CATHOLIC UNIVERSITY OF BUKAVU

B.P.285 BUKAVU

FACULTY OF SCIENCES

Department of Computer Science

Development of a messaging application for communication and detection of spam on a mobile operator, case study of Airtel, Vodacom and Orange.

Presented by : **MURHULA BYABUSHI Christian**

Dissertation presented and defended in order to obtain the degree of Bachelor in Computer Science.

Option: Network and Telecommunications
Degree: Final year of Bachelor

Supervised by: ***Hw.*** **MUGISHO MUSHEGERHA Youen**
Directed by : ***PhD.*** **Elie ZIHINDULA**

Academic year: 2022-2023

Contents

List of Figures

List of Tables

Introduction

0.1 Context and generalities

With the increasing of use of mobile devices in mobile telecommunication, the number of text messages sent every day has grown exponentially. According to *Statista*, a company that provides market and consumer data on a wide range of topics, including digital media and technology; the number of mobile messages sent worldwide in 2020 reached 3.5 trillion [34]. In the same case, with the raise of web pages and social media messaging applications like Whatshap, Teelgra, Snapchat Facebook, Instagram and many others, phone users can now send messages that are only based on text as in former time but also on video, audios which are more chestful comparatively [14].

For sure, for interacting with his partner more professionally, email message is the most mean used, but not in all cases since in some countries a given SIM Card of a telecommunication provider is used as a bank more than being an communication mean, so the importance to secure the communication lines of a mobile users in these countries. Along with the functions and interests that the mobiles messages encompasses in terms of conversing, money sending and receiving, there has been an increase in the number of spam messages that aim to deceive people into providing personal information, sending unwillingly money, menacing to death or taking other actions that benefit the scammer.

To address this problem, the development of a messaging application with advanced spam detection capabilities is crucial to set, filter messages and prevent the users. This dissertation focuses specifically on the development of such application for phone users and in general for mobile networks telecommunications technologies.

0.2 Problematic

In telecommunication domain, we use mobile devices or phones for sharing *SMS*, Email, chats by using some specific apps. Among all we use specifically *SMS* for personal and professional information sharing [24]. The SMS stands for Short Message Service, which is a text messaging service for mobile phones and other mobile devices. It allows users to send and receive short messages of up to 160 characters [25]. It is also possible to send or receive automatic SMS which are not sent by human, whereas by using web interface or API [20].

Time to time, more persons are receiving messages such as : "You won x amount of money send another amount to withdraw it", "Join me at x area to take your money but pay me the transport ...", "I'm *Sirene* Madam I have money for you", "I have a job for you" and many others fake messages. More of them are reported in this spams examples

link.

Furthermore, scammers go the point where they can introduce vulnerabilities in messages which exploit a weakness in the SMS messaging system to remotely install spyware on mobile devices commonly called *Simjacker* [9]. However, in marketing almost similar messages are used to sensitize people to by products and services which confuse users wether it is or not a spam message by leading users to ignore important messages or being more hesitant to engage with mobile marketing campaigns [8],[27].

Considering all above issues caused by spam, what are key technical challenges that could be addressed in messaging system which can effectively facilitate the communication as well as detecting and filtering spam messages ?

0.3 Hypotheses

According to the Oxford dictionary, hypotheses stands for a statement of the expected relationship between things being studied, which is intended to explain certain facts or observations [35]. An idea to be tested. Hence, using the content-based filtering techniques, which involves analyzing the content of messages and determining wether it is a spam or not would be considered as solution.

Firstly this would be done by utilizing the Machine learning algorithms which are : **Naive Bayes, Logistic Regression, and Supper vector Machines**. All these would be combined by the ensemble methods for making a more predictive model [38] including the preparation and classification techniques which avoid biased model [21].

Secondly, during the production steps, we should integrate the model inside of the system able to add technically in a blacklist or whitelist suspect users based on the specific probability of being a potential attack.

Overall, we will jump from Machine Learning as model (*MLaaM*) which is the output of writing ML algorithms run on data and represent what was learned by the algorithm on training data; to ML Model Software Deployment which encompasses all the activities that make a software system worthy to be used [17].

0.4 Delimitation and objectives

0.4.1 Delimitation

The present work aims to develop a messaging application for communication and detection of spam on a mobile network.

Geographically it focuses on all provinces of Democratic Republic Of Congo(DRC) where mobile phones are used and require techniques for implementation.

Besides, it does not function effectively across all languages unless the solution model has been specifically trained on those languages. As a result, it is challenging to claim its effectiveness in languages such as Swahili, Lingala, French, or even English. Moreover, achieving optimal performance often necessitates the involvement of a large population.

Indeed, the current work in terms of planning and execution has spanned a duration of nine months : From January to November 2023.

0.4.2 Objectives

This system present 2 types of objectives such as: functional and non functional.

As functional objectives, this system consist of developing a messaging application that can facilitate efficient communication between users; implementing the machine learning models which has the capacity of classifying the messages and preventing spam messages under a certain probability; designing and integrating a user-friendly interface for messaging application;

For non-functional objectives, it allows the user whose messaging application complies with relevant privacy laws and regulation to protect data information, reducing the attacks and frauds; Increasing the trustfulness of users and mobile services compagnie provider, optimizing the power resources of the user against threats posed by scammers.

0.5 Interest

Personally, this paper has allowed the author to gain knowledge and more experience in the field of mobile networks and messaging applications.

Socially, the developed system contributes to facilitating communication and reducing the impact of spam messages, which can be annoying and stressful for citizens.

Economically, this system of detection helps to save business server time and resources by filtering out spam messages allowing for more targeted marketing efforts.

Scientifically, the research achieved contributes to the advancement of the science in the domain of Mobile Networks, SMS messaging, and Machine Learning data processing and classification.

0.6 Research Methodology

Throughout this paper, the research methodology will be used to guide the study towards achieving its objectives. The research will adopt a descriptive research design to describe the development of a messaging application for communication and detection of spam on a mobile network. The study will focus on both qualitative and quantitative research methods [11]. The qualitative method will involve a literature review, interviews and analysis of collected messages. while quantitative method will focus on the development and testing of the messaging application.

The research will be conducted in two phases. The first phase will involve data collection through a survey questionnaire that can be completed on website, or can be directly provided to the web interface (API) by the mobile phone users for collecting their experience with messages and especially spams. Thus, the data collected will be analyzed using descriptive statistics [4] to identify the common types of spam messages and the frequency of occurrence, languages inside, and other attributes.

The second phase will involve the development of the messaging application using the data collected from the survey and the analysis of existing messaging applications. The development of the application will be guided by the principles of agile software development by using Python (Django framework) for *Back-end* and HTML,CSS and JavaScript for *front-end*. Then,the application will integrate the use of machine learning models based on selection's research of each other.

The evaluation of the messaging application interface will be conducted using both quantitative and qualitative methods. The qualitative evaluation will involve the measurement

of the application's accuracy and efficiency in detecting and filtering spam messages, on the other hand the qualitative evaluation will involve a user study to determine the usability and user experience of the application.

0.7 Work Plan (or Work Subdivision)

The work plan of this dissertation is divided into four parts. The first is the introduction, which provides a background information on the research problem. The second part consist of a situation analysis and assessment while the third part focuses on literature review and explanations on the methodology. Then the fourth part presents the practical result of this work. Finally the conclusion part summarizes the key findings and contributions of the study and presents limitations and provide recommendations for future research.

Chapter 1

Situation analysis and assessment on mobile phones

1.1 Introduction

In this chapter, we will focus on various aspects that enhance the comprehensiveness and practicality of this dissertation. It includes explanations of mobile messaging architecture, machine learning models, and spam messages in mobile world. Additionally, it provides an analysis of the architecture used by network operators, highlighting both positive and negative aspects of their approach to message handling.

1.2 Presentation of the working framework and definition of key concepts

1.2.1 Definition of key concepts

a) SMS(Short Message Service) :
The Short Message Service is a basic service allowing the exchange of short text messages between subscribers [26]. For supporting virtually all mobile devices, SMS is considered as a universal means of communication that enables users to communicate and function even though all users are not active simultaneously (asynchronous communication).

b) Enhanced Messaging Service (EMS) :
EMS has been created to allow the transmission of richer and more advanced messages. Unlike traditional SMS, EMS accepts not only text messages but also audios, melodies, and animations [25].

c) MMS (Multimedia Messaging Service):
MSS has been developed to facilitate the transmission of rich multimedia content in mobile messaging. Unlike SMS and EMS, MMS enables users to send not only text messages but also various types of multimedia files such as images, videos, audio recordings and even slideshows [25].

d) Spam message:
A spam message is understood as an unsolicited or undesired messages received on mobile phones which constitutes veritable nuisance to the mobile subscribers [39]. Clearly, this message can be sent with the intention of gaining financial benefits, collecting personal or organizational information such as security numbers, credit card details, or login credentials, and soliciting money by making false promises of future benefits or rewards that do not materialize.

e) Networks operators: The networks operators refers to companies or organizations that provide and manage telecommunication networks. These operators own and operate the infrastructure, such as mobile networks, fixed-line networks, or internet service provider (ISP) networks, that enable the transmission of user's information to another user of the network [15]

f) Artificial Intelligence (AI) : AI refers to the field of computer science that focuses on creating intelligent machines or systems that can perform tasks that would typically require human intelligence. For being practical, it encompasses algorithms, models and technologies that enable computers and machines to simulate human like cognitive processes such as learning, reasoning, problem-solving, perception and language understanding.

g) ML (Machine Learning) : Machine learning is a subfield of Artificial Intelligence that focus on the development of algorithms and models that enable computers to learn from data and make decisions or predictions without being explicitly programmed[43]. Clearly, when the data is labeled during the training, we refer to it as supervised model. If contrast, when the data is unlabeled and the model must discover patterns and relationships itself, it is an unsupervised model. Additionally, whenever it performs both the labeling and discovering patterns tasks, it refers to a semi-supervised model. Furthermore, there is the last type called reinforcement. This one, is used to teach a computer or an AI agent how to make series of decisions in an environment. Just like, we learn to play the game better by playing it over and over.

h) NLP (Natural Language Process): NLP is a subfield of Machine Learning that studies the human language and combing techniques from statistics, linguistics, life-hoods for making sentiment analysis, text classification, machine translation, question answering and text generation in a way that it can be understood computationally [7].

1.2.2 Presentation of the working framework

In the eastern party of DRC (South Kivu- and North Kivu) the usage of mobile phones has become more common, transforming communication and connectivity in the region. The DRC itself is a large country, covering over 2,345,000 square kilometers with the eastern provinces of North and South Kivu spanning approximately 59,483 and 65,070 square kilometers respectively [53]. According to recent statistics from *GlobalEdge* [1], an

[1]GlobalEdge : Created in 1994 by the International Business Center and the Eli Broad College of Business at Michigan State University (IBC), globalEDGE™ is a knowledge web-portal that connects international business professionals worldwide to a wealth of information, insights, and learning resources on global business activities

American company, around 95 million people were living in the DRC in 2022 [13], of which approximately 46.9% had active mobile phones based on *GSM* [2] research.

In this context, it is observed that more people in cities use mobile phones compared to those in villages, primarily due to limited accessibility. A research study conducted by *Target Canibet* [3]in 2015 focused on mobile connections in DRC cities including Bukavu, Goma, Kinshasa, Lubumbashi, and Matadi, found that among 1,000 people surveyed in each city, 9 out of 10 individuals were subscribed to a network operator. However, it was noted that approximately half of them subscribed to two operators, while a quarter subscribed to four operators, and 18% used the services of a single operator.

Furthermore, the recent statistics made by *DataReportal* [4] in DRC shows that the mobiles users continues to increase exponentially, merely because of new services provided by internet and Telecoms Operators, at the point that since 2021 to 2022, it is has been reported 3.6 million of new users between 2021 to 2022, a report that proves how much mobile phones is inevitable in this last decades.

1.2.3 Network coverage and infrastructure

In fact, two telecoms services exist in DRC such as : Fixed services (26%) and Mobile services (74%). The first one known as landline or wired services, involve the use of physical infrastructure; the second one which is popular is the mobiles services refer to telecommunications services provided through mobile networks. According to the Congolese Regulatory Agency (ARPTC), the DRC has four mobile operators - Vodacom RDC, Airtel Congo, Orange DRC and Africell DRC. Vodacom is the leader in the voice segment, with 35.2% of the market, followed by Orange (30%), Airtel (23.9%) and Africell (10.9%). In the mobile internet market, Vodacom has 37.44%, Airtel 31.25%, Orange 28.14% and Africell 3.17% [3].

Additionally, since the 190s, when the DRC witnessed the first installation of operator systems such as Celtel(now Airtel) and Vodacom, followed by Orange and Africell, the telecomunications sector has shown significant market growth, reaching 1 Billion in 2022$ and expanding at a rate of 21% per year according to *GlobalData* [5]. However, this growth necessitates the updating of the infrastructure, which includes various generations of technologies, namely the second generation, third, fourth, and fifth(under development).

In fact, the second generation have been deployed in various territories to enable more efficient **voice calls, data networks services, and introduce SMS for text messaging**. The infrastructure required for 2G networks includes the following equipments: 1) BTS(Base Transceiver Station) : Transmit and receives signals between mobile devices. 2) MSC(Mobile Switch Controller) : serves as the switching entity that connects calls

[2]GSMA (Global System Communications Association) : An industry Organization which represents the interests of mobile network operators worldwide created in 1982 to ease cooperation between countries deploying *GSM* (Global System fo Mobile) technology.

[3]Target Canibet: Reseach & Consulting Group working in DRC. `https://www.target-sarl.cd/fr/content/etude-sur-la-telephonie-mobile-en-rdc`

[4]DataReportal: A online Company designed to help people and organizations all over the world to find the data, insights, and trends they need to make better informed decisions produced by Simon Kemp, `https://datareportal.com/reports/digital-2023-global-overview-report`

[5]GlobalData: Expert Company of Analysis, innovatove Solutions

between mobile devices. 3) BSC (Base Station Controller) : manages multiples BTSs and controlling radio resources, managing handovers between cells and optimizing network performance, 4) AuC (Authentication Center) responsible for managing subscriber authentication and encryption keys to ensure communication between mobile devices and the network, 5) Home Location Register (HLR) the database that stores subscriber information such as phone numbers, authentication details, and service profiles, 6) Visitor Location Register (VLR) : The VlR is a temporary database that stores information about roaming subscribers within a specific area 7) MS (Mobile Station), including all the technologies used by the users's handset and has two parts : **Firstly, the mobile equipment which contains the radio equipment, the user interface, the processing capability and memory requirements for call signaling, encryption, SMS and the id of the mobile phone(equipment IMEI number). Secondly the Subscriber Identity module (SIM Card),** used in encryption of codes needed to identify the subscriber, storing subscriber's information, locate the user [12] as (+243 for each congolese number).
Indeed, all the 2G technologies covers a large distance varying between 1880MHz - 2700 MHz.

Besides, the third generation appears as revolution, **allowing multimedia messages, voice calls data, faster data speed**; however it requires a significant upgrade from the previous generation. Thus, the equipment involved in 3G technology includes : 1) BTS (Base station Transceiver) : Which plays the same role as for 2G; 2) Node B: Responsible for handling the radio interface and connecting mobile devices to the core network; 3) Radio Network Controller (RNC) : Controlling the Node B and managing the radio resources 4) Mobile Switching Center (MSC): The MSC is the central switching entity in the network that **connects calls between mobile devices**; 5) Serving GPRS Support Node (SGSN) : Responsible for managing packet-switched data services services and handling mobility for mobile internet access; 6) Gateway GPRS Support Node (GGSN): It serves as interface between the mobile network and external networks like internet; 7) The Home Location Register (HLR) and Authentication Center(AuC): plays the same role as in 2G; 8) Operations Support System (OSS) : It provides and functionalities for monitoring and managing the 3G network. Indeed, the 3G is appreciated for enabling higher- speed services and covers different frequency bands depending on countries, ranging between 850Mhz - 1700 Mhz [31].

Additionally, the fourth generation, commonly referred to as LTE(Long-Term Evolution) represents a significant advancement over previous generations in terms of infrastructure and services. This generation introduces higher data speeds, improved capacity, and better perfomance for mobile communication and data services. The upgrades in infrastructure include: 1) BTS and MSCs : These components remain unchanged from the previous generation 2) Evolved Packet Core (EPC) The EPC is a critical component of the 4G core network architecture which provides the packet-switched backbone that handles data traffic and ensures efficient data delivery between mobile devices and the internet or other networks; 3) Radio Access Network (RAN): The Ran is responsible for the radio interface between mobile devices and base stations; 4) LTE (Long-Term Evolution) : is the primary air interface enabling the high data speeds, low latency; 5) Back-haul Network : It connects base stations to the core network and internet infrastructure; 6) Spectrum Allocation: Hands over the mobile operator access to specific radio frequency

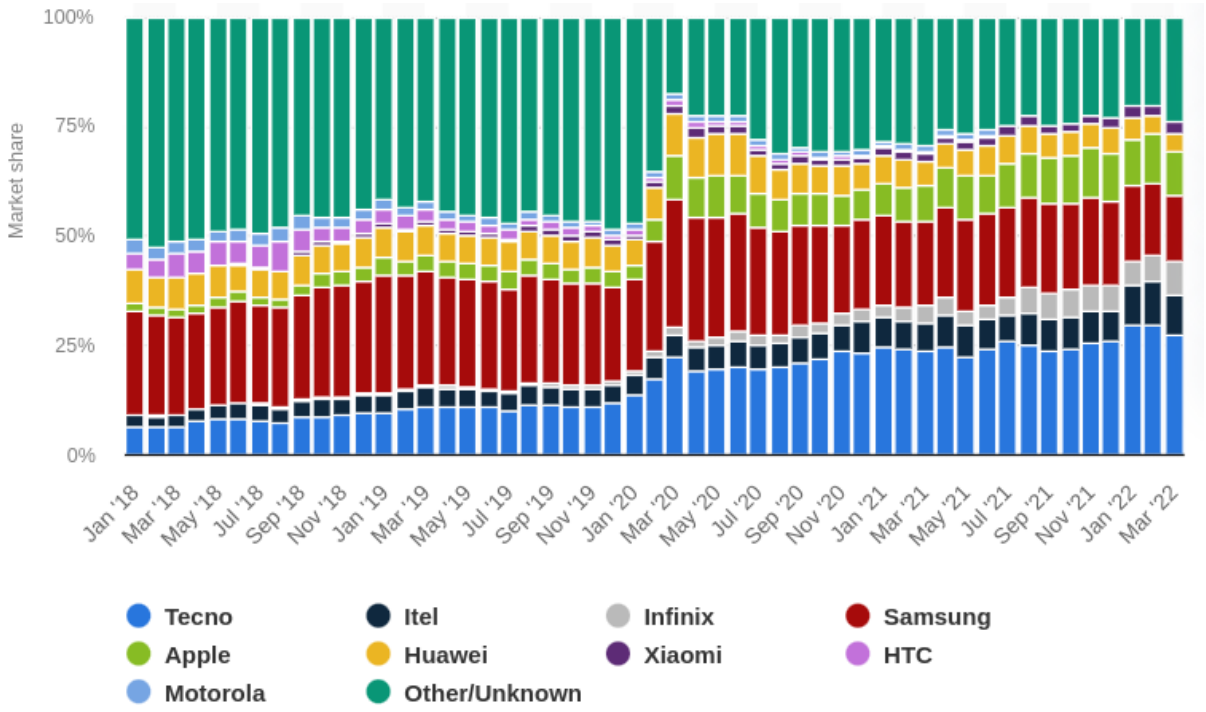

Figure 1.1: Market share of mobile device vendors in the Democratic Republic of the Congo from January 2018 to March 2022

bands; 7) Network Management System: These systems monitor and manage the 4G network, ensuring its smooth operation, performance optimization, and troubleshooting. However, it's spanning or coverage of 4G networks on frequency bands allowed in each country based on their preferences, ranging from 700MHz to 2600 Mhz. The higher frequency bands generally offer faster data speeds but may have a shorter range, while the lower frequency bands can provide broader coverage but with slightly lower data speeds [31].

1.2.4 Mobile Phone Models

Since the mobiles phones are essential tools for communication, there is a wide range of popular mobile widely used by citizens of the DRC. The popular models come from various brands and offer a range of features to cater to different user preferences and needs. Some of the popular mobile phone models in DRC include *Techno, Itel, Infinix, Samsung, Apple, Huwaei, Itel, HTC, Motorola*. As it can be seen on the figure 1.1, according to the recent statistics conducted by *Statistica*, Samsung was the market leader in terms of share from January 2018 to November 2020, but in 2022, Tecno has emerged as the market leader.

Furthermore, all these models provided above sell the telephone following different types which include mobile phones, offering the features such as touchscreen displays, cameras, internet, connectivity, and access to mobile apps; features phones, which are basic mobile phones used for calling and texting; smartphones used by the majority (around 35% in DRC), providing access to mobile internet, mobile apps, multimedia messaging, and various productivity tools; Tablets, used for reading and for the same functionalities as smartphones.

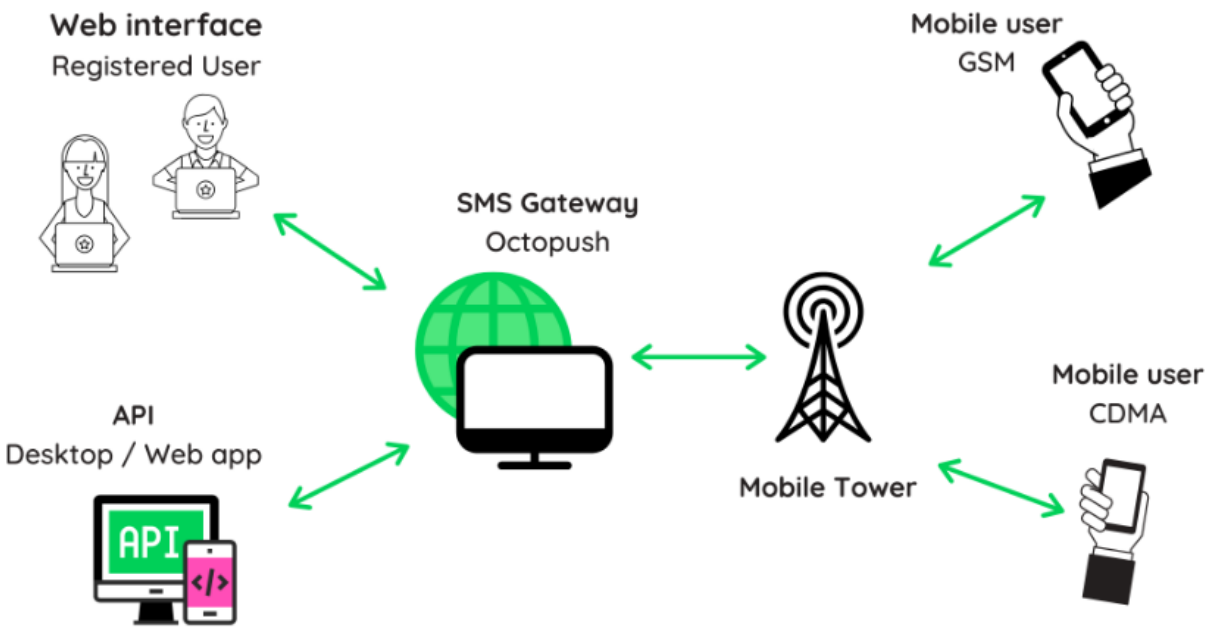

Figure 1.2: SMS Gateway Provider Architecture

1.2.5 Mobile usage and prevalence

In fact, each phone has its own unique set of characteristics that define its capacity and performance compared to others. Some phones come with specific applications that can be used independently, even without being connected to an operator, such as a camera, calculator, games app, and many others. However, other phones may not have such features.

To access the services provided by the operator, the phone's sim-card must be functioning, recognized by the operator, and capable of sending and receiving communication signals. **Of course, all these services work only if the phone's battery has sufficient power**.

Moreover, the services that citizen's subscribers benefit from are as follows :

Firstly, the Internet Access: The Internet services are used to connect people from different nodes. In fact, In accordance with the *WorldBank* [6] been used by 23% of the DRC's population in 2020. Nonetheless, it requires payment which proportionally gives mobile data usually expressed in Megabytes.

Secondly, the Text Messaging (SMS): Even though the internet is the most used for texting, the SMS remains a widely used form of communication, especially in regions with **limited internet connectivity or among users who prefer simple text messaging or do not have the internet connection**. Furthermore, with the architecture of GSM(Global System for Mobile Communications) invented in the second generation, sending messages became possible. Nowadays, the web environment has developed application interfaces (API) that connect external systems to operators for sending messages [19]. One of the platforms that offer these services connects its SMS gateway to the GSM operator, as seen in the case of *Octopush* [7] architecture shown in Figure 1.2

Thirdly, the Mobile Banking and Payments : With this services subscriber can make

[6]WorldBank : International Telecommunication Union (ITU) World Telecommunication/ICT Indicators Database `https://data.worldbank.org/indicator`

[7]Octopush : SMS platform for businesses connected with their audience

financial transactions; paying bills, transferring money conveniently.

Fourthly, the Mobile Entertainment: Mobile phones offer a range of entertainment options, from streaming videos and music to mobile gaming.

And finally, the others Mobiles apps: This party includes the health services, education, social medias applications.

1.3 Purposes of spammers in mobile messages

Most of the time, spammers prefer to promise recipients prizes and then ask for money to claim the offer. They also attack SMS gateways with *DoS* (Denial of Service) messages [2] whose goal is to overwhelm the system with unnecessary messages. Spammers send advertising and promotional messages based on company objectives, as well as SMS containing fake links or impersonating organizations to deceive recipients into taking certain actions or providing sensitive information [45]. Additionally, they may send SMS disguised as surveys to gather personal information for various fraudulent purposes.

1.4 Solutions

To address theses problems, it is necessary to involve various stakeholders, including network operators, app developers, regulatory bodies, and users. Firstly, it is recommended to implement mechanisms at the network level [18] to filter messages and block users involved in spamming. Secondly, users (subscribers) should be educated on how to analyze messages and report any one that is causing disturbances. Thirdly, regulatory measures should be enforced to establish stringent regulations and penalties for spammers and those engaged in fraudulent mobile activities. Fourthly, the development of apps that enable filtering, classification, and reporting in the subscriber side would be beneficial. Fifthly, a website can be set to collect messages whether spam or ham reported by users who have doubts about their legitimacy, and then Machine Learning models can be used for detection purposes. For this, supervised or unsupervised methods can be employed to classify and predicting whether a message is spam or ham.

1.5 Summary

Overall, with the growth of mobile technologies, subscribers benefit from diversified services including : text messaging, voice calls, mobile baking, entertainment apps, and many others. However, These advancements also bring new challenges, such as the development of spam messages that aim to disturb network subscribers with unwanted or threatening messages.

In DRC, especially in the eastern party, users face similar issues. This chapter emphasizes methods or techniques that can be used to address this problem and relatively reduce spam. One of the prominent techniques suggested is based on Artificial Intelligence, particularly Machine Learning algorithms.

Chapter 2

Review of the Literature and description of the approach

2.1 Introduction

This chapter delves into theory, methodologies, and machine learning techniques, including relevant algorithms and their deployment in the suggested solution. It also highlights the contributions of previous researchers in the field.

2.2 Revue of the Literature

Numerous researchers have extensively explored the subject of spam detection. Within this domain, some have directed their investigations towards the web environment, while others have delved into realm of mobile technologies.

Furthermore, these researchers have chosen to investigate the detection of spam across various communication channels, including emails and SMS, encompassing Multimedia SMS (MSS) as well. In the following sections, we comprehensively review the body of work that has been accomplished within this context as follows:

[22]. **Dr.V.M Veena K.Katankar** proposed a system that comprises an SMS gateway for transferring SMS messages after they have been stored and encrypted by the web server. This software operates through a web interface. Whenever a client sends a *POST* request, it is received by the web server, which is responsible for encryption or decryption if necessary. Subsequently, the gateway transfers the message as per its designated route. This solution proves to be particularly valuable in mobile banking and organizational marketing systems. Nevertheless, the author encourages other researchers to delve into channel services in communication and advanced encryption techniques.

[6] In their publication titled ***Short Message Service***, **Brown, Jeff and Shipman** members of IEEE, delve into several significant aspects. They start by exploring th growth of mobile phones and SMS services. They also examine the system architecture of SMS Centers and technologies used for message communication

Furthermore, they shine the spotlight on aggregators and services providers. These are the entities that enable users to send bulk messages, essentially sending messages with a large amount of text to a group of recipients. This includes the interesting capability of converting email messages into SMS.

Moreover, the article highlights that some of these aggregators may choose to collaborate with cellular networks. In this collaborative role, they act as intermediaries, connecting third-party entities that don't have direct relationships with cellular service providers. To achieve this, they employ a *SMPP (Simple Messaging Peer to Peer)* protocols.

[30] **Researchers A. Medani and A. Gani**, affiliated with the University of Malaysia, have published a comprehensive review focusing on security concerns and techniques related to mobile Short Message Service (SMS).

In their paper, they illuminate the process by wich a subscriber sends a message to another party while adhering to specific principles of the Over-The-Air (OTA) structure. This process involves transmitting the message from the sender to the base station and then forwarding it to the intended recipient trough the SMS Center (SMSC).

Crucially, they emphasize the importance of securing every SMS using *Public Key Infrastructure (PKI)*, ensuring end-to-end transmission security and safeguarding the message from unauthorized modifications. However, it's worth noting that the use of PKI can potentially impact mobile device performance due to the significant power requirements for the encryption process, and it may not guarantee integrity across all standards.

To address these security concerns within GSM systems, the researchers propose the implementation of *XML Key Management Specification* as a middleware solution. This middle ware system servers an intermediary, facilitating secure communication between mobile devices and enhancing overall system security for the benefit of clients.

[10] **Nikhil Kumar**, a reseacher affiliated with the University of New Delphi in India, published an article focusing on the topic of Email Spam Detection Using Machine Learning. In his study, he placed particular emphasis on comparing various machine algorithms, including ***Naive Bayes, Support Vector Classifier, AdaBoosting, K-Nearest Neighbour, and Bagging Classifier***. The objective was to predict wether an email was categorized as spam or legitimate (ham). To demonstrate his approach, he utilized an existing dataset available in the Kaggle workspace.

Through experimentation and parameter tuning, Nikhil found that Naive Bayes delivered promising results in terms of accuracy. However, he also pointed out a limitation associated with the Naive Bayes algorithm. This limitation is tied to its assumption of class-conditional independence, which implies that each feature is considered independent of the presence of the other features. In cases where this assumption does not hold, it can lead to misclassification of data points.

To address this limitation and enhance the performance of spam detection, the author recommended the use of **ensemble methods**. These methods involve the use of multiple classifiers for making class predictions, allowing for more robust and accurate results.

[33] In 2018, researchers Pavas Navaney, Gaurav Dubey, and Ajax Rana, who are affiliated with the University of Southern California and Amity presented a conference paper titled "SMS Spam Filtering using Supervised Machine Learning Algorithms".

Their study concentrated on a dataset comprising 5574 records, of which 4827 messages were categorized as "ham" (legimate messages), while 747 messages were classified as "spam" (unsollicited or unwanted messages).

The researchers applied three different machine learning methods to this dataset. Among these methods, it was observed that the *Support Vector Machine (SVM)* algorithm achieved the highest accuracy compared to the Naive Bayes and Maximum Entropy Classifier al-

gorithms.

[40] **Houshmand Shirani-Mehr**, a researcher in Machine Learning, published an article in 2013 titled : "SMS Spam Detection using Machine Learning Approach". His purpose was to address the spam filtering problem by utilizing ML algorithms. Therefore, He utilized a dataset from the *UCI Machine Learning Repository* repository [1] , which contained real SMS messages. In development, he employed the algorithms to tackle that problem such as : Naive Bayes with Laplace smoothing, Support Vector Machine, and Ensemble methods (*AdaBoosting and Random Forest*). As an improvement, the author added meaningful features such as the length of messages in terms of the number of characters and certain thresholds.

The results obtained after applying these methods to the dataset indicate that the SVM algorithm achieved the highest accuracy score.

[16] The authors of the article titled "SMS Spam Detection Using Machine Learning", namely **Gupta, Suparna Das and Saha, Soumyabrata and Das, Suman Kumar**. Their focus was on reviewing various techniques employed by other researchers in the realm of machine algorithms for SMS spam detection.

In their research, they adopted a similar approach by incorporating the *TF-IDF (Term Frenquency-Inverse Document Frequency)* method. This technique assesses the frequency of a word within a document and evaluates its importance in that document. *TF-IDF* is a well-known method for measuring word relevance in a collection of texts.

To assess the effectiveness of these techniques, the authors applied them to a spam dataset obtained from *Kaggle* [2] . After conducting their experiments and evaluations, the authors arrived at a noteworthy conclusion. They found that among all the ML algorithms they employed, the Naive Bayes algorithm consistently achieved the highest level of accuracy in SMS spam detection.

As shown above, many researchers have investigated the same topic using different approaches. Some have focused on security within mobile architecture, including message transfer processes, while others have concentrated on using Machine Learning (ML) models to combat the issue of spam. In general, these approaches are valuable to this project and serve as its inspiration at the point that many techniques related to these approaches have been implemented in this project.

However, what sets this project apart is its pratical approach involving specific society. Rather than solely relying on existing datasets from platforms like *Kaggle and UC Machine Learning Repository*, this project has actively engaged with people to collect data. It has also integrated some data from these platform datasets to enhance the quality of information.

Furthermore, this project harnesses the latest advancements in machine learning. It utilizes Ensemble Methods to achieve high levels of accuracy. Addionnally, it employs technical parameter tuning, including *GridSearcher and VotingClassifier*. In fact, Grid-

[1] **UCI ML** : The UCI Machine Learning Repository is a popular collection of datasets maintained by the University of California, Irvine (UCI). It serves as a valuable resource for researchers and practitioners in the field of machine learning and data mining. `https://archive.ics.uci.edu/`

[2] **Kaggle :** a platform for data science competitions and datasets. `https://www.kaggle.com/`

Searcher assists in identifying the most suitable parameters required for algorithm models.

Moreover, this project doesn't stop at model development, it extends to the deployment of the models generated through the processes. It provides backend *APIs* for certain platforms interested in learning from these results.

Additionally, it outlines the structure of GSM deployment, encompassing the SMSCenter's role in the mobile messaging system.

2.3 Tools and Techniques

In the mission of this project to create a messaging application that not only streamlines mobile communication but also tackles the pervasive issue of spam, this section serves as a technical guide. It will explore the tools and techniques at the core of the approach used.

Building an effective messaging application is like constructing a house- you need the right tools. Thus, this section discusses the software and technologies that form the foundation of our messaging app, including the programming languages, frameworks, and platforms utilized.

To combat spam effectively, this project is enlisting the help of machine learning. It delves into the world of data analysis and machine learning tools and frameworks : *numpy, pandas, matplotlib, scikit-learn,* that empower the authors to analyze, detect and prevent spam messages.

2.3.1 Messaging application development tools

In the realm of modern software development, the choices of development tools can significantly impact the efficiency and effectiveness of the project. When crafting application, the authors carefully considered the tools that will shape the foundation of the *backend* and *frontend* development. Actually, the *backend* references the environment where data of the app are stored, structured and more secured; while the *frontend* involves the space where techniques are developed to show to the user the interface comfortable for his understanding. Among all tools, some serve as programming languages and others as editors.

Hence, The choices made by the authors are *Python (with Django Framework)* as programming language and *SQL* for data structuring language in *backend* development and HTML, CSS and Javascript (with *Vue-Js Framework*) for *frontend* environment.

Back-end Development with *Python* (*Django*) and *SQL* :

is a powerful language appeared in 1980 firstly implemented by Guido van Rossum at *Stichting Mathematisch Centrum* in the Netherlands as a successor of a language called *ABC* [48]. It newest version is *Python 3* [3] which is available for all most environments, either *Macos* or *Linux* and *Windows*.

[3]Python : `https://www.python.org/`

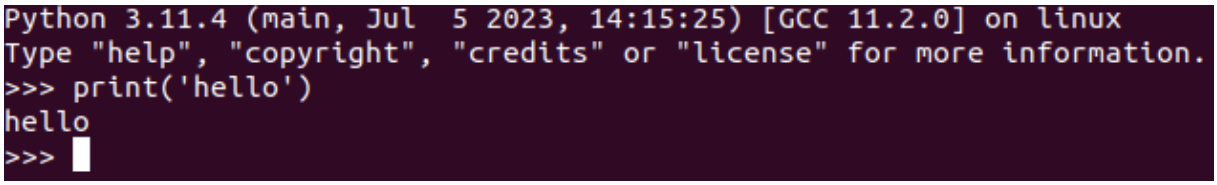

Figure 2.1: Testing if python is running on the OS (LINUX)

In comparison with *Java, C, C++, MATLAB*, *Python* is more readable, since it requires few lines of code which are clean. Next, it is a good choice for those who want to start with programming [5]. Technically it is versatile, since it used for a wide range of applications, from web development to mobile apps (with Kivy [4]), data analysis, machine learning, and scientific computing. Actually, the high reason that influenced the use of this programming language is its capacity to deal with data by proving scientists with many libraries.

Since the frameworks help to gain time in terms of development and structuring a project, *Python* provides many frameworks for web development. Notable these are Django, Flask, and more. Authors's choice, Django, is a high-level, open-source web framework for building robust and dynamic web applications. It's written in *Python*, which is one of the reasons for its popularity. It follows the ***batteries-included*** philosophy, providing a wealth of built-in features like authentification, URL routing, a powerful admin interface, and an ORM (Object-Relational Mapping) system, which simplifies database operations [1].

Actually, Django's ORM abstracts many *SQL* complexities, enabling developers to interact with the database using *Python* code, without needing to write raw *SQL* queries. This higher-level interaction simplifies database access and makes the development process more efficient. So, while *SQL* is at core of database interaction, Django's *ORM* providers a user-interface for developers, streamlining the development of data-driven web applications.
Suppose we want to retrieve all the employees in a database with salaries greater than $4000 using raw SQL. Here's what the SQL query would look like :

```
1 SELECT * FROM employees WHERE salary > 4000
```

While Django accomplishes the same task using the following code:

```
1 from myapp.models import Employee
2   employees = Employee.objects.filters(salary__gt = 4000)
```

How did we get there ?

In fact, the project must be running for doing that. For installing *Python*, just go to the official website for the download, no matter the OS version, either *MACOS* , *Windows* or *Linux/UNIX* since *Python* is portable. Indeed, the last version at the writing of this project was Python 3.12.0. To test if the it is working, just enter the command python as in figure 2.1.

[4]Kivy: `https://realpython.com/mobile-app-kivy-python/`

```
(base) christianresearcher@christianresearcher-HP-EliteBook-8470p:/$ sudo python3 -m venv SpamAppEnv
(base) christianresearcher@christianresearcher-HP-EliteBook-8470p:/$ source SpamAppEnv/bin/activate
(SpamAppEnv) (base) christianresearcher@christianresearcher-HP-EliteBook-8470p:/$ pip install django
```

Figure 2.2: How to install django ? Here, the name of *VE* is SpamAppEnv

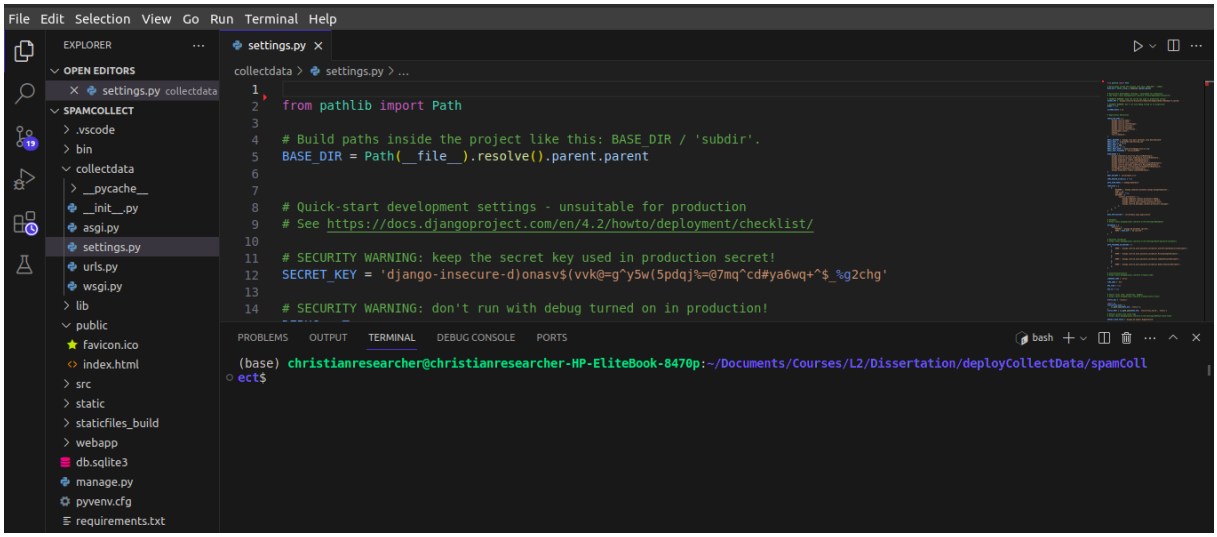

Figure 2.3: After the project and app are already created

Next, for installing Django, the steps remain pretty the same, going on the official page and following the guides as resumed in the as follows: For the step 1, Installing the virtual environment (*V.M*) [5] :
In Windows : `python -m venv myenv`; then active it by : `myenv\Scripts\activate`
The word *myenv* is just the name of virtual environment. But Why do we create it ? In fact, it is a best practice in Python development to manage dependencies, isolate projects, and maintain a clean and organized development environment.
In **MACOS/Linux** : Only the way of activating the *VM* changes. Then just by entering the following command:
`source myenv/bin/activate`; the virtual will be functioning.
Finally, as the *VM* is working Django, can be installed, by the command : `pip install Django`. At the writing of this project, the last version 4.2.6. The bit steps to follow for installing are demonstrated in the figure 2.2
Now, the project can be created, followed by the app inside, depending on interests. For that the commands : `django-admin startproject projectname` and : `django-admin startproject appname` can be utilized. The result is demonstrated in figure 2.3.
However, the configuration about the app created, ought to be made lest it should raise the error. Thus, in settings file, present in the project folder the properties called *INSTALLED_APPS*, as in the figure 2.4.
After, this we can just enter the commands :

```
1 python manage.py makemigrations
2 python manage.py migrate
```

For sure, the first command line says to Django the new changes, and the second applies or assesses them.

[5] How to create the VM ?https://realpython.com/python-virtual-environments-a-primer/

```
22  INSTALLED_APPS = [
23      'django.contrib.admin',
24      'django.contrib.auth',
25      'django.contrib.contenttypes',
26      'django.contrib.sessions',
27      'django.contrib.messages',
28      'django.contrib.staticfiles',
29      'appname',
30  ]
```

Figure 2.4: Adding the appname in *INSTALLED_APPS* dependencies

How does Django deal with the database ?

Django deals with databases through *ORM* as mentioned above. Its management involves to follow different steps such as:

Database Configuration : in the Django project's settings (usually in the *settings.py*, we have to specify the database we want to use. Django supports various databases types, including *PostgreSQL, Mysql, SQLite and Oracle* [6] . We have only define the database backend, connection details, and other options in the *DATABASES* setting. For instance the configuration of *MYSQL* will be like this :

```
DATABASES = {
  'default': {
  'ENGINE': 'django.db.backends.mysql',
  'NAME': 'the_db_name',
  'USER': 'the_db_user',
  'PASSWORD': 'the_db_password',
  'HOST': 'localhost',  # or the IP address of the MySQL
    server
  'PORT': '3306',  # MySQL default port
 }
}
```

Model Definitions : *Django* models are Python classes that define the structure of the database. Models are defined in the the ***models.py*** file. Each model class represents a database table, and each model field represents a table column. The syntax used for creating a database is as follows:

```
from django.db import models
class Employee(models.Model):
     name = models.CharField(max_length=100)
             salary = models.DecimalField(max_digits=10,
                decimal_places=2)
```

The corresponding *SQL* code is this:

```
        CREATE TABLE Employee (
         id INTEGER PRIMARY KEY,
         name VARCHAR(100),
         salary NUMERIC(10, 2)
```

[6]Setting Databases in Django : `https://docs.djangoproject.com/fr/4.2/ref/databases/#mysql-notes`

```
        );
```

Migrations : Every time a model (database table) is created or modified, *Django* need to notified for assessing these changes. As mentioned earlier, the command'**python manage.py makemigrations**' is executed in the terminal for managing the changes, '**manage.py migrate**' command is entered to apply all of these changes.

Database abstraction : With this technique, we do'nt no longer need to write *SQL* code, since by *Python* it is possible to interact with the database, and making a *queryset* request(*crud* [7]).Let's see how it works once again:

```
new_employee = Employee(name ="Eistein", salary=450000)
new_employee.save()
```

instead of doing this in SQL :

```
INSERT INTO employees (name, salary) VALUES ('Einstein',
   45000);
```

Indeed, we can see that even a non-professional in *SQL* can now deal with the database without any *SQL* code.
Furhermore, the use of abstraction techniques to interact with database, enhances its security of by guarding against *SQL injection*.

How does Django do for interacting with APIs ?

Django presents a useful package for interacting with the *API* called *DRF* [8] It allows : authentication policies including optional packages for *OAuth1a* and *OAuth2* [9], web browsable API and serialization [10] and deserialization that supports both *ORM* And *NO-ORM* data sources [32].

Front-end Development with *HTML*, *CSS* and *Js* :

The Front-end development is a crucial aspect of web development that focuses on creating the user interface and user experience of a website or web application. It involves using a combination of HTML,CSS, and Javascript to build the visible and interactive elements of a website.

The HTML (Hypertext Markup Language) is used as a maker of web pages by providing its structure and content. It uses a markup language with various tags to define headings, paragraphs, links, images, forms, and more [44]. For example a page with a heading and a paragraph should look like this in the content :

```
<div>
    <h1>Page Dev</h1>
```

[7]CRUD: Create Read Update Delete item

[8]DRF (Django Rest Framwork) : a powerful tools serving to interact with Apis. `https://www.django-rest-framework.org/`

[9]OAuth2 : (Open Authorization 2.0) : is a framework that allows third-party applications to access a user's data without exposing their credentials, such as passwords.OAuth 1.Oa is the old version of OAuth2

[10]Serialization : converting data into formats like *JSON*, *XML*, etc. The deserialization involves the reconstruction of the same operation

```
    <p>this is our page</p>
 </div}
```

Furthermore, the *CSS (Cascading Style Sheets)* is used for styling and layout. Thus, it controls the visual presentation of HTML elements. For example for our above code :

```
h1{
   color : blue; /* set the color to the element*/
   font-size: 24px; /* sets the font size */
   text-align: center; /* centers the element*/
}
```

Apart from that, the other powerful tool used in web development is *JS(JavaScript)*. It adds interactivity and dynamic behavior to web pages. It also leveraged for creating features like images sliders, form validation, animations and real-time updates without having to **reload the page**. The common frameworks used to extend its productivity are : React, Angular and Vue-js [51]. By updating the above *HTML* code, we can see the interactity created by JS:

```
<!DOCTYPE html>
<html>
<head>
 <title>JavaScript Example</title>
</head>
<body>
 <div>
 <h1 id="pageTitle">Page Dev</h1>
 <p>this is our page</p>
 </div>

 <button id="changeTitleButton"  onclick="changeTitle()">
    Change  Title</button>

 <script>
        // JavaScript function to  change the title
        function changeTitle() {
        // Get the h1 element by its ID
        var titleElement =  document.getElementById("
           pageTitle");

        // Check if the element exists
        if (titleElement) {
        // Change the text of the h1 element
        titleElement.innerText = "New Page Title";
          }
        }
 </script>
</body>
</html>
```

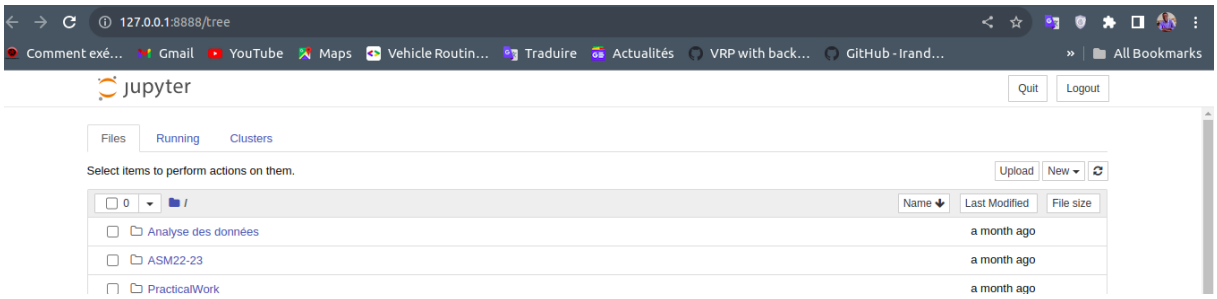

Figure 2.5: Start jupyter in conda kernel

In this example, the button with the ID 'ChangeTitleButton' and the function called 'ChangeTitle()' are added. The click on the 'Change Title' button executes the changeTitle function. Then, the function retrieves the `<h1>` element by its ID ("pageTitle") and changes its text to "New page Title". Thus, by this bit snippet code, *JavaScript* really appears interactive.

2.3.2 Machine Learning tools and frameworks

Machine learning involves learning from data, visualizing, analyzing it, and making predictions. To do this, we need the right tools. *Python* is one of best and powerful tools used for these tasks. It has a large and active community that continuously contributes to its growth. *Python* offers a variety of packages that assist data scientists in their work.

In Python, common packages used for working with data include **numpy, pandas, matplotlib, seaborn, scikit-learn, tensorflow**, and more.

The Integrated Development Environment (*IDE*) used for working in Python, even for all backend-development is : Visual studio. It is portable and downloadable from the official page(`https://code.visualstudio.com/`). Additionally, we have another *IDE* called *Anaconda* including *Jupyter notebook* which is used for machine learning and data science projects. It's designed to simplify package management and deployment by using a package manager called *Conda* which helps users to install, update, and manage packages, libraries and dependencies [46].

To start a new project we just execute in the terminal, the command : `jupyter notebook`, then a default browser configured just open directly the link. The page look like in the fig 2.5.

Data manipulation with *Numpy*, *Pandas*

Numpy Library

Numpy is a fundamental library for scientific computing with Python. It provides support for large, mutli-dimensional arrays and matrices, along with an assortment of high-level mathematical functions to operate on these arrays.

Actually, the reason of thinking about a new tool of computing in Python is because, all data structures it provides: lists for enumerating a collection of objects, dictionaries to build hast tables, are not ideally suited to high-performance numerical computation [50]. Basically, the usage of numpy asks for importing its modules like this :

```
1 import numpy as np  #np is a common alias used
2 In [4]: np.__version__
3 Out [4] : '1.24.3'
```

The structure and creation of a *NumPy* array. The fundamental data structure provided by the Numpy library for representing arrays is **ndarray** it refers to the array which is n-dimensional or multi-dimensional [52]. Thus, several means can be used to create an array as follows :
- With 1D(dimension):

```
In[8] : np.array([1, 2, 3, 4, 5])
#Creates a NumPy array from a given list or iterable.
Out[8] : array([1, 2, 3, 4, 5])

#2D array
In[12] : np.array([[1, 2, 3], [4, 5, 6]])

In[12]: array([[1, 2, 3],
               [4, 5, 6]])
```

```
In[10] : np.arange(1, 10, 2)
# Creates a 1D array from 1 to 9 with a step of 2
Out[10] : array([1, 3, 5, 7, 9])
#2D array for arange function, requires manipulation
```

Many other functions can be used for creating arrays, like : empty, zeros, ones, eye, etc.

Manipulation of arrays. The manipulation operation includes : The splitting, slicing, indexing,reshaping, arithmetical operations, concatenations, comparisons and many others. Let's dive into some of that operations as follows:

```
#We generate the integers from zero to twelve and
# re pack them into a 4x3 array
In [21] : np.arange(12).reshape((4,3))
Out [21] : array([[ 0,  1,  2],
                  [ 3,  4,  5],
                  [ 6,  7,  8],
                  [ 9, 10, 11]])
#Suppose we want to multiply each vector element by 3
In [24] :  a = [3,7,4]
          [4*x for x in a]
Out[25] : [12, 28, 16]
#Concatenate arrays vertically
In [29] : np.vstack(([1, 2, 3],[1, 2, 3]))
Out [29] : array([[1, 2, 3],
                  [1, 2, 3]])
#Concatenate arrays horizontally
In [31] : np.hstack(([1, 2, 3],[1, 2, 3]))
Out [31] : array([1, 2, 3, 1, 2, 3])
```

Let's mention that ndarray object is homogeneous since all elements within must have the same data type. The types allowed are : Float, int, bytes, str, number and complex

(for decimal complex numbers). To define, the type on the array on creation is made by **dtype** property like this:

```
In [55] : np.array([1, 2, 3, 4, 5], dtype='float')
Out [55] : array([1., 2., 3., 4., 5.])
```

Besides, dealing with slicing and splitting still depends on the dimension. Lets break in the code to see that :

```
In [56] : arr = np.array([[1, 2, 3],
[4, 5, 6],
[7, 8, 9]])

#split along rows (axis=0)
In [59] : split_rows = np.vsplit(arr, 3) # Split into threee
   1-row arrays
Out[59] : [array([[1, 2, 3]]), array([[4, 5, 6]]), array([[7,
    8, 9]])]

#slicing
In [60] : arr[1:3, 1:3]  # Get a 2x2 subarray
Out [60] : array([[5, 6],
                  [8, 9]])
```

Some others functions are used as follows : all, any, cov-corrcoef, dot, where, random(), randint() and many others. The details of usage are given on the official of Numpy package [11].

Pandas library :

Pandas [12] is an essential open-source Python library used for data manipulation and analysis. It provides functions for reading and writing data structures in various formats, including *CSV* and text files, Microsoft Excel, SQL databases, and the fast *HDF5* [13] format. Additionally, Pandas can **align data, handle missing data, allow reshaping and pivoting, perform data aggregation and transformation, and merge and join data sets** [28].
It offers many other capabilities for efficient data processing. To start with *pandas*, we have to create the pandas object like this:

```
import pandas as pd # pd is an alias name
```

Data Structures. Pandas has two main structures: Series and DataFrame. The first is one-dimensional array, while the second is a two-dimensional table that is similar to a spreadsheet or SQL. Let's break into an example:
- Series:

[11] Numpy, official page : https://numpy.org/
[12] https://pandas.pydata.org/
[13] HDFR (Hierchical Data Format version 5) : It is a versatile and high-performance data storage format commonly used in scientific and data analysis fields.

```
# Creating a Series from a list
In [46] : data = [1, 2, 3, 4, 5]
Out [46] : pd.Series(data)
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

- DataFrame:

```
In [48]: data = {'Name': ['Chris', 'Alice', 'Jojo'],
                 'Age': [25, 30, 22],
                 'City': ['Bukavu', 'Goma', 'Matadi']}
         pd.DataFrame(data)
```

Out[48]:

	Name	Age	City
0	Chris	25	Bukavu
1	Alice	30	Goma
2	Jojo	22	Matadi

Reading and Writing data. Pandas offers versatile functionality which deal with various sources and formats. The common sources include :

- CSV files: They are read and written like this :

```
#Reading from CSV
data = pd.read_csv('dafileName.csv')
#Writing to CSV
data.to_csv('new_data_file_name.csv', index=False)
```

- Excel files :

```
# Reading from Excel
data = pd.read_excel('dafileName.xlsx', sheet_name='Sheet1')

# Writing to Excel
data.to_excel('new_data_file_name.xlsx', sheet_name='Sheet1'
    , index=False)
```

- SQL Databases :

```
# Reading from SQL database sqlite
connection = sqlite3.connect('my_database.db')
query = 'SELECT * FROM my_table'
data = pd.read_sql(query, connection)

# Writing to SQL database
data.to_sql('new_table', connection, if_exists='replace',
   index=False)
```

In the same way, functions **read_json, read_html** are also respectively used for dealing with the JSON and HTML data.

- Data Cleaning. Pandas uses several functions, among them we have drop_duplicates(), fillna(). Which are used like this:

```
# Removing duplicates
df = df.drop_duplicates()

# Handling missing values
df = df.fillna(0)
```

- Data Exploration. Pandas provides methods for exploring the dataset, such as **head(), tail(), describe(), and info()**. Thus, the application will look like :

```
data = {'A': [1, 2, 3, 4, 5],
'B': ['X', 'Y', 'Z', 'X', 'Y']}
df = pd.DataFrame(data)
# Display the summary statistics of numeric columns
print(df.describe())

                A
count   5.000000
mean    3.000000
std     1.581139
min     1.000000
25%     2.000000
50%     3.000000
75%     4.000000
max     5.000000
```

Data Manipulation. Pandas performs the manipulation of tables as Excel using the pivot_table() and merge() function perform combination operation for analysis. Additionally it allows indexing, slicing, filtering, modifying data. Let's break into the demonstration:

- Pivot table :

```
In [72] : df.pivot_table(index='B', values='A', aggfunc='mean
   ')
print(pivot)
     A
B
X  2.5
Y  3.5
Z  3.0

#The operation computed is the average measurement
```

- Merging :

```
 #Create two DataFrames
 data1 = {'Key': [1, 2, 3, 4, 5],
```

```
'Value1': ['A', 'B', 'C', 'D', 'E']}
data2 = {'Key': [3, 4, 5, 6, 7],
'Value2': ['X', 'Y', 'Z', 'U', 'V']}
df1 = pd.DataFrame(data1)
df2 = pd.DataFrame(data2)

# Perform combination-like operation
result = pd.merge(df1, df2, on='Key', how='left')
print(result)

Key  Value1  Value2
0    1       A     NaN
1    2       B     NaN
2    3       C       X
3    4       D       Y
4    5       E       Z
```

- Slicing:

```
#Consider the example on the top about
#name, age, city
# Slice specific rows and columns
  df.loc[1:3, 'Name':'Age']
#From the second column till the threeth line
#From the Name till the Age column
#The result gives this:

Name   Age
1       Bob    30
2   Charlie    35
```

Assume that we want to apply some characteristic on a column :

```
# Create a new column based on an existing column
df['days'] = df['Age'].apply(lambda x: x * 360)
print(df)
# The result is like this:
      Name    Age     City        days
0    Alice    25      New York    9000
1      Bob    30   Los Angeles   10800
2  Charlie    35       Chicago   12600
```

- Modifying Data :

```
df.loc[2, 'Name'] = "Jonathan"
print(df)
#The df becomes like this :
Name   Age          City    days
0     Alice    25      New York    9000
1       Bob    30   Los Angeles   10800
2  Jonathan    35       Chicago   12600
```

Data Visualization With *Matplotlib* and *Seaborn*

Matplotlib

Matplotlib is a open-source, flexible and customizable Python package used for creating 2D and 3D [14] plotting having a high production-quality. In addition to this, it includes the capacity of saving images in different output formats (JPG, PNG, PS and others) [47].

To get started with it, we just go to the official page and download the dependencies [15]. However, for who are those using integrated development environment (*IDE*) like *Anaconda numpy*, *pandas*, and *matplotlib* are incorporated inside. When working with data visualization, the type of plot to choose should depend on the nature of the data. In the table 2.1 , the overview of types of plots and when to use them is given. However, some plots can be combined even though they are primarily made for whether categorical or numerical visualization.

The usage of *matplotlib* properties require a good manipulation of data, often done by the *numpy* and *pandas*. Let's break into examples of its usage:

First all, the import of *matplotlib* is required for any manipulation. It is done like this :

```
import matplotlib.pyplot as plt #plt is also an alias
```

```
import matplotlib.pyplot as plt
import numpy as np

# Data
x = np.arange(1, 11)
y1 = x
y2 = x**2
y3 = np.sqrt(x)

# Create a figure with subplots (1row and 3 columns)
fig, axs = plt.subplots(1, 3, figsize=(15, 5))

# First Subplot: Line Plot
axs[0].plot(x, y1, color='b', marker='o') # 'o' is a clue

#define legends
axs[0].set_title('Line Plot')
axs[0].set_xlabel('X-axis')
axs[0].set_ylabel('Y-axis')

# Second Subplot: Bar Plot
axs[1].bar(x, y2, color='g', alpha=0.6)
axs[1].set_title('Bar Plot')
axs[1].set_xlabel('X-axis')
axs[1].set_ylabel('Y-axis')

```

[14] mpl_toolkits.mplots3d : `https://matplotlib.org/stable/tutorials/toolkits/mplot3d.html`, tools used for generating 3D plots

[15] Matplotlib: `https://matplotlib.org/`

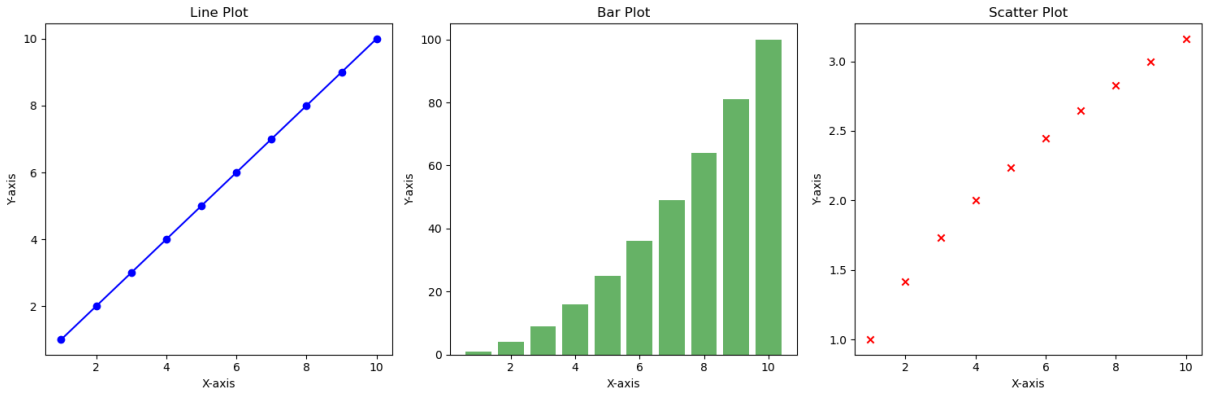

Figure 2.6: How to use *matplotlib* for visualization

```
# Third Subplot: Scatter Plot
axs[2].scatter(x, y3, color='r', marker='x')
axs[2].set_title('Scatter Plot')
axs[2].set_xlabel('X-axis')
axs[2].set_ylabel('Y-axis')

# Adjust subplot layout to prevent overlapping
plt.tight_layout()

# Display the figure
plt.show()
```

Additionally, *matplotlib* is used for creating 3D as follows :

```
        from mpl_toolkits.mplot3d import Axes3D
        import matplotlib.pyplot as plt
        import numpy as np

        fig = plt.figure()
        ax = fig.add_subplot(111, projection='3d')

        x = np.random.rand(10)
        y = np.random.rand(10)
        z = np.random.rand(10)

        ax.scatter(x, y, z, c='r', marker='o')

        ax.set_xlabel('X-axis')
        ax.set_ylabel('Y-axis')
        ax.set_zlabel('Z-axis')

        plt.show()
```

Notice that: *matplotlib*, uses *mpl_toolkits* for integrating with 3D, since at its release it was creating 2D plots only. Besides, the difference observed in the code at axis where the function *add_subplot* receives 111 value as the first argument. It just defines that the plot

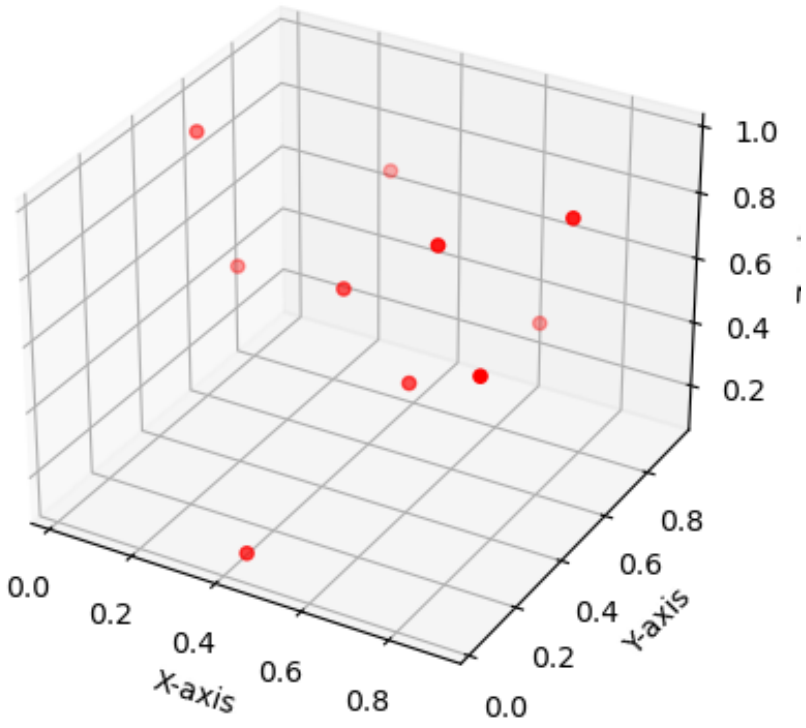

Figure 2.7: 3D plot with Matplotlib

will be a single for all 3 dimensions as shown in the figure 2.7.

Seaborn

Seaborn is another Python library for statistical data visualization, built on *matplotlib*. It just provides properties able to create more informative an visually appealing graphics [41]. To get started with it, we just visit the official page [16] which presents all steps required for installing. Alternatively, we can use an the integrated development environment *IDE* like *Anaconda* once again, which already includes it.

The following example shows how to combine both *matplotlib* and *seaborn*:

```
import matplotlib.pyplot as plt
import seaborn as sns # Its how to start the

# Create a Seaborn plot
sns.set(style="whitegrid")
tips = sns.load_dataset("tips")
ax = sns.barplot(x="day", y="total_bill", data=tips)

# Customize the plot using Matplotlib
ax.set(xlabel="Day of the Week", ylabel="Total Bill Amount")
plt.title("Average Total Bill Amount by Day")

```

[16] Seaborn: `https://seaborn.pydata.org/`

Type of data	Plot name	Utilities
Categorical Data	Bar Plot	It is deal for showing the frequency or count of categorical data. Example: Comparing the number of apples, bananas, and oranges sold at a fruit stand
	Pie Chart	Suitable for displaying parts of a whole. Example: Showing the composition of monthly expenses (eg. rent, groceries, utilities).
	Stacked Bar Chart	Useful for comparing categories while also showing their composition
Numerical Data	Histogram	Visualizes the distribution of a single variable or continuous data. Example : Analyzing the distribution of ages in a population
	Box Plot	Displays the distribution and spread of numerical data
	Scatter Plot	Depicts relationships between two numerical variables Example : Showing the correlation between the number of study hours and exam scores for multiple students.
	Line Plot	Shows changes in variable over a continuous range, often over time. Example : Plotting stock prices over several months to visualize trends.
Mixed Data	Violion Plot	Combines a box plot with a kernel density estimation to visualize the distribution. Example: Comparing the distribution of test scores across different schools.
	Heatmap	Useful for displaying relationship between multiple variables in matrix. Example : Analyzing the correlation between various factors (eg., age, income, and education level) in a survey.

Table 2.1: Choose the plot depending on data's nature

```
#Save the plot as an image
plt.savefig("seaborn_customized_plot.png")
plt.show() # Show the plot
```

The data used in the following code, named 'tips,' is provided for practice purposes. *Seaborn* is used to generate the plot, while *matplotlib* customizes it by adding a legend and saving the image generated in figure.

Machine learning with scikit-learn

Since Python programming language is establishing itself as one the most popular languages for scientific computing. Many and various libraries are developed inside making it more and more appealing [37].
Scikit-learn is an open-source and commercially(usage - BSD license) machine learning library for the Python programming language. It was initially developed by David Counapeu in 2007 as part of the Google Summer of Code project. Since then, it has grown to become one the most popular and widely used Machine Learning libraries in Python ecosystem [36]. The techniques and properties used inside of it, allows it to predictive data analysis by performing the classification, regression, clustering, preprocessing and more other tasks.
To get started with it as usually, the official page present all the steps for downloading all the packages or using *IDE* like *Anaconda* which encompasses it.

Before using its functions we have to initialize the by import the class which includes the methods and properties achieving a given task. For example, suppose that we are about to process data a class called *StandardScaler* can be useful **when the dataset contains features with different scales** . Then, it standardizes the data, making it have a mean of 0 and standard deviation of 1, assuring that all features have the same scale.

```
In[2]:  from sklearn.preprocessing import StandardScaler
        import numpy as np

        # Example data
        data = np.array([[1.0, 2.0, 3.0],
        [4.0, 5.0, 6.0],
        [7.0, 8.0, 9.0]])

        # Create a StandardScaler instance
        scaler = StandardScaler()

        # Fit the scaler to the data and transform the data
        data_standardized = scaler.fit_transform(data)

        print("Original Data:")
        print(data)
        print("Standardized Data:")
        print(data_standardized)

Out[2]:
```

```
Original Data:
[[1. 2. 3.]
[4. 5. 6.]
[7. 8. 9.]]
Standardized Data:
[[-1.22474487 -1.22474487 -1.22474487]
[ 0.          0.          0.        ]
[ 1.22474487  1.22474487  1.22474487]]
```

We just notice that, this time, all the standardized data have a mean close to 0 and a standard deviation close to 1. For that, the data become centered around zero and have now consistent units of measurement.
Additionally, the ***fitstransform()*** method on the object, defines both fitting and transformation process. Actually, the ***fit()*** method computes the mean and standard deviation of the trained data and ***transform()*** method scales the trained data based on the mean and standard deviation. Indeed, *fitstransform()* includes all both.
Actually, the explanations about training, fitting and others principles concerning a model are going to be tackled in the section 3 untitled :'Thinking in machine learning'.

2.3.3 Summary concerning the tools and frameworks

The completion of this project necessitated the utilization of various dependencies, tools, frameworks. These resources were instrumental in realizing the project's objectives. Notably, they were categorized into two main areas: those integral to the core functionality and others relevant to the user side, distinguishing the back-end from the front-end. Moreover, the project involved tools for data analysis and predictive modeling. The structure of these tools is presented in the table 2.2 for more clarity.

2.4 Description of the methodology and approach

Choosing a right methodology including approaches and methods to use, is an important task. However, the nature of environment defers and leads to challenging evaluation before determining the fit one. Thus, since this project investigates in Machine Learning project, the appealing methodology chosen for its achievement is inspired from MLOps(ML Operating systems).

2.4.1 ML operating systems

ML endeavors aim to deal with their projects from development till the production step. However, a large number of projects based on ML don't reach the final step according to their expectations. Actually, the paradigm of MLOps came to face this issue [23]. For that, it is a set of principles, best practices and concepts, and development culture that provide an end-to-end machine learning development process to design, build and manage reproducible, testable and evolvable ML-powered software.

	Tools	Roles
Data analysis and Machine Learning libraries	*Numpy*	Scientific computing library
	Panda	Data manipulation and analysis library
	Matplotlib	Python library used for 2D and 3D data visualization
	Seaborn	Another Python library for statistical data visualization, built on *matplotlib*.
	Scickit-learn	Machine learning machine learning library
Programming Languages (*Front-end* and *Back-end*)	*Django* Python	Python framework for developing web applications and *APIs*
	HTML	Maker of web pages by providing its structure and content
	CS	Styling the content
	JS	Rendering web pages interactive and dynamic

Table 2.2: Structuring the tools

Furthermore, MLops involves the union of many disciples, including Machine learning, software engineering (concerning DevOps too), data engineering as clarified in figure 2.8.

The Machine learning part is dealt by **data scientists** and have the principle role of building the models that address the business question or needs. While **Data Engineering** is conducted by data engineers making sure that data pipelines which are the core of the ML model life cycle, are in turn and clean. Next, the **Software engineering**, encompasses **software engineers** not highly concerned by the machine learning model building since they bring ML problem into a well-engineered product (into applications). In the other hand, **DevOps** is directed by DevOps engineers who bridges the gap between development and operations, granting that updates are continuously integrated and the development is still pursued (referring to *CI/CS*, Continuous Integration et Development).

All these disciplines work together to follow the ML life cycle, demonstrating that ML endeavors are on ongoing process within the same project for ensuring that it is more impacting in terms of results. As shown in the figure 2.9 the ML life cycle follows several steps resumed in the following points :

- Definition of the object and specification
- Data Collection
- Preparation and exploratory Data Analysis
- Model Training & Evaluation
- Model Deployment
- Model Monitoring

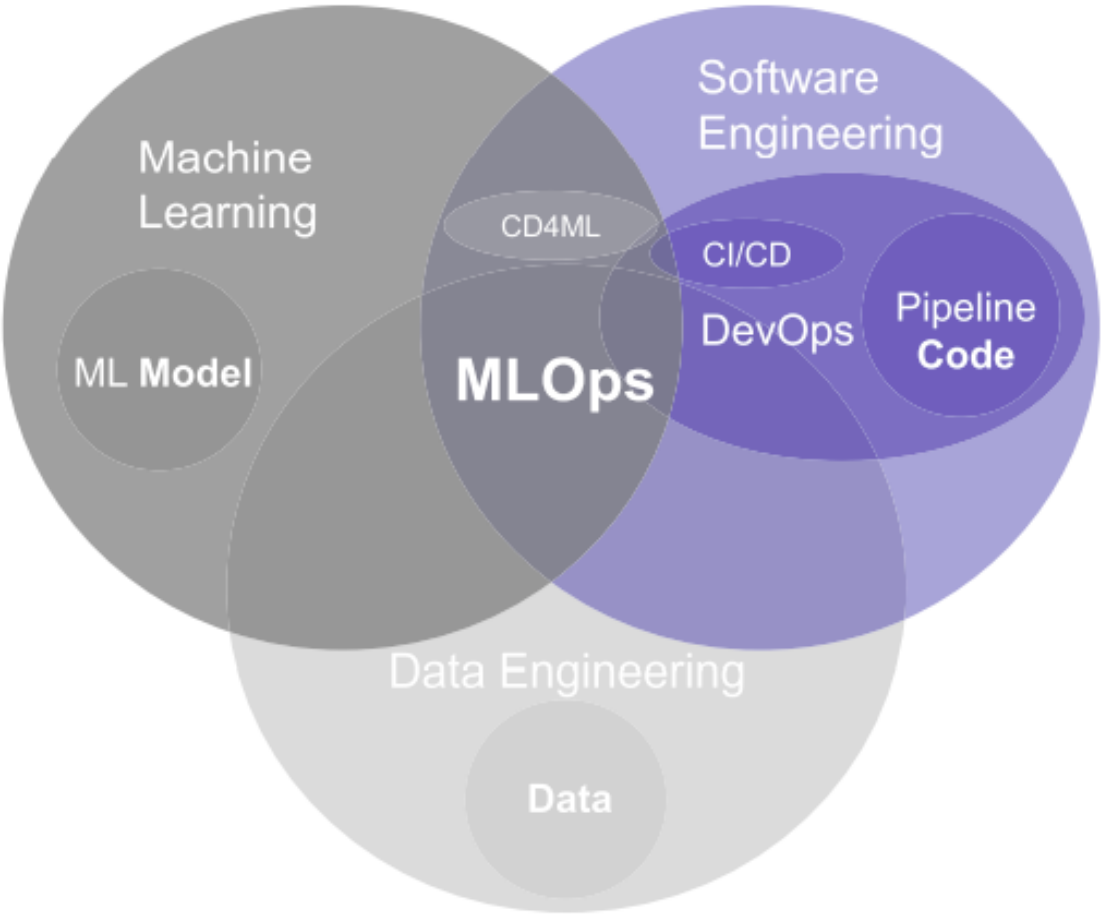

Figure 2.8: Intersection of disciplines in MLops

2.4.2 Definition of the object and specification

The definition of the object and specification is start stage where details about the problem is given in the clear way for facilitating the ML development. This task is associated to Business stakeholders who define the goal pursued.

2.4.3 Gathering data

At this step, raw data are collected from various data pipeline(sources) depending on interests of project and environment. Thus, data's source can be : (a) Databases, including data structured in relational databases like SQL and NoSQL databases. Actually, databases is mostly preferable source since they give structured data well-suited for analysis;

(b) Files (CSV, Text Files). This mean is highly used. It requires important analysis of the content for being aware of how and what part to extract (as spreadsheets for Excel), separated values (as for CSV files).

(c) APis. They are also the common use for structured data(in JSON, XML format) , leveraging web services for providing data to third parties or internal parties of the system [54].

(d) Web Scrapping. This mean refers to techniques used for collecting data from web sites (sometimes illegally) and structure it into spreadsheets, CSV or others simple means

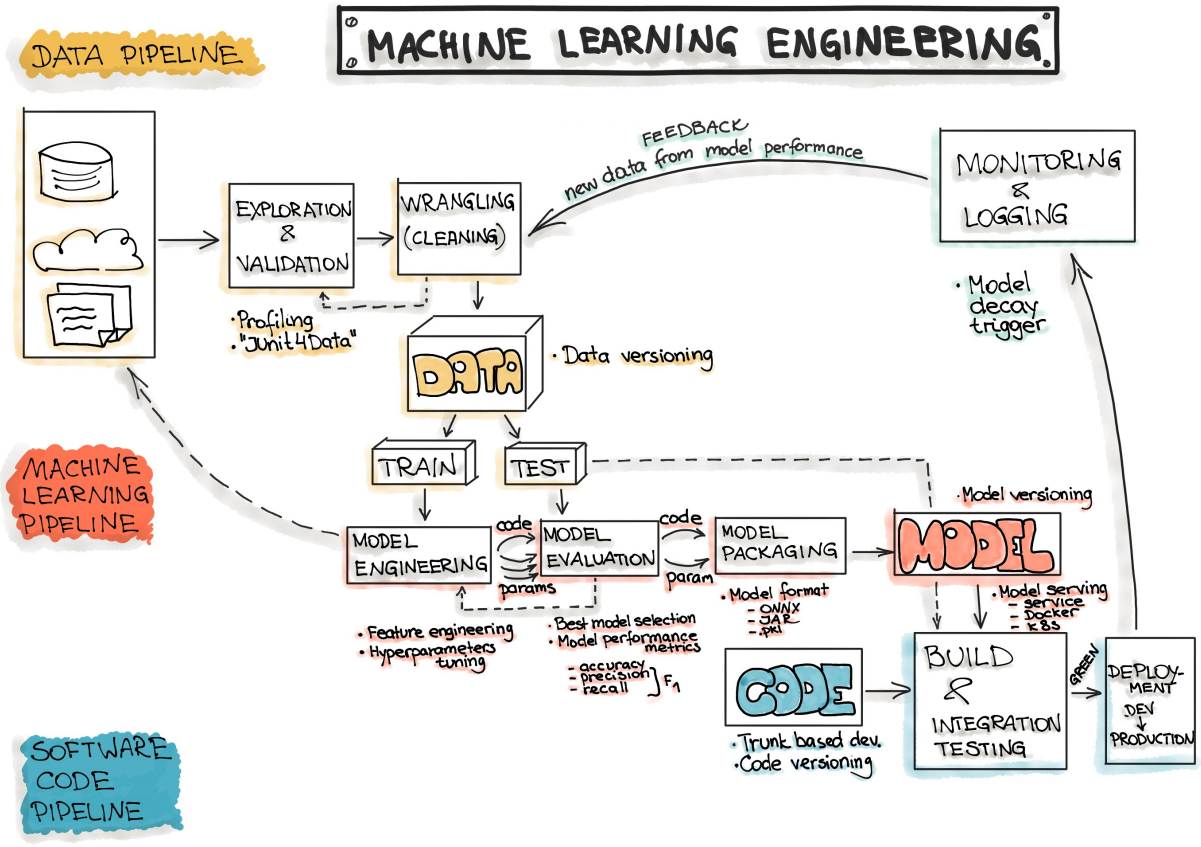

Figure 2.9: Machine Learning Workflow

[42]. It's more appreciated when data are not available from the APIs. That being, some applications performs like

(e) Images and videos : Images, videos are generated from cameras, satellites and other imaging devices. Thus, they are useful in specific tasks of ML like classification of taxonomy videos [29]. Many others means exist like surveys and questionnaires, audio data, however the guidance is problem to address.

Once the data are collected, it stored in **dataset** where they are subjected to manipulation and analysis.

2.4.4 Data preparation and wrangling

The preparation refers to two main tasks : The data pre-processing(wrangling) and data exploration . The data pre-prepocessing involves the learning of nature of the data, its characteristics, types format and quality, that consequently lead to better outcome. While, The data wrangling consist of cleaning necessary data.

On the other hand,**the exploratory data analysis** process aims to receive collected and cleaned data and facilitates the visualization that can be helpful in detection of outliers, identification of clusters and trends [49].

2.4.5 Model building and selection

Bibliography

[1] Marty Alchin. *Pro Django*. Apress, 2013.

[2] Iosif Androulidakis, Vasileios Vlachos, and Alexandros Papanikolaou. Fimess: filtering mobile external sms spam. In *Proceedings of the 6th Balkan Conference in Informatics*, pages 221–227, 2013.

[3] Paul Kimumwe Lillian Nalwoga Juliet Nanfuka Edrine Wanyama Wairagala Wakabi PhD Ashnah Kalemera, Victor Kapiyo. State of internet freedom democratic republic of the congo 2019. *Mapping Trends in Government Internet Controls, 1999-2019*, 2020.

[4] Allan Bluman. *Elementary statistics: A step by step approach*. McGraw-Hill Education, 2017.

[5] Andrey Bogdanchikov, Meirambek Zhaparov, and Rassim Suliyev. Python to learn programming. In *Journal of Physics: Conference Series*, volume 423, page 012027. IOP Publishing, 2013.

[6] Jeff Brown, Bill Shipman, and Ron Vetter. Sms: The short message service. *Computer*, 40(12):106–110, 2007.

[7] Erik Cambria and Bebo White. Jumping nlp curves: A review of natural language processing research. *IEEE Computational intelligence magazine*, 9(2):48–57, 2014.

[8] Guangquan Chen, Weijun Wang, and Xuan Zhou. A survey on sms spam filtering techniques. *Journal of Network and Computer Applications*, 80:149–159, 2017.

[9] Catalin Cimpanu. Simjacker vulnerability exploited for surveillance by at least one nation-state. *ZDNet*, 2019.

[10] Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, and Hamzah Al Najada. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):1–24, 2015.

[11] John W Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2014.

[12] Prof. Dantu's CSE. Cellular network basics. In *ADV. REAL-WORLD NETWORKS*, volume 14-760. University of North Texas, 2018.

[13] MONUSCO DRC. Protect, stabilize, consolidate peace, monusco's report. *United Nations Organisation Stabilization Mission in the Democratic Republic of Congo*, pages 1–4, 2015.

[14] Cori Faklaris and Sara Anne Hook. Oh, snap! the state of electronic discovery amid the rise of snapchat, whatsapp, kik, and other mobile messaging apps. 2016.

[15] Antonio Ghezzi, Marcelo Nogueira Cortimiglia, and Alejandro Germán Frank. Strategy and business model design in dynamic telecommunications industries: A study on italian mobile network operators. *Technological Forecasting and Social Change*, 90:346–354, 2015.

[16] Suparna Das Gupta, Soumyabrata Saha, and Suman Kumar Das. Sms spam detection using machine learning. In *Journal of Physics: Conference Series*, volume 1797, page 012017. IOP Publishing, 2021.

[17] Kennedy Ochilo Hadullo and DM Getuno. Machine learning software architecture and model workflow. a case of django rest framework. *American Journal of Applied Sciences*, 18(1):152–164, 2021.

[18] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Alexander G Gray, and Sven Krasser. Detecting spammers with snare: Spatio-temporal network-level automatic reputation engine. In *USENIX security symposium*, volume 9, 2009.

[19] Marko Hassinen and Smile Markovski. Secure sms messaging using quasigroup encryption and java sms api. *SPLST*, 3:187, 2003.

[20] Sunil Kumar Jangir, Manoj Kumar Sharma, and Pawan Kumar Gupta. Design and implementation of sms gateway api for mobile communication networks. *International Journal of Computer Applications*, 151(9):1–5, 2016.

[21] Thomas R Karl, Claude N Williams Jr, Pamela J Young, and Wayne M Wendland. A model to estimate the time of observation bias associated with monthly mean maximum, minimum and mean temperatures for the united states. *Journal of Applied Meteorology and Climatology*, 25(2):145–160, 1986.

[22] Veena K Katankar and VM Thakare. Short message service using sms gateway. *International Journal on Computer Science and Engineering*, 2(04):1487–1491, 2010.

[23] Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. Machine learning operations (mlops): Overview, definition, and architecture. *IEEE Access*, 2023.

[24] M Lavanya and KR Aruna. Sms spam detection using deep learning. *Journal homepage: www. ijrpr. com ISSN*, 2582:7421.

[25] Gwenael Le Bodic. *Mobile messaging technologies and services: SMS, EMS and MMS*. John Wiley & Sons, 2005.

[26] Gwenaël Le Bodic and Hatim Zaghloul. *Mobile Messaging Technologies and Services: SMS, EMS, and MMS*. ABC Publishing, 2022.

[27] Matti Leppäniemi and Heikki Karjaluoto. Mobile marketing: From marketing strategy to mobile marketing campaign implementation. *International Journal of Mobile Marketing*, 3(1), 2008.

[28] Wes McKinney and PD Team. Pandas-powerful python data analysis toolkit. *Pandas—Powerful Python Data Analysis Toolkit*, 1625, 2015.

[29] Uttam Kumar Roy Jubayer AI Mahmud Md Shofiqul Islam, Shanjida Sultana. A review on video classification with methods, findings, performance, challenges, limitatios and future work. 2020.

[30] A Medani, Abdullah Gani, O Zakaria, AA Zaidan, and BB Zaidan. Review of mobile short message service security issues and techniques towards the solution. *Scientific Research and Essays*, 6(6):1147–1165, 2011.

[31] Ajay R Mishra. *Advanced cellular network planning and optimisation: 2G/2.5 G/3G... evolution to 4G*. John Wiley & Sons, 2007.

[32] Anuar Manuel Nader Meljem. Django rest framework (drf) secure code guidelines. 2023.

[33] Pavas Navaney, Gaurav Dubey, and Ajay Rana. Sms spam filtering using supervised machine learning algorithms. In *2018 8th international conference on cloud computing, data science & engineering (confluence)*, pages 43–48. IEEE, 2018.

[34] Statista's own team of researchers and analysts. *Number of mobile messages worldwide from 2019 to 2023 (in trillions)*. https://fr.statista.com/, 2020.

[35] Chris Park. *A Dictionary of Environment and Conservation*. Oxford University Press, 1 edition, 2012. Online Publication.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[37] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[38] Sebastian Raschka and Vahid Mirjalili. Python machine learning: Machine learning and deep learning with python. *Scikit-Learn, and TensorFlow. Second edition ed*, 3, 2017.

[39] Muhammad Abdulhamid Shafi'I, Muhammad Shafie Abd Latiff, Haruna Chiroma, Oluwafemi Osho, Gaddafi Abdul-Salaam, Adamu I Abubakar, and Tutut Herawan. A review on mobile sms spam filtering techniques. *IEEE Access*, 5:15650–15666, 2017.

[40] Houshmand Shirani-Mehr. Sms spam detection using machine learning approach. *unpublished) http://cs229. stanford. edu/proj2013/Shir aniMeh r-SMSSpamDetectionUsingMachineLearningApproach. pdf*, 2013.

[41] Ali Hassan Sial, Syed Yahya Shah Rashdi, and Abdul Hafeez Khan. Comparative analysis of data visualization libraries matplotlib and seaborn in python. *International Journal*, 10(1), 2021.

[42] De S Sirisuriya et al. A comparative study on web scraping. 2015.

[43] Alex Smola and S.V.N. Vishwanathan. *Introduction to Machine Learning*. Cambridge University Press, Cambridge, UK, 2010.

[44] Jonathan Stark, Paco Nathan, John Papaconstantinou, Paco Lagerstrom, and Paco Hope. *Building Android apps with HTML, CSS, and JavaScript*. " O'Reilly Media, Inc.", 2010.

[45] Siyuan Tang, Xianghang Mi, Ying Li, XiaoFeng Wang, and Kai Chen. Clues in tweets: Twitter-guided discovery and analysis of sms spam. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2751–2764, 2022.

[46] Aria Teimourzadeh, Samaneh Kakavand, and Benjamin Kakavand. Application of python in marketing education: a big data analytics perspective. *Marketing Education Review*, pages 1–16, 2022.

[47] Sandro Tosi. *Matplotlib for Python developers*. Packt Publishing Ltd, 2009.

[48] LV Tulchak and AO Mapchuk. *History of python*. PhD thesis, 2016.

[49] Antony Unwin. Why is data visualization important? what is important in data visualization? *Harvard Data Science Review*, 2(1):1, 2020.

[50] Stefan Van Der Walt, S Chris Colbert, and Gael Varoquaux. The numpy array: a structure for efficient numerical computation. *Computing in science & engineering*, 13(2):22–30, 2011.

[51] Eric Wohlgethan. *Supporting web development decisions by comparing three major javascript frameworks: Angular, react and vue. js*. PhD thesis, Hochschule für Angewandte Wissenschaften Hamburg, 2018.

[52] Phd. Elie Zihindula. Courses of multivariate statistical analysis. (23):2–10, 2023.

[53] Pacifique Zikomangane. A free wireless network in the drc: An answer to internet shutdowns and exorbitant access costs. 2018.

[54] Michel Lutz Éric Biernat. *Data science : fondamentaux et études de cas. Machine learning avec Python et R*. Eyrolles, 61, bd Saint-Germain 75240 Paris Cedex 05., 2015.