pro Tool Reference – Emerging Hot Spot Analysis, Accessed 01.09.2017.

The Neighbourhood Distance and Neighbourhood Time Step parameters define how many surrounding bins, in both space and time, will be considered when calculating the statistic for a specific bin. Then, the hot and cold spot trends detected by the Getis-Ord Gi* hot spot analysis are evaluated with the Mann-Kendall test to determine whether trends are persistent, increasing, or decreasing over time. The results are symbolized with seventeen different categories describing the statistical significance of hot or cold spots and the location's trend over time.

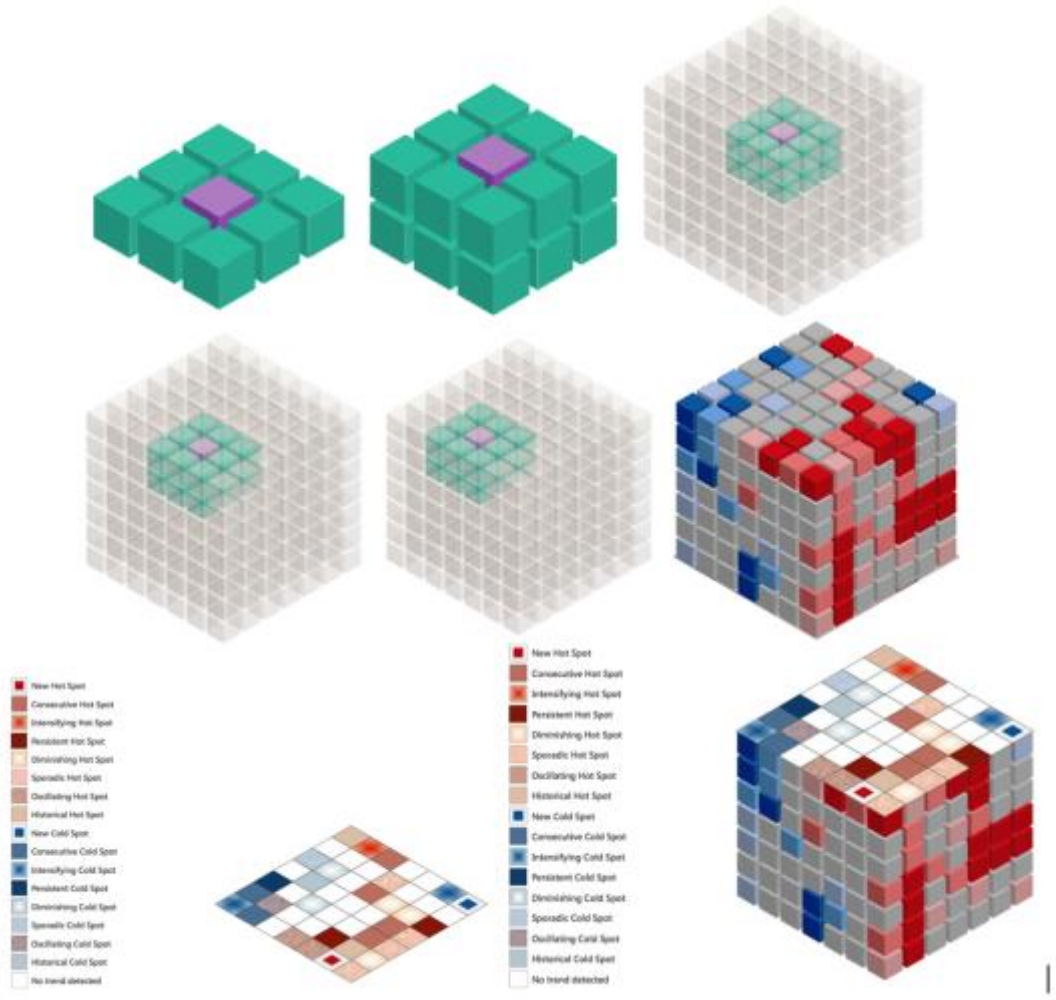

Figure 32: Emerging Hot Spot Analysis with default Legend

Hot Spot Analysis[25] and Optimized Hot Spot Analysis[26]

Given a set of weighted features, Hot Spot Analysis identifies statistically significant hot spots and cold spots using the Getis-Ord Gi* statistic. This tool identifies statistically significant spatial clusters of high values and low values. The z-scores (standard deviation) and p-values (probability) indicate whether the

[25] ArcGIS Pro Tool Reference - Hot Spot Analysis, Accessed 01.09.2017.
[26] ArcGIS Pro Tool Reference - Optimized Hot Spot Analysis, Accessed 01.09.2017.

observed spatial clustering of high or low values is more pronounced than one would expect in a random distribution of those same values.

Optimized Hot Spot Analysis executes the Hot Spot Analysis (Getis-Ord Gi*) tool using parameters derived from characteristics of input data. Similar to the way that the automatic setting on a digital camera will use lighting and subject versus ground readings to determine an appropriate aperture, shutter speed, and focus, the Optimized Hot Spot Analysis tool interrogates input data to obtain the settings that will yield optimal hot spot results. If, for example, the Input Features dataset contains incident point data, the tool will aggregate the incidents into weighted features. Using the distribution of the weighted features, the tool will identify an appropriate scale of analysis. The statistical significance reported in the Output Features will be automatically adjusted for multiple testing and spatial dependence using the False Discovery Rate (FDR) correction method.

Enrich Layer[27]

Enriches data by adding demographic and landscape facts about the people and places that surround or are inside the data locations. The output is a duplicate of an input with new attribute fields added to the table. This tool requires an ArcGIS Online organizational account and consumes credits.

When using Enrich Layer from the tool dialog box, the Country, Data Collection, and Variables parameters must be entered sequentially. Select a country to see data collections, select a data collection to see variables. Variables can be selected from multiple data collections by first selecting a set of variables from one collection; then select a different data collection and select another set of variables from the new collection.

This tool creates a new output feature class, which is a copy of the input with additional attributes added. The existing feature geometries and attribute fields and values will not be changed. Figure X shows the list of variables with which San Francisco dataset was enriched (Chapter IV, section 3.3.3)

Figure 33: Enrich Layer variable list

[27] ArcGIS Pro Tool Reference - Enrich Layer, Accessed 25.08.2017.